

DESCEND 2 EDITING



VON DER PIKE AN

	Type Name
15	POWERUP
16	POWERUP
17	POWERUP Homing
18	POWERUP MassDriver
19	POWERUP Napalm
20	POWERUP Fusioncannon
21	POWERUP Plasmacannon
22	POWERUP QuadLaser
23	POWERUP Superlaser
24	POWERUP Superlaser
25	POWERUP Frag
26	POWERUP Fusioncannon
27	POWERUP Plasmacannon
28	POWERUP QuadLaser
29	POWERUP Superlaser
30	POWERUP Superlaser
31	POWERUP Superlaser
32	POWERUP Superlaser
33	POWERUP Superlaser
34	POWERUP MassDriver

English
v.1.0.469

Top [?]

MovementType: None

- Set user flag ein to TRUE
 - UserFlag: ein
 - True/False: TRUE

Script 000: Start Waffe

- Owner: waffe (Object)
- Event: Collided (with IT)
- If the following condition is met:
 - AND
 - (IT is a player or player weapon) is
 - Bool: IT is a player or player w
 - Object: IT
 - Operation: is TRUE
 - (User Flag ein) is TRUE
 - Bool: User Flag ein
 - UserFlag: ein
 - Operation: is TRUE
- Then perform the following actions:
 - Set movement type for waffe to Physics
 - Object: waffe
 - MovementType: Physics
 - Turn ON spew from waffe at gunpoint
 - Object: waffe
 - GunNum: 0
 - SpewType: Blue Fire
 - Mass: 0.00
 - Drag: 0.00
 - PhysicsFlags: [NO FLAGS SET]
 - IsRealObject: FALSE
 - BlobLifetime: 1.50
 - BlobInterval: 0.15
 - SpewLife: 30.00
 - Size: 4.00
 - Speed: 20.00
 - Minimize: TRUE
 - SpewHandle: None
 - Turn ON spew from waffe at gunpoint
 - Turn ON spew from waffe at gunpoint
 - Turn ON spew from waffe at gunpoint
 - Turn ON spew from waffe at gunpoint
 - Turn ON spew from waffe at gunpoint
 - Turn ON spew from waffe at gunpoint
 - Generate object waffe timer event in 3
 - Object: waffe
 - Time: 30.00
 - TimerID: waffen_timer
 - Set user flag ein to FALSE
 - UserFlag: ein
 - True/False: FALSE

Script 002: ist timer beendet?

- Owner: waffe (Object)
- Event: Timer (TIMER ID) Went Off
- If the following condition is met:
 - (TIMER ID) = (waffen_timer)
 - TimerID: waffen_timer
 - Operation: =
 - TimerID: waffen_timer
- Then perform the following actions:
 - Set movement type for waffe to None
 - Object: waffe
 - MovementType: None
 - Set user flag ein to TRUE
 - UserFlag: ein
 - True/False: TRUE

To the
Community.

CONTENTS

❖ A Quick Word in Advance

❖ Main Section The Program

❖ Section A - Creating a Level From Scratch

001 - Editor's Instructions.....	20
002 - The First Room.....	23
003 - Change the Room.....	27
004 - Texturing.....	29
005 - The First Level.....	31
006 - Add A Second Room <>.....	34
007 - Reactor Room <>.....	37
008 - Reactor <>.....	40
009 - Lathe a Room <>.....	43
010 - The Bend Tool <>.....	45
011 - Basics.....	48
012 - A Simple Room.....	51
013 - Start a Level <>.....	54
014 - Texturing <>.....	56
015 - Lighting <>.....	58
016 - Test The Level <>.....	62
017 - Add New Rooms <>.....	64
018 - Objects & Player Starts <>.....	66

❖ Section B – Deepening into D3Edit

019 - View Shortcuts.....	69
020 - The Geometry Panel.....	70
021 - Set up D3Edit & Quicktest.....	84
022 - Backup Editor Settings and Profiles.....	89
023 - Multiplayer Defaults Modes.....	94
024 - Apply Tools menu.....	97
025 - Object Info Dialog (OI).....	98

❖ Section C – Advanced Building

026 - Building a High Quality Shell <>.....	107
027 - Multiple Layers <>.....	110
028 - Multiple Extrusion <>.....	114
029 - Combine Multiplanar Extrusions <>.....	118
030 - Share Space.....	122
031 - Vertex Operations on Faces.....	125
032 - Select Segments.....	126
033 - Bend – Basics.....	128
034 - New Style Bend, v39.....	130
035 - The Bend Function <>.....	133
036 - Redesigned Bend Function in v40.....	135
037 - Bend: Mathematics.....	137
038 - Bend, Objects Included.....	139
039 - Bend Without Bend <>.....	141
040 - Lathe – Deepening.....	142
041 - Extrude – Depression.....	145
042 - Extrude: Zero.....	151
043 - Mirror, Mirror...<>.....	158
044 - Room within a Room <>.....	161
045 - Make a Sphere.....	163
046 - Complexity Made Easy <>.....	171
047 - Primitives.....	182
048 - Create rock Formations.....	188
049 - Cave Style Tunnel.....	190



❖	Section D – Debug	
	050 - Concave Faces.....	198
	051 - Non-Planar Faces.....	199
	052 - T Joints.....	201
	053 - T-joints: Second version.....	202
	054 - Verify your Level.....	206
	055 - Verify your Mine.....	211
❖	Section E – Terrain	
	056 - Terrain Implementation.....	218
	057 - Create Outside World <>.....	221
	058 - Outside World – Update <>.....	227
	059 - Building.....	229
	060 - Player Start In The Outside World.....	231
❖	Section F – Textures	
	061 - Custom Textures.....	236
	062 - Apply Custom Textures <>.....	238
	063 - Custom Textures: Tips.....	242
	064 - Make Textures Tileable.....	245
	065 - Partially Transparent Textures.....	248
	066 - Animated Textures - Screenshots.....	254
	067 - Animated Textures - Homemade <>.....	261
	068 - Textures @ 256x256.....	262
❖	Section G – Sounds	
	069 - Play sounds.....	265
	070 - Insert Sound Sources.....	268
	071 - Terrain Sounds.....	269
	072 - Sound Sources On The Terrain.....	270
	073 - Music In Your Custom Descent 3 Level.....	272
	074 - Sound: Avoid Problems.....	276
❖	Anticipation: Getting DALLAS to Work	
❖	Section H – Doors & Objects	
	075 - Doors Tutorial <>.....	285
	076 - Custom Doors <>.....	288
	077 - Sound For Your Custom Door.....	294
	078 - Custom Objects 1 <>.....	295
	079 - Custom Objects 2 <>.....	297
	080 - Custom Objects 3 <>.....	301
	081 - Custom Robots Part 1 <>.....	305
	· Papacat's Custom Weapon Workshop.....	307
	082 - Introduction & Overview.....	308
	083 - The Powerup GAM.....	310
	084 - The Weapon GAM.....	311
	085 - A Simple Primary Weapon <>.....	315
	086 - The Ship's GAM.....	319
	· OOFEdit Excursus.....	321
	087 - OOFEdit: Terminology.....	323
	088 - OOFEdit: Features.....	324
	089 - OOFEdit: Getting started.....	325
	090 - OOFEdit: A Powerup.....	328
	091 - OOFEdit: Turrets <>.....	330
	092 - OOFEdit: Animation <>.....	333
	093 - Ship Tutorial.....	337
❖	Section I – Scripting	
	094 - Scripting: Hello World.....	343
	095 - Scripting: Introduction.....	346
	096 - New Dallas Search Functions.....	348
	097 - Make Weather.....	356
	098 - Refined Player Respawn.....	358
	099 - Teleporter <>.....	359
	100 - Create Lighting <>.....	360

101 - Wind Scripting.....	362
102 - Music Regions.....	363
103 - Spew!.....	366
104 - Broken Glass With Shards <>.....	368
105 - Scripting a Switch I.....	372
106 - Scripting a Switch II.....	375
107 - Alert!.....	378
108 - Scripting An Inventory Cloak <>.....	381
109 - Reactor Gamma: Script Companion.....	385

❖ **Section J – Designing**

110 - Level Design Rules.....	388
111 - Single Player Level Building.....	391
112 - Single Player.....	393
113 - Anarchy.....	395
114 - Anarchy.....	397
115 - Team Games.....	399
116 - When Great Levels Die.....	401
117 - Eye Candy.....	408
118 - Weapon Balance.....	411
119 - Why Build Missions?.....	413

❖ **Section K – Single Player**

120 - Level Goals.....	416
121 - Robot Centers.....	417
122 - Building According To Plan.....	419
123 - Paths.....	421
124 - BNode Tutorial.....	425
125 - Set BNodes.....	427
126 - Player Respawn Points.....	431
127 - Include Load Screen.....	432
128 - Level Briefing.....	434
129 - Ship Choice In Single Player <>.....	439

❖ **Section L – Master Class**

130 - Optimize Doors.....	442
131 - Dynamic Light.....	447
132 - Group Files.....	450
133 - Customs, again.....	454
134 - Building with Objects.....	455
135 - Use Mercenary Bots.....	460
136 - Scripting & Matcen <>.....	463
137 - Partially Scale Rooms.....	467
138 - Composite Objects <>.....	469
139 - Recycling.....	475
140 - Open Mercenary Levels.....	478

❖ **Appendix – Useful information**

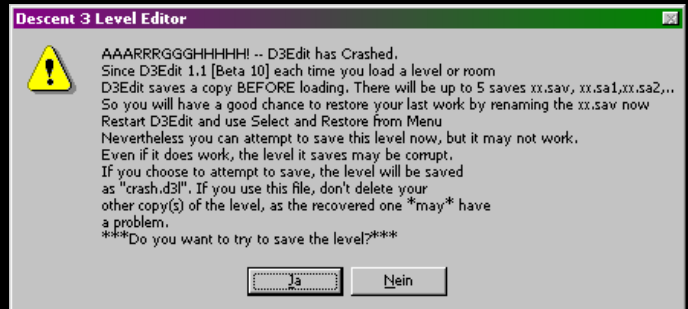
List of all D3 Tools.....	480
House Of Mirrors.....	485
In-Game Help For Testing.....	486
D3-Edit Limits.....	488
Application of D3 Levelinfo.....	489
Documentation Of the Doors In Descent 3.....	492
Have Custom Objects Visible In D3Edit.....	517
The Gam Format: The Specification File.....	518
Epilogue	
Shortcut List.....	540

<> Indicates that an example file belongs to this chapter.

A Quick Word In Advance...

The purpose of this work is to provide as complete a picture as possible of how to build levels for Descent 3, which are not only well constructed and error-free, but also attractively designed.

I started this file with the tutorials which relate to www.descent3fischlein.de condition. In my opinion, it is more practical to have a single work to work through. I revised everything, added missing information, including graphics to help explain it. The files that belong to the respective chapter are also numbered accordingly. Since there are more tutorials on the internet, I decided to incorporate them here and translate them if necessary, so that a comprehensive work is created that I can add to at any time; In the meantime, every scrap of information on the topic that I could get my hands on has been included in this file. However, this results in



You'll see this dialog more often... it's probably a good idea to get into the habit of buffering more often. A pleasant side effect is that you do this with all programs, and it's not a mistake 😊

that one topic or another is covered several times. Topics further back usually require knowledge of the previous ones. When translating, I tried to stick to the original as much as possible, including the humor if possible, but where there are simply terms that cannot be translated 1:1, I tried to preserve the meaning of the text.

Of course, the rights of creation remain with the respective authors!

There are conventions in this document that are intended to make it easier to use:

filename.ext	Words in this font refer to files or file names and their paths.
Ctrl+Alt+Del	keyboard shortcuts; Uppercase & lowercase letters are synonymous, Shift is specified separately. For numbers without additional information: always those from the number block!
To Current	Buttons to click or titles of dialog boxes.
Go there	Link - this allows you to move within the document.
File->New...	This font indicates a program menu choice, or something displayed by the editor; for example in the status bar or dialog boxes.
Is -AGhl	This displays inputs or outputs, including program code or lists.
<i>It is as is</i>	Comments from me are written in 8 points.
Tx	This refers to the buttons in the toolbar.
Danger:	You should take information in red-framed boxes seriously - really!

The editors used are 1.1 [Beta 10 | Atan P.0_1_39(p)] - called v39 for short; and the 1.1 AV 40, called v40 for short, which brings many innovations, improvements and more convenience.

So Let's Go,

Ragil Ral

Main Section - The Editor

Ragil Ral

Do you know the fairy tale of the land of milk and honey? Good! Because as with all great things, an obstacle has to be overcome before the actual learning, trying and building can take place. If you skip this section you can still do it, but it will take significantly longer - unless you have acquired knowledge elsewhere that applies here.

What is it all about anyway?

Anyone reading this wants to create missions for Descent 3 and learn how to do it now, damn it. And of course as quickly as possible, 🙏 that's quite clear.

Requirements

All you really need is basic computer knowledge and a little English (or the dictionaries at your desk), the rest is in here and started with this line. Furthermore, a little patience is required, as if you really want to learn something, you can't just chew through a topic once; Anyone who plays a musical instrument will know what I mean... practice makes perfect, here too. Try around. And for heaven's sake, practice, practice, practice! But that doesn't mean that you should always do the same thing, but rather that you simply try out what you've learned extensively. A calculator proved to be a very practical tool. Of course you also need D3Edit.

Basic elements

In Descent 3 everything is made up of polygons, called 'faces', which have 'edges' as edges and 'vertices' as vertices. Objects like our Pyros are also made of them (they are actually space files that have been given 'life'). That's why the editor is specialized in these, and everything you do with the editor (from

Always get the latest editor! It's in the Descent forum, at
"D3Edit - Development - Forum / D3 Editing Tools":

<http://www.descentforum.de/forum/viewtopic.php?t=2577>

You will also get other tools there that you will need later.

Objects such as bots, powerups or Scripting aside) is to manipulate these basic elements in such a way that the end result is as cool a level as possible. The individual elements are numbered which starts with 0.

Current and Marked

'Current' is a little difficult to translate: 'current', 'currently valid', 'present'. This refers to a selected (clicked) element, and before an element can be edited, it must at least be 'current'. This differs from 'marked' in that when you click in an active view, the element that lies under the mouse cursor becomes 'current', corresponding to the operating mode. Marked elements, however, remain marked, even when the work mode changes.

It should also be pointed out that the game (being the machine program that it is) actually only performs mathematical operations on what you create as a level. Therefore, mathematical laws also apply and it is an advantage to know them. This makes it easier to avoid errors or to understand how a vertex works or why there are errors.

You'll get to know the individual buttons, menus and whatever else is lying around later. For now, we will only discuss the rough structure of D3Edit.

General Tips For D3Edit

I learned while making this file that it's better to think BEFORE you do ANYTHING. You may have seen the dialog box in the introduction, and you probably don't want to see it right after you've been creative for an hour and haven't saved anything. Before you click any button, make a menu selection, or do any serious fiddling with the basic elements of the level, save your stuff. This will also be pointed out again and again. Since v40, the editor creates backup copies, called levelname.sav and levelname.sa1 until levelname.sa5. The higher the number in the ending, the younger. They are created when you open a level.

The Status Line

It is located in the lower part of the program window and shows you all sorts of information, such as which operating mode you are in, where your cursor is in the editor, etc., etc. Since v40 you can set a value for a specific function here.

R: -1/0 F: 150/256 P: -1/0 E: 0/4 V (idx): 0/4 (136/242)	Grid: 0	0, -4, 70	0, 0, 0	Vertex mode	Marked: V: 0 F: 0	Conc: 0.0010	PtE: 0.000250
--	---------	-----------	---------	-------------	-------------------	--------------	---------------

Ready

You will gradually find out what means what. Make her your friend!

Units

There is a unit system in the editor. I don't yet know exactly how big such a unit actually is (tests so far lead me to believe that a unit could be about a yard), but they are decisive (well, they are units). If you stand directly against a wall with the pyro in front, the field of vision is about eight units wide. The displayed coordinate grid shows you where you are; its grid constant determines the values with which the elements are manipulated in the editor. You can change the grid constant either via the **PgUp/PgDwn** buttons or via the context menu, always for the respective orthogonal view. In addition, you can also enter the lattice constant directly via **1** until **7** (Select the number row above the letters).





You can come with me **G** Turn off the coordinate grid completely and thereby move verts, faces and everything else by 0.1 units, which can cause difficulties. Avoid editing constructions that are smaller than 1 unit.

Behavior When Editing

Don't be frugal. Anything that is too much can be removed, but if something is really missing somewhere, patience is usually required; However, this does not apply in all respects. Sometimes you have to be careful where you add something; However, this is then (usually) pointed out. Save your work more often under different names so that you can always access older files, i.e. older versions. Every now and then an idea comes to you, but it can no longer be realized with the expanded room and/or level, then something like this is really practical. When the level is completely finished, you can still make the big bet **Delete** button takes place.

Modes

D3Edit has four different operating modes, each of which regulates certain matters, and you often have to switch back and forth between them while building levels. This can be done using key combinations or the buttons in the toolbar.

Hotkey	Button	Mode	Purpose
Ctrl-R		Vertex Mode	Creating and editing vertices, creating faces
Ctrl-F		Face Mode	Creating faces, editing and texturing them
Ctrl-G		Object Mode	For handling objects such as powerups or bots
Ctrl-H		Path Mode	To give 'intelligent' objects certain specifications

BytabIf you switch between Vertex Mode and Face Mode, you were first in one of the other modes
tabone in vertex mode.

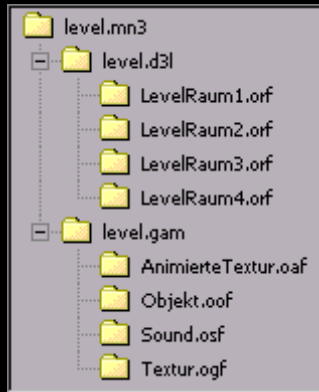
In fact, the modes overlap a little in their possibilities, but that's fine



Files

A Descent 3 level consists of various resources, not just the level.

First of all, the file formats: rooms get the file extension .orf, get the levels .d3l; then there is still .oof for objects, .oif and .oaf for textures and even .dll, in scripting; There is also a format for audio (.osf) and some more. The finished, playable level then has - as you probably know - .mn3; This file is a compilation of all the resources belonging to the level and is created using a tool that comes with D3Edit.



The structure of a level file is roughly reproduced here on the left.

Tip:

When testing, it is advisable to move all missions in the D3 mission directory away beforehand. Simply create a folder in the mission directory and put everyone in there so that only your level is in the Mission directory is located. Descent 3 loads much faster and you can find your level straight away.

In the editor you create rooms from the basic elements mentioned and then put them together into levels, possibly with an outside world; this will then be the level file. The individual rooms can be saved separately, which is practical - for example, you can create a lamp with the right textures, save it, and then install it at any time wherever you need light. There are also space files to download, do a little digging in the forums. And by the way: the doors are also their own rooms, just like all objects!

But not just the level file; you can build your own sounds, textures and objects into the level. In the end it's all shoved into a file, and that's it. .mn3, the playable level. You can also use DALLAS (which will be discussed in detail later) to program 'life' into a level, so to speak, and even determine the behavior of the bots; then another one will come. .dll and a .msg in addition. The AI that was programmed into the game has interfaces that can be accessed via DALLAS. If you want to know more about game AI, I recommend 'Developing an Artificial Intelligence Engine', by Michael van Lent and John Laird, created at the Artificial Intelligence Lab, University of Michigan; the boys have at the Descent 3 AI co-programmed. (tools/coding/BotsInDescent3&QuakeII.zip)

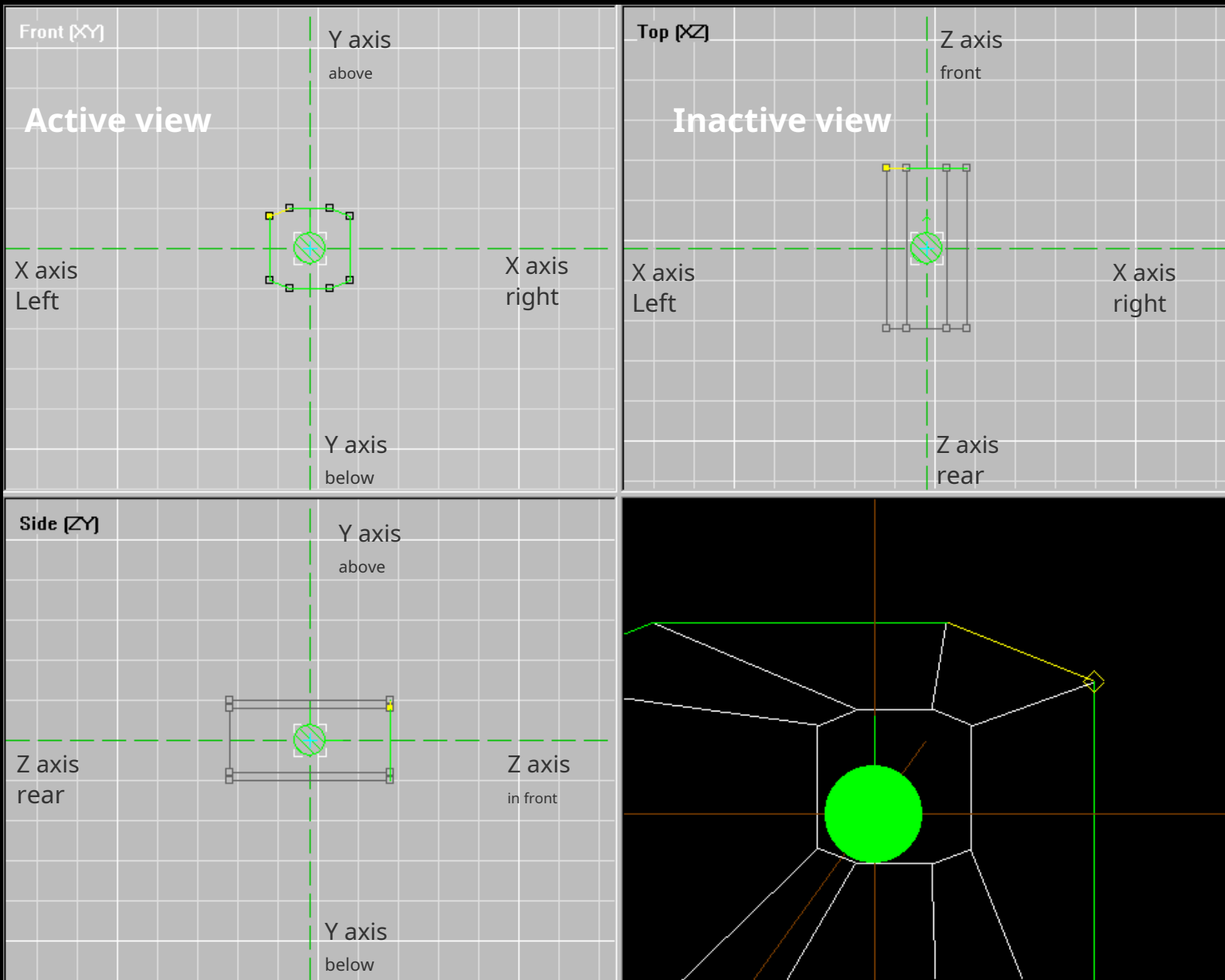
Ready To Take The Plunge?

Goes up File->New->New Level - Default Room. With this, the editor creates a valid (even playable!) level; This is ideal for trying things out, as you don't lose anything if things go wrong.

The Views

There are two- and three-dimensional views, and they differ, not only in what they show, but also in their function. The 2d views are actually correctly called 'orthogonal' views ([Latin orthogonium < Greek orthogonion] (Geom.): rectangle) and they only ever show two axes - but then there are three of them. They only offer the line view. The 3D views, on the other hand, have three display types: Lines, Textured and Lines, and Textured.

2d Views



To work in a view, you first have to activate it by clicking in it once; it then becomes a little brighter and the name becomes white. The next click will already have an effect.

This is where you make the constructions and insert elements. You can see the grid in the screenshot below, this is the coordinate system of the editor. It can with **G** can be switched on/off and determines the increments in which movements or changes take place. Increment means here that when elements are moved or rotated, this occurs with certain fixed values (table). You can also change the origin point of the grid using the button on the left. But be careful in the 2d

Views: **Ctrl-click** also puts the reference center into an active view! You can do it with the button from the construction panel (Open with) but **set** again. The views are always seen from the player starting point.

Increment list			
Grid size Units	linear increment Units	Angle increment Degree	Angle increment Quadrant shares
1	set Grid size	1.4063	Quad/64
2		2.8125	Quad/32
5		5,625	Quad/16
10		11.25	Quad/8
20		22.5	Quad/4
50		45	Quad/2
100		90	Quad
Grid off	0.1	correspondingly set grid size	

3d Views

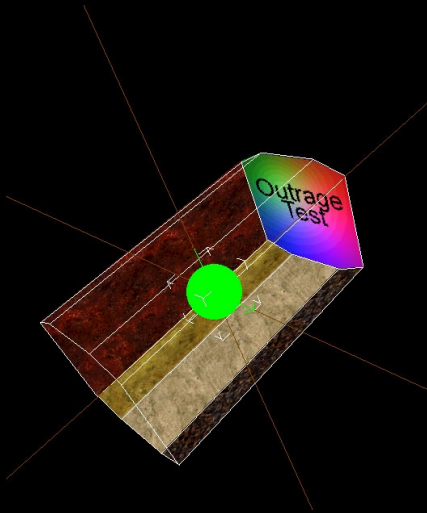
The 3D view shows the room/level and gives a preview of what it will look like in-game. It shows you a perspective view of your work. Here you manipulate what you have created: you select areas and texture them, or edit them further in one of the 2D views. However, you can also make limited changes to the geometry here.

Move in the 3D view - always go where you want

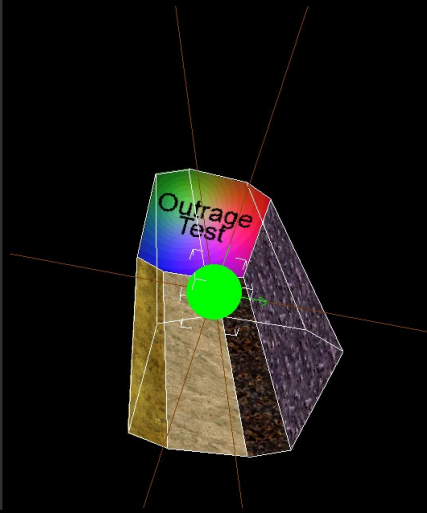
With **Ctrl-click**en and hold/drag in a 3D view to rotate it. If you've tried it out, you'll have noticed that sometimes it's not that easy to get what you want into view.

If you move the mouse horizontally, it rotates horizontally; if you move the mouse vertically, it rotates vertically. The reference point here is always the display!
It takes a little practice, but it's like riding a bike.

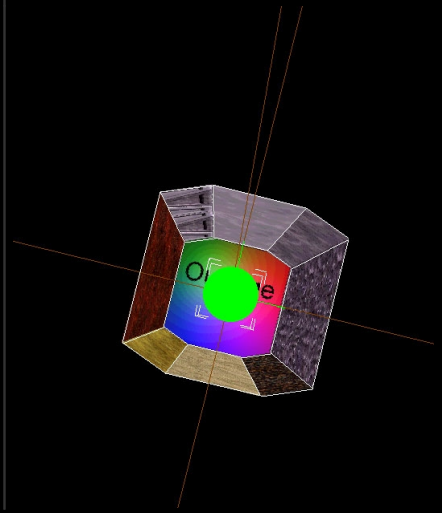
All axes look somewhere. Move the mouse horizontally and rotate the display until you get to a level.



A layer is now looking at you.
Now move the mouse perpendicular.



Now you look directly at an axis and see the axis cross, here YZ.



This is just a simple room, but as complexity increases you have to maneuver more and more precisely in the 3D views.

With **Shift-click**en/hold and drag to move the view.

There are other navigation options, these can be found in the context menus; learn the keyboard shortcuts for it. I find it most practical **Ctrl-Shift-C** 😊

The Views 😊

There is the 'World View', the world view and the 'Room View', the spatial view. In the world view you can see the entire level, with the objects used and, depending on the setting, a different number of rooms (set in File->Settings).

The World View

only has the 3d view. You can texturize and add rooms here, and the treatment of the outdoor terrain also takes place here. If you have several rooms in the level, the Room View always shows the current room.

The Room View

shows you either the room that is current in the world view, or a separate room that you are working on. Only here can you make extensive changes to the geometry or insert objects.

In order to have full control, you now need knowledge about the...

Context menus of the views

This allows you to show or hide things; with the 3D views you can determine the type of display. Pay special attention to **center...** and **Move camera to...**, this will bring things back to your screen if you happen to be totally lost. Center centers the 3D view on the corresponding object, which means you then rotate/pan around it. Move camera just brings it into your field of vision. Additionally there is also **AltGr-C**, this resets the view and centers it on the current room.

Room View - 2d view

✓ Show attached rooms	
✓ Show room center	
Show valid Multiplayer mine area	
✓ Show <u>v</u> ertices	
Show <u>n</u> ormals	
✓ Show <u>o</u> bjects	
Show <u>g</u> rid	G
✓ Snap to grid	
Adjust grid size	▶
Display Path Nodes	
Display AI-Nodes	
Display current room AI-Nodes only	
Center current <u>f</u> ace	Shift+C
Center current <u>o</u> bject	
Center current <u>r</u> oom	C
Center <u>o</u> rigin	Home
Select Face by Number	

Room View - 3d view

Textured	
✓ Textured with Outline	
Show animated texture frames	
Wireframe	
Draw marked Verts	
✓ Show object polymodels	Ctrl+P
✓ Display Path Nodes	
✓ Display AI-Nodes	
Display current room AI-Nodes only	
Center current <u>f</u> ace	Ctrl+Shift+C
Center current <u>r</u> oom	C
Center <u>o</u> rigin	Home
Move camera to Current <u>R</u> oom	Shift+M
Move camera to <u>C</u> urrent Face	Shift+C
Move camera to Current <u>O</u> bject	Shift+G
Move camera to Current Object's <u>V</u> iew	Ctrl+Shift+G
Move camera to Current <u>N</u> ode	Shift+D
Select Face by Number	

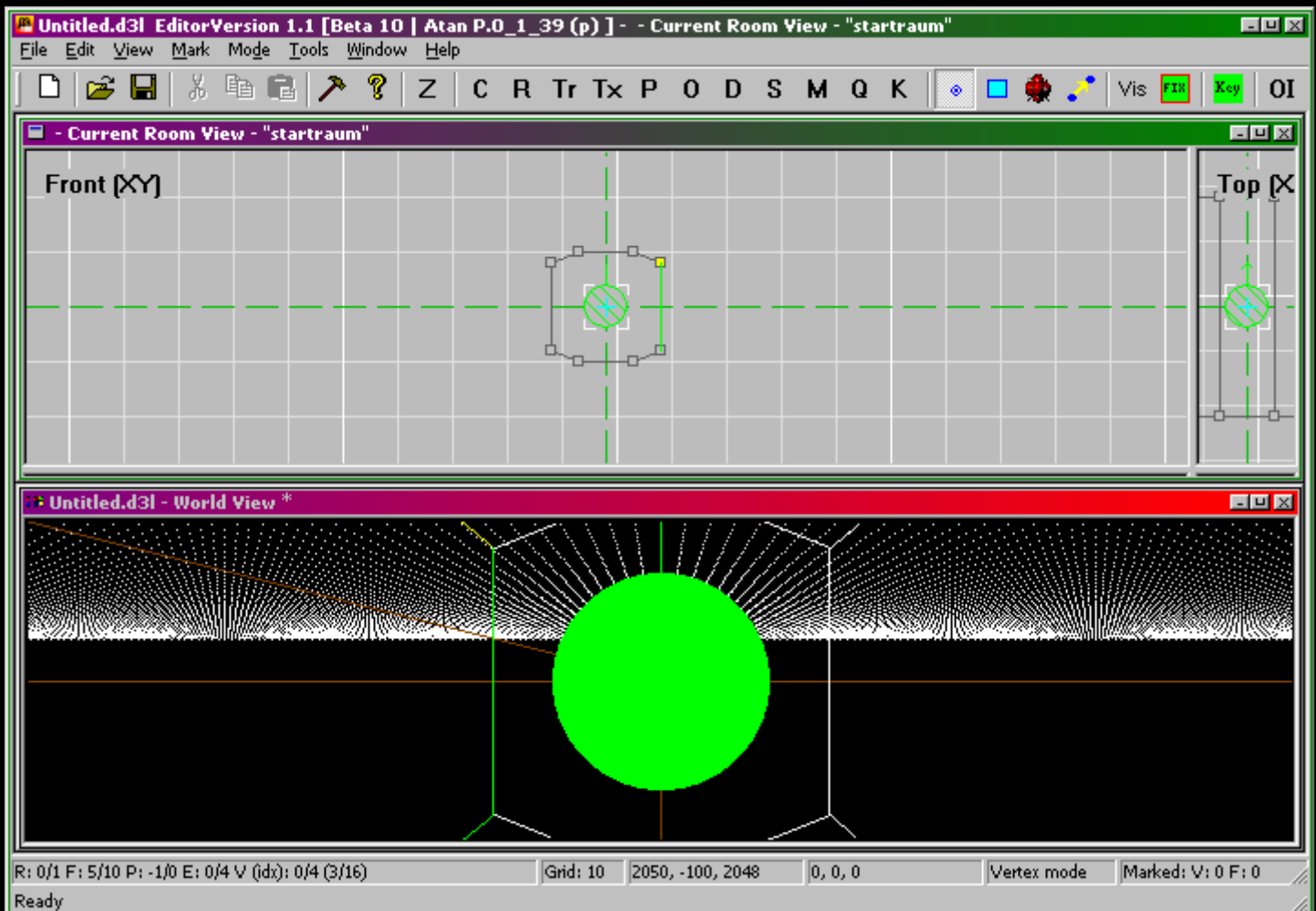
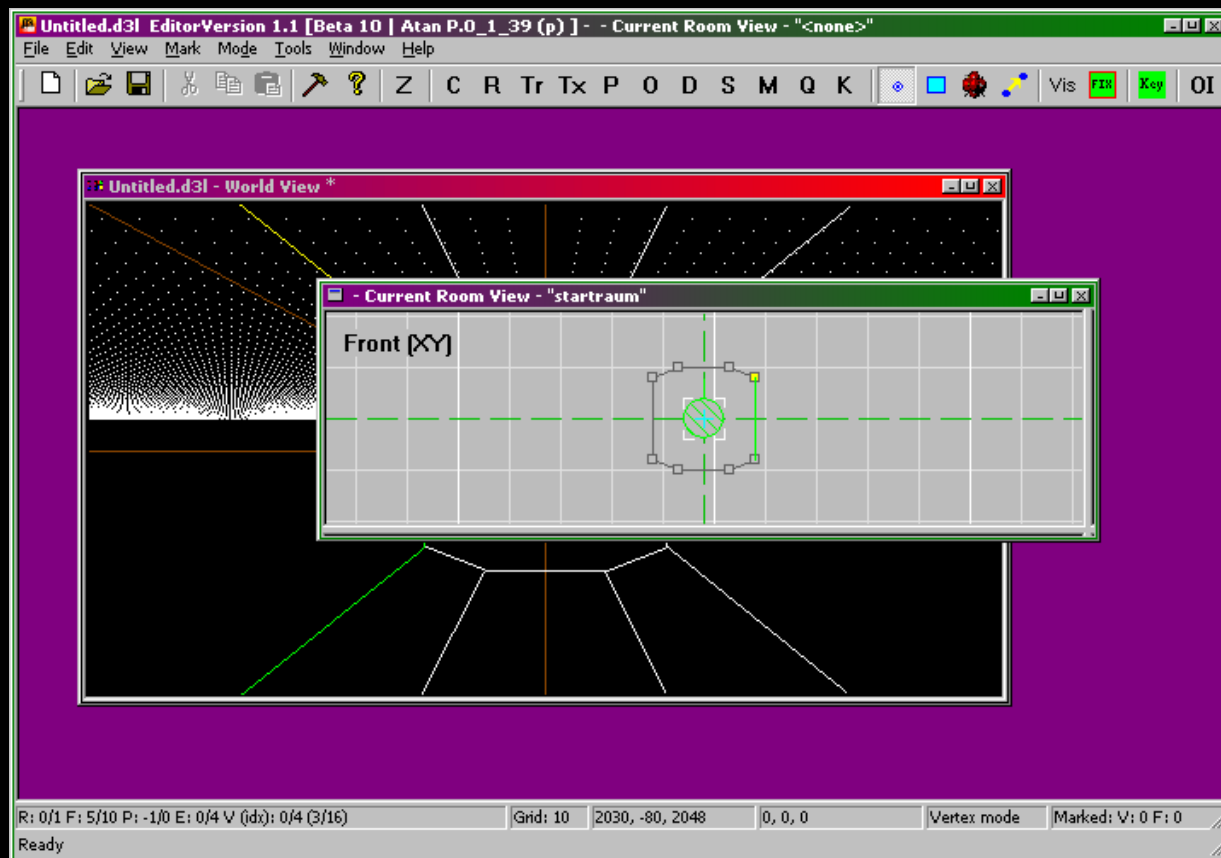
World View

Textured	
✓ Textured with Outline	
✓ Show animated texture frames	
Wireframe	
✓ Draw marked Verts	
Show terrain	Ctrl+T
✓ Show Sky Dome	
Show One Mine Room from Terrain only	
Display current room view	
✓ Display Path Nodes	
✓ Display AI-Nodes	
Display current room AI-Nodes only	
Center <u>m</u> ine	Home
Center current <u>r</u> oom	C
Center <u>o</u> rigin	
Move camera to <u>C</u> urrent Face	Shift+C
Move camera to Current <u>O</u> bject	Shift+G
Move camera to Current Object's <u>V</u> iew	Ctrl+Shift+G
Move camera to Current <u>N</u> ode	Shift+D
Show Sats	
Set Move/Rot Scale for Objects	▶
Set Move/Rot Axis for Objects	▶
✓ Display Axes at Current Object	
Select Room by Number	
Select Face by Number	
Select Object by Number	
Select Portal by Number	
Current <u>L</u> evel Room	

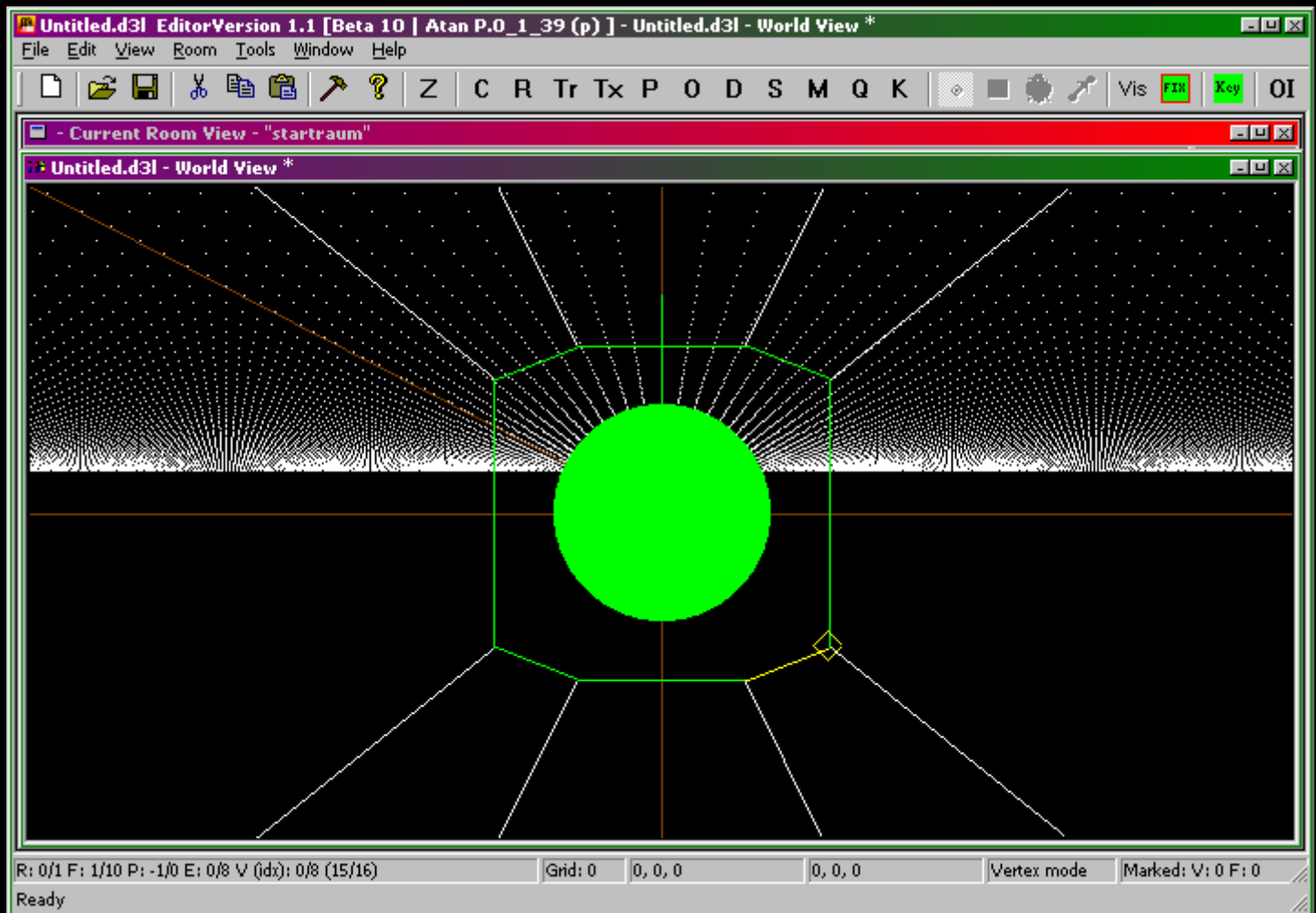
Another tip about the views, because you will soon notice that such a monitor offers very little space:

Under Window there is a choice Tile, which arranges the open file windows evenly within the program's window.

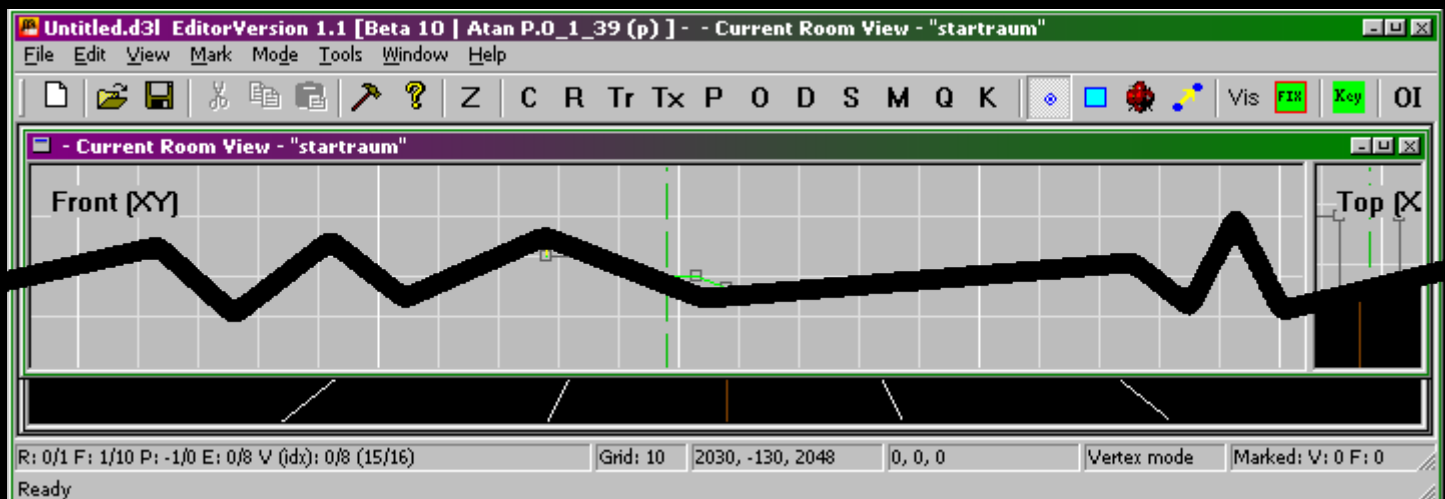
This will be the active window (here with the green title bar) as supreme arranged.



This allows you to open the windows so that the title bar of the window with the room is at the top and that of the world view is below it.



This has the advantage that you can always see which room you are in or what name it has, since when you switch to another room - in the world view - the room view always shows the current room and its name. In addition, you can then shorten the room view a little so that you are back in the world view with one click. If you have more than two windows open, you have to be with them **Ctrl-Tab** You can use it to switch through the windows.



World View - Wireframe view color code

If you move in the world view in wireframe mode, the lines of the level display have different colors. These mean:

Orange
indicates selected rooms.

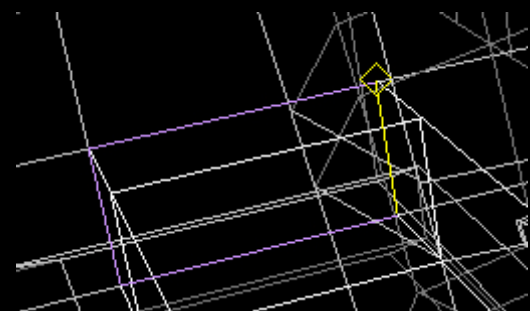
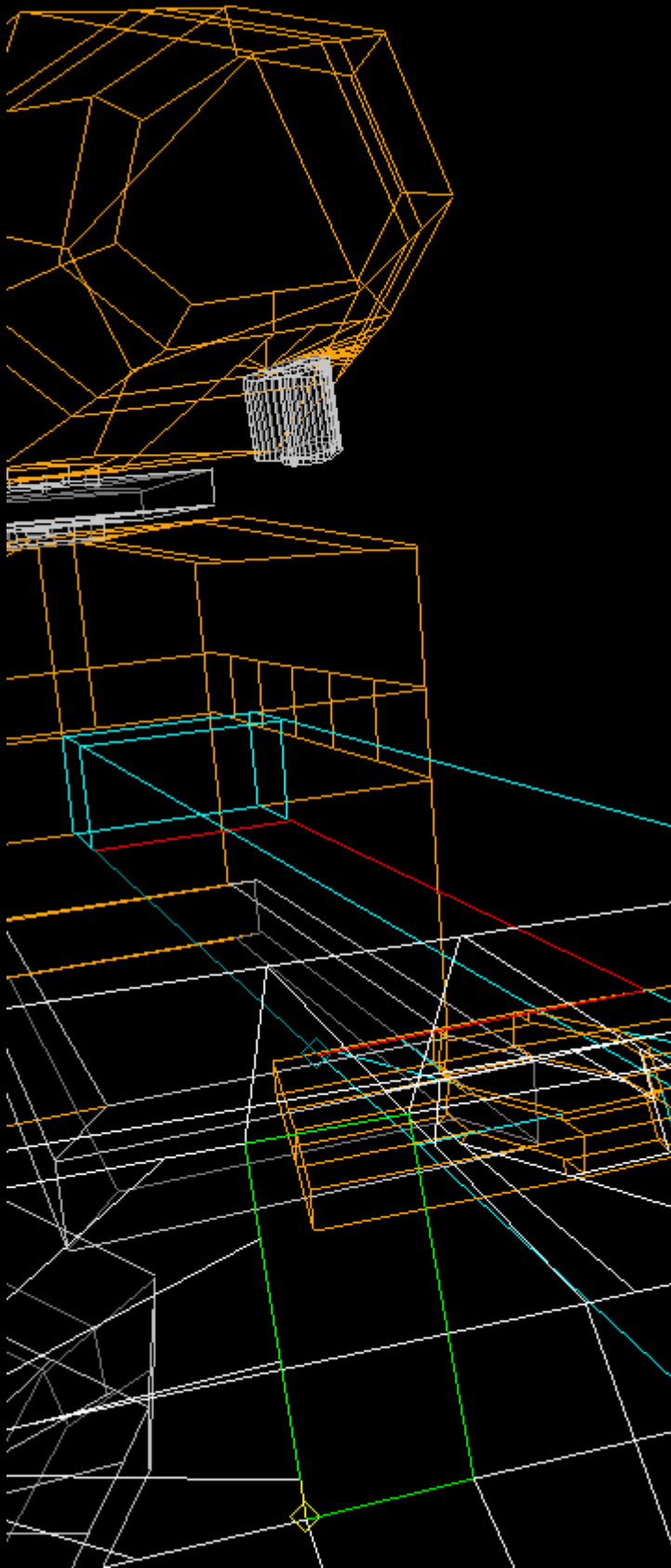
Not-so-white
the standard representation.

Turquoise
Marked space.

Red
Marked face of the marked room

White
Current Room; this is then displayed in the room view.
Green the Current Face in the Current Room, with Edge and Vert (the diamond) in **yellow**.

lavender
Portals if they
clicked (Fig
right).

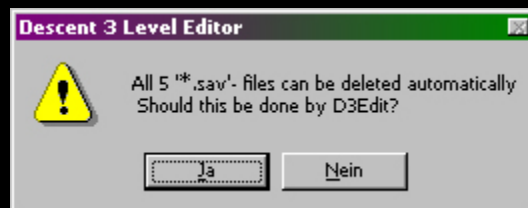
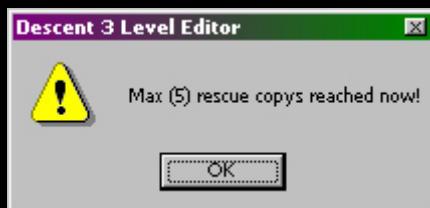


Update v40: the .sav*files

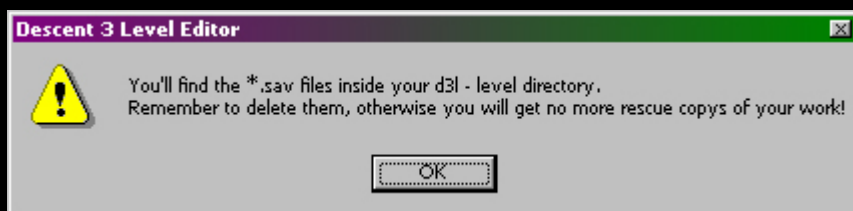
Since version v40, D3Edit creates a copy of it (up to six copies) when opening a room or level. These files are called <levelname>.sav and .sa1....sa5. The age of the backup copies is also listed in this order; the higher the number, the younger the file (right image).

If the maximum number of backup copies has been reached, D3Edit indicates this (bottom left image) and then offers to delete the backup copies automatically (bottom right picture);

Dateiname	Erstellt	Geändert
temprob.d3l	20.07.2008 14:08	01.10.2008 15:41
temprob.gam	23.07.2008 14:03	31.08.2008 12:51
temprob.qpd	26.08.2008 17:15	31.08.2008 13:41
temprob.sa1	01.10.2008 15:52	31.08.2008 12:42
temprob.sa2	01.10.2008 15:52	01.10.2008 13:38
temprob.sa3	01.10.2008 15:52	01.10.2008 15:16
temprob.sa4	01.10.2008 15:52	01.10.2008 15:21
temprob.sa5	01.10.2008 15:52	01.10.2008 15:25
temprob.sav	01.10.2008 15:52	31.08.2008 12:40
wide01.orf	23.07.2008 14:18	23.07.2008 14:18

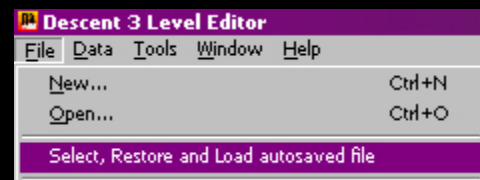


If you want to keep the backup copies, you have to copy them away or move them now. If you move them, you must click 'No' when asked to delete, otherwise D3Edit will issue an error message and will not open the level or room.



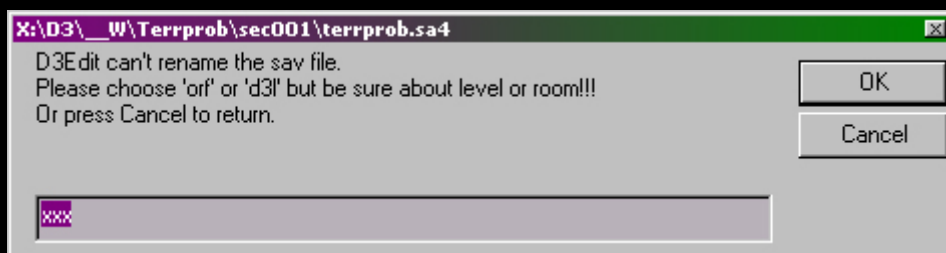
You will see this dialog if you click 'No' when asked to delete.

If something goes wrong, voteFile->Select, Restore and Load autosaved file and follow the instructions that come next.



It may be that D3Edit wants to know whether the restored file version was a room or a level, then this dialog box appears:

then enter the corresponding ending here.



This means there is a safety net for the first time.

Section A—Create a Level From Scratch

This is Descent3Fischlein's translation of Schplurg's walkthrough tutorials 'The Level'.

001	Editor's statement	This is supposed to be a short explanation Be an editor.	Schplurg	20
002	The first room	Here we show how to create 'Vertices' placed, 'Faces' created and the 'extruding' function explained.	Schplurg	23
003	Change the space	Add textures and the space change.	Schplurg	27
004	Texturing	The room is given a better look and light - Added textures.	Schplurg	29
005	The first level	An explanation of how to calculate the light in the room and you can see the level from the inside for the first time.	Schplurg	31
006	A second room add	Shows how to create a second room add and how to add one Builds energy tanker.	Schplurg	34
007	Reactor room	Here you can see how to create a room in a different way and added to the level.	Schplurg	37
008	reactor <i>without script!</i>	Here you can see how to work with the lathe tool and position things and player adds starting points.	Schplurg	40
009	Lathe a room	The lathe tool is used again and a room is created with it. This is then created using the snap function positioned.	Schplurg	43
010	The bend tool	Here we explain how to create curved tunnels and how to use them incorporated into the level.	Schplurg	45

This walkthrough was written by Robo and is slightly shorter than Schplurg's 'The Level'.

011	Basics	Basic concepts and views	Robot	48
012	A simple room	First start of the editor, first room	Robot	51
013	Start a level	Create a level correctly	Robot	54
014	texturing	Add texture to the room	Robot	56
015	lighting	Adjust and calculate lights	Robot	58
016	Test the level	One.mn3create and test	Robot	62
017	Add new rooms	Explanation of BOA & rooms attach	Robot	64
018	Objects & Player Starts	Essential for a level	Robot	66

[Toc](#)

= The Level =

By **Schplurg**

This is a complete walkthrough that creates a complete Descent3 level. Descent3Fischlein has translated the ten tuts into German.

It is recommended that everything EXACTLY is carried out as described, otherwise (at best) unforeseen results can be expected. I personally worked through it several times until I developed a 'fluid' feel; I also produced some very strange errors -Verify Mine: error-free, but then fly out of the level through the walls. It was also interesting that I was somewhere between '007 - Reactor Room' and '008 - reactor' lost the floor (in the literal sense!) - no idea where - and had to put it back in afterwards. Now, a few months later, of course I know what I did wrong...

If the first five tuts are too fast for you, I recommend this [Robo's Walkthrough \(from page 47\)](#), which basically covers the same thing but goes into a little more depth.



Add a few nasty bots and the testing will be more entertaining



001 - Editor's Instructions

Schplurg

I'm not repeating verbatim what Schplurg from Gameedit wrote in English!

(Descent3Fischlein)

The editor is a very good program that can be used to create high-quality levels. However, you have to get to know him first. I would like to briefly describe the D3Edit here.

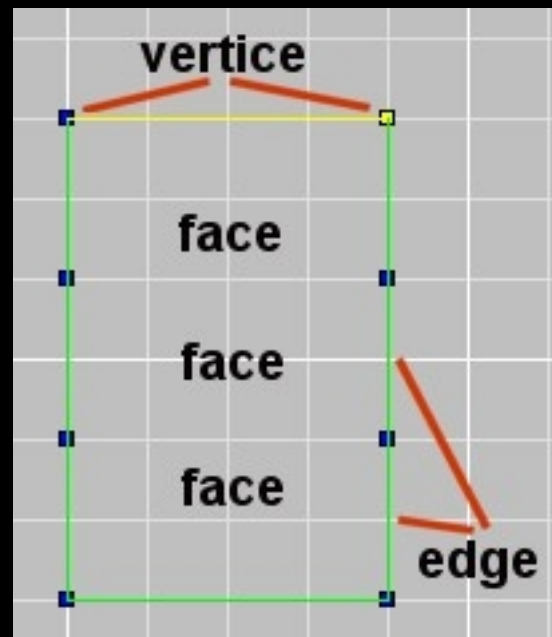
Faces

A face is two-dimensional and consists of three or more vertices. Imagine a room where you are sitting and look at a wall. It has four vertices and four edges. However, it is also possible to create a face with multiple vertices and edges.

Example:

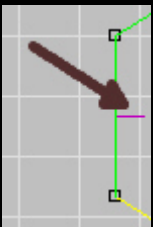
Here you can see what a face can look like. A face doesn't only need to have four vertices. It is also recommended that you add several vertices to a face or work with several vertices right from the start, because you can then split the face and design the textures differently. But more on that later. The vertices must also be on the same plane, otherwise you will create a 'nonplanar' face.

A face is a series of vertices and edges, there is a first, second, third



Shell

A 'shell' is a group of several faces that close off the space so that you cannot fly out of the level or room (see the level from outside). That would be a 'Bad Shell', which will be displayed in D3Edit under 'Verify Mine'. I will explain to you how to correct these 'bad shells' in another topic.



Normals

I don't think there's much to say about this. A 'Normal' is a small magenta line that indicates which side you can see a face from. A 'Normal' should always point into the interior of the respective room, otherwise you can fly out of the room.

Rooms

In every Descent3 level there is at least one room (should be several). The rooms are connected to each other using 'portals'. You can also connect the rooms using 'Doors', which may be better.

Objects

Objects are Powerups, Robots, Player Starts, Barrels, Soda Machines. These can be placed anywhere in the level.

Textures

Textures are images that you can use to design your level. With light textures you can illuminate your level effectively, but you have to think about where to place a light source to get a good effect.

Modes

Vert Mode - (Ctrl+r) - Vertices in the grid place, select, mark, copy & Insert, move vertices in the grid.

Face Mode - (Ctrl+f) - Insert Faces & delete, select, mark, copy /Insert, move faces in the grid

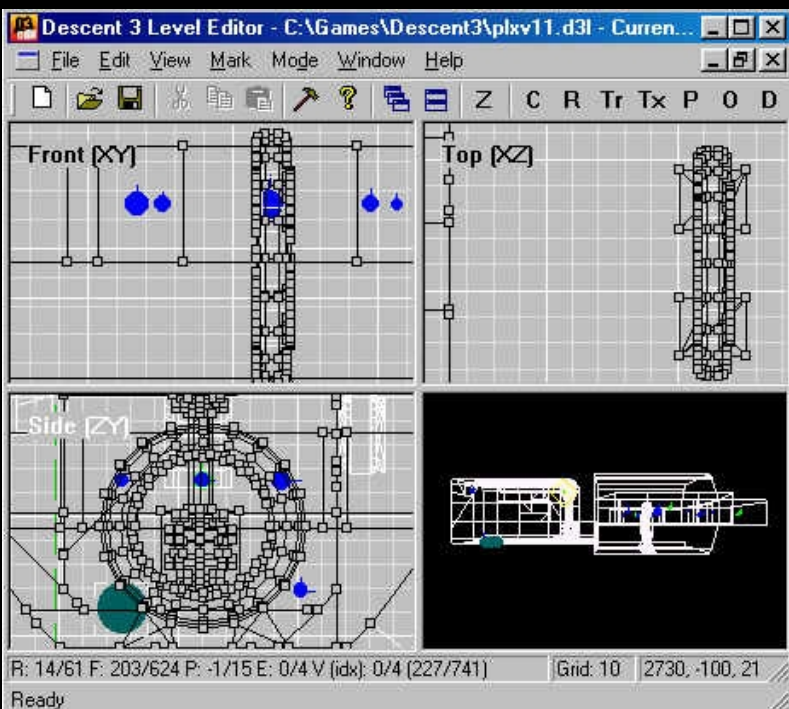
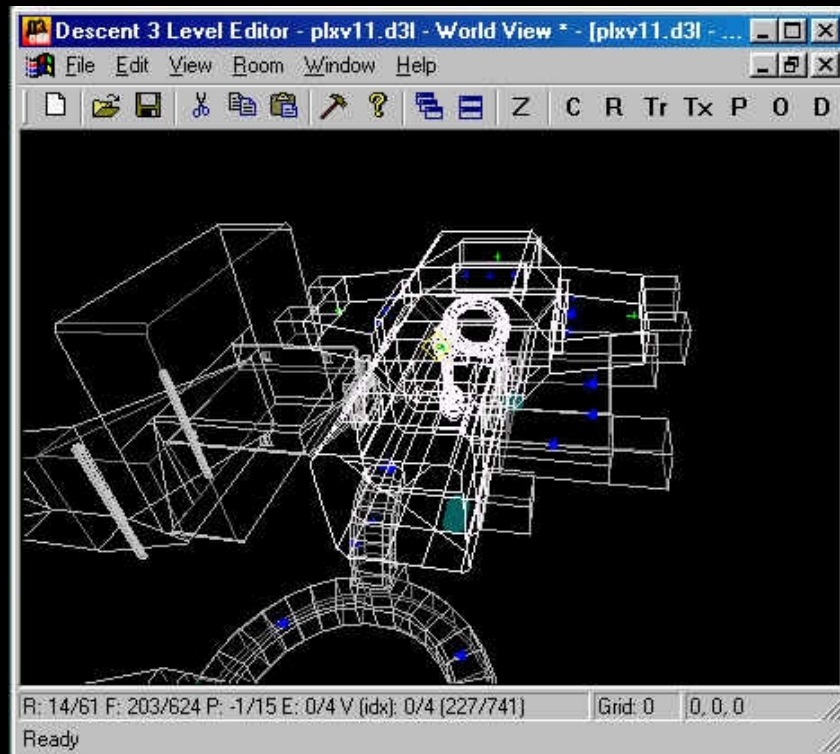
Object Mode - (Ctrl+g) - insert objects, delete and move around

Path Mode - (Ctrl+h) - Paths for robots

Create, see chapter110
- BNodes.

Views in D3Edit

World View is the view in which one can see the whole level (picture on the right). You can move it in all directions, create new rooms, delete rooms and view them with textures. The picture is my level Planet X. If you click in a room, it is highlighted in white and the view changes to **Room view** to the clicked room. If I now in the menu View->Display Current Room View select then you will see the picture on the left. I turned off the display of normals in the image.



Room view edit room. Here you can see four windows, one the front from above, one from the side and the fourth is similar to that of World View, except that you only see the current room. Here you create rooms, shape them, texture them and insert objects.

The Grid

This is the grid in the background. You can change the grid size if you are working on a very large space or want to move vertices by a certain unit of measurement; namely under View->Adjust grid size/rotation angle->select or below Custom...enter your own values.

The axes

You have an X, Y and Z axis. You need these three axes to create a three-dimensional space. Here's a picture:

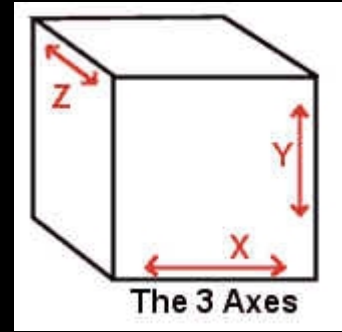


Image source: Gameedit

So that should be enough for now, the various functions of D3Edit are described individually.

Back to Section A

002 - The first room

Schplurg

Here we are now creating the first room step by step. You should have a screen resolution of 1024x768 pixels or higher so that you have enough space to work.

Open D3Edit and you should see the following:

If there are any
There are menus that are
open on the main screen,
you can do this
can be closed by pressing
the buttons in the
Toolbar used
(highlighted). This
Buttons open the following
menus:

Tx Texture menu

P Path menu

O Object menu

D Doorway menu

S Sound menu

Tr - Terrain menu

R - Geometry menu (extruding, lathing, bending...)

Q - Quicktest menu, a relatively new facility to level quickly and without much effort to test

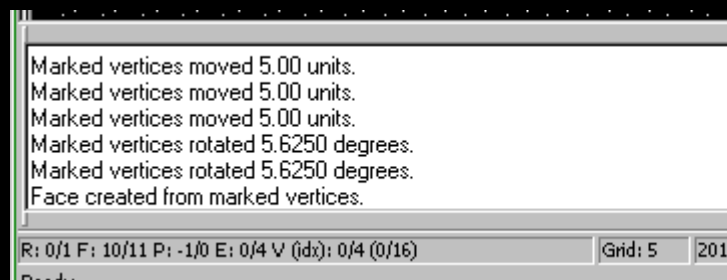
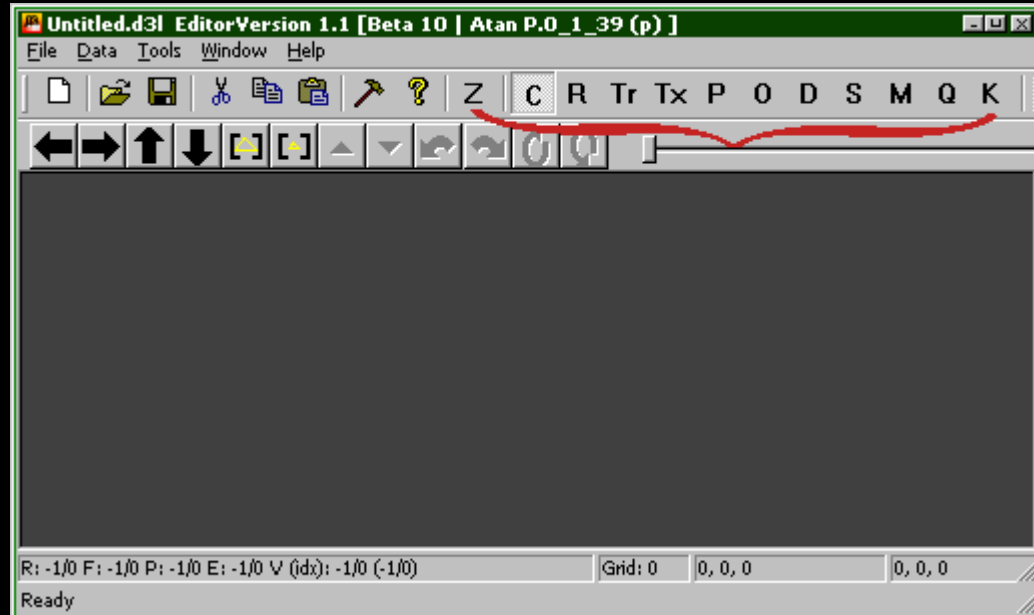
C - Camera Slewer; This is the menu bar below the standard menu bar (at the top of the picture see). You actually hardly need this at all, because you can also do it with key combinations, but do it little by little.

Z - Switches the Z buffer on or off. The textures are displayed better, you need to but more system performance!

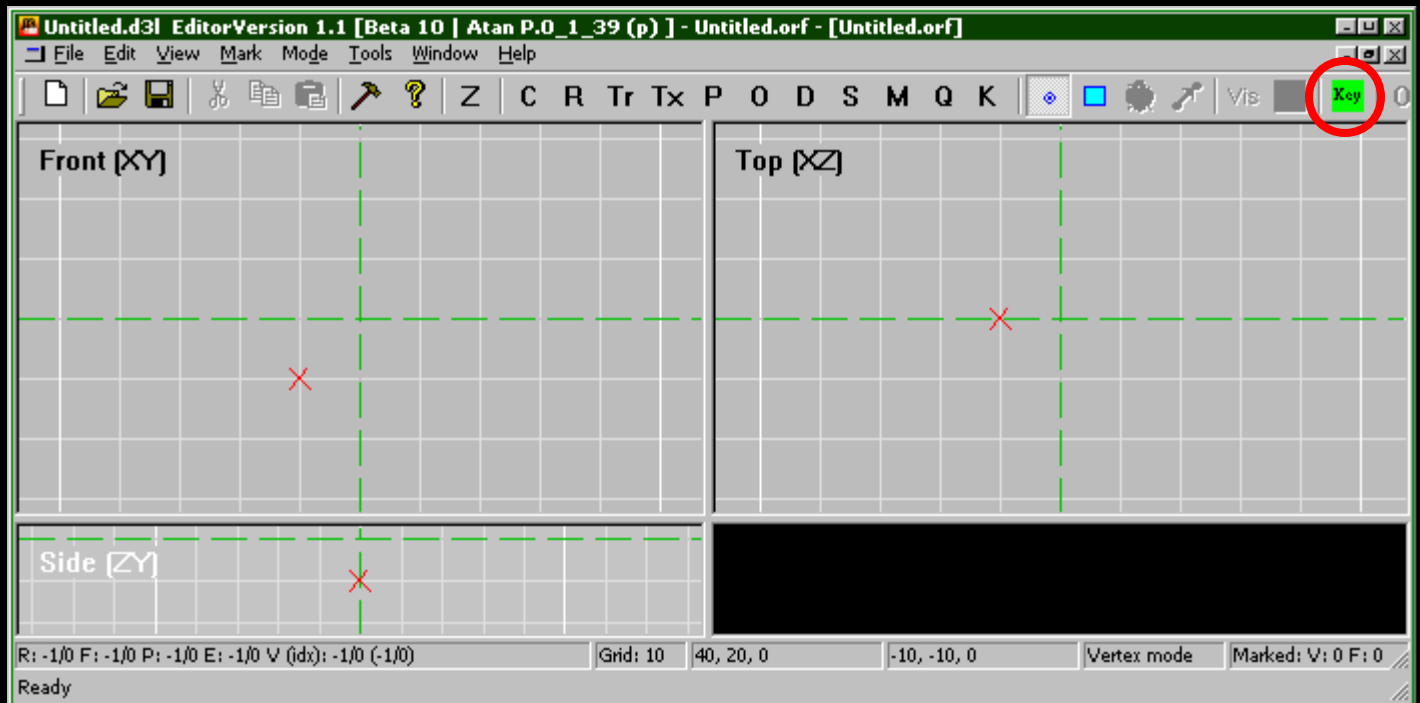
K - A tool to use selection criteria to determine what should be marked on or off (more on this later). For example, you can specifically select faulty faces or T-joints.

M - Messages, here you can see what you have done recently or the program (picture on the right).

Start!



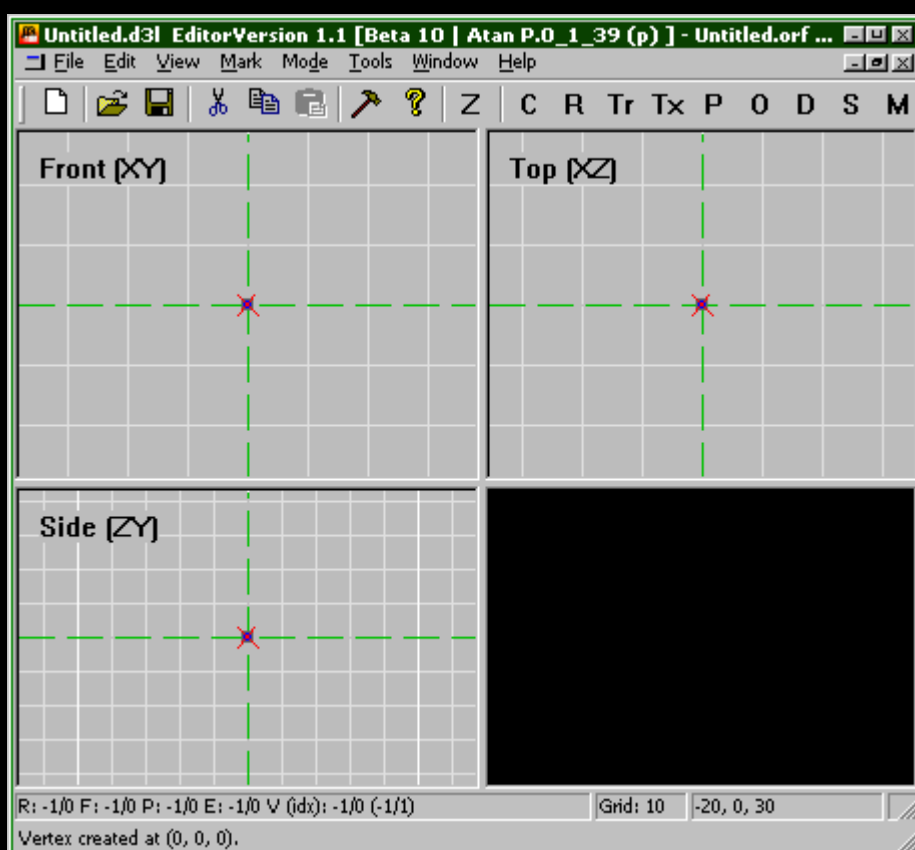
Select in menu File->New... New Room and click OK. Now the Room View has opened, maximize the window. You now see what is shown in the following image. the Red one **X** is always the insertion point and the green/white dashed lines represent the center of the level. At the bottom of the status line you can see which grid is set (Grid: 10), next to it the position of the cursor is indicated (-10, -20, 0) and another field next to it shows the position of the insertion point (-10, -10, 0).



At this point there is a hotkey list that you will need now.

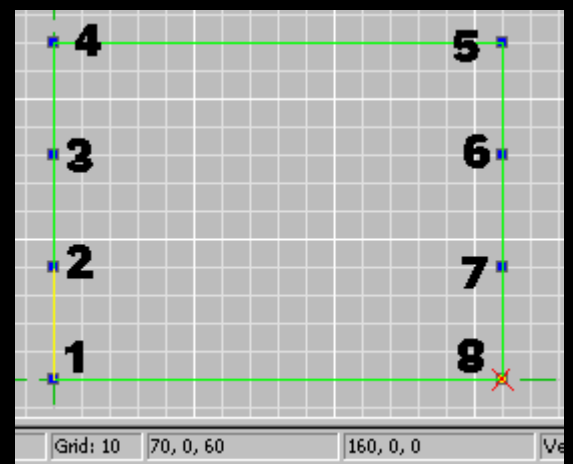
But it is also available again in full at the end of the file (page 541), and can also be viewed directly in D3Edit using the button circled in red (see image above).

<p>View Control cycle through windows = Ctrl + Tab</p> <p>pan = Shift + left mouse zoom = Shift + Ctrl + left mouse zoom = mouse wheel</p> <p>scroll = left a scroll right = d scrolling = arrow keys zoom in = w zoom out = s</p> <p>grid smaller = [grid larger =]</p> <p>Modes select Face ctrl + f Vertex ctrl + r object ctrl + g path = ctrl + h</p> <p>place Vertex (Vertex mode) = ins place Face (Face mode) = ins place Face (Vertex mode) = shift + ins</p> <p>Selecting next... Vertex = v Face = f room = r portal = p select previous above items add shift + (key)</p> <p>Mark items in Vertex, Face, and object modes mark selected = space unmark all = u invert markings = i mark all = m mark worldview = m</p>	<p>Mark multiple items Dragging a rectangle around Vertices and Faces will mark all that lie entirely within the rectangle. alt + drag to unmark</p> <p>Faces: n = flip marked face(s)</p> <p>Moving marked items (Vertices and objects in 2D views) move = shift + arrow key</p> <p>... numpad keys left = 4 right = 6 up = 8 down = 2 twist left = 1 twist right = 3 objects rotate left = 7 objects rotate right = 9</p> <p>Editing copy Face = ctrl + c paste Face/verts = ctrl + v paste on top (of copied item) = ctrl + shift + v</p> <p>Moving placed rooms rotate = alt + mouse slide = alt + ctrl + mouse</p> <p>stop moving = old</p> <p>Other save = ctrl + s</p> <p>texture stretch = < texture stretch = ></p>
--	---



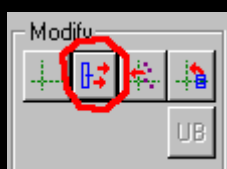
Now switch to vertex mode (**Ctrl+R**), and sets the insertion point at the center point at "Top (XZ)", as in the picture on the left. After that is done, press the button **Insert** or **Into the**. Now the room contains the first vertex, repeat until you have as many vertices as in the picture below (also the order). When this is done, all vertices must be marked! (Button **M**

or **Mark->mark all**)

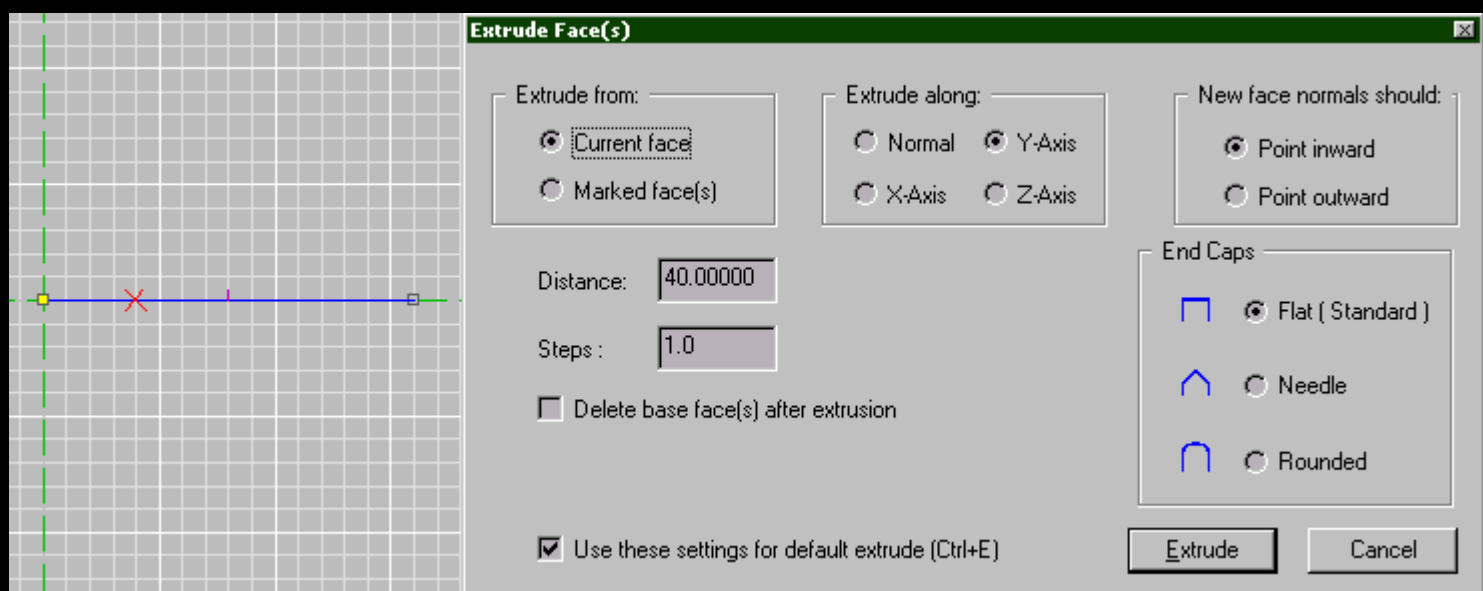


Now switch to face mode (**Ctrl+F**) and again **Insert** or **Into the** press. You have already created the first face. The face is now the '**current Face**'.

extrude



Next we turn the face into a room. Press the **R** button in the toolbar; a menu opens. Now click on the button as shown in the image on the left. The following dialog box will open if you have an editor older than v40:



'Extrude from' Here you can choose which face it is and how high the room should be ('Distance'). Note that positive or negative values determine which direction!

'Extrude along' In which axis should the space be extruded.

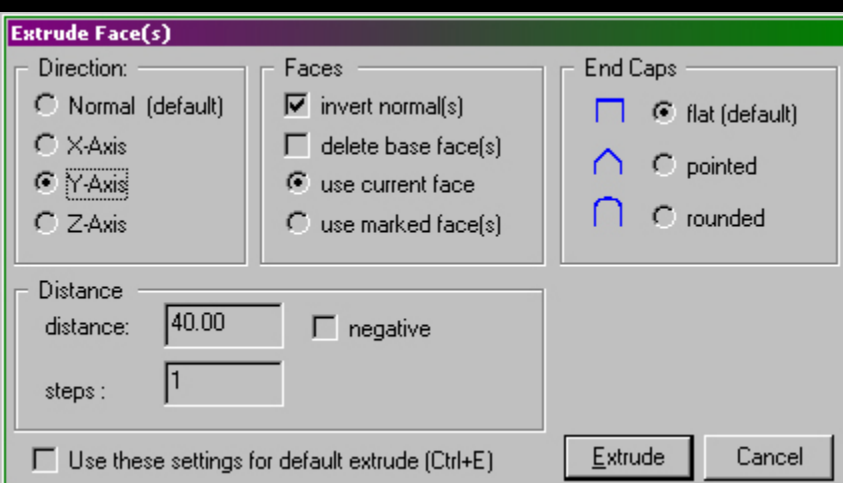
'New Face normals should' which direction the faces should point.

'Delete base Face(s) after extrusion' determines whether the original face should be deleted or not. 'Use Settings as Default' you can check the box or not; is it set these values for the default extrusion (**Ctrl+E**) used.

Apply the settings from the image and click on the Extrude button. The first room is finished in terms of geometry.

Update for v40:

The Extrude dialog has also been modernized as part of the revision of the Extrude function, then take the settings from this image.



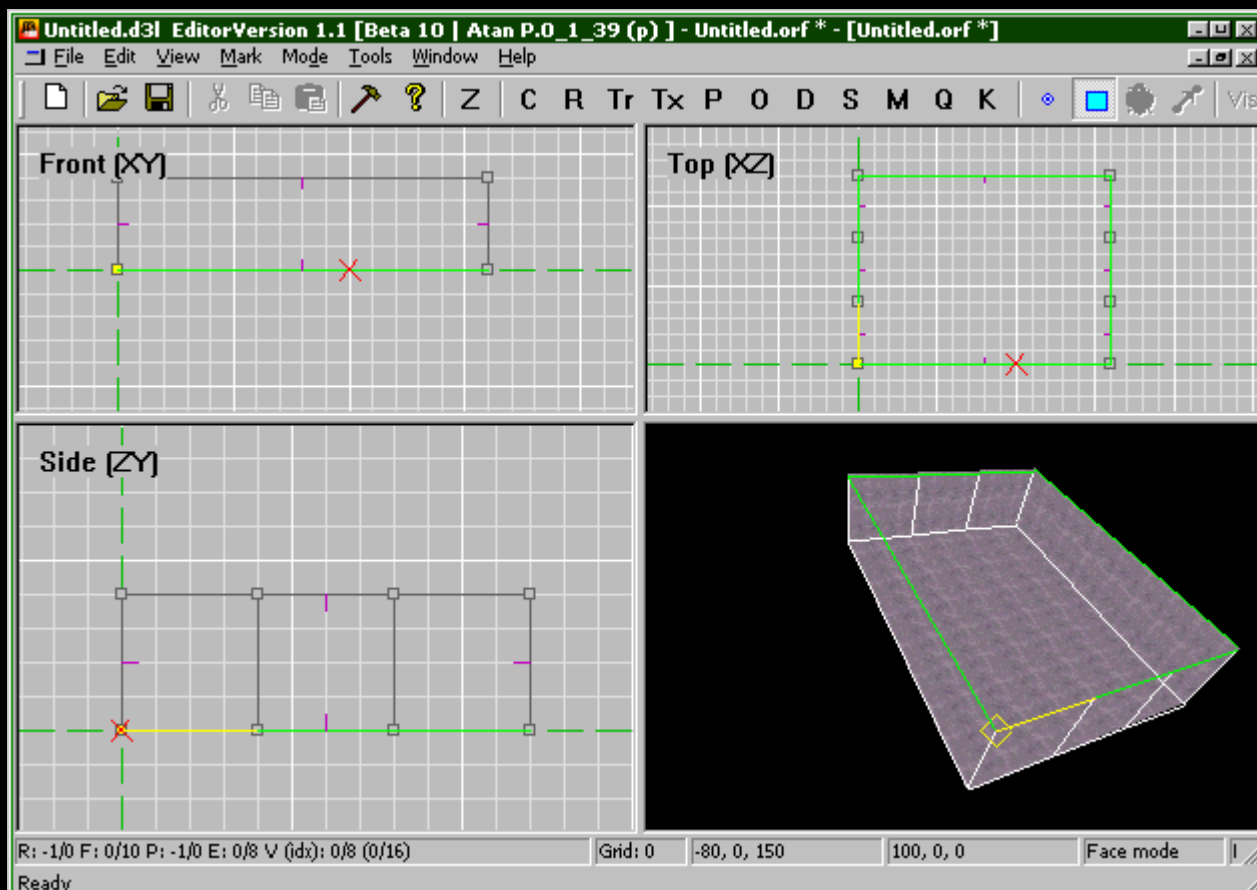
'Direction' Here you indicate where they are going Extrusion should go.

'Faces' By default, the new face normals always look in the same direction as the original face. 'invert normal(s)' turns them around.

'Distance' Here you specify how far, how often and in which direction you extrude; checking 'negative' reverses the direction.

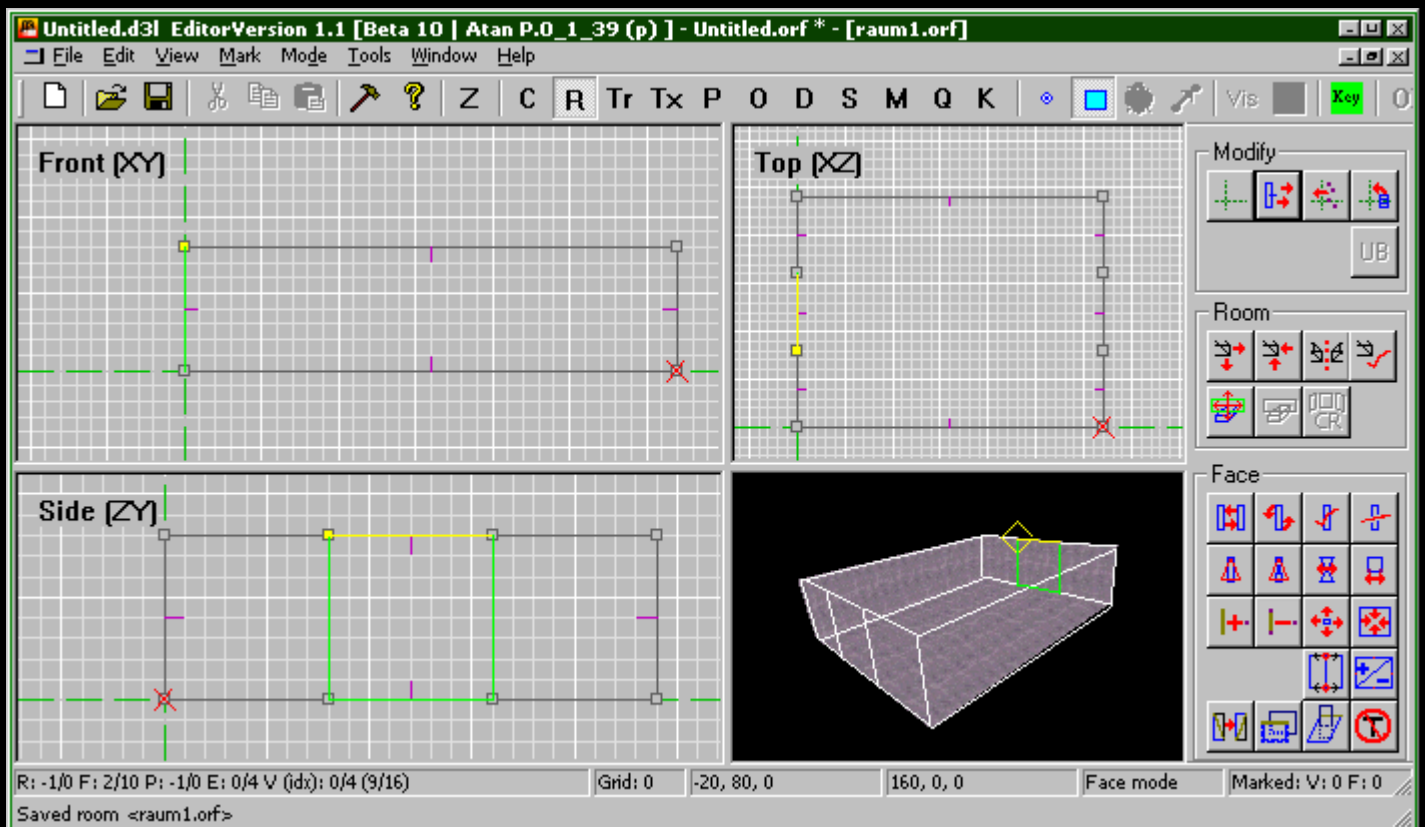
So that's it, the first room is created. The topic "Changing the room" continues! Saves the room with the name "raum1.orf"!!!

Back
after
Section
A



003 - Change the room

Schplurg

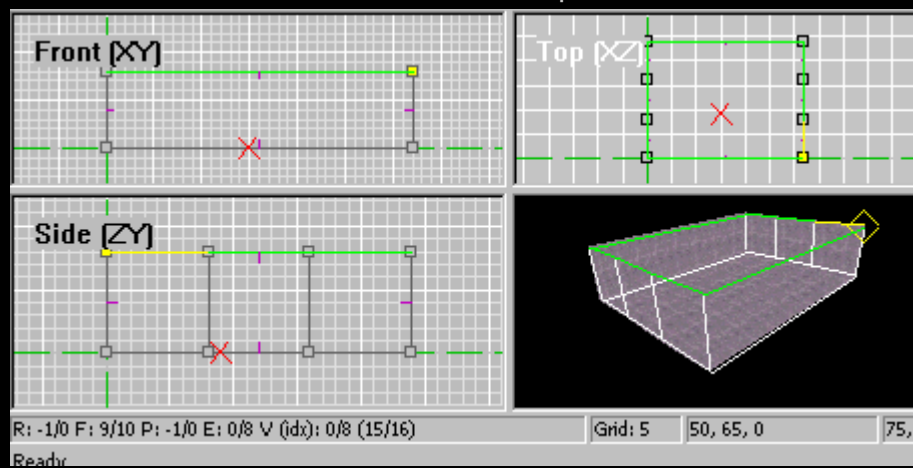


This part follows the topic '[The first room](#)' at! Open D3Edit & then your room (raum1.orf). Your room should look like the picture above. Let's take a closer look at the room. In "The First Room" we set the room height to 40 units, now let's check that. The center line is visible in the Front(XY) and Side(ZY). Note: I have set the grid to 5 (see image below in the status line, "Grid: 5").

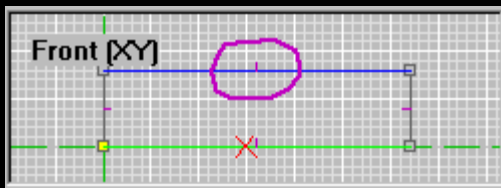
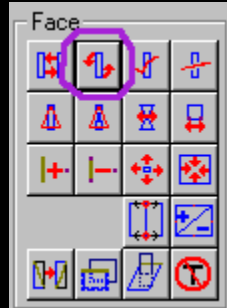
There are 8 grid squares between the ceiling and the floor, per grid square 5 units (Grid 5) therefore the room is 40 units high since $8 \times 5 = 40$.

Note the current face, which is green in the room. Look at the output in the status line from the previous image; the areaQ:2/10 says: the first number is the number of the current face, the second is the total number of faces. If you press the button now **F** pressed, the Face 7 becomes the current Face. It counts up and starts again at 0. **Shift+F** does exactly the opposite 9...0.

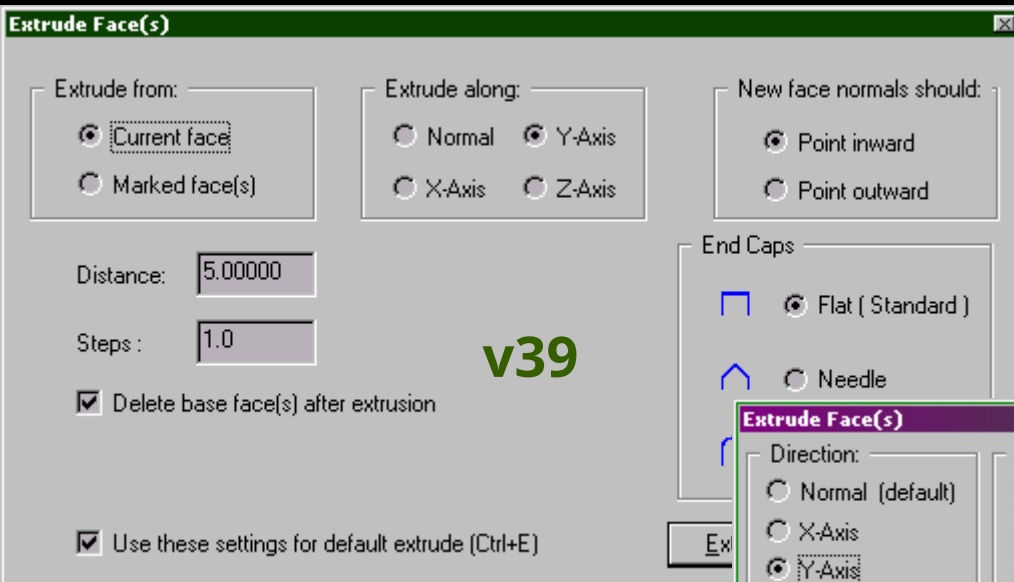
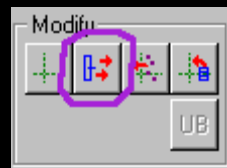
We now need a light source in the room, otherwise you wouldn't be able to see anything later when you fly into the room. We use the Extrude function again. Select the face from the ceiling (previously with **Ctrl + F** Switch to face mode !!!), that's possible as follows: click on the Top(XZ) view until the top face at Front(XY) turns green. You can also do it with the **F-Done** button! Now the face with space, or in menu **Mark.... Mark/unmark**, to mark. Now click a few times at Front(XY) outside the room, now you can see in the image on the right that the face is marked.



In the previous image the face is already selected. The status line below shows F:9/10, that means the current face is 9. Note that the "normal" points downwards or into space. The first time we used the Extrude function, the "Normal" was pointing up. We have to turn the face around and we'll do that with the **flip**-Function (picture on the right, keyboard shortcut **N**). You can see the result in the small picture below.

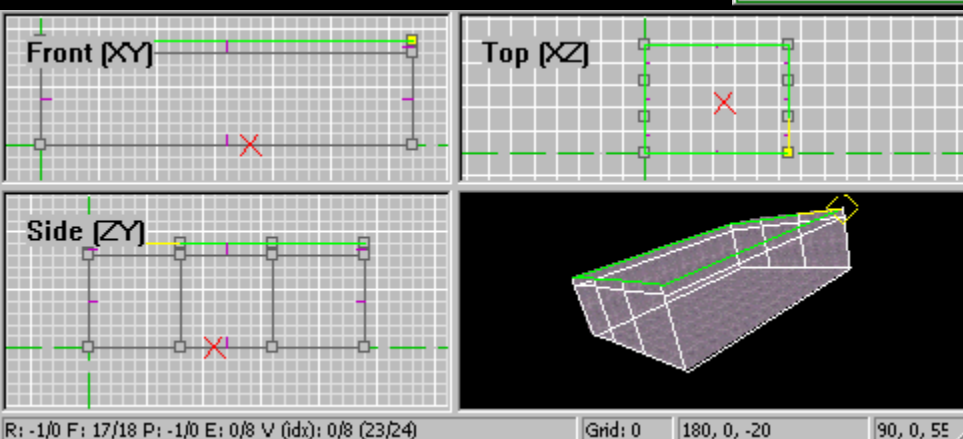
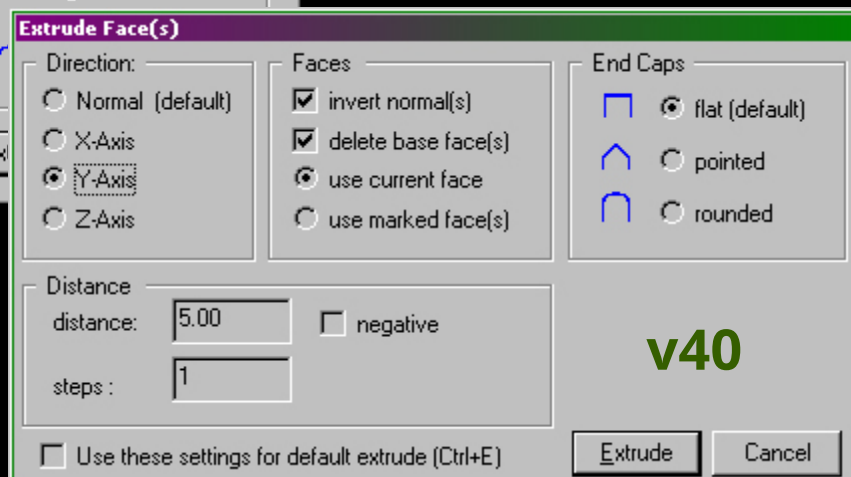


Now we can move on. Make Face 9 the current Face again, then call the Extrude function:



Enter the same what you see in the picture on the left. We will now expand the room by 5 units upwards and want the original face to be deleted. If you are working with the new version, take the other image.

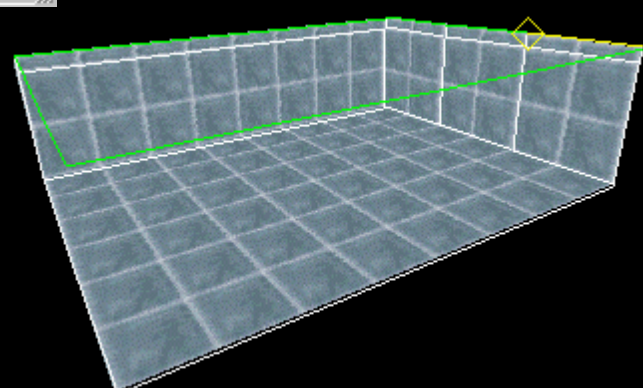
You can see the result below. So now we have the room almost finished. We now have to texture it, switch to face mode (**Ctrl+F**) and mark all faces with your **M** or in the menu **Mark...** Mark all presses. Close the geometry menu by clicking on the



button **R** clicks. Now just click on **and** **texture** menu opens, click on the button **Choose...**, in the following dialog box **Metal Textures** and then looks for the texture

AdvMetalPanel1. Now click on the texture and drag and drop it into the texture list 'Custom'

push. Then closes the dialog box **Texture Palette** and clicks on **AdvMetalPanel1**, in **Custom**, the texture is now the current texture. Click the button **To Marked**. Right click in the room preview and select **Textured with outline**, now you can look at the room with textures. It should look like this. Save the space and you're done. Back to Section A

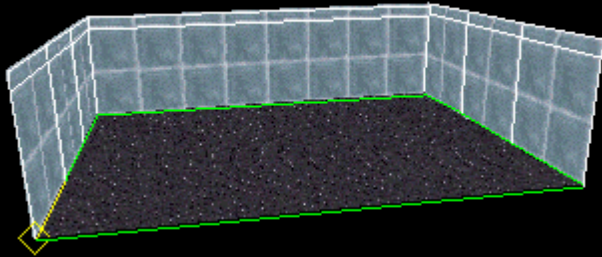


004 - Texturing

Schplurg

All images I use here are from Gameedit

For this part you need "raum1.orf" from the topic Changing the room!
Open D3Edit and then your room.

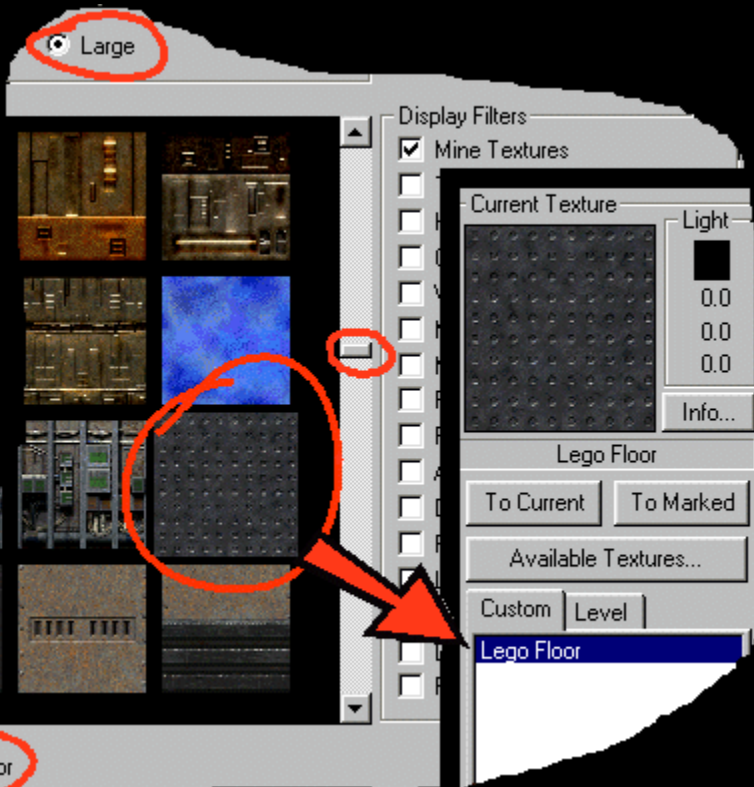
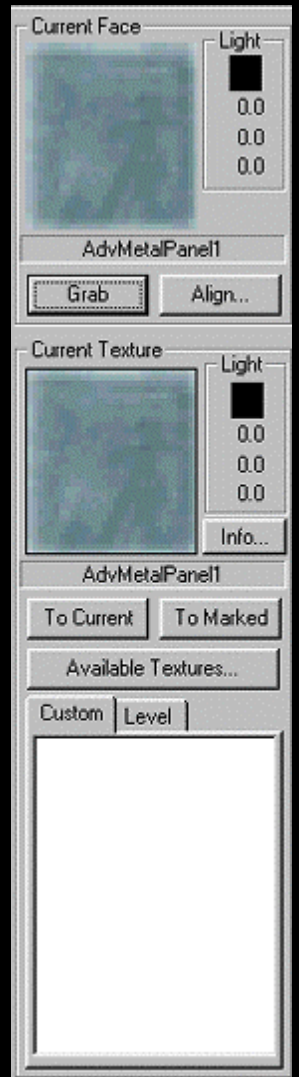


(floor already changed)

First change with **Ctrl+F** into face mode, we stay in this mode for the whole topic. We would like to slightly change the look of the room. In the picture on the left the floor has already been changed, I would now like to explain that to you.

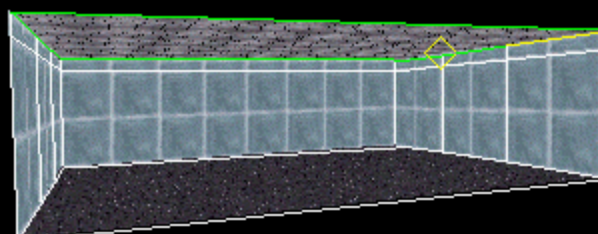
Uses the **f**-button around the face, as can be seen in the picture, to make it the current face. You can also do this with the mouse. Click the button **TX** to open the Texture menu, which should resemble the image on the right. Now click on **Choose** (newer versions otherwise Available Textures). In the following window (**Texture Palette**) select **Large** and click **Mine Textures** at. Then look for the texture 'Lego Floor', this is what you put in

the field, see picture below.



Now click on the texture in the field and then on the button **To Current**, Minimize the Texture Palette window and look at the room in Room View, it should look like the first picture. We now texture the ceiling. In the Texture Palette window, search for '**LWceil.tga1**'. Now you just have to repeat the steps as described above.

Does your room now look like the middle of the picture below? Good! Now let's illuminate the room a little more.



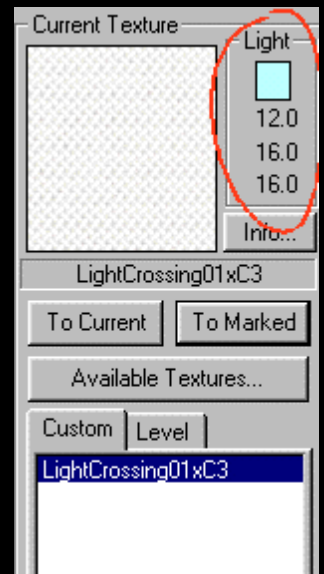
Lighting - the short version

To explain the lighting in more detail here would go beyond the topic. I'll explain this in more detail under another topic, because you can give a level a special touch with the right lighting. See my level Planet X. :-)

First clear all checkmarks in the window **Texture Palette** (we minimized these earlier!) below **Display filters**. Now you click on **Light Textures** and look for the texture '**LightCrossing01xC3**'.

Now look at the image on the right, the area circled in red provides information about the color and RGB color values (the numbers below Light). When you click on the button **info...** if you click, you'll get a little more information about the texture, but you can't change anything. The RGB color values are intended to help you estimate the color and brightness. It makes a difference whether you have to illuminate a large or small room.

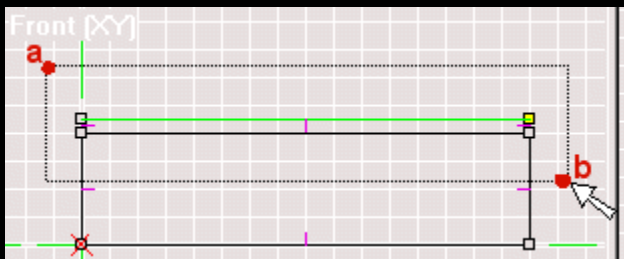
Now you know why we changed the room in the previous topic. If we had added a larger area, it would be much too bright in the room! That should be enough to adequately illuminate the room.



Texturing multiple faces

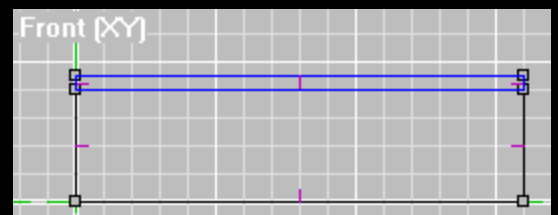
There are a few ways you can texture a face. You can texture each face individually, like we did with the floor and ceiling, or you can texture multiple faces at once.

(colors changed for better view)



Faces and verts can be marked in the respective mode by drawing a rectangle with the mouse around the faces or verts to be marked. Do the same as in the left picture (we are still in face mode). The **a** stands for starting point and **b** stands for end point. If you're at that point **b** When you have arrived, release the mouse button again. You can see the result in the picture below on the right.

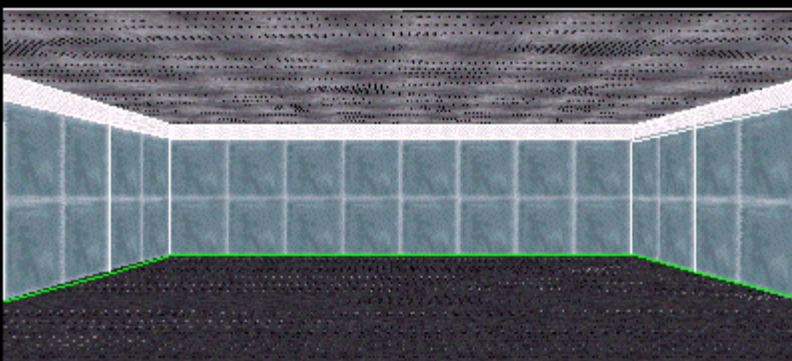
But wait! We also marked the ceiling, we have to unmark it. Click in the top view until you have the ceiling face as the current face (outlined in green), now press the space bar or in the menu **Mark... Mark/Unmark**. You can choose the method that seems easiest to you. Once you've done that, click on the button in the Texture menu **To Marked**, complete.



You can see the result in the picture on the left. Now you should save the room, because there is no "undo" function in D3Edit!

In the next topic we will use space as a level (mn3) and calculate the lights. It will be called "The First Level".

Back to Section A



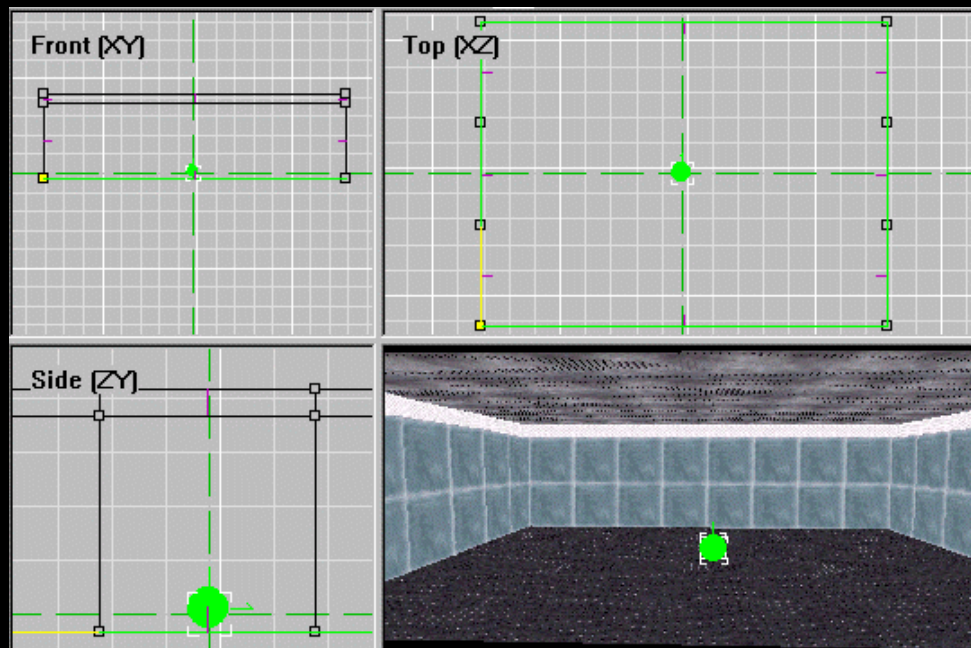
005 - The first level

Schplurg

All images I use here are from Gameedit

For this topic you need the space from 'Texturing'! Opens D3Edit and then the room "raum1.orf".

Choose in the menu File/New.../New Level - Import Room and up **OK** click. In the following dialog box look for your file "raum1.orf". This room should look like the picture below.



Here we see that a green dot has been added, which is the first player starting point. There are these points in every level, there can be several but at least one.

in the menu Window, which looks like this (your menu may be slightly

different look) busy

us for the time being only the lower one part of image.



You can navigate here if

You have opened several rooms. This way you can keep track of which room you are currently working on. Now on Untitled.d3l - World View click, now we are in the world view. Here you can view your levels completely in 3D or switch to another room and make it the current room. Now let's save the level first, because D3Edit tends to crash abruptly. Choose in the menu File/Save as..., look for the directory in which you save the levels and name the level "thelevel.d3l".

After you've done that, go back to the File Menu and select Edit Level Info....(newer editors: File > Level Settings...) In this window you can, among other things, set the name for your level; You will see it when you start the level or in-game (map, etc.). Fill in the fields and click **OK**.

Look for errors

Goes back to File Menu and selects Verify Mine. This menu part is extremely important for troubleshooting! If everything here is at 0 then you have worked perfectly. In the lower part you can see **BOA: NOT VALID**, if you were to play the level now, you would get an error message when starting the level "BOA not valid!", we will change that right away. Goes back to the menu File and there you choose Compute BOA Vis Table, you confirm the next dialog box with **OK** and the table is already calculated. Now again Verify Mine, you can see that now **Wow: valid**. Now save the level again! If you add a room, repeat this process. Now close the room "raum1.orf". You can answer the question of whether you want to save the room with no. We haven't changed anything to the room.

Lighting

Lightmap spacing:	<input type="text" value="5"/>	<input checked="" type="checkbox"/> Combine lightmap faces
Iterations:	<input type="text" value="9999999"/>	<input checked="" type="checkbox"/> Volume lights
Ignore limit:	<input type="text" value="0.005000"/>	<input type="checkbox"/> Super detail

Now let's make light in the level

Here I will explain to you how to calculate the light. Goes to the menu Window and choose there Lighting. I'll first explain the individual input fields.

Lightmap spacing-is the distance between the light pixels. Higher settings produce coarser light, lower settings produce realistic light and shadow effects. (= the resolution of the lights)

iterations-specifies how many lights should be calculated. Here you should always choose the highest number, because then you are sure that you have really calculated all the lights and that the level looks better.

Ignore limit-Light is not calculated as soon as the intensity/brightness falls below the specified value. The lower the number, the more realistic.

Combine lightmap faces-D3Edit attempts to combine 2 or more faces into a single "lightmap". Be sure to leave a check mark here.

Volume lights-indicates that a dark corner remains dark. You can then hide in this dark corner with a ship. I did it that way on Planet X.

Great detail-sets the detail level finer. Is not necessarily recommended as the light calculation takes considerably longer. You could try it out.

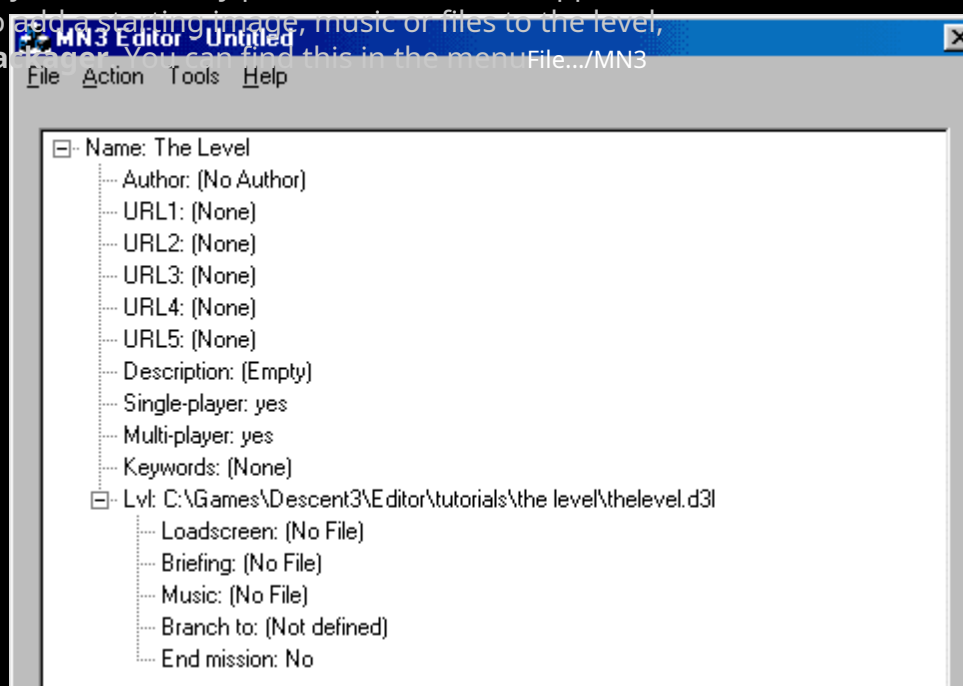
Set all values as in the picture at the top right. Then click in the box **Entire Mine on Count Lights**, in the next dialog box **NO** click (I want you to see how the lights are counted). Now click on the button **Light It**. In the following dialog box click **Yes**. Now a dialog box will appear telling you that the calculation can take a very long time. It depends on how big your level is, on Planet X, with the outside world, it took 18 hours and I have a P3 500 MHZ. When the calculation starts, you think D3Edit is done, don't be fooled, D3Edit starts the calculation! (if you have a tool that shows processor utilization, you can see that work is still being done) After that's done, save the level.

The MN3 Packager

A level must be as `mn3` be saved or you can also say packed. That's what happens with that **MN3 Packager**. You can also add a starting image, music or files to the level, that's what you do with it **MN3 Packager**. You can find this in the menu `File.../MN3 Packager`. (newer versions:

Tools > mn3edit)

Right-click on Name in the following menu on Edit mission information. Here Give the level a name (the name that you then see in the level list in Descent3), enter the information about the author and click OK.



Now you choose in the menu **Action.../Add level**. Now up **Browse**, under level filename, searches the level.d3l and **OK**. The MN3 Editor should look like the image above. The URL information must only be entered when the level is completely finished. This means uploading the level to my homepage or any other website so that I can then download it via these URLs (in Vortex, for example). You will also find out which URL is entered. Now save the level, **File/Save as...** goes to your mission directory (your **Descent3/missions** directory) and saves the level under the name "derlevel.mn3". Closes the MN3 Editor and D3Edit.

Now start Descent3, **New Game** and starts **The Level**.

Here you can see another picture of your level. **Congratulations** this is your first level that you created!!!!



Update:

Nowadays, level packing is much easier, thanks to the quick edit. However, doing it the old way cannot be compensated for in terms of experience 🏆

[Back to Section A](#)

006 - Add a second room <>

Schplurg

For this topic we need the file "thelevel.d3l" from **The first level**. Open D3Edit and then "thelevel.d3l"

Note: We first need to fix a bug in our previously created level. Every time you create a level you have to align the room with the grid, otherwise your entire level will be next to the grid. Every level designer should take this into account!!!

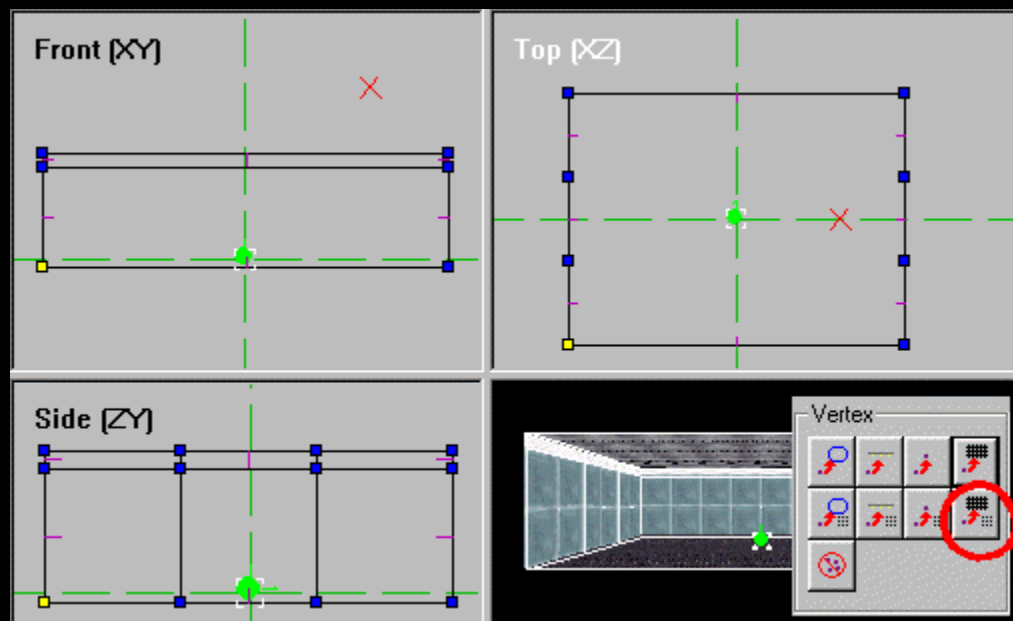
We had a room with that New Level Import Room function was added to a new level and the room was not added exactly to the grid, but between two grid field points. Assuming you try to create or edit other rooms, errors may occur.

What we have to do now to eliminate the error: First check whether you are in Vert **mode** located

(**Ctrl+R**). Now mark all verts with the key **M**. Or do it with the mouse, hold down the left button and drag from the top left to the bottom right. Make sure that a vert is yellow (that's the current vert), this vert becomes

then aligned with the grid and takes all marked verts with it.

Take a look at the picture on the right, that's how it should be look.

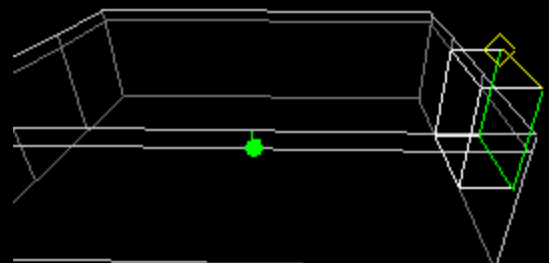
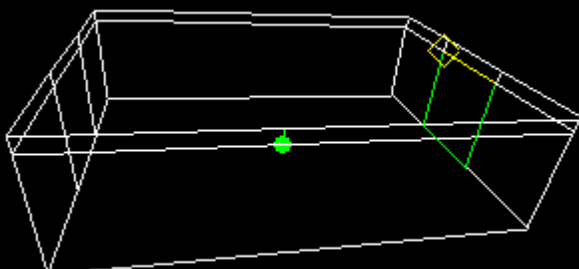


Now open the "**geometry**

bar" with the button **R**. Under Vertex we find two buttons for which there is no information in the status line. We are interested in the lower one of the two, outlined in red in the picture. When all verts are highlighted, click this button and we have aligned the first axis. Repeat the step in the other two windows so that the room is on the grid in every axis. Now our room is aligned with the grid.

Thanks to **Sirian** for this tip!

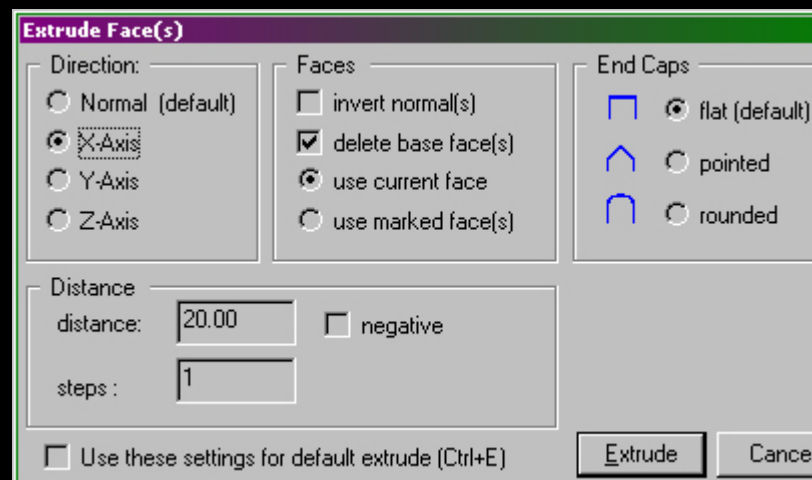
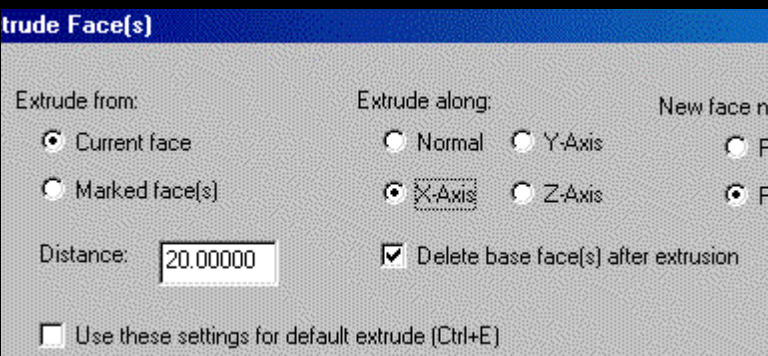
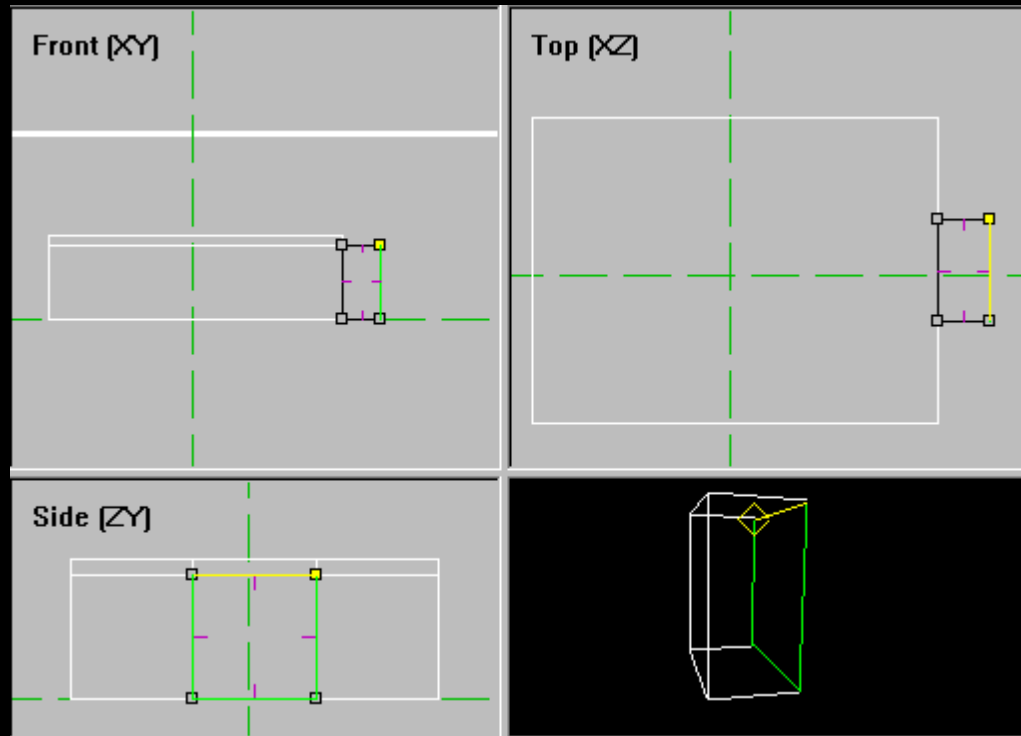
Back to the second room! Make sure you are in World View. There are several methods to add rooms, the simplest is the function **Add Room at Current Face**. You can find them in **RoomMenu** at the top. Make the Face the Current Face (outlined in green), as in the left picture, then select **Add Room at Current Face**, the result then looks like the picture on the right.



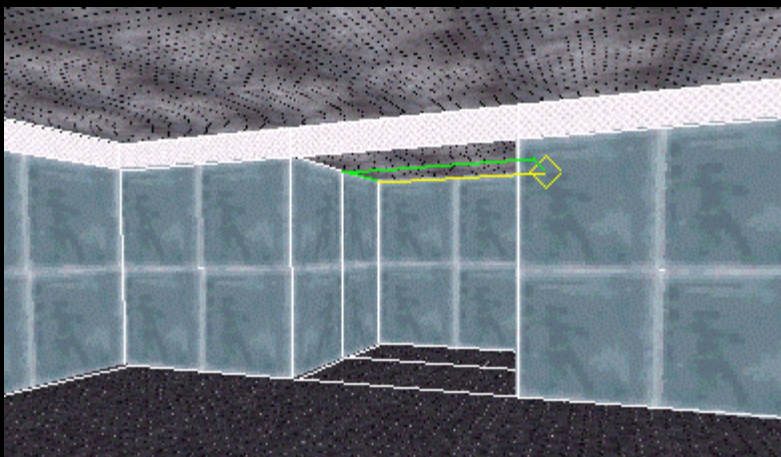
Note that the new room is now the Current Room. Click with the right mouse button and select **display current room viewer** the Key combination **Ctrl + tab** to open the room in room view.

We have a room here (picture on the right) with them Units of measurement 20 x 40. We will now fill the space with the extrude Expand function by 20 units. I have already explained this function under 'Change the room'. We're doing it a little differently here. Note that the outer face is the Current

Face is and clicks the Extrude button in the Geometry Bar. Fill the dialog box according to your editor version as shown in the image and extrude the space. Left image for v39 and older, right for v40.



Now we have to set the textures. Opens the Texture Bar (-button you now look below, where the textures are actually in the window, it is empty. Click on Level and here you can see all the textures that are already built into your level. Now design the room so that



it looks like the first room (except for the lamps on the edge of the ceiling). You can see the result in the picture on the left. Now save the level in the World View!

Room Properties

Name:

Flags:

<input checked="" type="checkbox"/> Refueling	<input type="checkbox"/> S1
<input type="checkbox"/> Secret	<input type="checkbox"/> S2
<input type="checkbox"/> Red Goal	<input type="checkbox"/> S3
<input type="checkbox"/> Blue Goal	<input type="checkbox"/> S4
<input type="checkbox"/> Green Goal	<input type="checkbox"/> S5
<input type="checkbox"/> Yellow Goal	<input type="checkbox"/> S6
<input type="checkbox"/> Current face is a goal face	

So, the room is now almost finished. We now want to assign a task to the room. We could use the room as a tunnel to another room, but that will come in the next topic.

Opens the menuWindow.../Room Properties. You have to be in the **Current RoomView!!!** Under **Surname**(bottom left picture) we click on the button and name the room **Energy 1**, then sets the flag like in the picture. With this we have created an energy tanker. Now we calculate the lights in our level, we can do that as shown in "**The first level**" is described.

Save the level and create onemn3-File to see the result of the work.



In the next topic we will use the extrude multiple times and create an amazing space!!!

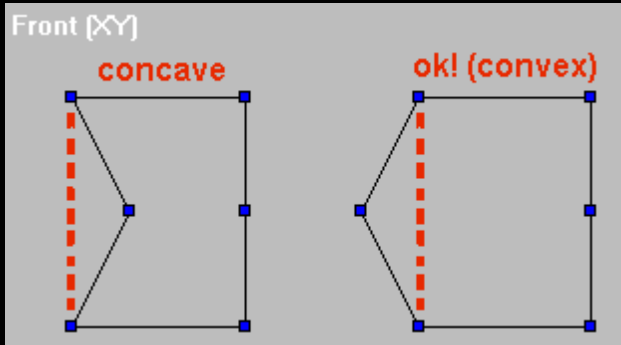
Back to Section A

007 - Reactor Room <>

Schplurg

For this topic we need the file "**thelevel_1.zip**" from **The second room**. opens D3Edit and then "thelevel.d3l". Click on in the menu **File/New...** and chooses **New Room**. We have already dealt with the extrude function in previous topics. Now let's create a more complicated room. There are limitations in D3Edit though, i

I would now like to describe one.



Concave faces

A face is concave when a vert that is part of a face points into a face. (smaller picture, left)

We will create a Convex Face in this topic and then fix it. You have to divide the Convex Face in half. Creates a face as shown in the larger image. (see: The first room)

This face is at the end of a long room, rather than the floor. Therefore, we edit the face in the front view. Insert the verts one after the other against the clock, because this way we get a face in the direction we need. Now let's do it **Verify Room** over it, will "Found 1

"concave faces" displayed.

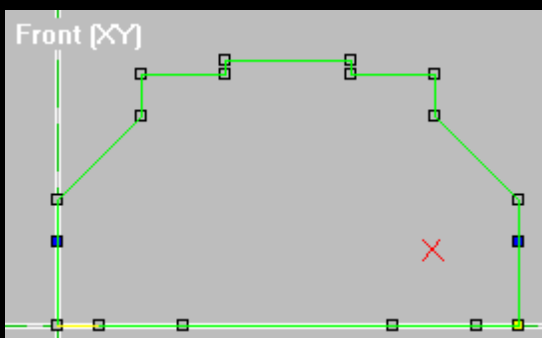
Split faces

Around the face error-free create, must we this in several small ones Share faces. The let's do with that

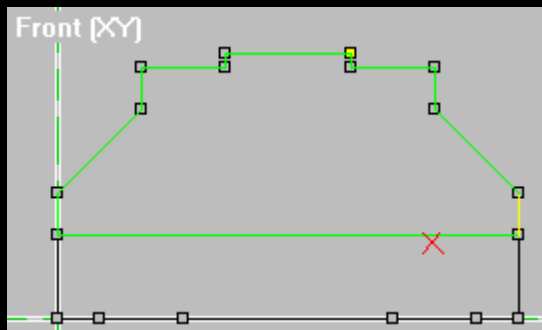


"Split" function (picture above) in

Geometry menu (**R**). Let's mark the two verts. We use the mouse to draw a box around the two verts, we can see the result in the smaller image below. As you can see, the two are verts 40 units from the floor, that's where the room will be put together later.



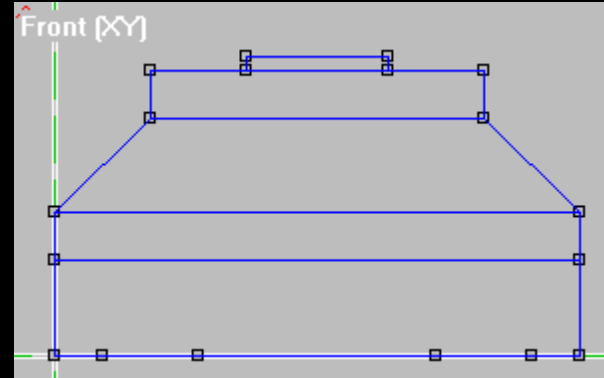
Check whether the face is current, now the button **Split Current Face** (the one outlined in red in the small picture above). The result can be seen in the next picture (left). If you are now in the menu **File.../Verify Room** If you click on it, you can see that there is still a concave face, but we did that because the connection to the second room has to be placed later.



Divide the rest of the face so that it looks like the picture on the right.

After that's done, let's take another look

to see if we still have a concave (File.../Verify Room) and you will see that there are zero errors in our room. We don't need to divide the lower faces on the floor yet.



Extrude from:

☐ Current face

☒ Marked face(s)

Distance:

☒ Use these settings for default extrude (Ctrl+E)

Extrude along:

☐ Normal

☐ X-Axis

☒ Y-Axis

☒ Z-Axis

☐ Delete base face(s) after extrusion

New face normals should:

☒ Point inward

☐ Point outward

Now we can go ahead and create a room from the face. We do this again with the Extrude function. First we need all the faces mark (key **M**), now click on a face (we need a current face) and take a look **Side(ZY)** check whether the normals all point in one direction. Now this opens

Extrude dialog box (see 'The first Space') and fills everything as in the corresponding picture; above for v39 and older, right for v40.

Click on the Extrude button and the result should look like the image below.

Extrude Face(s)

Direction:

☐ Normal (default)

☐ X-Axis

☐ Y-Axis

☒ Z-Axis

Faces

☒ invert normal(s)

☐ delete base face(s)

☐ use current face

☒ use marked face(s)

End Caps

☐ flat (default)

☐ pointed

☐ rounded

Distance

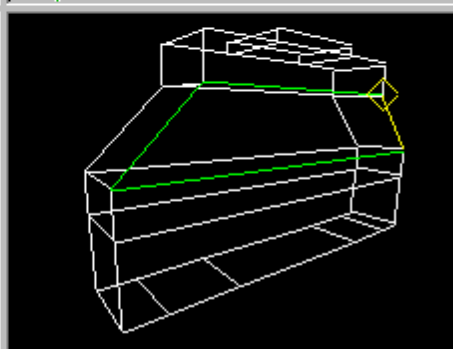
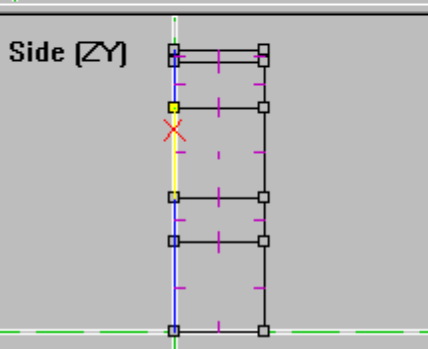
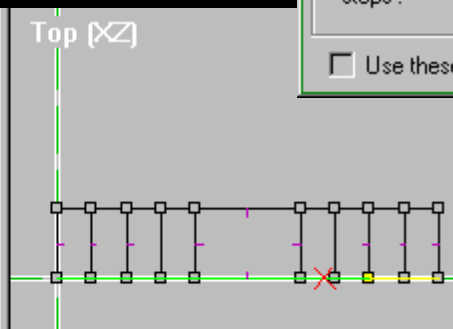
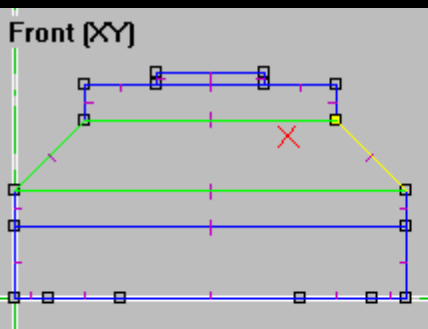
distance:

☐ negative

steps:

☐ Use these settings for default extrude (Ctrl+E)

Extrude **Cancel**



The original faces are still marked. The key **U** press to unmark, now we mark the face that has just been created, we use the extrude function more often. After your faces


have marked, click on

Turning faces because otherwise you won't achieve the desired effect. Look at the normals because they should face outwards. If you look at the room now, you will notice that there are faces in the middle of the room, we will delete these at a later date.

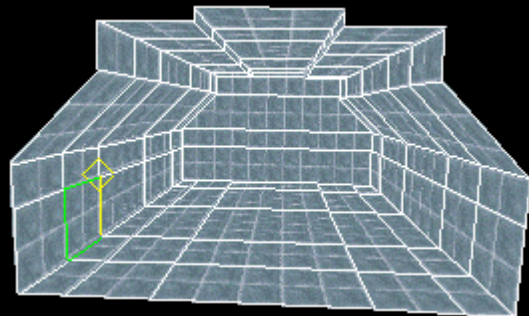
We have to extrude the space four more times, namely:


- 1.40 units in the Z axis.
- 2.80 units in the Z axis.
- 3.80 units in the Z axis.
- 4.40 units in the Z axis.

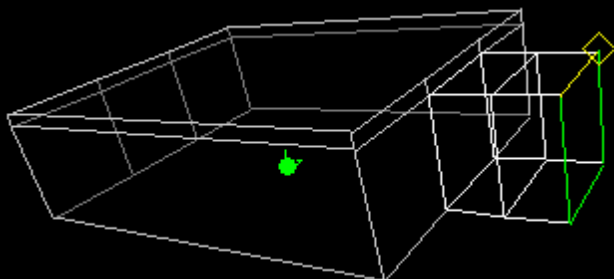
We do this in the same way as described above. After you have done this, select the unnecessary faces in the Preview room, mark them and delete them. You can also do it in the 2D view, but you have to check which ones you delete!! Now your room should look like the picture on the right. Now we have to create the duplicate verts

delete, we do this with 

Now save the room!



Now we have to add the room to our level. Marks all faces with the button  and makes the face that can be seen in the previous picture the current face. Now switch to the world view and make the face the current face, as can be seen in the picture on the left. This is the point at which the rooms come together.

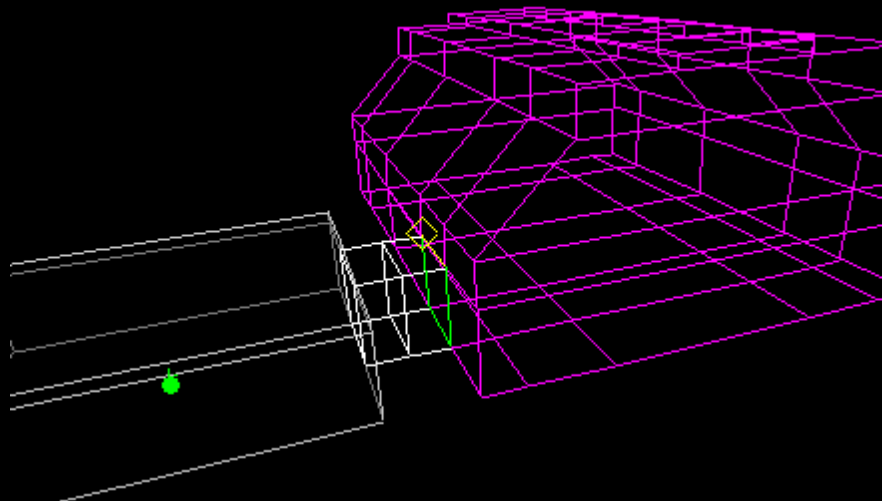


Now we go to the room menu and Click onPlace Room at Current Room.

In the picture on the right you can see the result.

Now into that againRoomMenu and upAttach roomclick.

We made the two portals the same size so that the rooms fit together exactly. You don't have to make the faces the same size, but the risk of a mistake is smaller and we can change the entrance beforehand design (columns, beams, etc.).



Makes the new room the current room and switches toWindow/Room Properties. Now give the room a name**The ingot**and closes**Room Properties**. If your nowVerify Mine

If you let it run, you will see that you have no errors in the level, except for the boa. Now calculate the light with the settings as in the picture on the left.

Lighting	
Lightmap spacing:	5 <input type="checkbox"/> Combine lightmap faces
Iterations:	9999999 <input checked="" type="checkbox"/> Volume lights
Ignore limit:	0.005000 <input type="checkbox"/> Super detail

Now save the level and test it.

Back to Section A

008 - Reactor <>

Schplurg

For this topic we need the file "thelevel_2.zip" from 'Reactor Room'.

In the previous topic we added a room to our level. In this topic we will put a reactor in the room, we will do that with the "**Lathe**" Done the tool. We will also insert a player starting point.

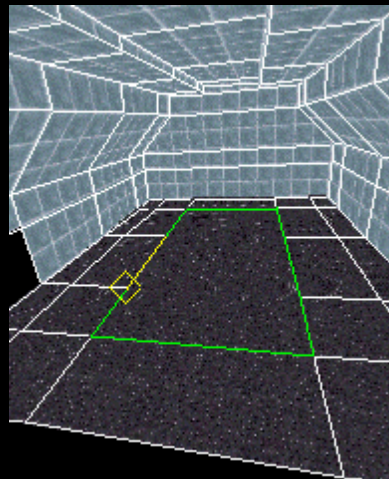
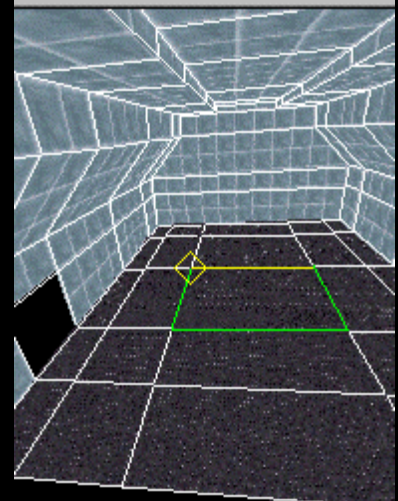
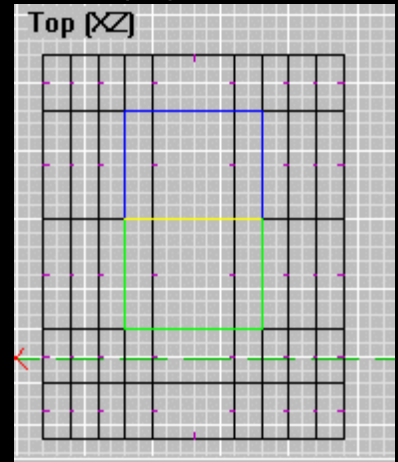
First we use the zoom button to see the room as a whole **S**. We will now build a raised platform on which the reactor will later stand.

Combining Faces

To combine two faces, they must be the same size, as you cannot combine faces that would result in a concave face when combined. You also get a note from D3Edit that this is not possible.

Click on the first face and mark it (button **Space**). Now that clicks adjacent face so that it is the current face. **Don't mark!** It should look like the picture on the right.

Now click on that **Combine Face** Button (left, marked red). Now make the next face the current face and click on this button again. You can see the result in the middle picture on the left.



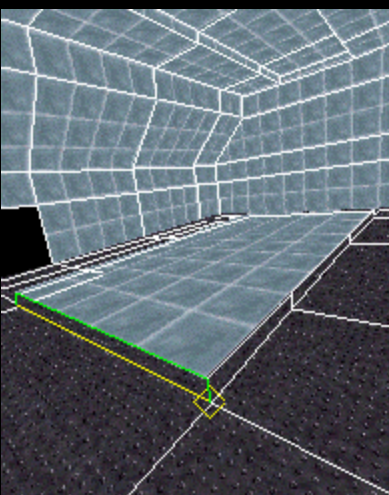
Now we will use the Extrude function to create the platform. We extrude the face by 5 units (Extrude From = Current Face, Extrude along = Normal, New Face normals should = Point outward). Put another tick in the box "Delete base Face(s) after extrusion" so that D3 doesn't die hidden faces have to be calculated. Now save the level!

Lathe tool

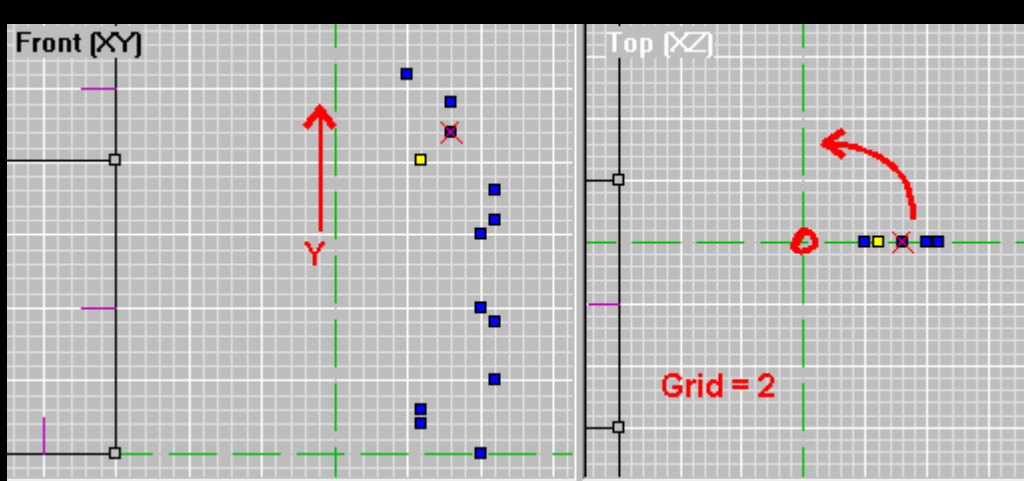
This tool is one of the most powerful in D3Edit. Creating round or square parts is no longer a problem. You can also apply what you learn with this tool in a 'real' 3D program.

The tool does the following, it takes some placed verts and moves them around a center line and creates faces.

Now switch to vert mode and check that all verts are unmarked. We should shift our field of reference. In the following picture it is outside and on the right side of our room. It is better to set things up outside of rooms as it is clearer. When we're done, we'll move it to the right position.

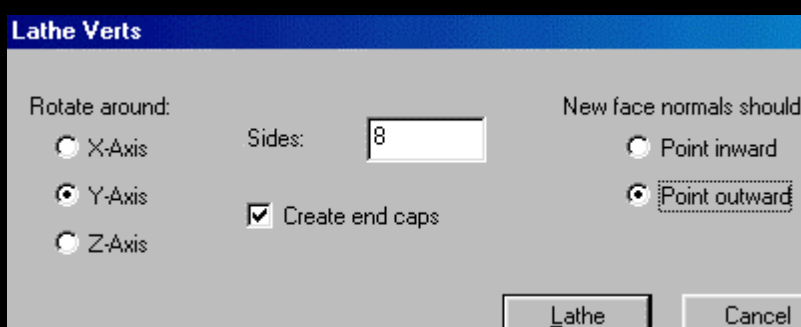
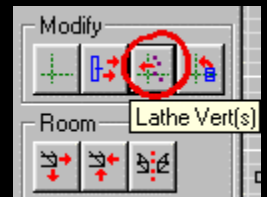


the extruded platform



Place the verts like in the picture on the left, from top to bottom.

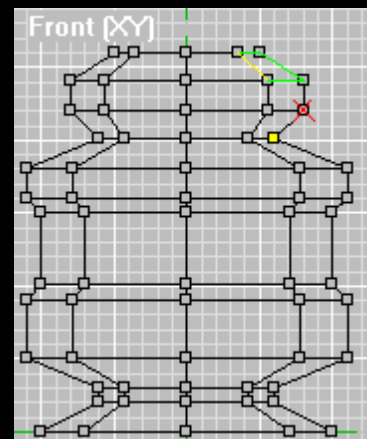
Opens the Lathe Tool, by your on the button from the right Image clicks.



Most of the options are self-explanatory. Create end caps since we didn't place the verts directly on the y-centerline. This space is then encompassed by a faceend cap is called. If we don't create them, we will have a hole at the top and bottom. Sides means that if we look at the reactor to see 8 sides

are.

Now press the Lathe button and see what happens. (Picture right)



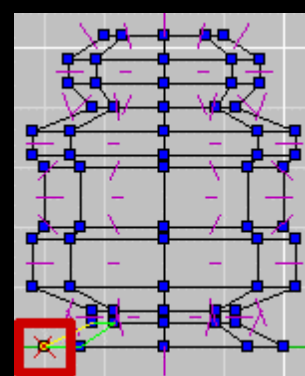
Now we come to the UV's (texture alignment). This is a very important feature to get the textures right position. Imagine you have a brick texture and several faces that contain this texture and are adjacent to each other. If we don't align these textures, the joints from the brick texture would not match.

Now we want to start aligning our textures (is simply). First, texture the reactor. You can texture it however you like. When you're done, press the button **M** to highlight all textures in the room. Now click on in the texture bar **Align...**, a dialog box like the picture on the left opens.

Select the option above **Marked Face(s)** and clicks the button **Default UVs**; The textures in your room are now "optimized".

Moving the reactor

First we have to deselect all faces using the button **U** (Unmark). Now we switch to vertex mode, press the button again **U** press. We continue to work in the **Front (XY)** opinion. Mark all verts from the reactor and make one of the lower verts Current (picture on the right). Simply click on the face and press the button **v** select one of the bottom ones on your keyboard.



Now we have to get the face number of ours

Find out the platform on which the reactor should stand. To do this, click on the platform and look for the face number in the status line. (picture above)

Write the number down or remember it. **annotation:** You will still need the edge number, but in this example it is 0. If you want to check this, click on a face like in the picture and press the button **E**. In the status line you can then see which Edge is currently active. (E: 0/4 then means Edge 0 of 4 is current)



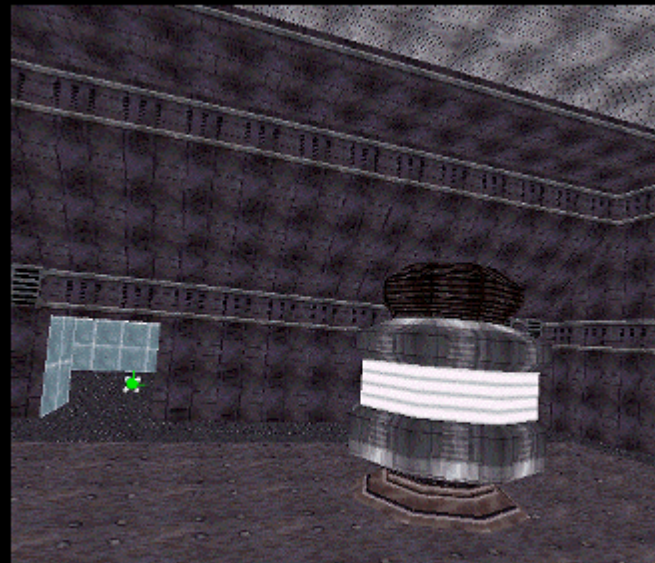
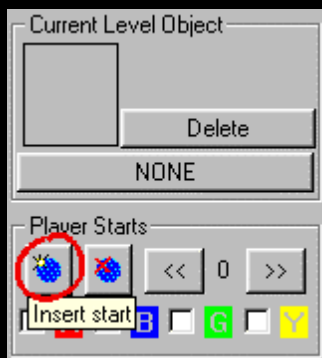
We now click on the button like in the picture on the left and will then be asked for the face number (enter 105). Now we are asked for the edge number and enter 0. Our reactor is now on the same level as our platform. Now set the grid to one in order to then place the reactor in the middle of our platform. Now move the reactor with the buttons **4** and **6** on your **NumBlock**, with the help of the standard below, you can put it in the middle.

Inserting a player starting point

If you see the small green dot in the picture on the right, that is a player starting point. We will now place a second one in our reactor room. Opens the object bar with the button **in t** in the menu bar.

Positions that **X** with a click of your mouse or join in **Ctrl + cursor keys**. You have to do all three Views to see if that **X** also lies in the room. Now press it

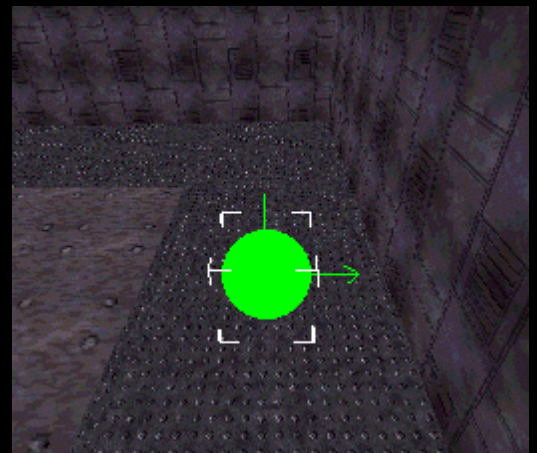
Button that is outlined in red in the picture on the left. Now you have two player starting points in the level.



You can see which direction the player is facing by the green arrow. I still have to rotate mine, so the player starting point has to be current. You can see this in the white lines that surround the object. You can move this point using the keys **8th**, **4**, **6** and **2**. You can rotate the object using the keys **7** and **9**, on the number pad.

Now the light has to be calculated. Sets the following: **Lightmap spacing = 4**, **Iterations = 99999999**, **Ignore limit = 0.000500** and check the box

Volume lighting. Once you've done that, save the level and test it in Descent.



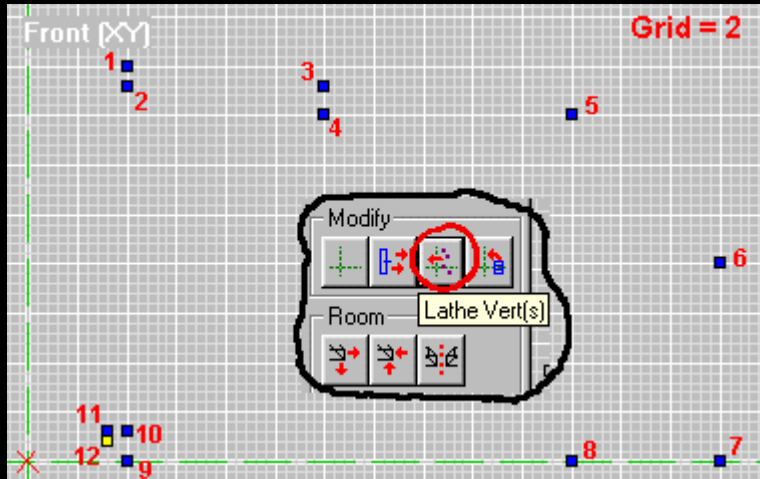
Back to Section A

009 - Lathe a room <>

Schplurg

For this topic we need the file "thelevel_3.zip" from 'Reaktor'. Opens D3Edit and the file "thelevel.d3l".

We had already worked with the lathe tool in the previous topic. We have a reactor in created in a room. Imagine if the reactor was a huge room and the faces were facing inwards, then you could fly into it. We will now create a room using the Lathe Tool.



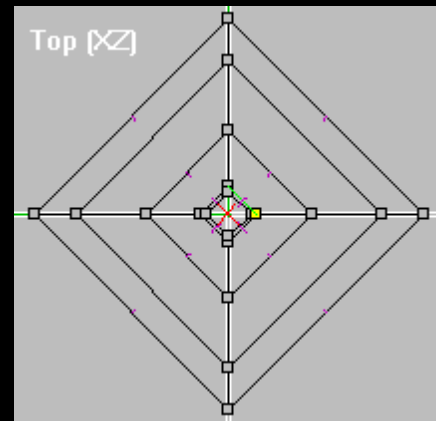
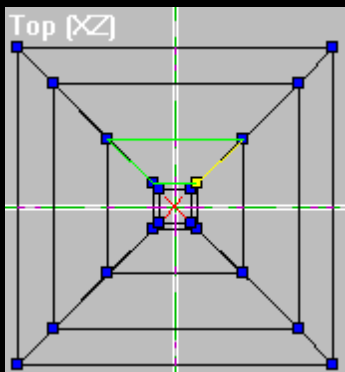
Now we use the tool to create a room with 4 sides. Open a new room and set the verts like in the picture on the left.

Note the distance between the vertical reference line and the nearest verts. This place will be with the **end caps** filled.

Saves the space.

Opens the Lathe dialog box. Make the following settings: **Rotate around=Y**, **Sides=4**, **Create end caps** put a tick **Normals= Point inward**. Click on that **Lathe** Button.

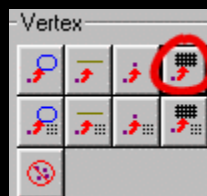
You can see the result in the picture on the right. Now switch to the Top view (XZ) and mark all verts with the key **M**. We still have to rotate the room. Press the buttons **1** and **3** on your num pad and turn it until it looks like the picture on the left.



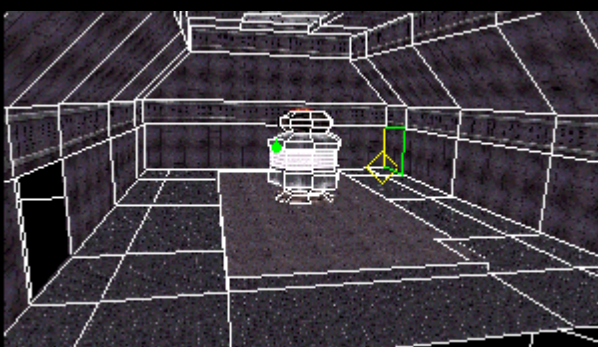
Next we have to align the room with the grid. If you look closely, you can see that the verts are not on the grid.

Note: When I did this, I had to set the grid to 2, otherwise I always had non-planar faces!!!

So set the grid to 2 and make a vert the current one (yellow). Now click the button in the picture on the right, which is outlined in red. Now the current vert is dragged to the grid and so are all the selected ones. If you now in the menu **File.../Verify Room** If you let it run, you can see that our room has no errors. Saves the space. Now we can texture the space.

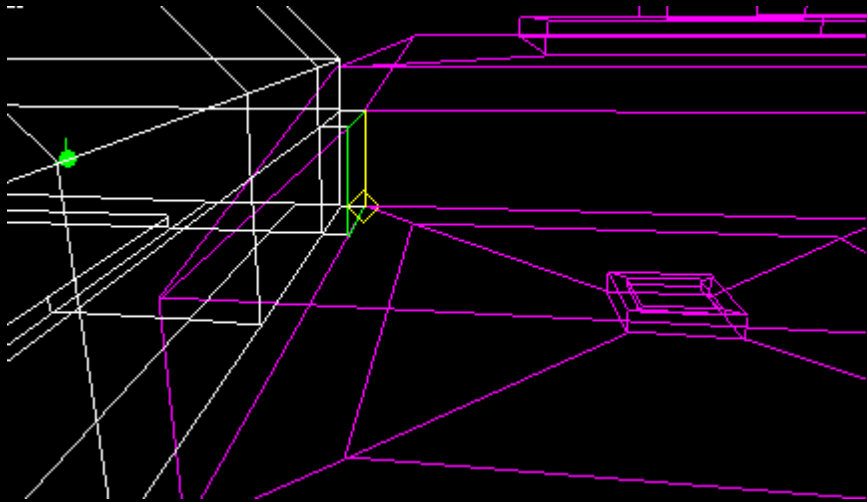
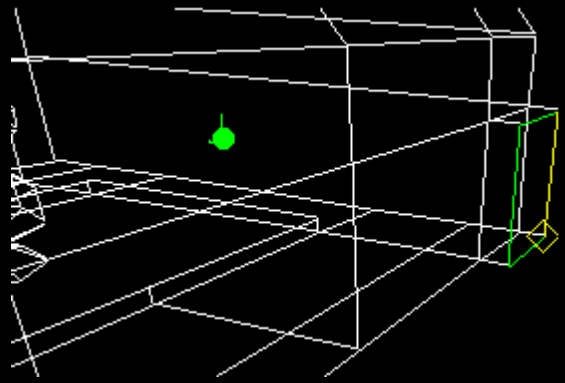


Adding the room to our level



Switch to World View and click on the reactor room. Now switch to the Room View view. We need to prepare the portal to the new room. Now click on the face that can be seen in the picture on the right and extrude it along the X-axis by 10 units, Normals Point Outward (so that the room is not directly connected to our reactor room). After that's done, texture the new faces.

We now make the face that can be seen in the picture on the right the current face. Now we switch back to our room that we want to connect and mark all faces in face mode (button **M**). Now we have to make one of the four sides of the room the current face. We switch back to World View and click on the menu Room on Place Room at Current Room. Now is our space that we tie up want, marked purple and is in the middle of our new room. Now go back to thatRoom Menu and select Snap Placed Room. Now the room will be oriented differently, we can influence this by pressing the button **V** Press on our keyboard to change the current vert, which is marked with a yellow square in the picture.



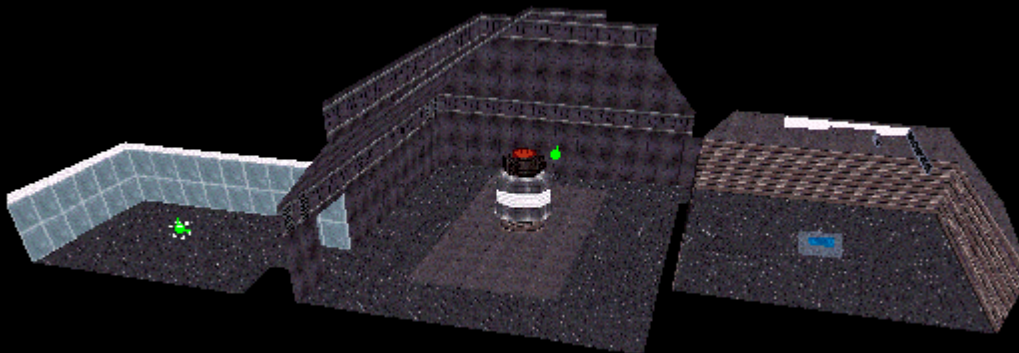
Try it out a bit to see how the room can be connected and how you can align it. Snap it until it looks like the picture on the left. If that's the case, then back to thatRoomMenu and up

Attach roomclick, now the room is attached to our level.

Now save the level.

Now calculate the BOA (lights), you should know how to do that knowledge. Now you can test the level.

Here is a picture that shows what it should look like by now.

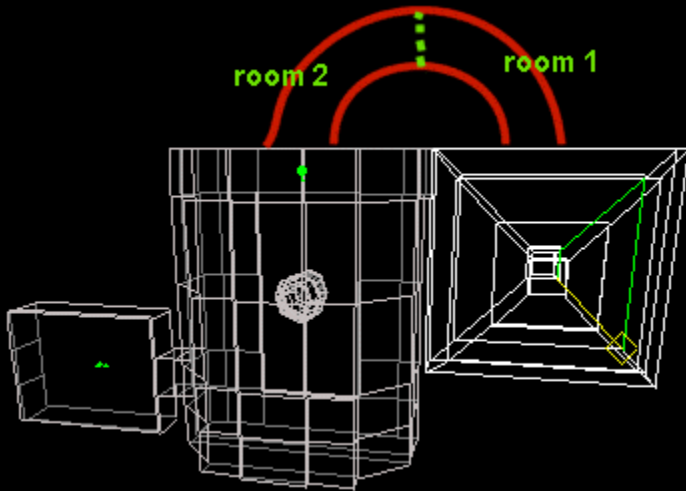


[Back to Section A](#)

010 - The Bend Tool <>

Schplurg

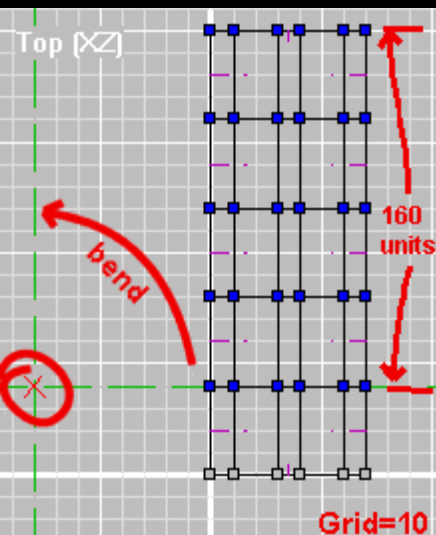
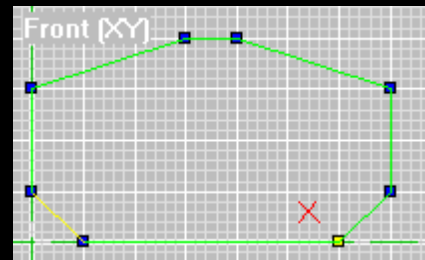
For this topic we need the file "thelevel_4.zip" from 'Lathe a Room'. Opens D3Edit and the file "thelevel_4.d3l".



In this topic we will create a tunnel that looks something like the image on the left. To be more precise, there are two tunnels, each at 90°, one curved to the right, one to the left. There are also other ways to create tunnels, but there will be more about that in the "Tips and Tricks" section.

Creates a new room and in this one, in the front (XY) view, a face. You can do it

Design it however you like, just make sure that it is not concave and that the height of the face does not exceed 40 units (because of the connection to the existing rooms). You can see my face in the picture on the right.



Now create 5 segments of 40 units each with the Extrude function. (Z-axis) Once you've done this, your tunnel should look like the picture on the left.

In this picture you can see further information. The Red one **x**, which is circled, is the point around which the tunnel is "bent". Only the verts that are marked will be "bent". The horizontal line (green) must lie exactly on the bottom marked row of verts. This is where the rotation begins. To move the green line, press the **Ctrl**-key and click with the left mouse button. Set it up like in the picture on the left; 80 units distance. Once you've done that, save the room `base_tunnel.orf`, we will need this again.

Now click on that **Bend Verts** Button. In the following dialog box you click **Y - axis**. In the input box **Distance** you enter 160, that

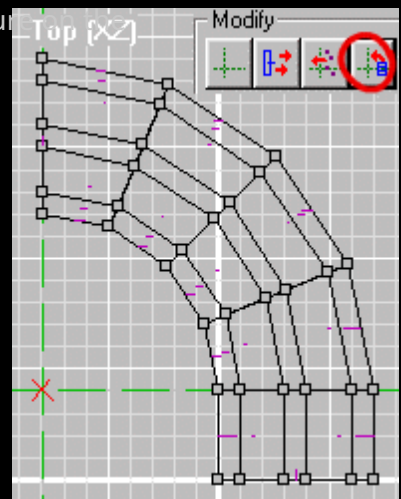
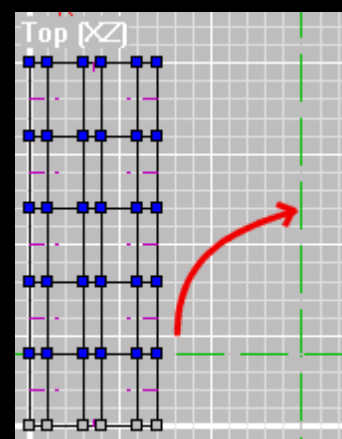
160 is calculated as follows: We have 4 segments, one segment is 40 units long, $4 \cdot 40 = 160$.

In the input box **Angle** you enter 90 because we want a 90° rotation. When that's done, press the **Bend**-button. Your tunnel should now look something like the picture on the right. Save the room again, this time under the name `tunnel_rechts.orf`.

update

Opens now `base_tunnel.orf`, this way we don't need to create a new room. We will now create the left half of our 180° tunnel.

Set the green auxiliary lines as in the picture on the left (80 units to the left). Now click on that again **Bend Verts** button and then click **Y axis**. In The Field **Distance** you give **160 (Negative 160!!!!)** one, you can try something different, but then you will be everything



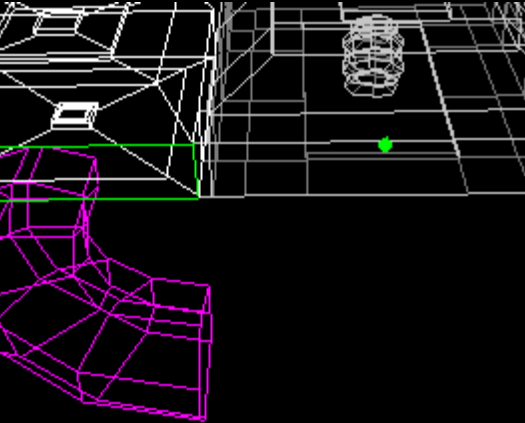
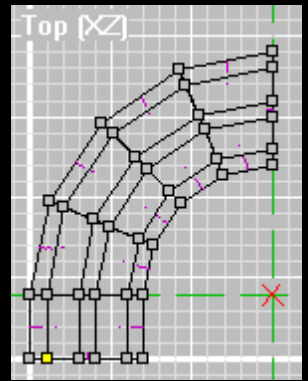
create possible ones, just not what you want. I can't explain this phenomenon to you either. Ok, at **Angle** you enter 90 again. Press that **Bend** button again, you can see the result in the picture on the right.

Saves the room as `tunnel_links.orf`.

Now we have to add our tunnels to the level by opening the room

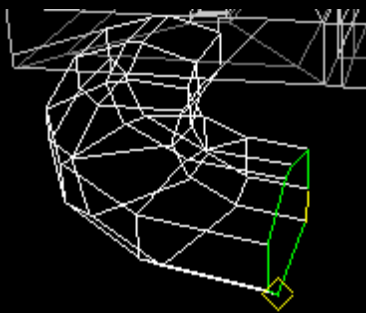
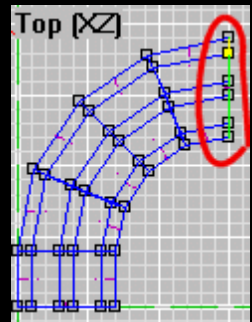
`tunnel_rechts.orf`. Switch to face mode, mark all faces and do

the very first face to the current one. Go to World View and make the face where you want to add the tunnel the current one. (picture on the left)

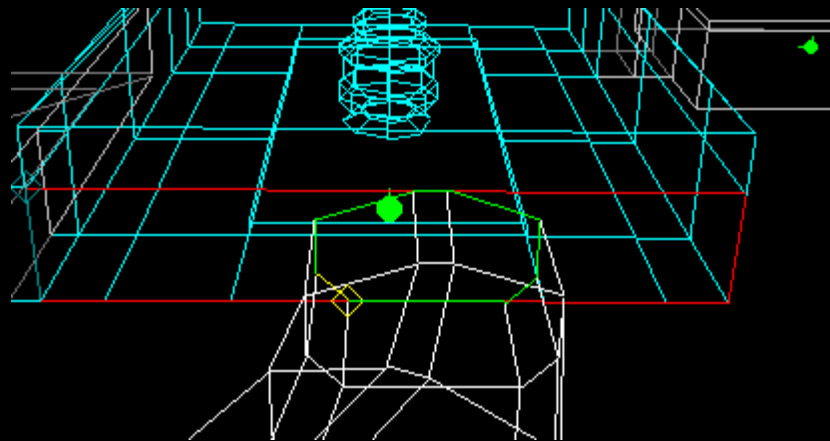


Now switch to the menu `Room/Place Room at Current Room`. Ok, our tunnel is correctly placed, choose `nowRoom/Attach Room`. Now the first part of the tunnel is in the level. Now open the room `RoomTunnel_links.orf`,

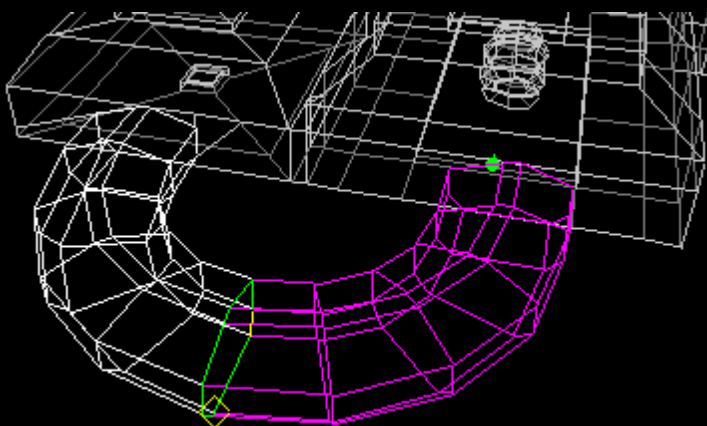
marks all faces and makes the face marked red in the image on the right the current one. Now switch back to World View and make the face shown in the image below the current one. Now select in the room menu `Place Room at Current Room` and then `Room/Attach Room`.



Now we have to create a portal that connects the tunnel to the reactor room. To do this, mark the face of the reactor room in the world view. First make the corresponding face the current one, then mark it with the button **M**. Now make the face of the tunnel the current face. The face of Reactor room, that marked is shown with a red border (picture on the right).



Go to the `nowRoom`-Menu and select `Join Rooms`. You will now be asked whether the space should be in the middle of the face, click on **NO** or **No**, since our faces are flush with each other.



Perfect!!! The tunnel is now complete. Texture it with some lights, calculate them and test the level.

Back to Section A

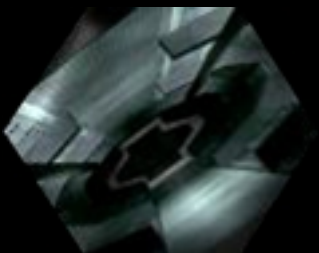
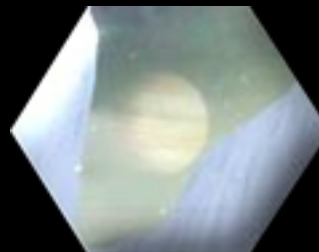
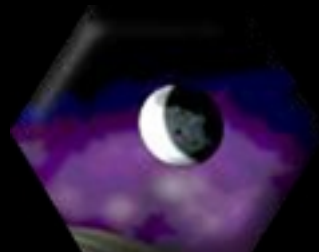


Robo's walkthrough basically covers much the same thing as Schplurg in his first five tuts, but in a little more depth.

If Schplurg's The Level was too fast for you at first, this is the right place for you.

The original is in English and can be reached at the following address:

<http://www.planetdescent.com/d3edit/>





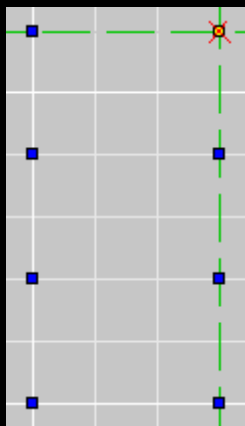
011 - Basics

Robot

Before you start getting good at something, you need to know the basics. For example, before you speak English fluently, you start with simple words. The same applies to level building. You have to know the basics to be successful. Definitions are a good starting point as you can start to learn terms.

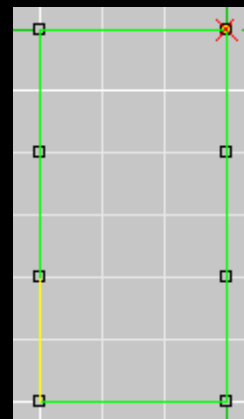
Vertex:

An invisible, zero-dimensional point that connects edges to form a face. In D3Edit they appear as small colored squares displayed, which are gray if unchecked, blue if checked and yellow if current (these terms will be explained later). Also short **Vert** called.



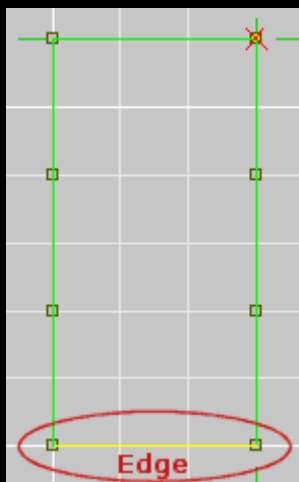
Face:

A polygon with a minimum of three vertices. They are represented by black lines when unchecked, blue when checked and green when current, similar to the vertices. Levels are simple polygons arranged to form walls, floors, ceilings, pipes, reactors - everything you see.



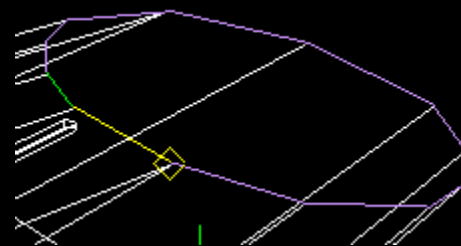
Edge:

An edge is a line that connects two verts in a face. They are always yellow, but you only see them when you have a current face. Easy to understand once you try it.



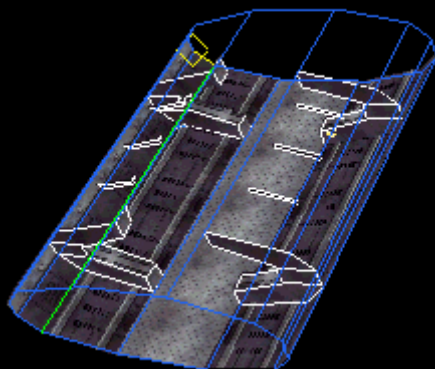
Portal:

Not what you think. A portal is a pair of Faces, which two rooms connect. She are in-game invisible, except You turn them into force fields or Similar. She can also be used to trigger events. In D3Edit they are purple when selected.



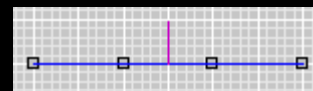
Shell:

These are the external faces in every room, which are the Ah, that's very difficult to understand (and explain). inner your level from the dark abyss separate, which we like to call the " - I have it in the picture on the right the shell faces marked, so You see better can that it the outermost ones Faces are.



Normal:

A normal is a small purple line or vector that looks out of the face in the direction in which the face is pointing. This is important to know for several reasons; the the Face is only visible from one direction. One page is with whatever you want textured, metal, water, wood or with the face of your next victim, the other side is invisible. You can actually fly through it! 😊



Think of a clock: if the hands go clockwise, then the dial = the face looks towards you, the clock is 'visible'. If the hands go counterclockwise, you look at the clock from behind and don't see anything.

Okay, those are the main terms, now let's get to the interface. This part might get a little confusing 😊

panels

There are currently eleven panels in D3Edit, each with its purpose. They are accessed by clicking on the corresponding buttons. Here is a list of all the buttons, their associated panels and a short description:

C	Camera control	This is used to manipulate the camera views.
R	geometry	For handling polygons, verts and edges. This works on the outside area.
Tr	terrain	
Tx	Textures	Manages the textures
P	Paths	Adds paths for robots and cameras.
O	Objects	Organizes the objects.
D	Doors	Adds doors between rooms.
S	sound	For handling the sound sources in your levels.

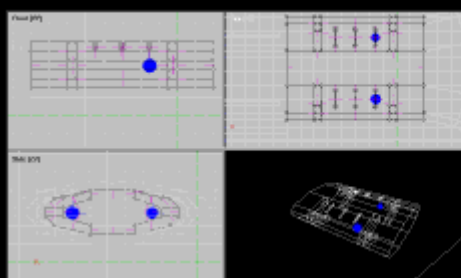
Since D3Edit version Dev8 there is also:

M	Messages	Here you can view messages from D3Edit without having to open Notepad.
Q	Quick test	A great tool for quickly testing your level.
K	mark	Allows you to select faces according to certain criteria.

It might be helpful to print this out and pin it somewhere for easy reference. After a while you'll know it by heart 😊

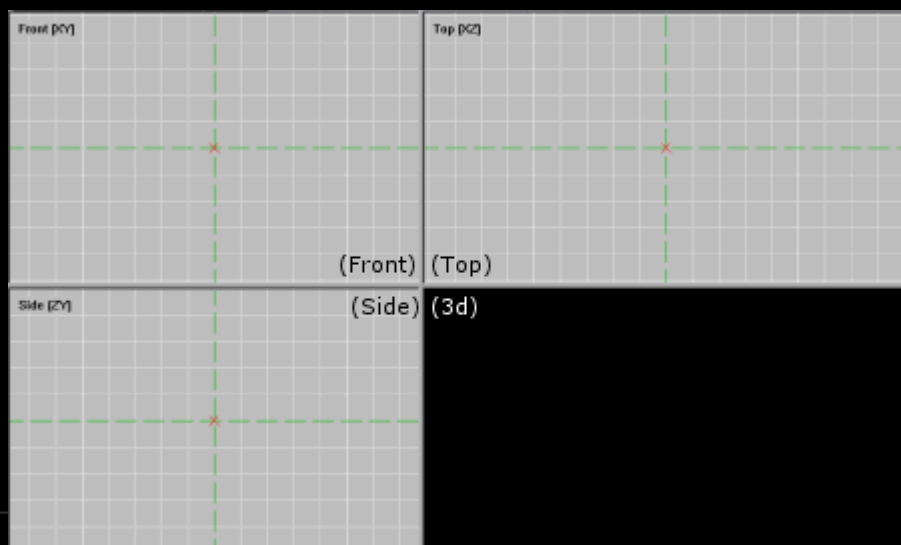
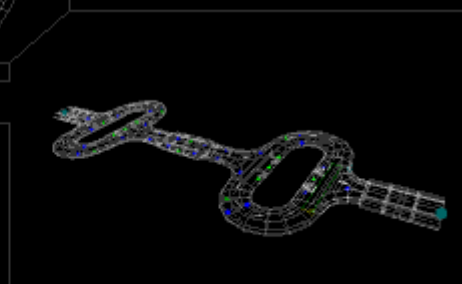
Viewports

There are four views: Top (Top, XZ), Side (Side, ZY), Front (Front, X-Y) and the 3d view. They show your rooms from different perspectives. (Picture right)



Room View

World View



Current Room View / World View:

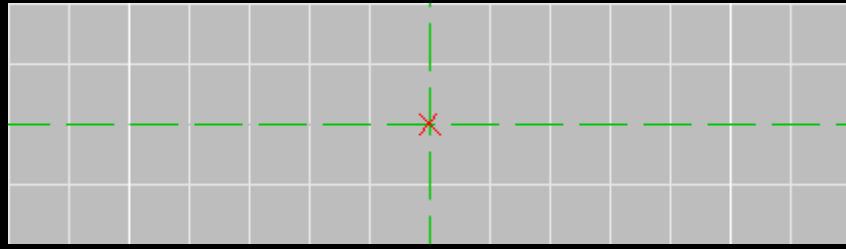
Here on the left you can see the two views in D3Edit, each titled. The top view is this **Current Room View**, the lower one **World View**. If you want to see the whole level, go to the World View, but if you have one

If you want to look at the room that is part of your level, you go into it **Current**(more selected)**Room view**. This is a bit confusing, I know. You can only have one level open, one room selected in the level and 50 rooms that are separate from your level. The separate rooms have their own room views.

The grid and the axes:

To help you position everything exactly, D3Edit has the Grid. It can be found in the 2d views (Front, Top, Side), its size can be adjusted and it can be turned on or off. Not difficult to understand, very different from other D3Edit

Features.



Because you're playing a 3D game, you have to do everything **do in 3d**. To better deal with the grid, you need to understand the axes:

- **X**–Left Right
- **Y**–Up and down
- **Z**–Next previous

This means that the directions in XY are shown in the front view, the directions in XZ in the top view and those in ZY in the side view. You'll figure it out if you experiment a little.

That's the basic know-how. And we continue...

[Back to Section A](#)



012 - A simple room

Robot

If you have downloaded and unzipped D3Edit, you will see a settings dialog like this when you first start the program:

That is **my** Settings window. I suggest you copy the settings, but you have to point to the Descent 3 directory if it's in a different directory than here. After this is done, the D3Edit starts normally.

Turn off all panels (using the buttons in the toolbar) and go File...New

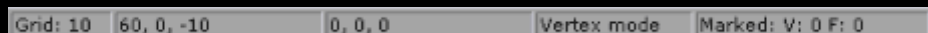
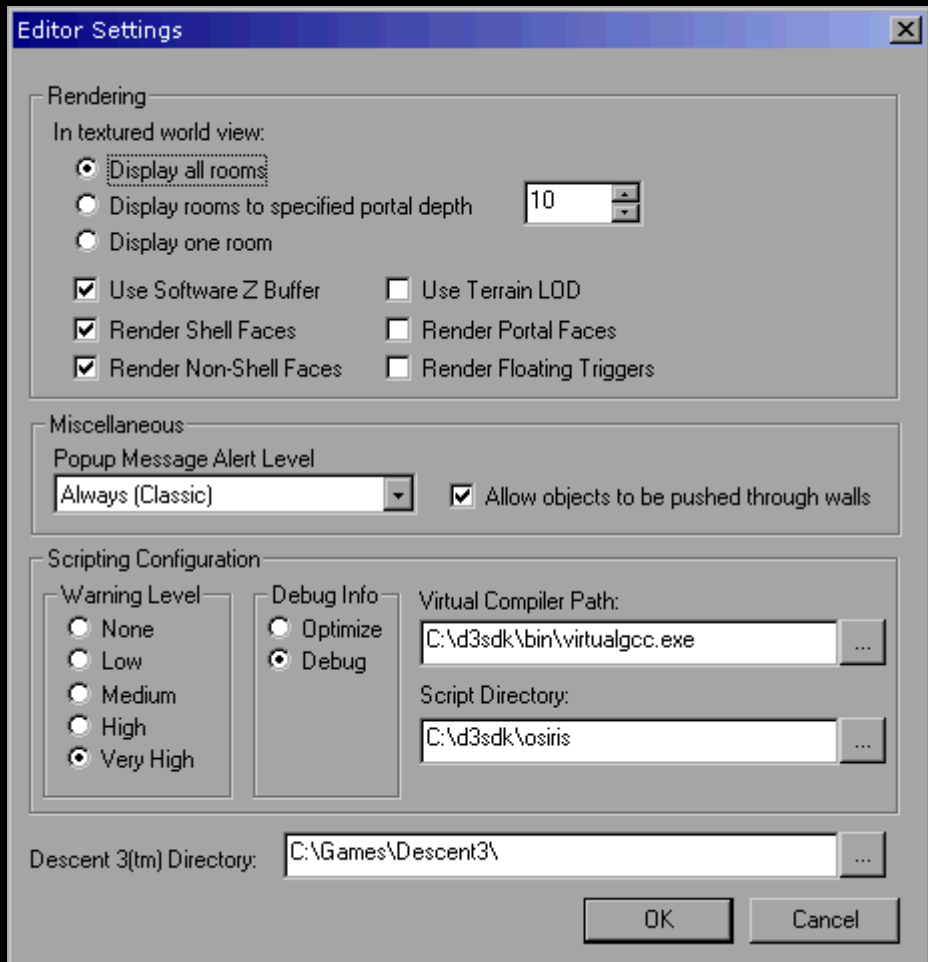
and choose New Room. Now

let's get to work. Creating faces is as simple as connecting dots, where the points are verts. Let's try it. Make sure you are in vertex mode by going to the

Look at the status line:

It shows you the position of yours

Cursors, your set grid size, which mode you are in and how many faces or verts are marked.



Then click twice in the top view to make it active and the red **X** to be displayed and press **Into the** or **Simple**. What you should now see is a small blue square, which is a vertex. To delete one or a group of vertices, press **U** to mark everything first and click and drag a box over the verts you want to get rid of - they will then turn blue, indicating that they are marked. Then just **Del** press -

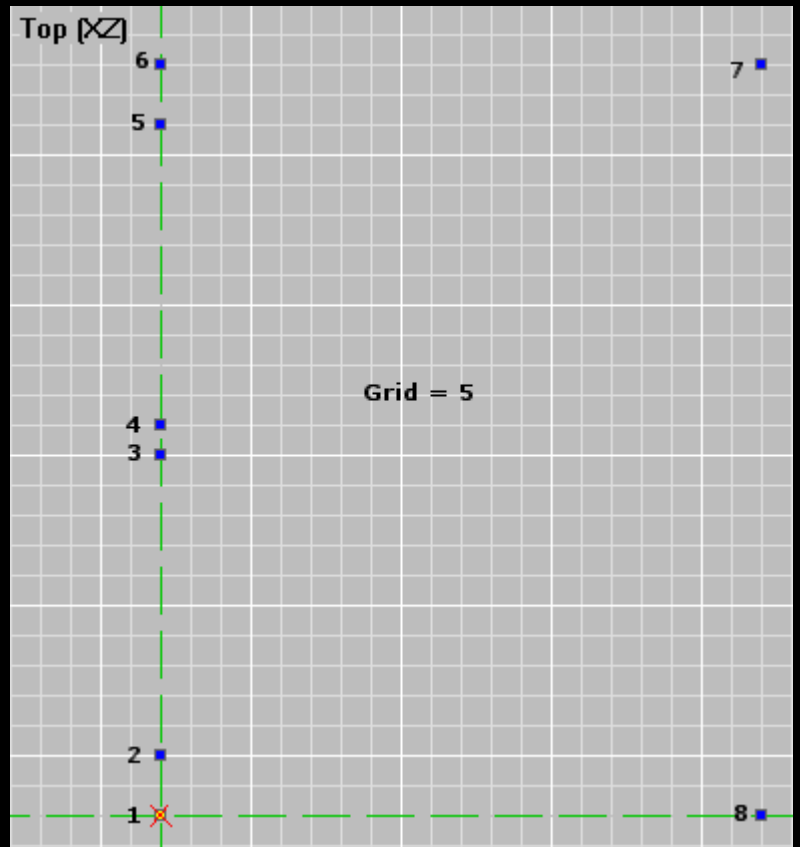
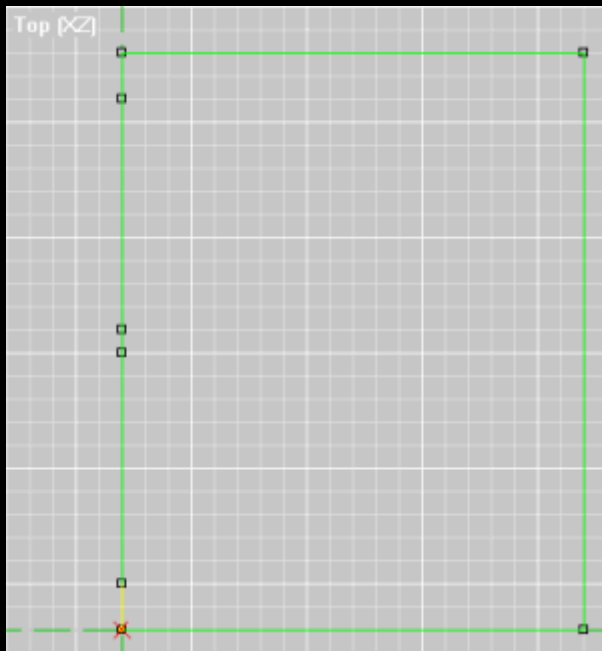
Delete everything you have done so far and place the verts in their correct position and order according to the grid, which means inserting them clockwise. Make sure you have it set to top view or you'll be in trouble. If you make a mistake, use the previous instructions. **Pay attention to the grid size!**

To change the grid size, press **Picture on** to make them smaller, and **Image Ab** to make them bigger. As the size changes, you will need to zoom in or out to see the grid lines by pressing **Ctrl-Shift-Left mouse button** and move the mouse back or forth, or use the keys **S** and **W**. Alternatively is also possible **Ctrl-mouse wheel**.



As you can see on the right, the verts create a rectangular outline that represents the shape of the room. Now go by pressing **Ctrl-F** into face mode, check by looking at the status bar and making sure the verts are still highlighted (blue), and press **Into the**.

You should now have a face like this:

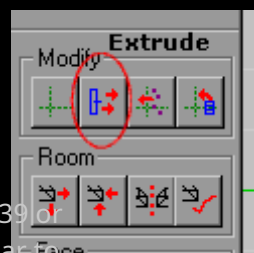
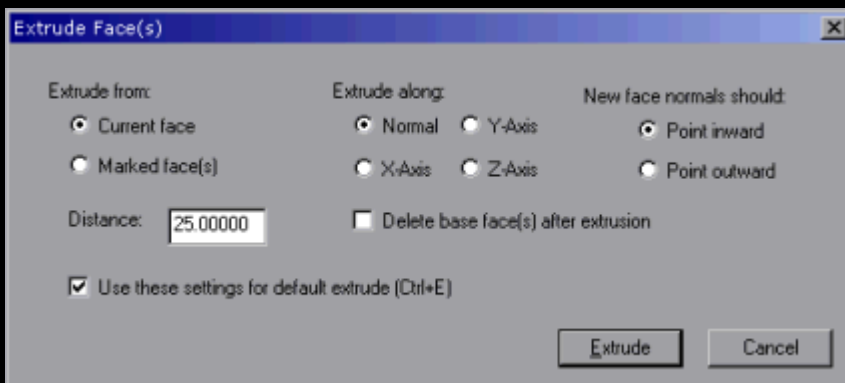


Your first face 😊

This is the system used to create faces from scratch. All you have to do is place the verts in the order you want the edges to be. Here it is 100(x) by 125(z) Descent3 units, reuse the grid to remeasure.

Now we'll make it a room, with a floor, walls and a ceiling - using the Extrude function, which is used to manipulate individual faces. Click on the button **R** and what you see is the geometry panel, with its many buttons -

Click on your new face to make it current (green), then click on the Extrude button (right).



If you are still working with v39 or older, then a dialog box similar to this one will appear, set the options according to the picture on the left.

Extrude From: Current Face/Marked Face(s)–Extruded from a single current face, or several marked faces. Around

To extrude multiple faces, you must have all of these faces selected and one of them current.

Distance: xx.xxxxx– How far the faces should be extruded. As a thumb measure: 10 units correspond to 1 ship length/width/height.

Extrude Along: Normals/X-Axis/Y-Axis/Z-Axis–The direction in which the face(s) will be extruded. If 'Normal' is selected, the face will be extruded in the direction the normal faces.

New Faces Normals Should: Point Inward/Point Outward – Determines the direction in which the normals of the new faces should point; In the case of Inward they point inwards (faces visible from the inside) or outwards in the case of Outward (faces visible from the outside).

If you have the new v40, your dialog box will look like this:

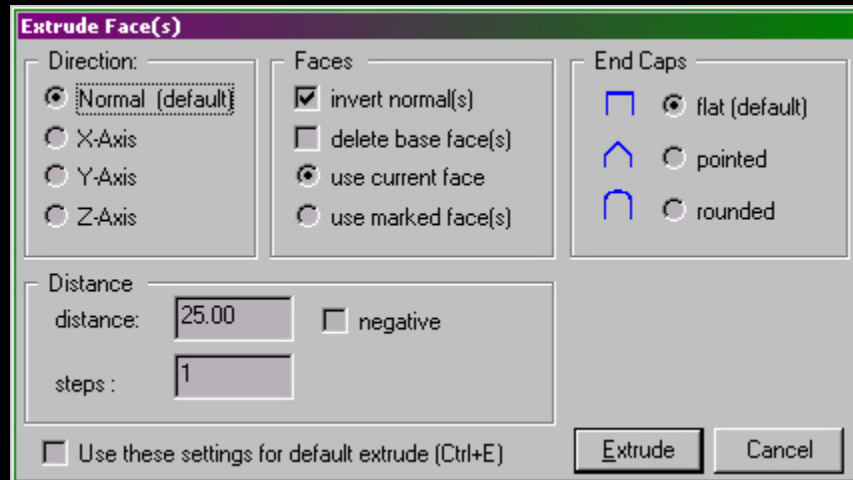
Faces-Here you enter your base face, whether from the current or from the marked face(s). Here you also say in which direction the normals of the new faces should look; by default they face the same direction as the base face.

Distance-Here you specify how far.

Ticking negative reverses the direction.

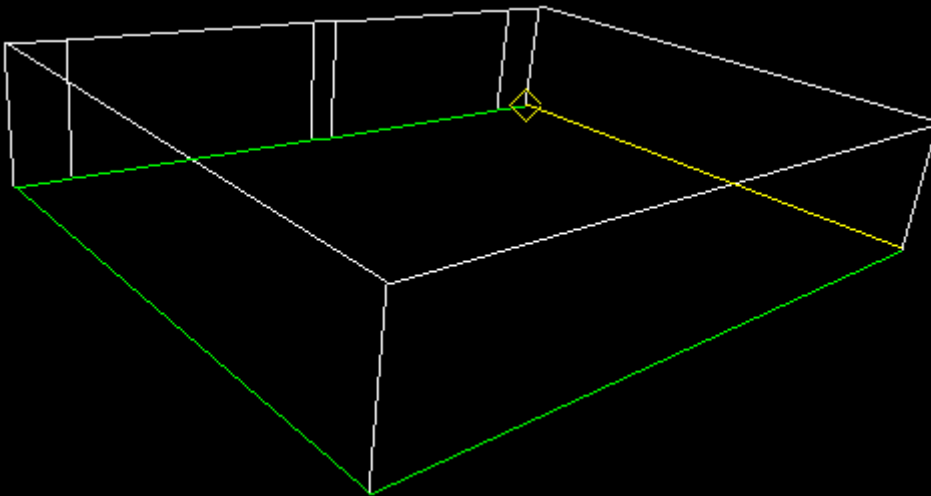
Direction-The direction in which it should go.

If 'Normal' is selected for Direction, it goes in the direction of normal from the face.



Adopt the settings from the image that applies to your version,

Then press **Enter**... and then you have it:



Woohoo! A room! As you can see, it has a floor, a ceiling and walls - one of which is split into segments.

If you measure the space with the grid in all 2d views, it should measure 100 x 25 x 125 units. If so, go to File... Save Room As and save the room as

box1.orf.



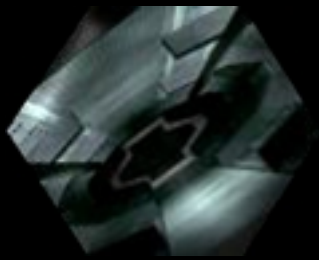
TIP

If you have more than two vertices in a face that lie in a line, then the verts in the line - not the ones at its ends - create separate faces when extruded. In our example space we have six vertices in a line, and after extrusion we have five separate faces. Over time, you will reap the full benefits of this approach.

That's it for now, next we'll make the room a level



Back to Section A

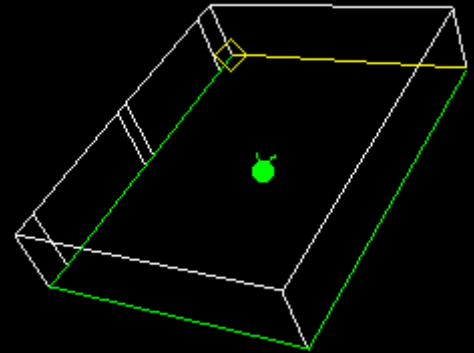


013 - Start a level <>

Robot

If you've gotten this far, I think it's safe to say you're interested. That's good, because now things get even more interesting -

Open D3Edit and go to File...New, choose New Level - Import Room. Find box1.orf and click OK to open it, then go to Room View. I bet you're thinking the same thing every first-timer is thinking: *What the #!%£&*% is that green dot doing? Something doesn't fit...*



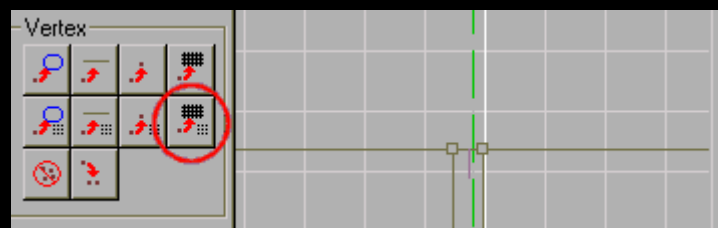
Definitely everything is in order. This green dot is the player starting point (Player start) or spawn point. It's here because you're no longer dealing with a simple room, but with a level.

We're not done yet, we still have a few things to edit and a problem to solve. *gasps* What? A problem? O_o

When in doubt, look between the lines:

If you take a closer look at your room, you may notice that all of the verts and faces are not aligned with the grid. This means that if you wanted to mount something on a wall, like a light fixture or something similar, that would be completely impossible to do. All we need to do is get the room back on the grid before adding anything else. If we adjusted it afterwards, it would create a lot of errors - depending on how complex your level is. This is because aligning moves your space, and if you have already added rooms, they will be separated.

Go to vertex mode (**Ctrl-R**), check this by looking at the status bar and press **M** to mark all verts. Then click on any face to select it, this will make a face green and a vert yellow (both are then current) and press this button in the geometry toolbar (button **R**):



Click on the top view, then on the button, then in the side view and then in the front view, each time pressing this button. This brings the verts back onto the grid in all axes, as only two are shown in each view. This should have solved the problem, if not, try again.

Now go follow Window...Untitled.d3l - World View, what you should now see is your level on a black background. Then go follow up File...Save Level...and save the level as level1.d3l. If you are using a dev or beta version of D3Edit, right-click on the black background and turn off Show Terrain. If you are using a different version, press **Ctrl-T**, both methods make those annoying white dots disappear. Save again.

Now let's check out a few things, the following procedures will set up your level so you don't have to mess around later.

Give me a name:

Go to File... Level Settings..., and forgiveLevel name a name. You'll see it in the middle of the screen when the level loads. I know that the layout of this dialog box varies depending on the editor version, here D3Edit Dev8th. Wear underDesignersYour name in and out copyright(with the year), and give your level a description if you want. Finally, carry the value for themGravityone, which causes napalm blobs and spew to fall downwards. I like her at 16.1

Can you count so far?

Next go to File... Show Level Stats. The box that pops up contains the collected statistics for your level. Currently the level has ten faces and 16 verts, with one object starting the player. However, some people find that their completed levels have very high stats. One of my levels has 21 rooms, 1460 faces, 2128 vertices, 22 portals and 42 objects. And that's just a small level! You can specify these stats in the text file for your level.

Problem time:

Now we will check the level for errors. Go to File... Verify Mine. As you can see, all the numbers are zero, which is good, we have no errors. If there were any, there would be a non-zero number in the corresponding line, e.g.1 bath shell.

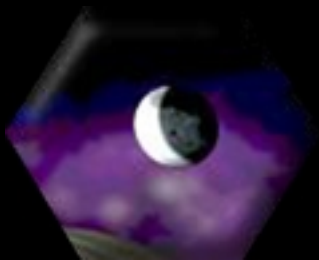
Finally, go afterFile... Compute Boa Vis Tableand select OK. (Boa stands forbigOleArray) This calculates which spaces of points are visible in other spaces, so D3 doesn't have to draw the entire level when you fly around in it - just what you see. This is absolutely essential when building, testing and finally releasing - it always has to be calculated. You can find out more about the Boa Vis table and its relationship to rooms under Adding New Rooms.



When you release a final version, or an update, it's a good idea to make a checklist of things to change, add, or remove from the level.

Next we're going to do a bit of texturing, it's time to get rid of this default texture and add some metal. (Or brain!)


Back to Section A



014 - Texturing <>

Robot

Alright, time to do a little texturing! This should be short and sweet.

Open `level1.d3l` and go to World View. Right-click in the background and select **Textured with outline**. Activate the texture panel with the button . This panel is used to texture the level, paint it so to speak, align, pick up and select textures

This is what it does:

dig: Takes the texture from the current face and puts it into the Current texture box.

Align: Allows you to rotate, move, stretch and textures to squeeze.

To Current: Inserts the current texture into your current face. **To**

Marked: Inserts the Current Texture into all selected faces.

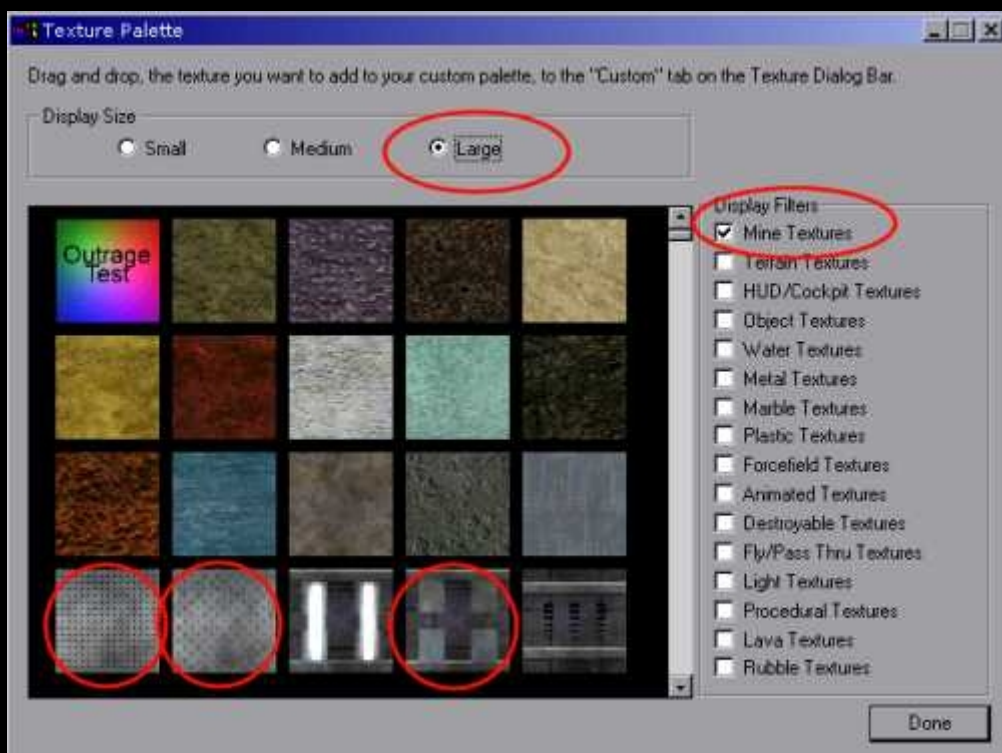
Propagate: Causes textures and/or their orientation to change 'spread' other faces.


Choose: Pops up a list of all textures.

Custom box: Contains all the textures you choose from the textures list have chosen.

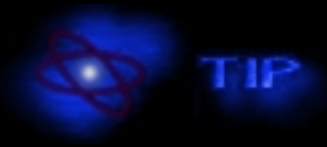
Level box: Contains all textures included in the level.

Don't worry, using this panel will come naturally to you once you get started.



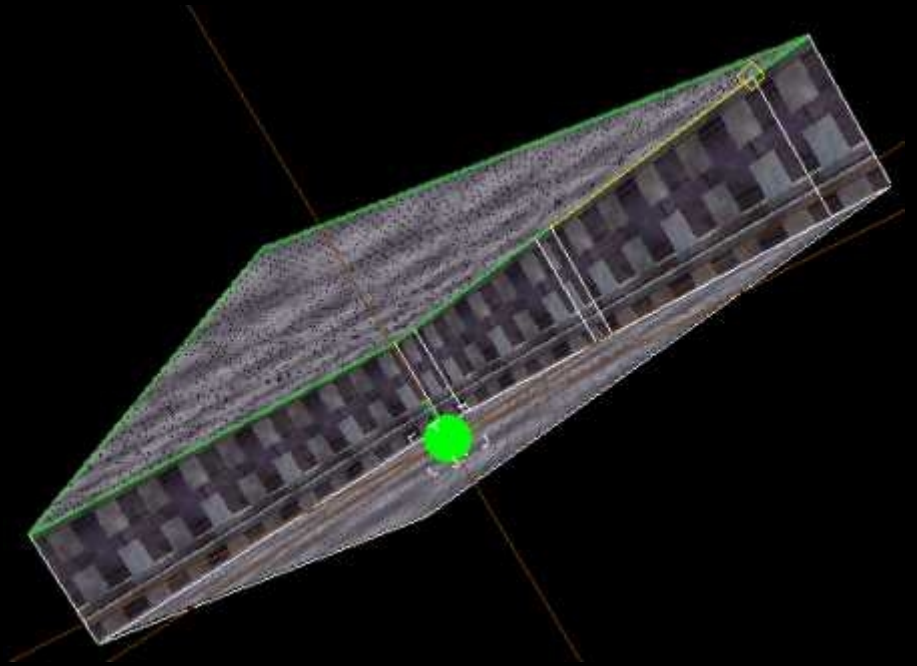
Click on **Choose...** around the . Open the texture list. Then choose **Large** and then **Mine Textures**. As you see There are 949 textures to load, so be patient. To use a texture, it must be in your Custom Texture box - to do this, click on a texture and drag it over the box, then release it. The textures we need are: **LWceil.tga1**,

LWflor.tga1 and **LWout2.tga1**. They are all near the beginning of the list so they are easy to find.



Always plan before you do anything. This also includes texture schemes. Last-second changes to geometry can produce errors if not executed properly.

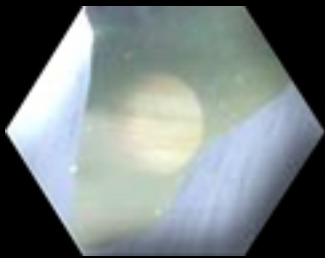
Close the list and go to the Current Room View. Go into face mode (**Ctrl-F**) and press **M** to mark all faces, then select Lwout2.tga1 as the Current texture and then press To Marked, this will apply this texture to all marked faces. Then click on the floor of the room to current it do, select Lwflor.tga1 as current and press To Current. Do the same with the ceiling but use LWceil.tga1. You should be in the 3d view (right click) Textured with outlinecome to this result:



Save...

Finally, things are starting to look good. Next we will illuminate our level and then undertake a first test flight. Continue!

Back to Section A



015 - Lighting <>

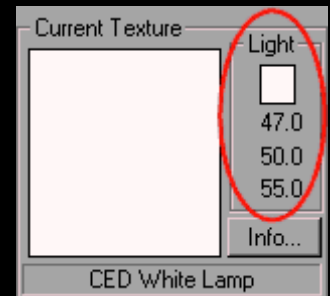
Robot

We have now textured the level to make it more visually appealing. Now we will refine this even further.

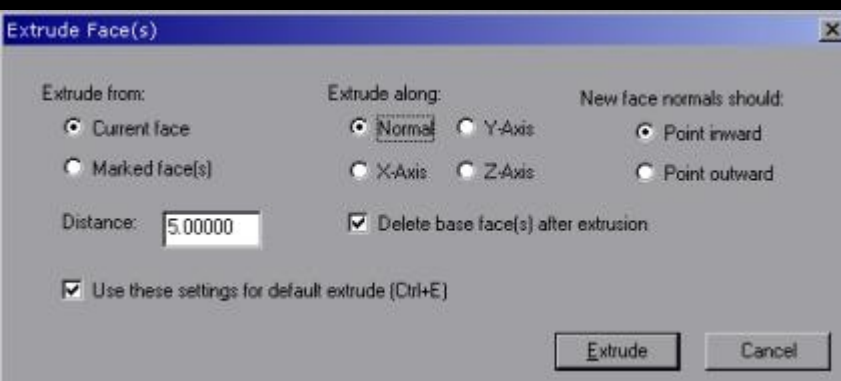
In D3 you will notice that some rooms are darker than others, and some may have colored spots on the wall. This is because of D3's Fusion Engine. This contains a nice little feature with which you can light up your rooms as you want, sometimes creating pretty levels. D3Edit uses this to help you light your levels easily and simply, thanks to the Radiosity program.

Light is emitted by Light Textures in specific amounts and colors. It is very similar to the RGB color model, three numbers that produce specific hues and brightnesses.

On the right we have a Light Texture called CED White Lamp. I sometimes use this texture in large, open areas to create lots of light. The different values of red, green and blue light can be seen mixed together. For example: 17,17,17 is moderate brightness, 4,4,4 is dim light and 255,170,170 is a light pink peach. The square above the numbers shows the color of the light, be careful not to blind anyone, like in my first levels! Ray Charles would have to wink at my second level :P

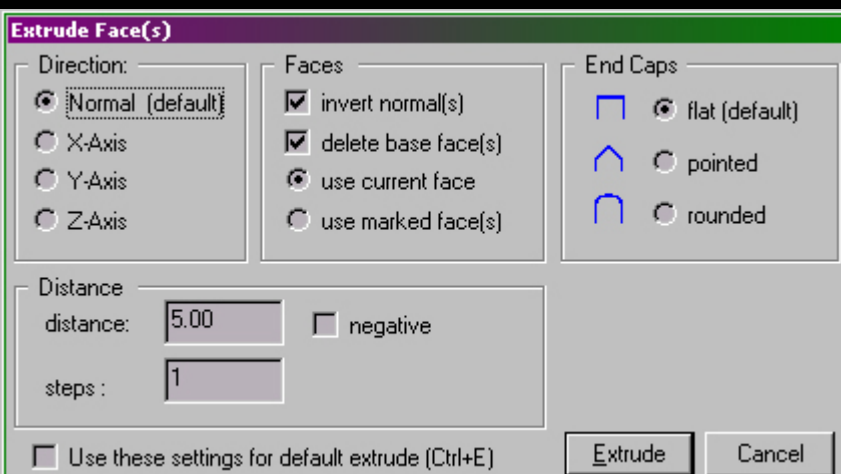


Let there be light:



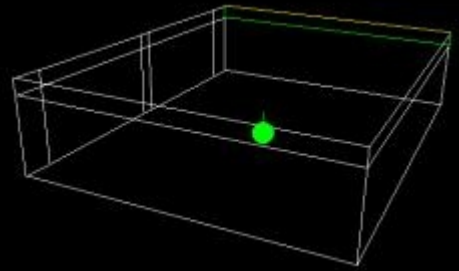
Open level1.d3l and go to the Current Room View. Go into face mode (**Ctrl-F**) and select the blanket (make sure it's actually the blanket), press the **Space bar** once to mark it. Press **N** to flip the face, which causes the normal to point up (can be done in one of the 2d views

checked), then press **U** about everything to mark off. Select the ceiling again and click on the Extrude button, adopt the settings from the image on the left corresponding to your editor version (upper for v39 and older, lower for v40) and press Enter:



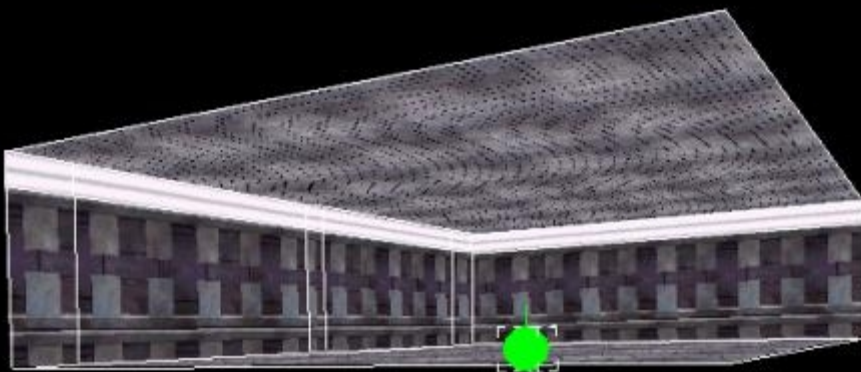
The result should look something like the one on the right. By extruding the ceiling we have raised the 'roof' in our room, giving us a set of smaller faces at the top which we will turn into lights.

You'll notice that this is similar to what Schplurg did in his tutorials. That's because it's easier for you to do and it's almost stress-free.



Texture light:

Open the texture selection list with Choose..., click Light Textures, then find and transfer LightCrossing03x. This will be our light texture. Drag a box over the top set of faces we created to highlight it (blue), and make sure you have this



Once you have selected a texture – press **To Marked**. This quickly and easily applied the texture to all the highlighted faces, but this also texturized the ceiling. So select the ceiling face, in the Current Texture box select LWceil.tga1 and press **To Current**.

You should now get the following:



Try not to have too many bright or large lights in your levels. This can become hard on players' eyes after a while, making them tired (and giving them headaches!). This will also spice up your lighting and make your level look better if you want it to be more atmospheric.

One last thing still needs to be done, namely the light calculation program. Go to World View and save your level, then follow **Window...Lighting** to open the lighting dialog. Apply the settings from the image on the next page and click **Light it!**

Lightmap spacing: 5–Ignore it, it's best to leave it as it is.

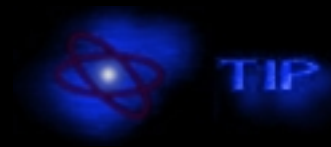
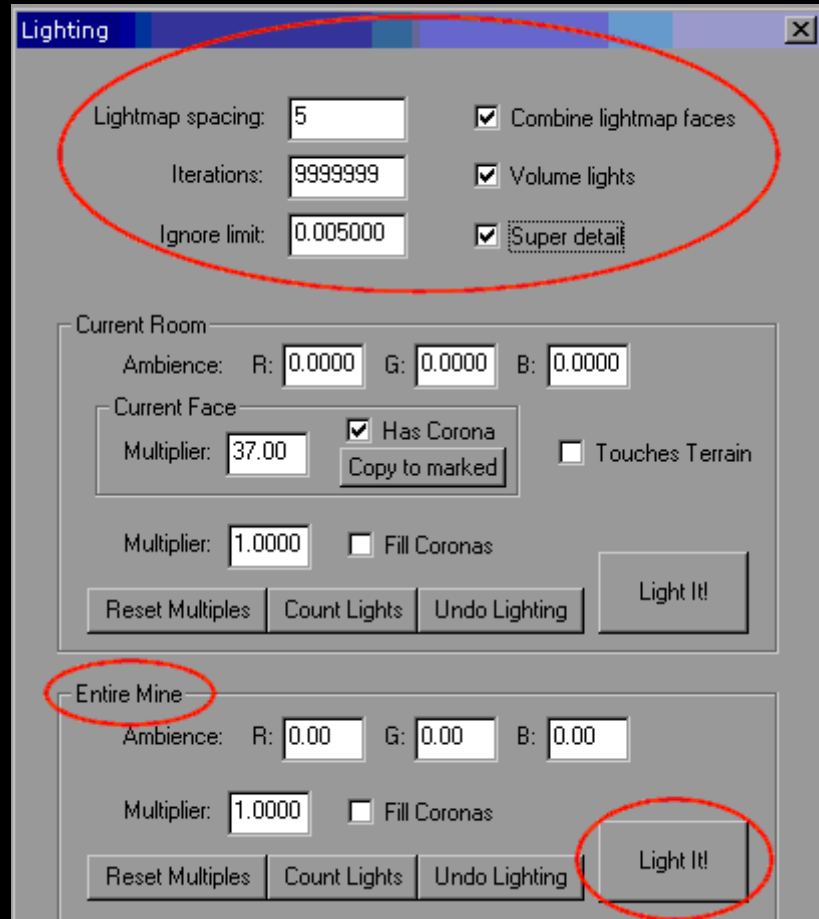
Iterations: 9999999–Specifies how often the light should be reflected from the faces of the level. **Ignore Limit: 0.005000**–The point at which the lighting is so dim that it is not used.

Combine Lightmap Faces: Yes/Nosmoothes the lighting across the faces in your level. It's best to leave it that way.

Volume Lights: Yes/No–When on, ships and objects are affected by the light. When off, they appear at full brightness.

Great detail: Yes/No—When on, the light is emitted from the entire area of a face that has a light texture. When off, the light only comes from the center of the light textures.

Depending on your computer's performance, these calculations should be completed in a few seconds. In large levels with lots of lights, this can take hours
(it can also take days and weeks...)



Volume Lights and Super Detail affect the calculation time for lighting, but also the appearance of the level. Volume Lights is a must, but if your level looks the way it should, it might be a good idea to check out Super Detail.

Save...

Next we will test fly the level.

Additional information: Lighting Guide

At this point an explanation of the lighting parameters, written by Chris "Gwar" Ledwith and Jason Leighton, based on the Lighting FAQ from Outrage; both worked at Outrage. I was able to find this tut via www.archive.org.

Ragil Ral

Texture Light

The numbers next to the texture are in the unit "watts per square foot". This means that the size of the face carrying the light texture affects how bright the light shines. Just as a small light bulb doesn't give off as much light as a large one, a smaller face doesn't glow as brightly as a large one (assuming both faces have the same texture).

Texture Reflectivity

is in the range from 0 to 1.0. Reflectivity determines how much light is reflected when incoming light bounces off the texture. This value is very important if you want to create a specific look for your level (be it rusty, rocky, chrome, etc.). The higher the value, the more light is reflected and the more the texture behaves like a mirror.

Here is an explanation of the various lighting features of the editor. Each option has two recommended settings or areas - one for designing and one for releasing.

Input fields:

Light map spacing	iterations	Ignore limit
Specifies how far apart the lightmap pixels are. The smaller the value, the better your shadows will be, but this also requires more memory and increases the lighting calculation time.	Specifies how many faces cast light. The higher this value, the better your lighting, but it also takes longer Lighting process is complete. If you are making a production level (i.e. one that you would release), set this value really high. To illustrate, some retail levels in D3 use 1 million iterations.	This is an important field. This is what the radiosity engine says: "If you shine light on a face and it still doesn't make much difference, save yourself the time." The smaller this value, the better for lighting, but again the time required for lighting increases. If you're just testing the lighting and want to get a feel for what it looks like, leave the value at 0.005.
Design: 20 (default value) Release: 5 to 8	Design: three times (number of lights present) Release: a large value, large enough to achieve full convergence	Design: 0.005 (default value) Release: 0.0005 or 0.00005 is even better

Checkboxes:

Combine Lightmap Faces	Volume Lights	Great detail
The radiosity engine will attempt to combine coplanar faces into a lightmap. This does two things: a) it reduces the amount of memory used for the lightmaps, and b) it makes the seams between the lightmaps disappear, an unwanted rasterizing effect. Combine Lightmap takes some time, hence	Determines the lighting in 'dead air'. This is the free space in the level; this will color your ship accordingly as you fly through the tunnels and halls. Without this option the ship will always have full brightness, even if it is in the shadows. This increases the time for the lighting process by approximately 33%.	Another important option. When it is off, light is only emitted from the center of the face; when it is on, it is emitted from the entire face area. This provides much, much more accurate lighting but increases the computing time for lighting by a large amount. Use it only for release.
Design: Off Release: On (default value)	Design: Off (default value) Release: On	Design: Off (default value) Release: On

Buttons:

Count lights in mine-simply count the light sources in the mine and offer to set the Iterations field to the value Lights*3 for you.

Light It!-Performs the lighting calculation. May take a while (minutes to hours) depending on the options you have set. When this is done, a dialog pops up saying that the lighting is complete. Don't worry if the number of iterations executed doesn't reach the value you entered; 100% convergence is all that matters. SAVE your level afterwards!

Back to Section A

Since today, in 2008, about eight years after this text was written, the computing power of PCs has increased quite a bit, it is advisable to adjust the values accordingly. Personally, I usually set 1 as the value for Lightmap Spacing.

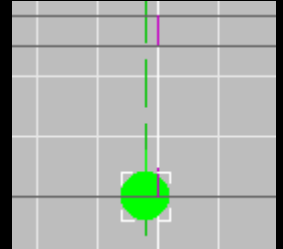


016 - Test the level <>

Robot

In this tutorial we will now prepare the level for a test flight in D3, but first - we have to fix something that could get annoying.

In the picture on the right you can see that the player start is quite close to the ground - actually in the ground! When you start the level, your view will be divided by the floor, and you will be between the room and the house of mirrors.



What we need to do is raise the Player Start to a height of 10 units above the ground.

Open `level1.d3l` and go into object mode (**Ctrl-G**), then click on the button **O** to bring up the Objects panel. In object mode you can select, move around, rotate and delete objects. Click the player start a few times to select it, then click in the front or side view and press once **8th** on the number pad to move it. This moved the player starting point up by 10 units. Problem solved.

Save and open MN3 Edit (*I use this in these tutorials because Quicktest is used in some Versions are often unreliable.*) One.mn3-File is the file that contains and organizes your levels, load screens, scripts, music, briefings and other files - all in one nice package, and then compressed. You can do that in there.d3l-Specify file/level set name, description, up to five auto-download URLs, keywords to set game modes and specify

whether or not the level can be played single or multiplayer.

Click **File...New**. Under Name, enter the name you previously gave the level to, your name, and - if you want - a description. Leave the auto-download URLs blank as the level is not yet suitable for multiplayer. It should look something like this:

Name	Level 1
Author	Mark 'Robo' Roberts
URL	
Keywords	
Description (40 chars or less)	A level from 'Robo's D3Edit Tutorials'
Allowed game modes	<input checked="" type="checkbox"/> Single player mission <input checked="" type="checkbox"/> Multiplayer mission

Next go to **Action...Add Level**. Click the browse button next to the level filename box, find your level and click OK. Click OK again to close this dialog box and proceed

File... Save As. Save your level to your D3 mission directory and you're done.

Start Descent 3, start a new game and find your level. Have fun!



Some people (including me) have found a bug in MN3Edit, but I don't think it affects everyone. It seems you can't save over a file twice, make sure the previous version is deleted or save with a different name if you have problems.



(016-tut7.zip)

[Back to Section A](#)

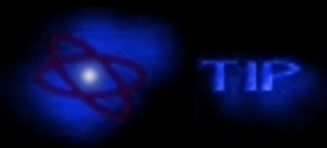
017 - Add new rooms <>

Robot

When you fly around in Descent 3, your graphics card draws all the faces, textures and lights as quickly as it can and sends them to your monitor. A lot of details can demand a lot from your system when processing them. To remedy this problem, D3 uses the BOA Vis Table (**big Ole Array**); I already mentioned this in 'Starting a Level'. When you calculate them in D3Edit, it writes into the level which rooms you can see from each point on the level, so when you play D3 only draws the rooms you can see. It checks very effectively.

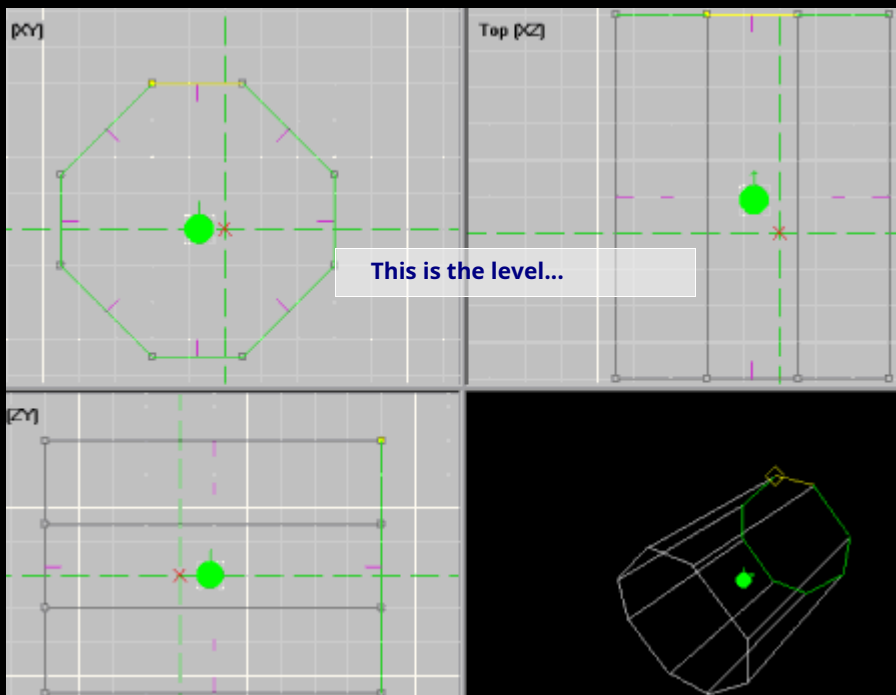
Anyway, there are two things you have to do to make this work. First you need to calculate and save the BOA Vis Table in D3Edit before testing (obviously). But secondly, you must have your level divided into rooms, if not - it's useless!

These spaces don't have to be tiny and you don't need many, but what you do need are enough spaces to keep your polygon count down and the frame rate up. I've found that a lot of rooms tend to look a little cluttered and tend to create more mistakes which are harder to fix, so how many rooms does it take? Well, I tend to make them reasonably sized and individual. I separate bends and straight tunnels, dogfighting rooms and powerup rooms - anything that has a specific purpose. You'll have to do energy centers, wind tunnels, and rooms with fog or sound separately anyway, because these options affect the entire room. But before we separate rooms, we first have to build them -

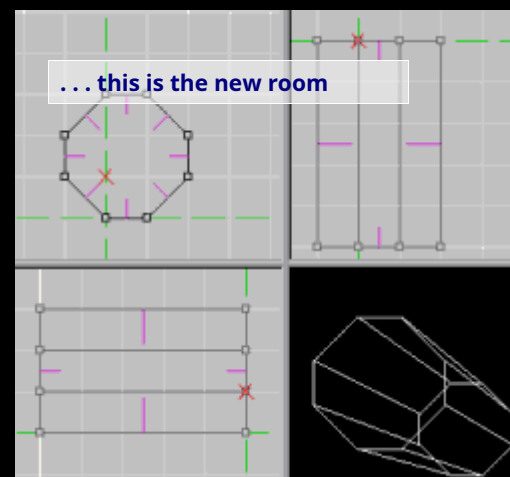


To cycle through all rooms in your level, use the **R**-button (In World View).

On the right I built a large octagonal room and converted it into a level... the player is already starting positioned. I have a similar one below octagonal room, much smaller than the one in the level, we'll hang it there. Open both files from the .zip and go to the room view of the room. Go into face mode (**Ctrl-F**) and mark all faces in this room. Then do one of the end faces

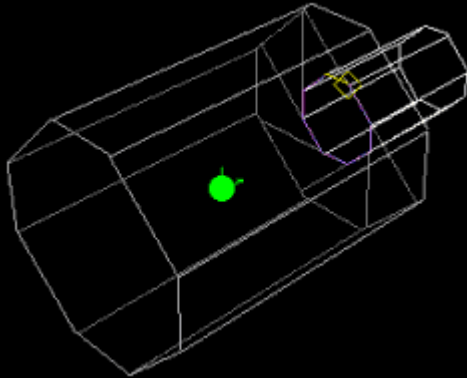


click or use the **F**current button (green). Now we are ready to add the room. Make sure you have all the faces in the room marked and an end face current, then go into the world view of the level. Go thereRoom...Place Room At



Current Room. Do you see the smaller, purple room? This tells you where you will insert the space into the level. If you don't want to include the space, go aheadRoom...Unplace Room, otherwise chooseRoom... Attach Room.
And this is what you should get now:

File: 017-addingrooms.zip



We have now successfully added a room to the level. If you press, you will notice that the lines of one of the rooms are drawn with a brighter white, indicating that it is current. D3Edit connects the current face of the level with the current face in the room.

In the picture our new room is current, and a little brighter. But what is the purple face doing there? This is a portal. Portals are the boundary faces between spaces and are invisible unless you script them to behave differently. You can turn them into force fields to block rooms, and you can turn them into triggers to turn something on or off - or trigger a scripted event.



In more complex situations, when you have multiple rooms that need to be attached in different places, you need to know how to attach the room.

D3Edit attaches the last room you viewed that has marked faces and a current face, so if you clicked into the wrong room view you have to go back and view the room you want to attach again before attaching. Confused concept... for some -

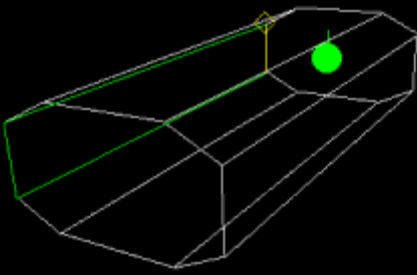
Do you remember the BOA Vis Table? Time to use them. Go toFile...Compute BOA Vis Table and click**Yes**.

You'll need this skill if you want to build a fully playable level with lots of rooms, so use it often and properly 😊

Back to Section A

018 - Objects & Player Starts <>

Robot



Here we will add a second player launch ourselves into the small space here on the left, as well as a single object.

The space is small but adequate for what we want to do. Use the space from the .zip, because I have preset a few things.

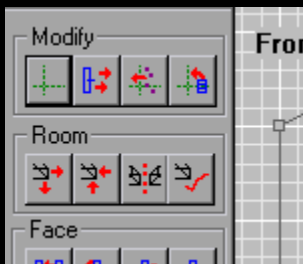
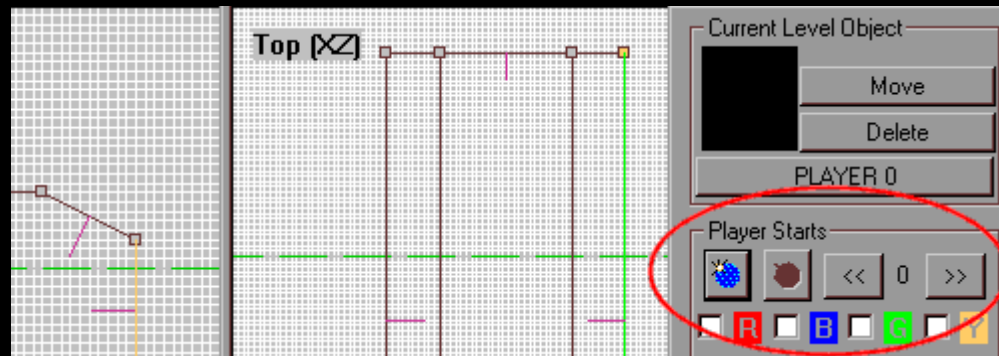
So, let's get started.

In this room I have a player start at the end as this is a

level is. Your room must be a level before you can insert player starting points; D3Edit wouldn't let you otherwise anyway.

Go to Current Room View

and open the Objects panel with the **O** button. The part of the panel we're interested in at the moment is this red one circled here:

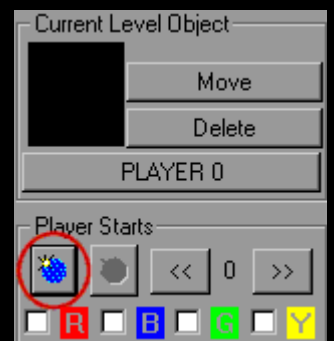


Open this Geometry-

Panel and click on the button at the top left: Set Reference Frame. Enter in the fields from left to right: **2042,-93** and **2090**. This causes a few lines to intersect near one end of the tunnel. These lines are the reference frame and are used to position player starts, objects, waypoints and nodes. You can see in the picture above just another example of a reference frame.

The place where the lines intersect is where the object will be placed, if you click in the views you could move it, so be careful. After the reference point is set, click on the button circled in the image on the right in the Objects panel.

You should now have a second player starting point in the room, which you can see in your views. If you press the '>>' button to the right of the other one, you have selected the player starting point (which is then written in the button above).



Note that none of the four checkboxes are ticked. This means the current

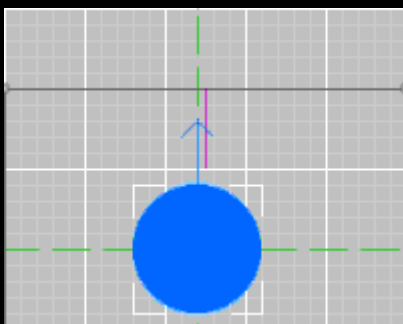
Player starting point can be used by everyone from all teams. If you

red If you tick it, it can only be used by the Red Team, same with **blue**. If you **both** If you tick, both teams can use it, but neither the green nor the yellow team. Clear?

Do you see the arrow sticking out of the object (left)? This is the direction your Pyro will face when you spawn from this point, meaning you will then face a wall. Do we want that? Nope. -

What we're going to do now is rotate the object back to the correct direction. Go to Object Mode (**Ctrl-G**) and set the grid size

50. I hope your reference point is still set, because the object will be rotated around it. Make sure the object is selected and press four times **7** or **9** in NumBlock. And here we are - in the right direction 😊

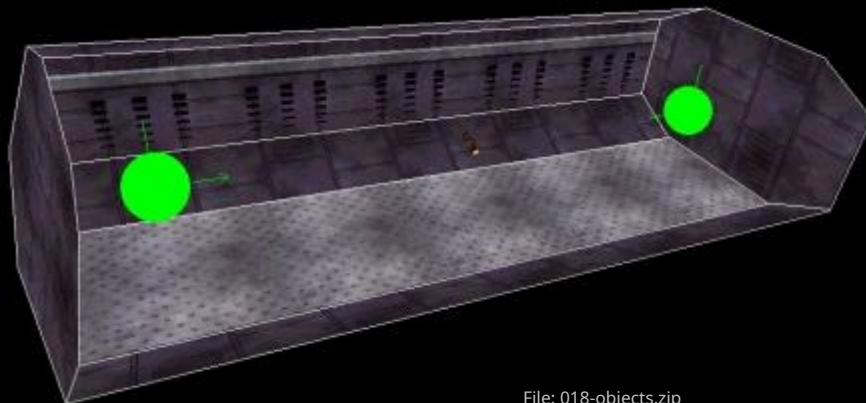
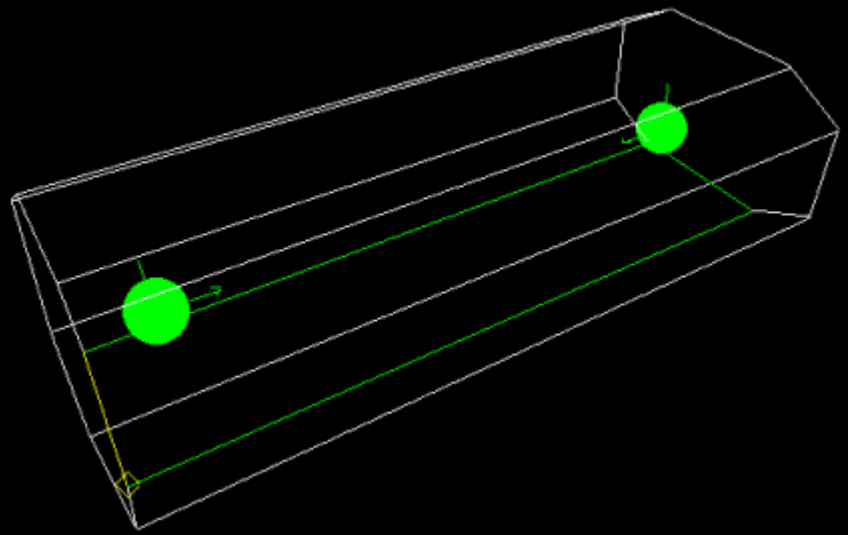


You should have something like that now, and be good **save**:

Now it's time to get to powerups and the like. Set up your reference point **2042,-93** and **2050**, where we will place our object. Click on Available Objects in the Objects panel, which is a dialog similar to that for textures. Tick powerups and wait for them to load.

It basically works the same like the textures panel; draw one individual Frag Missiles into the Custom Object box and close the selection list. Select the Frag, click Mine and then press Insert (keyboard or button).

Save...



File: 018-objects.zip

Back to Section A

Section B–Deepening in D3Edit

019	View shortcuts	This feature is probably indispensable	Ragil Ral	69
020	The geometry panel	All buttons of the geometry bar individually, with examples! explained.	Papacat	70
021	D3Edit & Quicktest furnish	Setting up the Quicktest tool.	WillyP	84
022	Editor settings and profiles	Setting up the Quicktest tool.	(LL)Dark	89
023	Standard mods	Explains how to use other game modes enables...	[DCG]Roadrunner)	94
024	Apply tools menu	Setup and configuration of the Tool menus.	(LL)Atan	97
025	Object info dialog	Explanation of the OI dialog.	(LL)Atan	98

[Toc](#)

019 - View shortcuts

Ragil Ral

Without them it would hardly be possible to work reasonably quickly in D3Edit. Luckily there is this green button that shows the hotkey list. So this list doesn't exist



Shortcuts

View Control

- cycle through windows = ctrl + tab
- pan = Shift + left mouse
- zoom = Shift + Ctrl + left mouse
- zoom = mouse wheel
- scroll left = a
- scroll right = d
- scrolling = arrow keys
- zoom in = w
- zoom out = s
- grid smaller = Page Down
- grid larger = Page Up
- grid 1,2,3,4,5,6,7
- center Room and reset View = RAlt + c
- center current face = Shift + c
- center Mine/Room = c

Mode select

- face: ctrl + f
- vertex: ctrl + r
- object: ctrl + g
- path: ctrl + h
- object or path node : LClick

Selecting next...

- vertex = v
- face = f
- object = o
- room = r
- portal = p
- path = h
- node = n
- select previous above items add shift + (key)

Mark items in vertex, face, and object modes

- mark selected = space
- unmark all = u
- invert markings = i
- mark all = m
- mark worldview = m

Marking multiple items

- Dragging a rectangle around vertices and faces will mark all that lie entirely within the rectangle.
- alt + drag to unmark

Modify items

- Moving marked items (vertices faces, objects, nodes in 2D views)
- move = shift + arrow key
- used numpad keys
- left = 4
- right = 6
- up = 8
- down = 2
- twist left = 1
- twist right = 3
- objects rotate left = 7
- objects rotate right = 9
- Rotate around current vertex = CTRL+NUMPAD1 / CTRL+NUMPAD3
- Rotate around current face center = CTRL+ALT+NUMPAD1 / CTRL+ALT+NUMPAD3

A big advantage of this hotkey list is that it doesn't disappear when you do something in the editor, which is the case with all other lists the application focus the case is.

R: -1/0 F: -1/0 P: -1/0 E: -1
Shows Shortcuts

020 - The geometry panel

Papacat, updated to v40

All buttons in the Geometry Bar are explained here with examples.

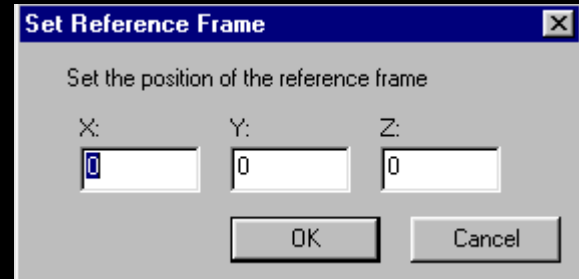
Modify:

Set reference frame



Opens a dialog box where the new position of the reference point should go.

I prefer **Ctrl-click** to move the reference point. If you also need the 3rd dimension set you have to go to another 2d view **Ctrl-click** en. Typically you're just dealing with a layer to add verts to, or an axis to rotate or bend.



extrude

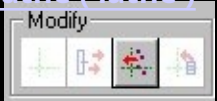


required: a current or one or more marked ones

Faces

Extrudes a face into a 3d object. Great for columns, tubes, platforms... etc. see also Hydra's tutorials on extruding (Section C, Advanced Building)

Lathe ('lathe')

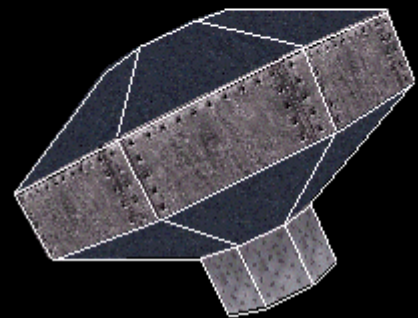
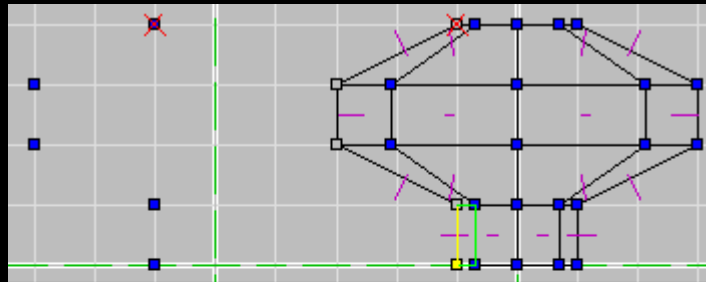


required: several marked vertices

Creates a 3d object by copying the selected verts in a circular pattern.

See also Hydras Tuts (the best I've seen on the subject [Section C, Advanced building])

I prefer to lath objects separately and then to fit in. And it's not just for objects; use segments for rounded ones Halls or to round corners.



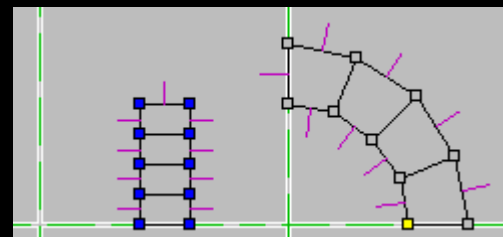
Bend



required: several marked vertices

Bends the selected verts over an axis. See Schplurgs and the other tuts on the topic.

It's good for bending tubes or columns, but has some side effects. After you bend something, you have to reset the UV's of the textures and realign them. See 'Bending without bend' and 'Just no to bend', an alternative to Bend tool. Another would be the lathe tool.



Unbend



If something goes wrong when bending, this button will help. It is the only undo that D3Edit knows



Room:

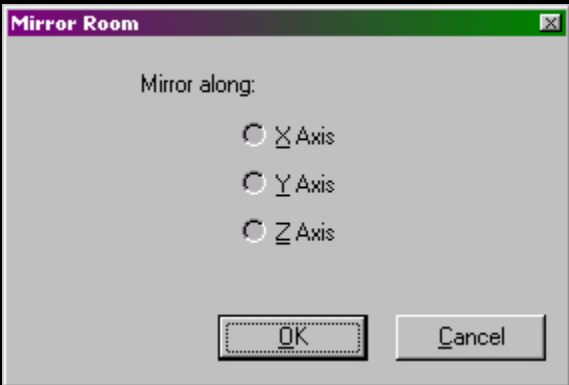
Expand/Contract Room

This allows you to inflate or shrink the entire room. Be careful with the portals!



Mirror Room

This allows you to mirror the entire room along one axis. In the dialogue you have to decide on an axis.



Verify Room

Calls Verify, the checking program.



Split room

In this way you divide up rooms, become even more so [No029 - Share spacetreated](#).



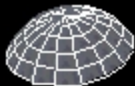
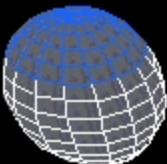
Create Room from Marked Faces

This turns a group of marked faces into a new one. or created. A new room view opens, with a room containing all the faces that were marked at the time of the call.

Left: Marked faces ->click button->results on the right.

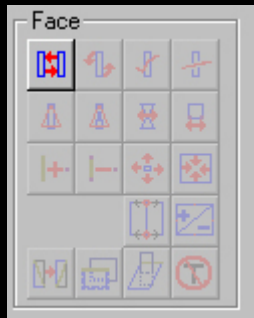


required: marked faces



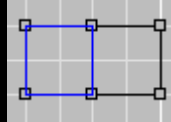
Face:

Combine Faces

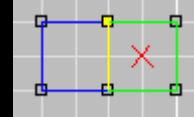


Use this to reduce your face count. Attention: You can/may only have one face marked when you use the button. Alternatively: click on a face next to the current **Ctrl-Shift-Click**.

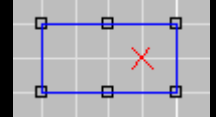
1st step
mark a face



2nd step
a close face
make current



3rd step
Run tool (=button click)

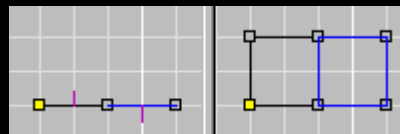


Flip Faces

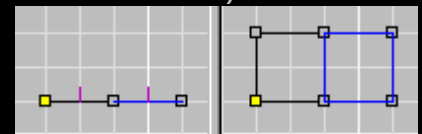


This flips all selected faces (=their normals). Keyboard shortcut is **N**.

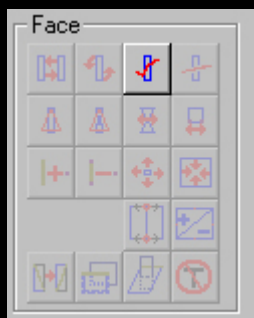
1st step
all faces that are being shot
have to mark



2nd step
Run tool (=click button
or **N**)



Face planar check



Checks marked faces for flatness and then offers to split them into triangles for you. The new faces get the default texture, so you have to rework them all.

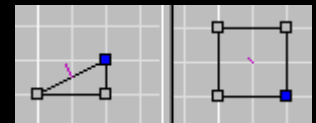
Good for fixing smaller faces, but you have no control over how the faces are split.

With four verts and a slight bend in the face, it won't do anything.

Five or more could make things messy (Example 3). Only part of the face would have to be split. The next two tools will give you control over splitting.

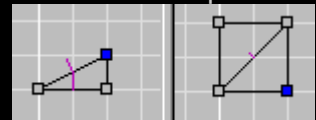
example 1

Non-Planar Face

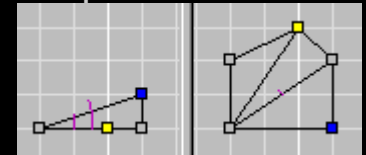


Example 2

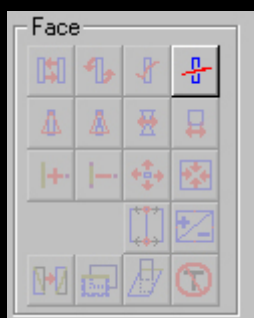
After the split



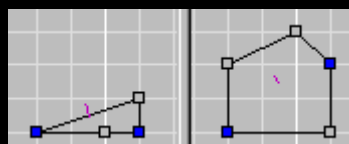
Example 3



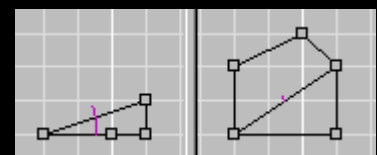
Split Current Face



1st step
Mark the verts that should be
connected by the new edge

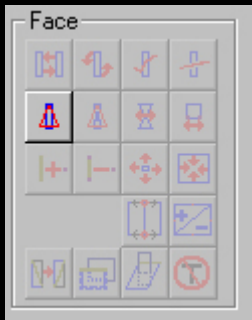


2nd step
click on the face to make it
current. Click button to get
this result



*needed: exactly two
marked verts in the same
face*

Triangulate Current Face

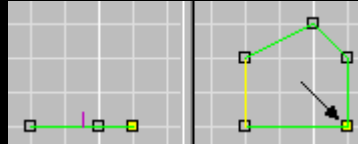


This divides a face into triangles, with the new edges coming from the current vert.

It works with both planar and non-planar faces.

1st step

Mark the verts through the new one
Edges should be connected



2nd step If necessary, set the correct Vert current (v); Clicking the button gives this:

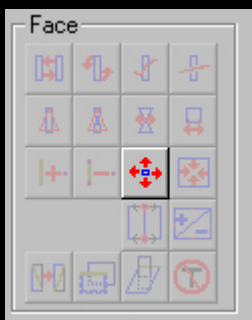


Triangulate Non-Planar Faces

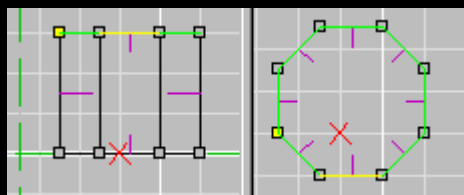


This splits all marked faces in the same way as 'Face Planar Check', but without checking and offering the change first. It finds all marked faces that are non-planar and triangulates them.

Expand Face



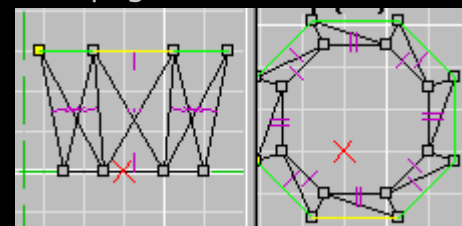
Enlarges the current face. Do one current and press the button. Each click enlarges it further while keeping the proportions the same. Quite good, but has one disadvantage, see example:



We'll start with this one simple column here. Do the top face current.

After clicking twice you will get this. All pages which are enlarged

Face concerns were split. One thinks because they were non-planar. Sometimes that's true, but not here.



Update for v40:

Meanwhile, Expand and Contract Face are the adjacent faces mostly not anymore split.



You have to go back and use Combine Faces to reconnect them to keep the face count down.

Contract Face



Contracts the current face while maintaining its proportions. It has the same tendency to split faces whether they need it or not, so be careful.

Expand and Contract also work with completely incorrect faces, see Error: Reference not found - Error: Reference not found.

Add Vertex

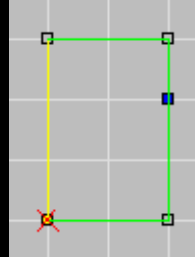


required: exactly one marked Vert and a current Face

Adds a highlighted Vert to the current Edge. It's good for fixing T-joints like in this example:

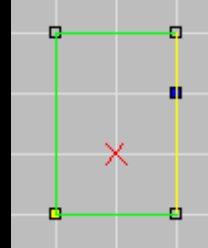
1st step

Highlight the vert that added to the face shall be.



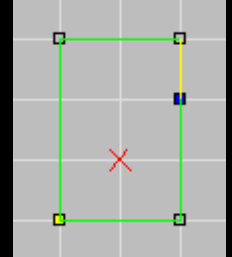
2nd step

Do the face current and press **F** to the Edge current to receive.



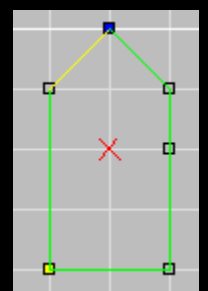
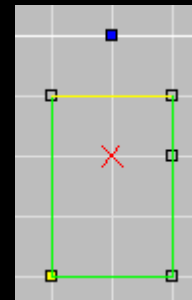
3rd step

Run tool.
Now there are five edges.



The vertex does not have to be on an edge of the face. You can also use this to give a face more shape.

But be careful. The tool does not check for non-planar or concavity.

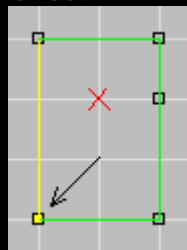


Remove Vert

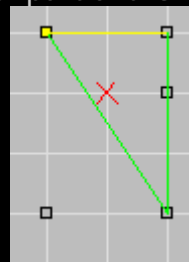


This removes the current Vert from the current Face.

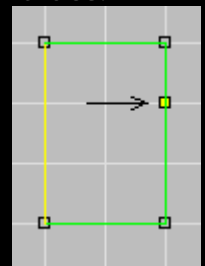
1) Do the face current and press **V** until the vertex current is to be removed.



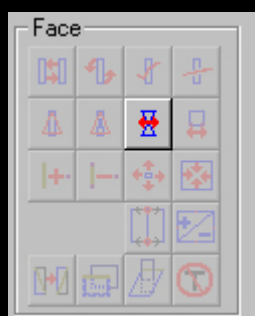
2) Run the tool and you will get something like below. Note that the vert is still there, just no longer part of the face.



3) It can also be used for T-joint fixing by removing the unused vert from the parent face.

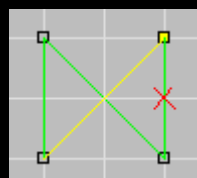


Twist Face

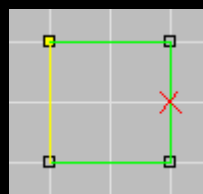


Fixes a twisted face that occurs when creating a face whose verts were not inserted in a sequential order. This usually happens when you create a face and append it to something that already exists. For faces with five or more verts you will need to use 'Swap Face Verts'.

Do it face current,



Click button then gives this:



!Note!

1. Check where the normal points afterwards.
2. Reset the textures to default UV's. Otherwise they will be distorted.
3. It doesn't check for planarity - you have to.
4. It doesn't check for concavity - you have to.

Swap Face Verts



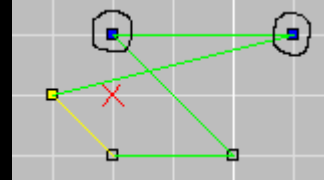
*needed: two marked
verts in the same edge*

To fix twisted faces with five or more vertices. It swaps the two marked verts with each other. I prefer to make faces with no more than four verts and then join them to avoid this. With five verts it's easy; with six or more it is possible to have more than one spinner. You are better off if you avoid this situation as much as possible.

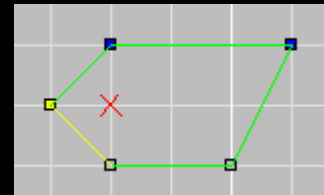
!Note!

1. Check where the normal points afterwards.
2. Reset the textures to default UV's. Otherwise they will be distorted.
3. It doesn't check for planarity, you have to do that.
4. It doesn't check for concavity, you have to do that too.

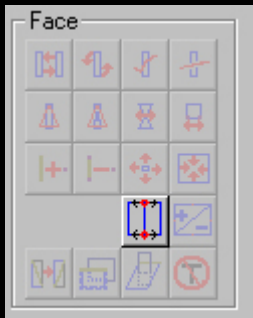
First mark the verts that should be swapped. Then do it the face current.



Press the Button for this result:



Split Marked Faces Verts



required: marked faces

Similar to the 'Split Current Vert' button, only here all vertices of a given selected face group are split.

A bit difficult to show with a screenshot 😊

Expand / Contract Marked

This function is still quite new and has its own peculiarities.

It is there to shrink or enlarge selected face groups. One click on the button allows factors to be entered directly.

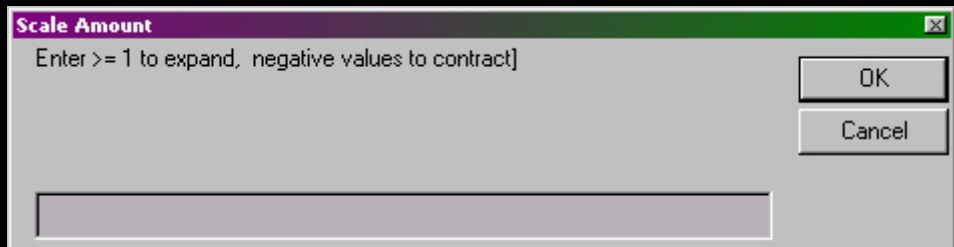


required: marked faces

However, a face bandage is torn apart; Remove Extra Verts must then be carried out.

Update:

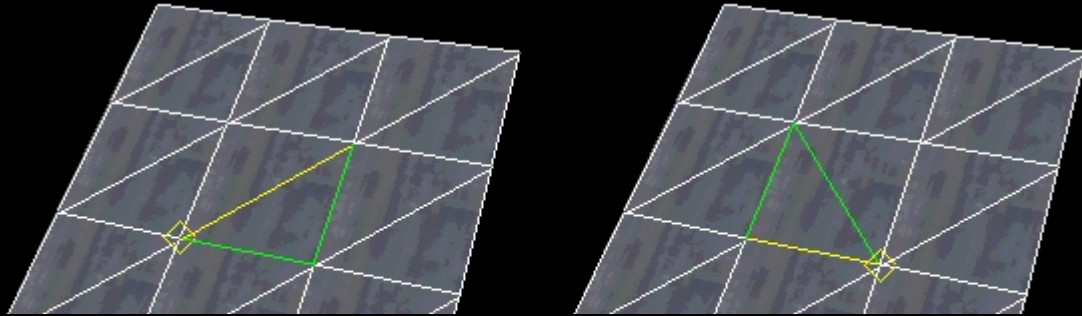
As of v40 the dialog looks a little different, the third option (All marked Faces Centerpoint) is no longer available.



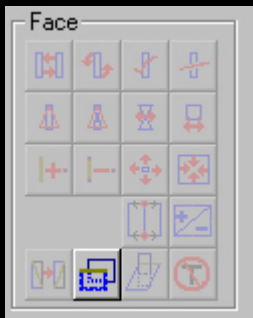
Turn Triangular Faces



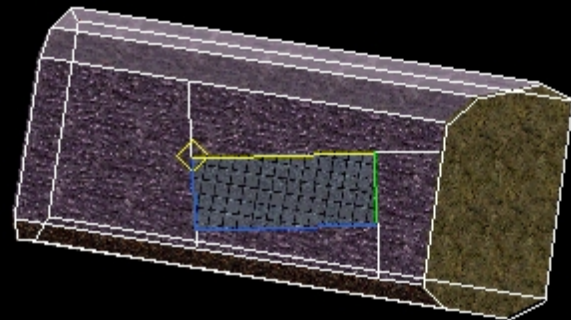
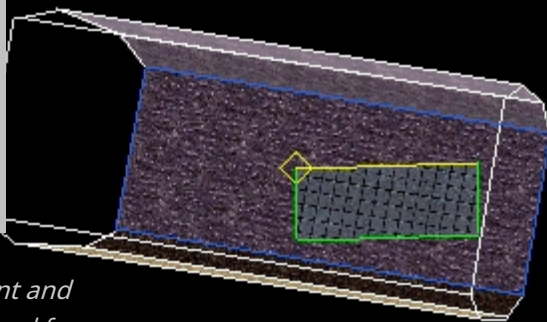
Changes the 'attack' of two triangular faces on their shared edge. This must be current.



Clip Faces



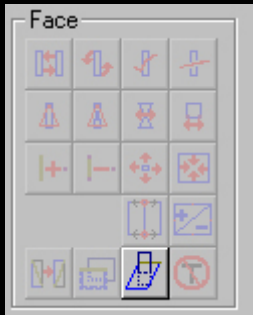
D
U
'K



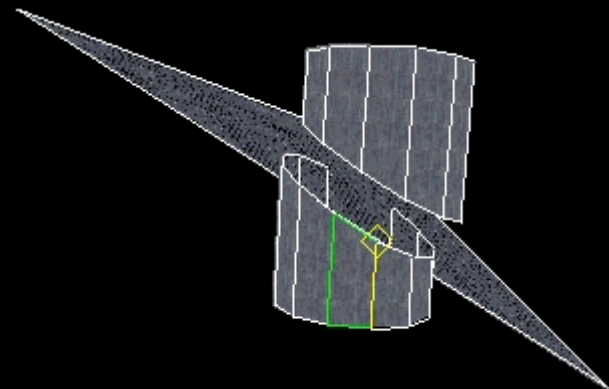
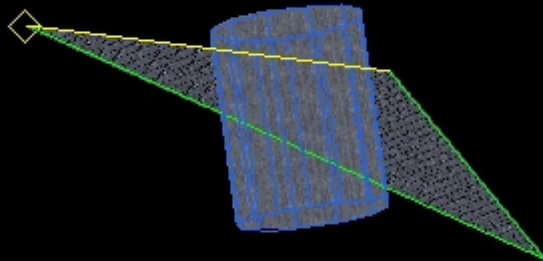
required: a current and exactly one marked face

What is important here is that the two faces must be plane-parallel.

Split Faces at Plane



One of the greatest functions ever - a face bandage is cut with another face:



required: at least one egg marked Face and a current

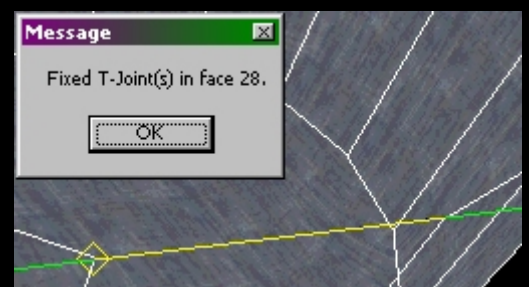
Marked faces are the 'cutting material', the current face takes on the role of the 'blade'

Kill T Joint



Eliminates T-joints in Faces.

Make face current, click button -> Verts are added to the current face until there are no more T's.



Edge:

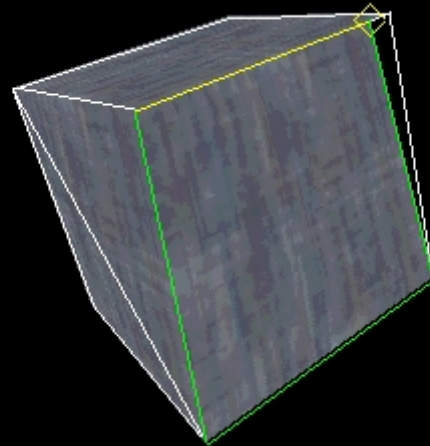
There are only three buttons in the Edge box, but they are described here anyway:

Expand / Contract current edge

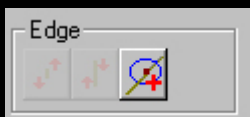


These buttons extend or shorten one Edge.

If faces become no, they split



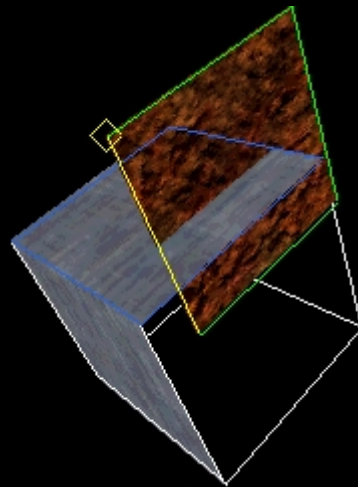
Intersect current edge with marked face



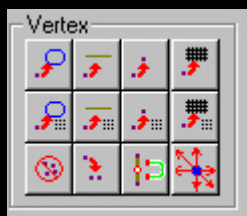
This function determines the intersection of a given current edge with a marked face and sets a vertex in the current edge.

On the left the initial situation, on the right the inserted vertex.

required: exactly one marked face and a current edge



Vertex:



In the Vertex box you will find, among other things, all the 'Snap' tools. These tools are very powerful. Master them and building complex structures will be a pleasure instead of a Sisyphean task. Use them for precision to get repaired faces and bad shells back on the grid.

The snap tools move the selected vert(s) to a defined target (edge, vertex, grid point or plane of a face). They don't attach them, so you still have to 'Remove Extra Verts' or 'Add To Current Edge' (whichever seems more appropriate).

The first four move this or that marked Verts to the target. The second four move that current Vert to the target and take all marked verts with you.

Snap Marked Vert To Plane Of Current Face



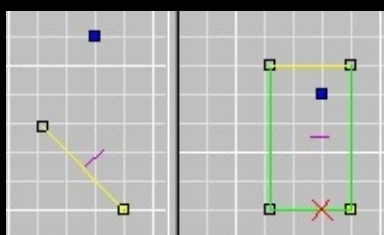
When you point to the tool it says 'Snap marked to Face', but always read the entire description in the status line. It snaps to the PLANE of the face. It can happen, but it doesn't have to; see examples.

It's good for accuracy, but also useful for getting non-planar faces flat.

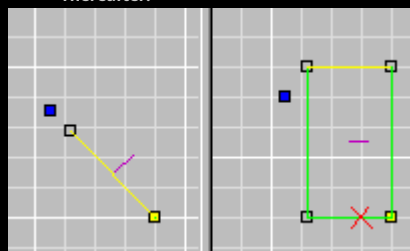
Select the vert(s) and make the face on whose level you want them current. Click button.

The Vert(s) move perpendicular to the current face.

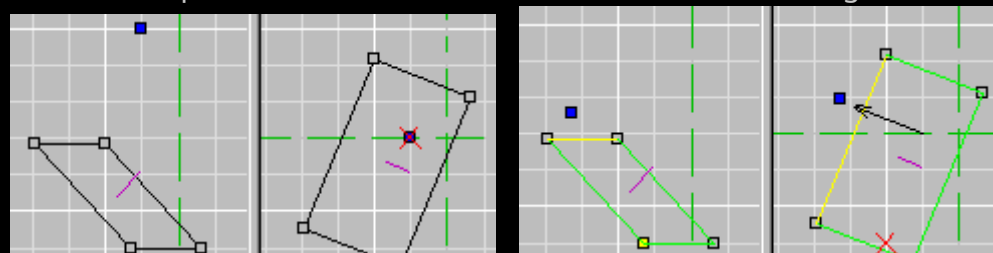
Previously:



Thereafter:



In this example I have tilted the face towards the vert. Through this



You can see that the grid has no influence on the movement of the verts. That's why I recommend that you always save before using this tool. The results are not always what you expect.

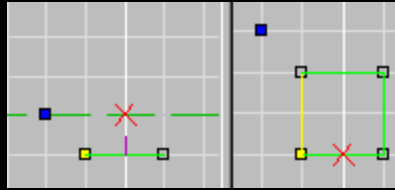
Snap Vert To Line Of Current Face



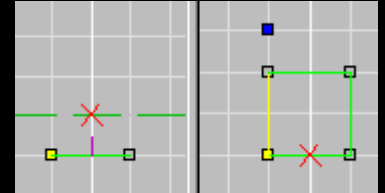
This moves the marked vert/s to the line of the current edge, not necessarily to the edge itself. Imagine the edge extends to infinity on both sides; the vert(s) move perpendicular to the line of the edge.

Mark the one moving vert/s.
Make the face current and press **E** to the desired one
Edge current to make. button click.

Previously

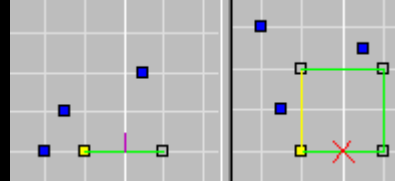


Afterward

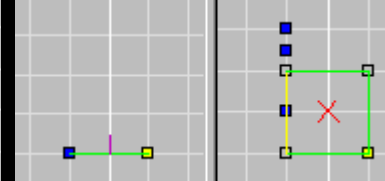


If you have several Verts moves, each takes his own own path, the not in relation to the others.

Previously



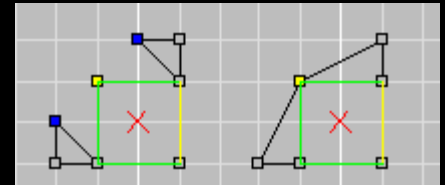
...



Snap Marked Vert(s) To Current Vert

Mark vertices that you want to move. Make the target face current. Use **V** to make the target vertex current. Click button.

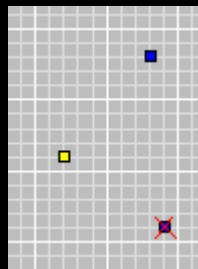
In the example you can see that this can be a great way to fix bad shells (Left Before, Right After).



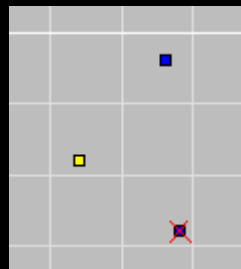
Snap Marked Vert To Grid

Moves the highlighted vert/s to the next one *active* grid point. Good for getting back on the grid, but I recommend only doing this once to keep control. See below.

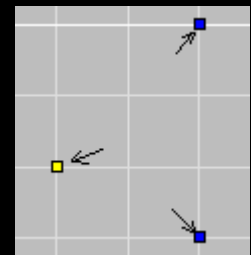
A few deployed Verts with the Grid size 1.



Grid size to 10 changed.



Apply tool. Here you can see how they move to the next grid point.



The next tools ask you which face, which edge or which vert you want to have as a target. Therefore, you need to understand the information contained in the status line:

R: 45/56 F: 6/42 P: 0/3 E: 0/4 V (idx): 0/4 (11/48) Grid: 1 2048, 250, 2430 0, 0, 0 Face mode Marked: V: 0 F: 0

D3Edit starts enumerations with 0; four faces are numbered from 0 to 3.

R: 45/56	Room #45 of 56 is current	Grid: 1	Grid size is 1
Q: 6/42	Face #6 of 42 is current	2058, 250,2430	Grid point where the mouse is currently
P: 0/3	Portal #0 of 3 is current	0,0,0	Current grid point (redX)
E: 0/4	Edge #0 of 4 (of the current face) is current	Face Mode	Mode display: Face, Vert, Object
V:(idx) 0/4(11/48)	Vert # 0 of the current face Vert # 11 of the current space is current	Marked V:0 F:0	How many verts and faces marked are.

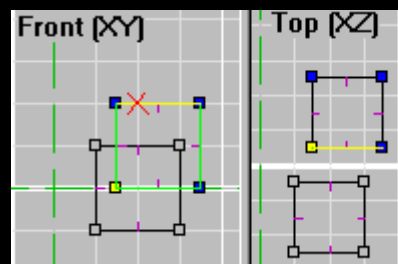
You should keep under observation; it's a good way to make sure you haven't accidentally marked something.

Snap Current Vert To Specified Face

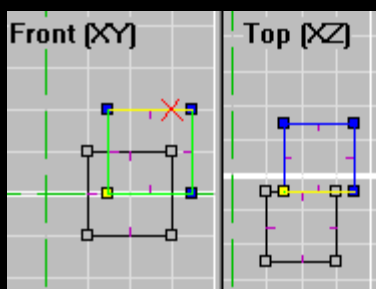
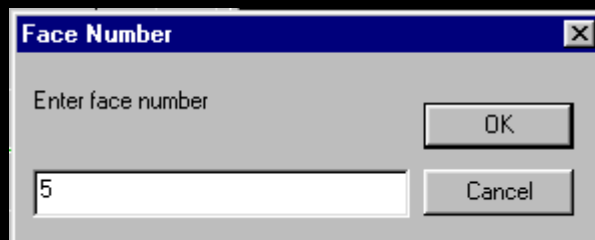
This 'snaps' the Current Vert to the specified face, taking all marked Verts with it. In this example I have two cubes in a room.



Before you mark verts and make a current, find out the number of the face you want to 'snap' to.



I mark all the verts of the cube I want to move. Make a face Current, whose vert you want to have current, it doesn't matter which one. Then use **v** to get the Vert current. Apply the tool. You will then be asked for the face number where you want to move the verts (remember, it moves to the face's layer, not the face itself). You are also asked about an edge, but that seems to be the case to have no influence (verified with v039 and v40).



Click **OK** and see what happens.

Snap Current Vert To Line Of A Specified Face



For this tool you need to know the number of the target edge and the number of the face that edge belongs to. The tool moves to the imaginary line (taking all the marked verts with it) of that edge, not necessarily to the edge itself. Remember that none of the verts will be connected automatically if the edge is hit; You have to do that yourself.

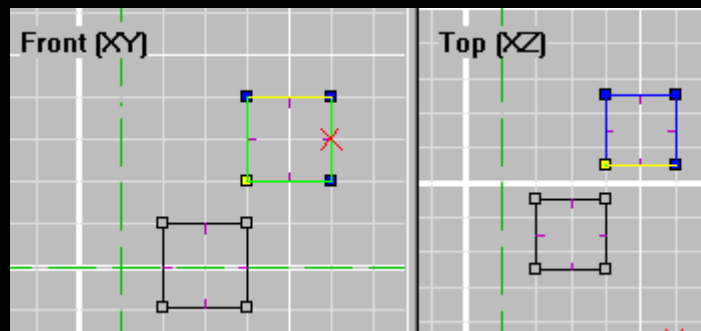
Snap Current Vert To Specified Vert



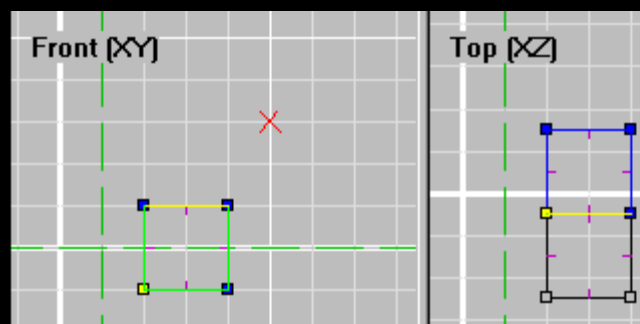
This is probably the tool you will use the most. To use it you have to set the number of the target vert relative to the Space (not relative to his face).

V (idx): 3/4 (8/16)

Mark all the verts you want to move.
Make one current.



I've circled the target vert here. This will line up the dice. Select the tool and tell it which vert you want to move the current vert to. Don't forget, they won't be put together before you Remove Extra Verts execute.



Snap Current Vert To Grid Point



Moves the Current Vert to the next grid point, taking all marked Verts with it. The goal is then the next oneactive(=visible) grid point of theactive2d view (top, front or side)

Remove Extra Verts



Should be self-explanatory...removes unnecessary vertices. When this function is executed, constructs are "welded" together, so to speak.

Split Current Vertex



This function does the exact opposite: a current vertex is split into several; This allows you to essentially 'separate' a face construct:

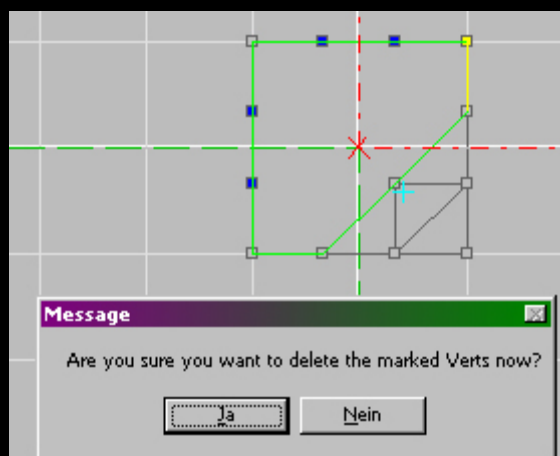
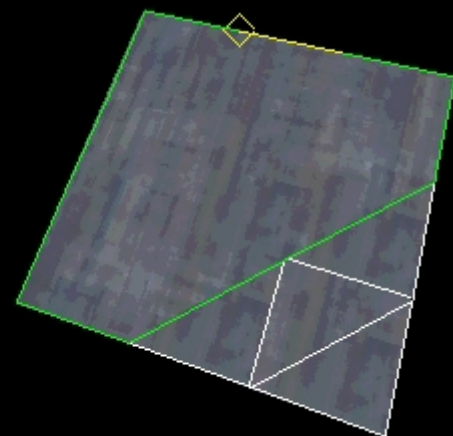
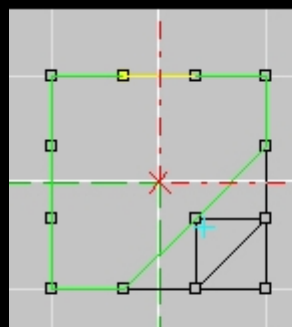


Left: before Split,
Right: after split, one
vert moved

Remove Unused Verts from Edges

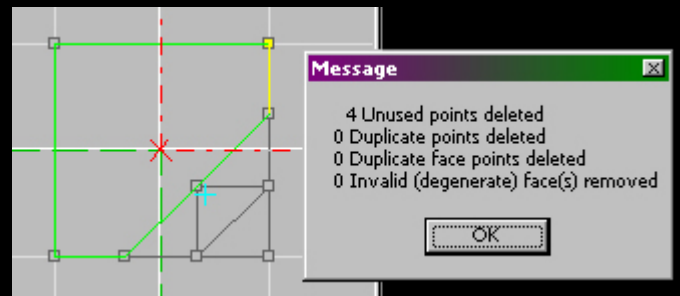



New since v40. This function removes unnecessary vertices from otherwise straight edges:
Faces were here united, with Extra Vertices remained in the edges.



If you execute the function, the unused verts are marked.

If you confirm this with Yes, the verts will be deleted and a short report will be displayed.



This feature ensures cleanliness in the level 

There has also been an innovation here since v40: the 'strength' of the anti-edge vert magnet can be determined in the status bar:



In the Status line stand on the right from Marked V:0 Q:0So

in addition Conc: which indicates the tolerance for concavity, and PTE: which is the parameter for the function. A **click** This opens a drop-down list field from which a value can be selected. This value determines how far away a vert must be from a given edge to be considered a 'corner'.

The range of values is from 0.000001 until 0.0025, default value is 0.000250. This is usually sufficient, but it can happen that continued editing of the geometry has created verts that are further away from the ideal edge; then it helps to increase the value here.

If that no longer helps, you have to do it manually again.

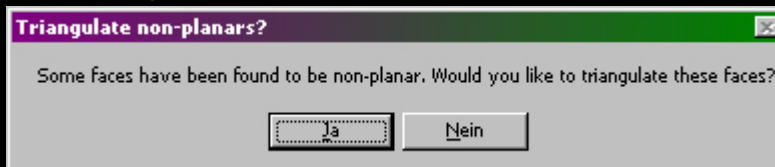
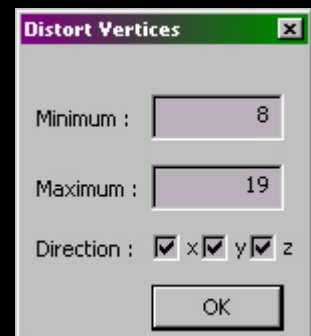


required: one or more marked verts

This button is also brand new: the Vertex Shaker.

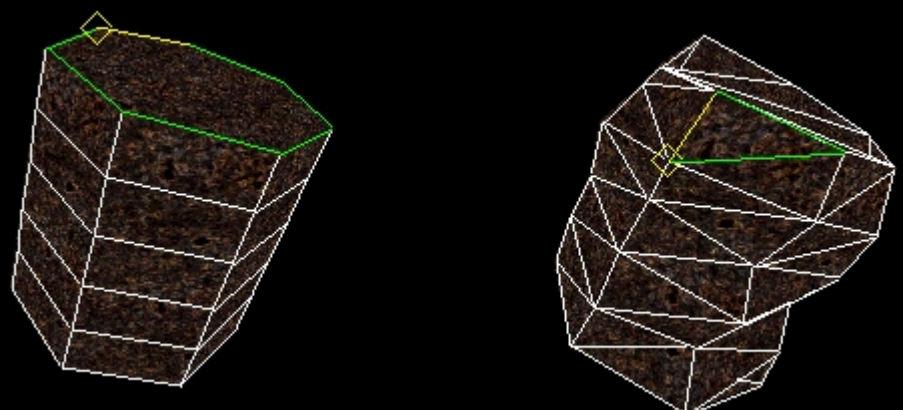
This function allows you to move selected verts by a random amount in any direction. This dialog pops up:

You can choose freely under Direction. If non-valid faces arise, you will be asked whether you want to split them. This makes the creation of



irregular forms
Child's play!

Example of application to a seven-sided cylinder; left before - right after:



Back to Section B

021 - Set up D3Edit & Quicktest

WillyP

revised

D3Edit

Many new features have been added since the first release of D3Edit added. I highly recommend that you download the latest version. The best place to get them is

www.descentforum.de/forum/, there in the 'English Embassy' section go to the topic 'Important: D3 Editing tools'. The current version is 1.1 AV40, avoid using outdated versions.

The older versions had some serious stability problems, which have largely been eliminated with the new version. And of course there are new, valuable additional features. Anyway, there is still no autosave and undo is limited to a few areas. The advice to save and back up often is still valid! Something can always happen

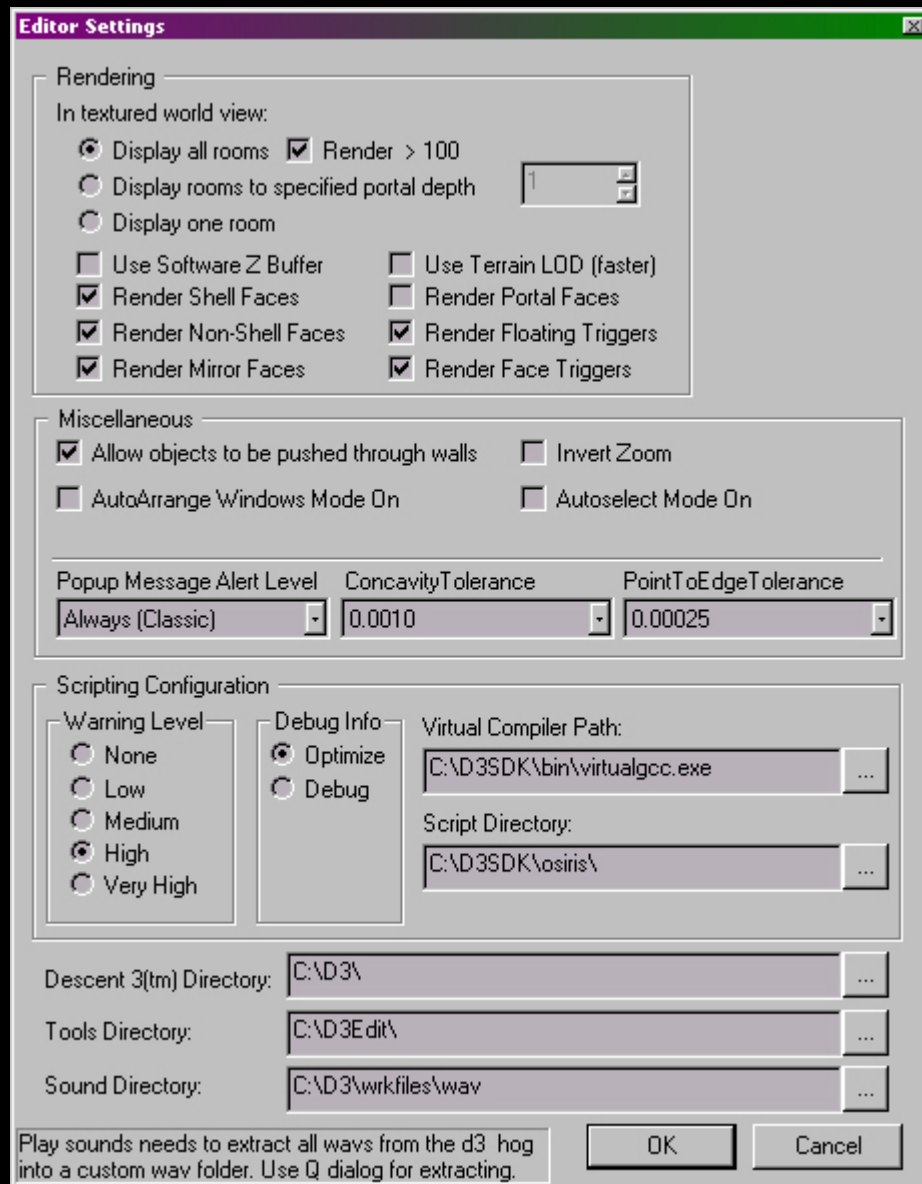
go wrong.

After you have unpacked it, there are only a few things you need to set up before you can start building levels. Under File go to Settings,

which is a dialog box called **Editor Settings**. For now we just need this Descent 3(tm) Directory put it where you installed your D3; is the default path C:\Games\Descent3\.

Optionally you can also Tools Directory in order to be able to work with external tools and files from D3Edit. Then go after Tools->Customize->Add and surf to your tool.exe, the .gam-File of your level, your favorite tutorial (for example this file), in short, every file type to which a corresponding application is assigned. You can even create a link page in html containing links to your favorite Descent pages! (You will 'All Files' have to select to see other files than just the executables.). Click on Open->OK. If you're not sure which file represents your tool, check the properties of its shortcut.

You can now start building levels, but... don't open your level yet, I suggest you first set up Quicktest, which replaces the mn3 packer and is much easier to use than this one.



1. the Quicktest Toolbox: Fill out all fields.

Now we can set up Quicktest, a good tool that comes with the latest versions of D3Edit and is launched from the editor. Open it with the button from the toolbar. First, create a new profile: Click **New** and give a name.

The file name of the mission is included in the MN3 name. The name that is visible in the mission selection of Descent3 is included in the mission. If you have additional files, such as textures, objects, etc. you can include them. Click **Insert >>** to load. You can view individual files, select files and contents of entire folders, more on that later.

2. Data: Load table files and mission hog

Click **Data** -> **Table File Manager**. The Base table file should be **Table.gam**. This allows you to use the original D3 textures, objects and other goodies in your level. If you have a custom **gam** for your mission

If you have one, download it as **Mission table file**. If

You *not* have one, make sure there isn't one there either. If you already have one, **mn3** for this level created, you can now load it: **Data** -> **Configure a Mission Hog** -> **Mission Hog File**:. If you *none yet* **missionary.hog** saved, then leave it empty for now. However, a simpler procedure has emerged, see 'Error: Reference not found'.

3. Open an existing level or create a new one.

Now you can open your level or build a new one. Save the level with a short file name. Only (preferably lowercase) letters and numbers, yes?

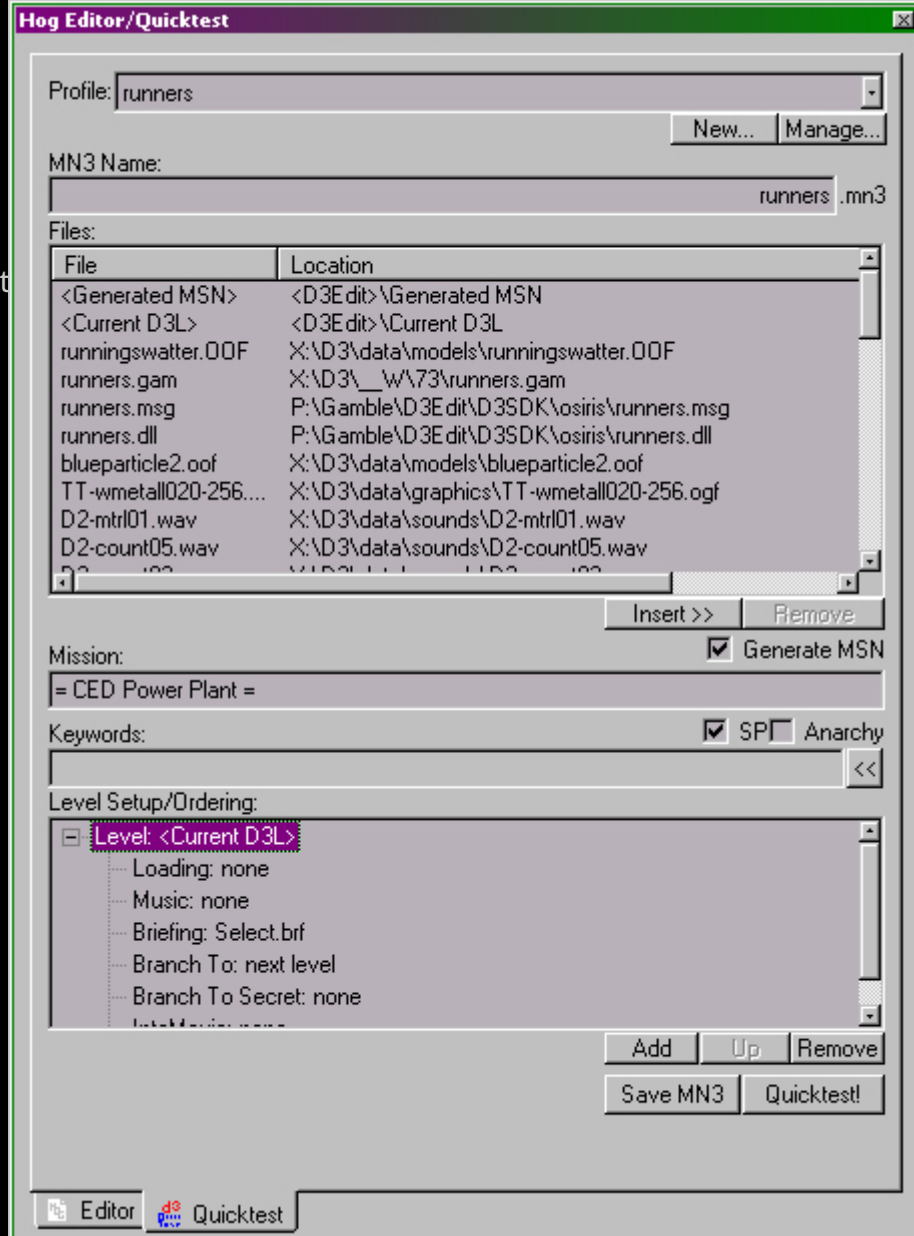
4. Create a mission .hog with Quicktest

Back in Quicktest make sure that **Generate MSN** and **SP** are ticked, then click **Anarchy** and then you can choose keywords from the list, unless the level is intended to be single player only. Save your level and click **Save MN3**.

You have now created a mission! We're almost done, just a few more steps.

5. Load your new one.mn3 as Mission Hog.

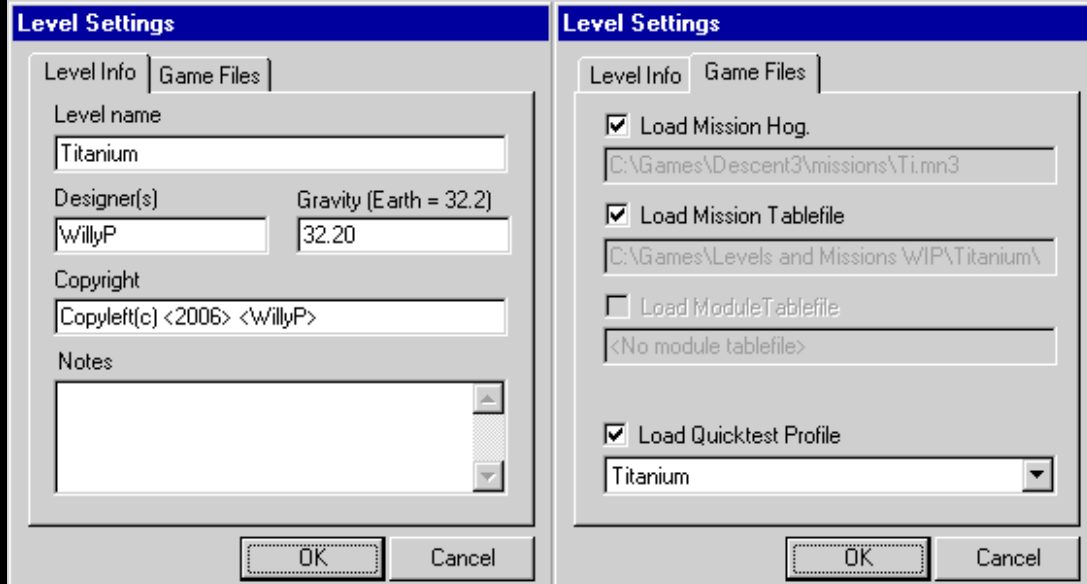
If it's a new mission, you have to do yours.mn3 previously as a mission.hog load. First, close your level. Go to **Data** -> **Configure a Mission Hog**. Press the button **Change**. Browse to your Descent mission directory and select the.mn3 that you just created and click **OK**. If you have an existing one.mn3 in step 2, skip this. Again, if the procedure from 'Error: Reference not found' is applied, this step is obsolete.



6. Open your level, open the level settings.

Now you can open your level again. Choose the World View. OpenLevel settings under Files and fill the fields under the Level Info-tab. In the Game files-Tab you can now Load Mission Hoghook

(also check Load mission table file if you have one), then check Load Quicktest Profiles and select your profile from the drop-down list (All in **Level Settings dialog**). This may be a lot of steps, but usually you only have to do it once.

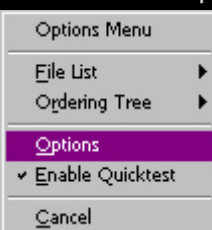


Load custom elements with Quicktest

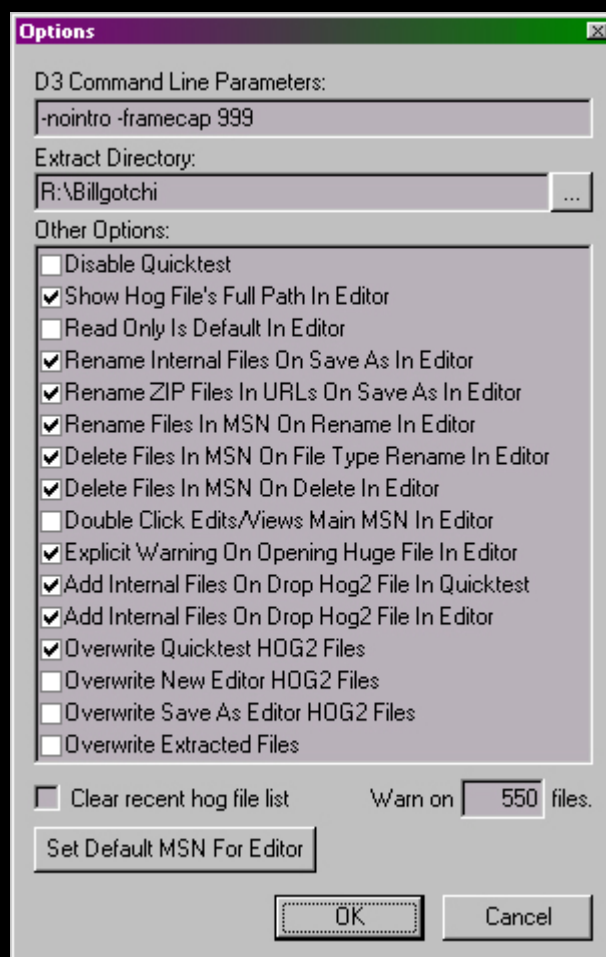
If you want custom (official translation of 'custom', really) If you want to have textures, music, sounds, robots, etc. in your level, you have to put them in the.mn3be included. The same applies to scripts, they also have to be in the level file. If in the mission.gamIf an object is entered that will be used in the mission, then it must be included in the level file, otherwise D3 will crash immediately when loading! Click the button **Insert >>**, You can then load the custom files individually or an entire folder. (Inserting a new file with the same name will overwrite the old one; but be careful, you won't get a warning about this!) Have you inserted your files and **Save MN3** clicked, it is already playable when you start D3 manually.

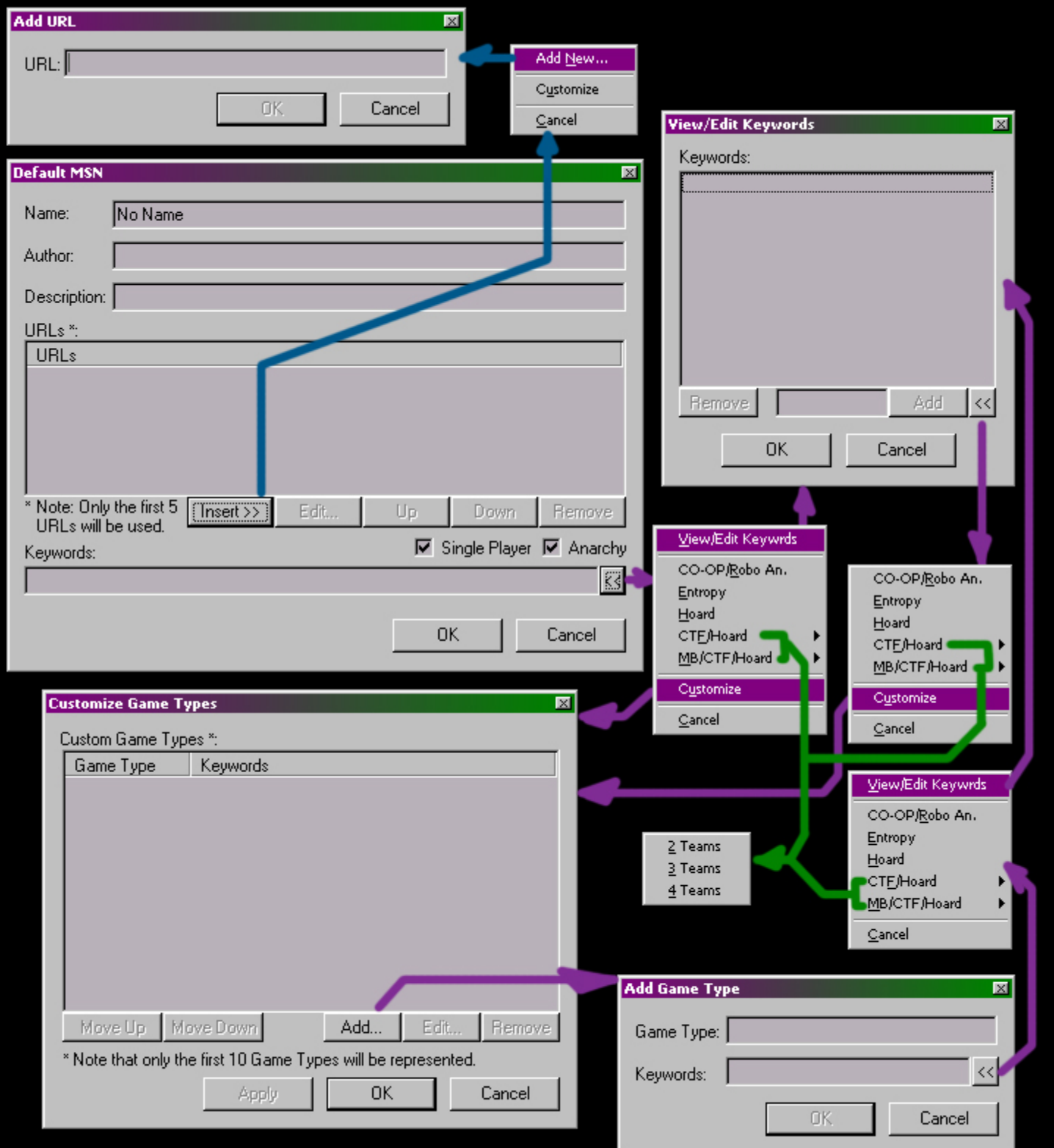
Test fly your level!

If you want to test the level, first make sure you have saved it. If you **Quick test!** If you want to use the button, you may need to set the options; To do this, right-click on an empty area of the quick test panel and select in the context menu that pops up (on the left) Options then you will get this dialog box (right):

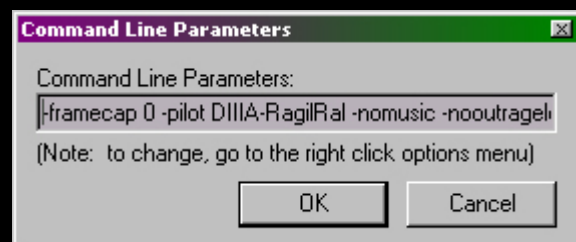


Here you can pass parameters to D3 and make a number of settings. Under the button **Set Default MSN For Editor** There is another dialog hidden that allows you to specify download URLs for your mission and also enables expanded treatment of keywords. This dialog is quite branched and you can call up every other one from almost all sub-dialogs, so there is an overview of how they are connected on the following page:





Once you have made all the necessary settings, you can use the **Quick test!**-Button test your level. Quicktest then updates your `mn3and` and shows you the command line parameters again (right). Then Descent3 starts, choose your level from the list and off you go. See also the appendix: [In-game testing aids](#).



Updating your custom files

From time to time, while building your levels, you will see the files you put in the .mn3 you want to make changes. For example, if you have another entry in the .gamfile or changed the animation of a door.

Your files do not need to be reinserted after you make changes to them. Just save yours. gamfile with the same name, or press Save in DALLAS to get an updated.dll to compile, etc...

If you have inserted them before, they will be updated! For textures and objects, save the .mn3 and close/open your level to see these entries in the corresponding list fields (*Note:*

You become yours

.tga-Images in one.oif texture file and one.orf-Room into one.oof-Object have to convert after you have changed it.) Here, too, the simplified procedure is of course 'Customs, again' applicable.

Back to Section B

022 - Backup editor settings and profiles

(LL)Dark

Unfortunately, D3Editor has the unpleasant property of saving all of its settings in the registry. Unfortunately, it's not just about the settings of the user interface and the few path details in the settings, but also about the quick test profiles created by the user. Depending on the level, these profiles can be very extensive. Larger single player projects can contain hundreds of files.

If Windows is reinstalled, a new registry will be created and all editor settings and profiles will be lost.

The only solution is to back up the relevant registry branches.

If you deliberately reinstall Windows, this can of course be done by hand, but who ever thinks about such trivialities and if the registry is defective, there is often nothing that can be saved anyway.

After consulting with Atan, we came to the conclusion that it would be too time-consuming to build a backup function into the editor because it is simply needed too rarely.

Other tasks are more important.

So I quickly put together a batch file that automatically carries out this task every time the editor is closed.

Of course, the prerequisite is that you always start the editor via this batch file. Admittedly, batch files (also called batch processing) are a relic from the DOS days and don't look nice due to the lack of a graphical interface, but they can do a lot and are relatively easy to create even for users who are not familiar with programming.

So ideal for our purpose 😊

Step by step:

1. Open the folder in which your `sd3edita.exe` lies.
2. Click with the *right* Right-click in the window and select under in the context menu `New-> Text document`
A new file with the name will be created `New text document.txt`
3. Rename this file in D3 Editor `START.bat` OR `D3-Editor-START.cmd` around. You answer the dialog that appears *Yes*.
4. Now again with that *right* Click on the file and in the context menu `Edit` choose. A text input window will then open.
5. Enter the following text here:

```
echo off
if exist %EditorSettings-Save.reg goto 1
echo No saved editor settings found d3editAv.exe

regedit /e EditorSettings-Save.reg HKEY_Current_USER\SOFTWARE\Outrage echo Editor settings
have been saved in the file EditorSettings-Save.reg exit

:1
regedit /s EditorSettings-Save.reg
echo saved editor settings found and entered d3editAv.exe in the registry

regedit /e EditorSettings-Save.reg HKEY_Current_USER\SOFTWARE\Outrage echo Editor settings
have been saved in the file EditorSettings-Save.reg exit
```


6. Now click on **File->Exit** and confirms the following query with **Yes**.
7. Now you have to make sure that you always start the editor with this batch file.
To do this, create a shortcut to the file on the desktop.
Click with the *right* Mouse click on the link and select in the context dialog
Characteristics.
In the index card *shortcut* assumes her **Run->Minimized**.
In the same tab you can also set a different symbol for the shortcut. (e.g.
that in the **d3edita.exe** supplied icon for the editor)
8. You should delete your original shortcut for the editor so that there are no
mix-ups. Only if you always start the editor via this batch file will your current
settings be saved!

That's it.

From now on, only start the editor via this shortcut.

You will notice that after starting the editor via the batch file, a second program entry
can now be seen in the taskbar.

If you click on it you will see a DOS window with a status message.

This program is the batch file and you are NOT allowed to quit it!!!

It closes itself when you close the editor and saves your editor-specific registry
entries in the file **beforehandEditorSettings-Save.reg**. You can find this in the same
directory as the batch file.

Every time the batch file is started, the registry entries saved in this file are entered back into
the registry.

After reinstalling Windows, all you need to do is start the editor via the batch file
and your program will work as always.

Explanation of commands:

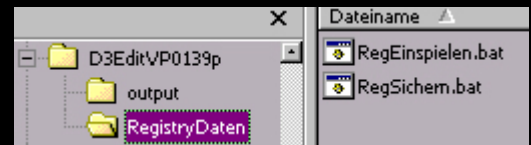
echo off	By default no messages are output
if exist %EditorSettings-Save.reg goto 1	if the fileEditorSettings-Save.regin the If the program path of the batch file is found, jump to jump label 1, if not, continue here. (This command is only used to ensure that the program runs smoothly the very first time the batch file is started or if the fileEditorSettings-Save.reg is not available for other reasons)
echo No saved editor settings found	This message is displayed in the DOS window
d3editAv.exe	Starting the editor
regedit /e EditorSettings-Save.reg HKEY_Current_USER\SOFTWARE\Outrage	When the editor has ended, the entries of this registry branch are in the file EditorSettings-Save.regSaved.
echo Editor settings have been saved in the EditorSettings-Save.reg file	This message is displayed in the DOS window.
exit	Ending the batch program
:1	Jump mark 1
regedit /s EditorSettings-Save.reg	the entries from the fileEditorSettings-Save.regare written back to the registry.
echo saved editor settings found and entered into the registry	This message is displayed in the DOS window
d3editAv.exe	Starting the editor
regedit /e EditorSettings-Save.reg HKEY_Current_USER\SOFTWARE\Outrage	When the editor has ended, the entries of this registry branch are in the file EditorSettings-Save.regSaved.
echo Editor settings have been saved in the EditorSettings-Save.reg file	This message is displayed in the DOS window.
exit	Ending the batch program

Of course, it is also possible to store the batch file somewhere other than the editor directory and the registry backup can also be saved in another directory or in several places.

Then it is necessary to provide all files with the appropriate path information.

But it is also a bit more convenient; If you save the D3Edit installation as a whole, the whole thing can be regulated as needed. To do this, the process used by Dark is split into two parts and the resulting batch file is integrated into the D3Edit tools menu. This is of course more work, but a little more elegant.
But one by one...

First, create a folder in the D3Edit directory and name it so that you will later know what it is for. The batch file(s) are set there.



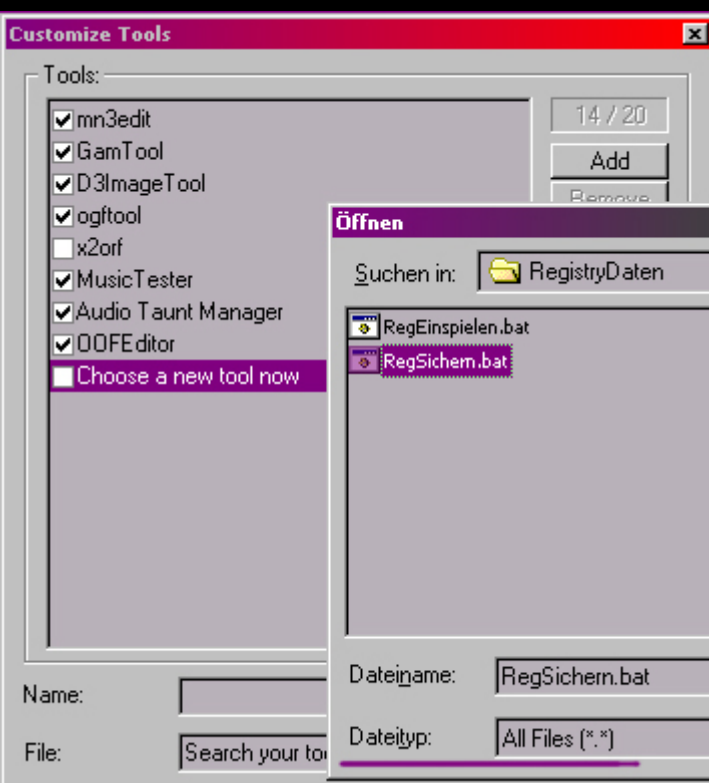
Batch file to backup settings

First create an empty batch file as described by Dark and put the following code in it:

```
@echo off
regedit /e RegistryData\EditorSettings-Save.reg HKEY_Current_USER\SOFTWARE\Outrage echo Editor
settings have been saved in the file EditorSettings-Save.reg pause nul
```

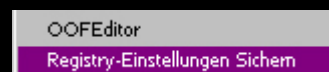
The `pause` serves only to pause the batch processing and wait for you to press a key before it exits; If you want, you can leave it out, then a little window will just flash briefly - in this case you will too `echo` from line three do not see. The other commands have been explained previously. `Aexit` is not necessary here, the batch ends itself.

Integrate into the tools menu



This batch file should now be accessible directly in D3Edit: Follow along **Tools**->**Customize...**, click there **Add**. In the dialog that opens, set the file type: **All Files (*.*)** otherwise you won't see the batch files. Select your backup batch if you want **Surname**:

Enter a text that will then be displayed in the tools menu.



If you were paying attention, you will have noticed that in the second line of the batch file the path points to the folder that you previously created. This has the following reason: when something is called from the tools menu in D3Edit, the directory where the editor is located is the working directory. So you would just

```
regedit /e EditorSettings-Save.reg
HKEY_Current_USER\SOFTWARE\Outrage
```

write, the backup would be stored where D3Edit is located. This also means that you can specify a path here where you want the backup to go; approximately `%userprofile%\desktop\<filename>` if you want them slapped on the desktop.

Restoring the backup file

First of all, you obviously have to put the D3Edit directory back on your newly installed system, unless you had it on a different partition to begin with. For this reason, it is recommended to store the batch files within the D3Edit directory, because this way they will be automatically taken with you when you re-import after a new Sys installation, as will the backup file that was created using them.

Now there are two options. The easiest way is to navigate to the directory where the registry backup file is located, right-click on it and select 'Merge'. You will then be asked whether you really want that, to which the answer is 'yes'; If the data has been written correctly, a corresponding message will appear - this means that D3Edit is fully ready for use again.

Of course, you could also write a batch file to restore the registry data:

```
@echo off
regedit /s EditorSettings-Save.reg
echo saved editor settings found and entered into the registry. echo D3Edit is now started,
continue with any key...
pause > zero
.. \d3editAv40.exe
```

Here you notice again that the paths are different: none in the second line, but one when calling the editor (last line), namely the indication to go one directory higher (..\). This has the same reason why a path must be specified in line two of the backup batch: namely the working directory used. This is always the directory from where the program or batch was started. Since the batch is in the same directory as the backup file, you only need the file name and no path to restore it, but you do need it to start D3Edit - this is one directory level higher than the batch.

Aexit is not necessary here because the batch terminates itself after executing the last statement. And whoever noticed it, that `@before` the introductory `oneecho off` is only used to suppress the display, since with the `echo off` Yes, all subsequent messages are suppressed, but not this first instruction.

However, this solution requires that you consciously create backup copies; If you want it automated - like Dark solved it - but with a little more interaction, use the following batch:

```
@echo off
regedit /e EditorSettings-Save.reg HKEY_Current_USER\SOFTWARE\Outrage
echo Editor settings have been saved in the EditorSettings-Save.reg file. Any key to start D3Edit...

pause > zero
.. \d3editAv40.exe
```

Then set up this batch as a shortcut to start the editor as Dark described in point 7.

Back to Section B

023 - Multiplayer Default Modes

[DCG]Roadrunner)

I would like the following game mod settings here, using the example **Movieworld** ,explain for the level editor.

Hoard
CTF
Entropy

Basics:

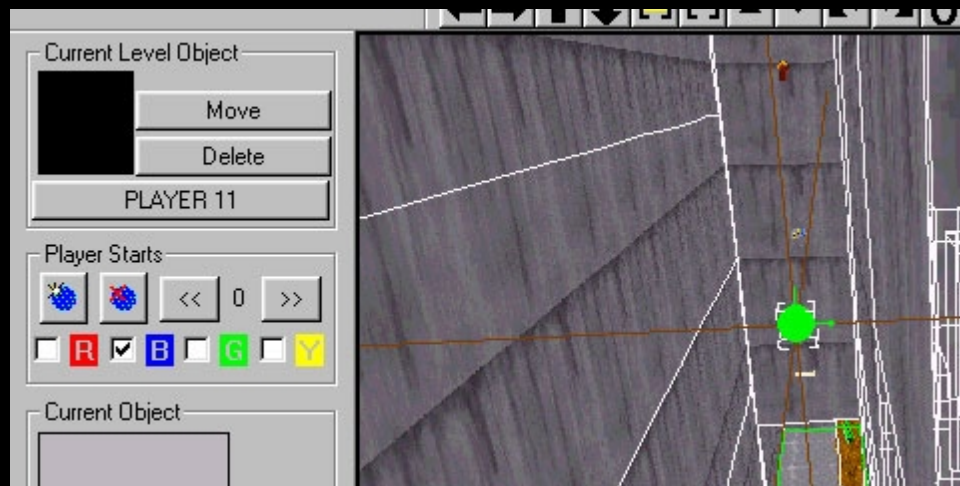
Everyone has a little different ideas about it as far as the division of the rooms in these types of games is concerned. Therefore, the following is my opinion on the setting of the rooms. Movieworld is certainly not a good example of Hoard, but since the settings for Hoard and CTF are similar, the mod is there. The most important thing of all is the settings window 'Window - Room Properties'. All necessary settings are here performed.

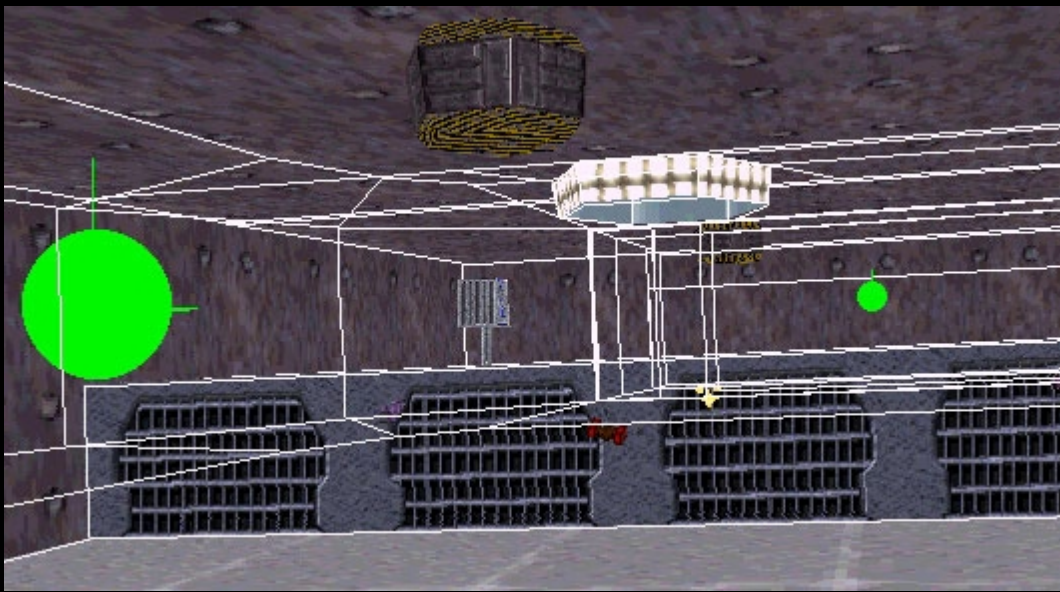


Now select a room that will be the dropping point for the balls. Name it like in picture 2 for example '**Kino2 Blue**'. In Movieworld, these drop points (there are two) are behind the screens. Once you have chosen the room, select in '**Room Properties**' Red goal. This makes this room a goal room. If you don't want to define all areas in your goal room as goals, mark (**don't mark face!**) one of the surfaces and select in '**Room Properties**', '**Current Face is a goal Face**'.

It's the same with CTF, except you need two, one red gate and a blue gate. This wasn't a problem at Movieworld as there are two cinemas (intentional). Behind the blue cinema is the blue gate and behind the red one is the red one. In the game it looks like this: you have to bring the captured flag across the level. Which shouldn't be so easy when there are a lot of opponents. So it's no use if two goal areas are next to each other. Oh yes, the textures for these mods are provided by Descent. You can texture as normal so that your rooms look good even in Anarchy. When selecting the player starting points, care should be taken to ensure that, for example, a player from the red team cannot start directly at the blue flag. You can achieve this by assigning a team color to the starting points.

This starting point at the bottom of the hallway is assigned to the blue team (picture on the right). Of course there are also neutral rooms like in the next picture. In Movieworld the ventilation room.



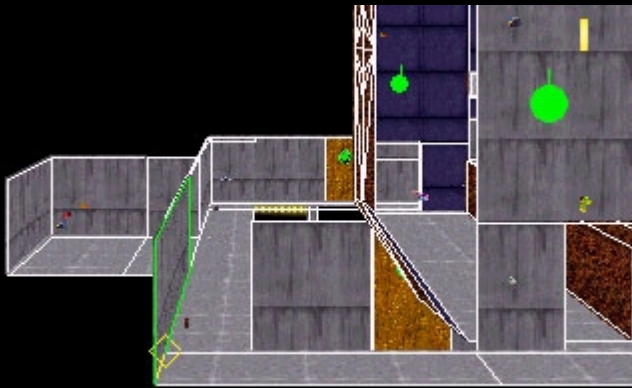


Here you simply choose both colors to give every team the opportunity to give to start here.

Now to Entropy!
The mod that is played less often, but can be a lot of fun if the level is good. There are three types of rooms in Entropy.

Laboratories, repair rooms and per team

Energy tanker. All of them will also be overWindow/Room Propertiesforgive. In Movieworld I assigned 3 laboratories, 1 energy room and 1 repair room per team. Of course, you can also distribute each type of room multiple times. The viruses that need to be collected will later be in the laboratories. The other rooms speak for themselves. In Movieworld, as in CTF and Hoard Goalrooms, the room behind the screen is one laboratory per team.

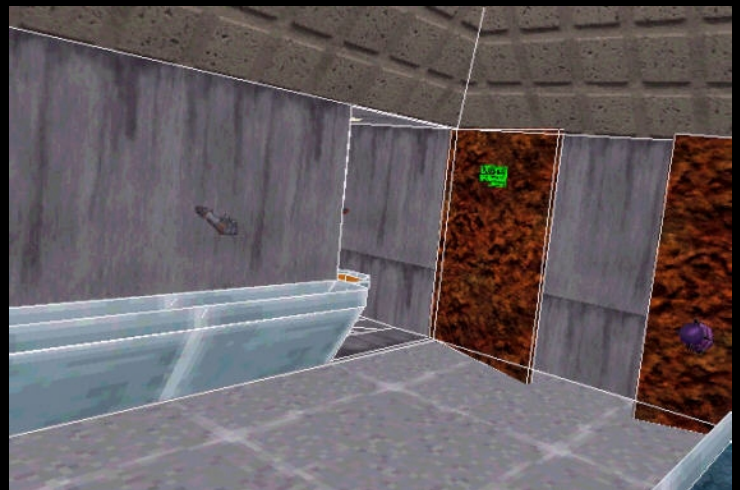


The repair aisle is the aisle directly in front of it. And the energy tank room is to the right of the repair corridor. The next

Laboratory is the toilet and the demonstration room.

The setting of these rooms looks like this:

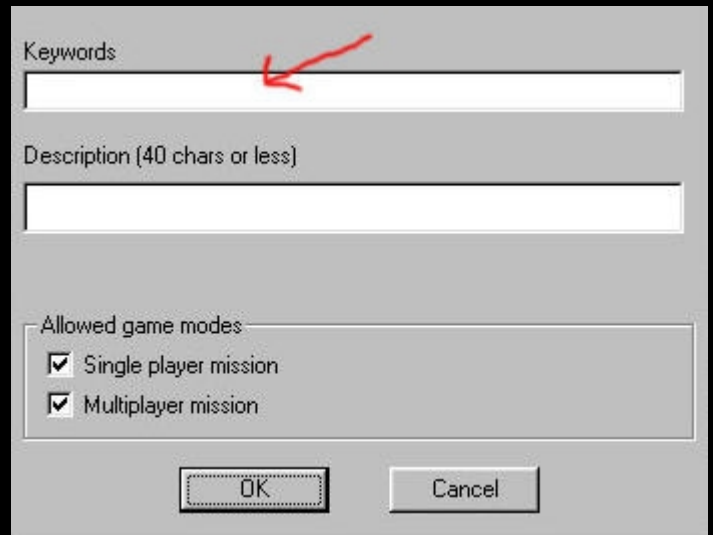
- S1=Laboratory Red
- S2=Energy tanker red
- S3=Repair room red
- S4=Laboratory Blue
- S5=Energy tanker blue
- S6=Repair room blue



In Entropy the setting is the player starting points**VERY**important. It cannot be that a red player starts in a laboratory from the blue team.



If you now pack your level with the MN3 packer, you still have to enter the following keys (right):



Keywords

Description (40 chars or less)

Allowed game modes

- ☒ Single player mission
- ☒ Multiplayer mission

OK Cancel

For a Hoard (level with a goal) - GOALS1
For CTF - GOALS2, GOALPER TEAM
For Entropy - ENTROPY

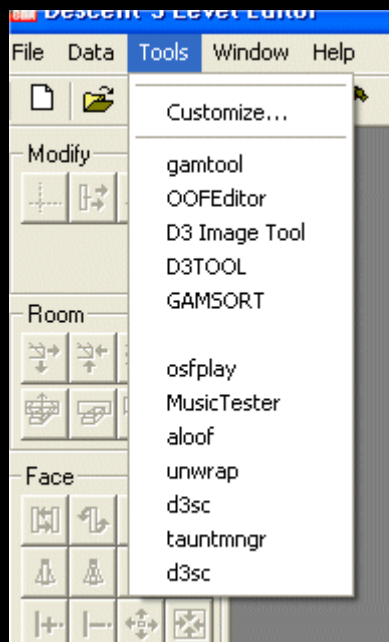
For a level like Movieworld it would look like this:
GOALS2,GOALPERTEAM,ENTROPY

So that's it actually.
As MUDDY always writes so nicely 'A T-Joint' free level building.....
[DCG]Roadrunner

Back to Section B

024 - Apply Tools menu

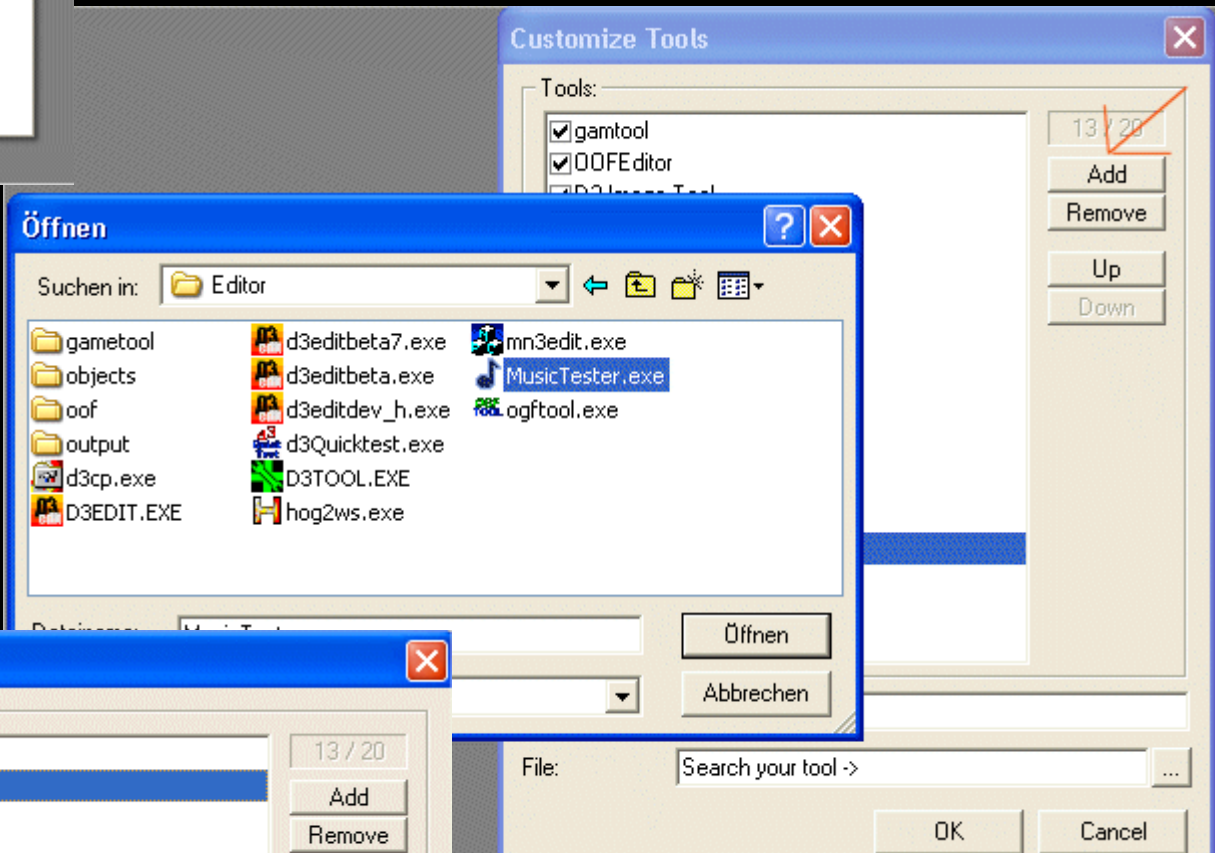
(LL)Atan



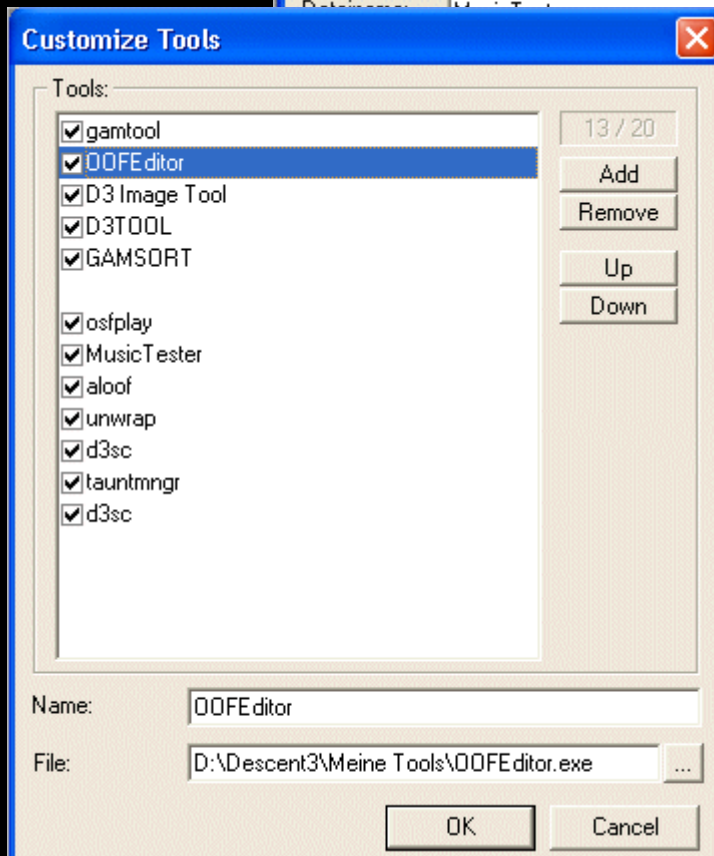
all create a directory for the D3 tools and put them there tools to put down. Then you set up the tools menu for quick access and can easily start your tools without having to go to the Windows interface. The picture on the left shows a nü.

When you start D3Edit, these entries are empty. mize you can then put together your tools:

Selection window with which we can browse through our folders and select the desired tool. (below)



After Selection that will be tool into the list registered.



It is possible to enter your own name for each tool. The position within the menu is also adjustable (left).

Ticking off an entry causes it to be deactivated in the menu, although it remains visible.

Back to Section B

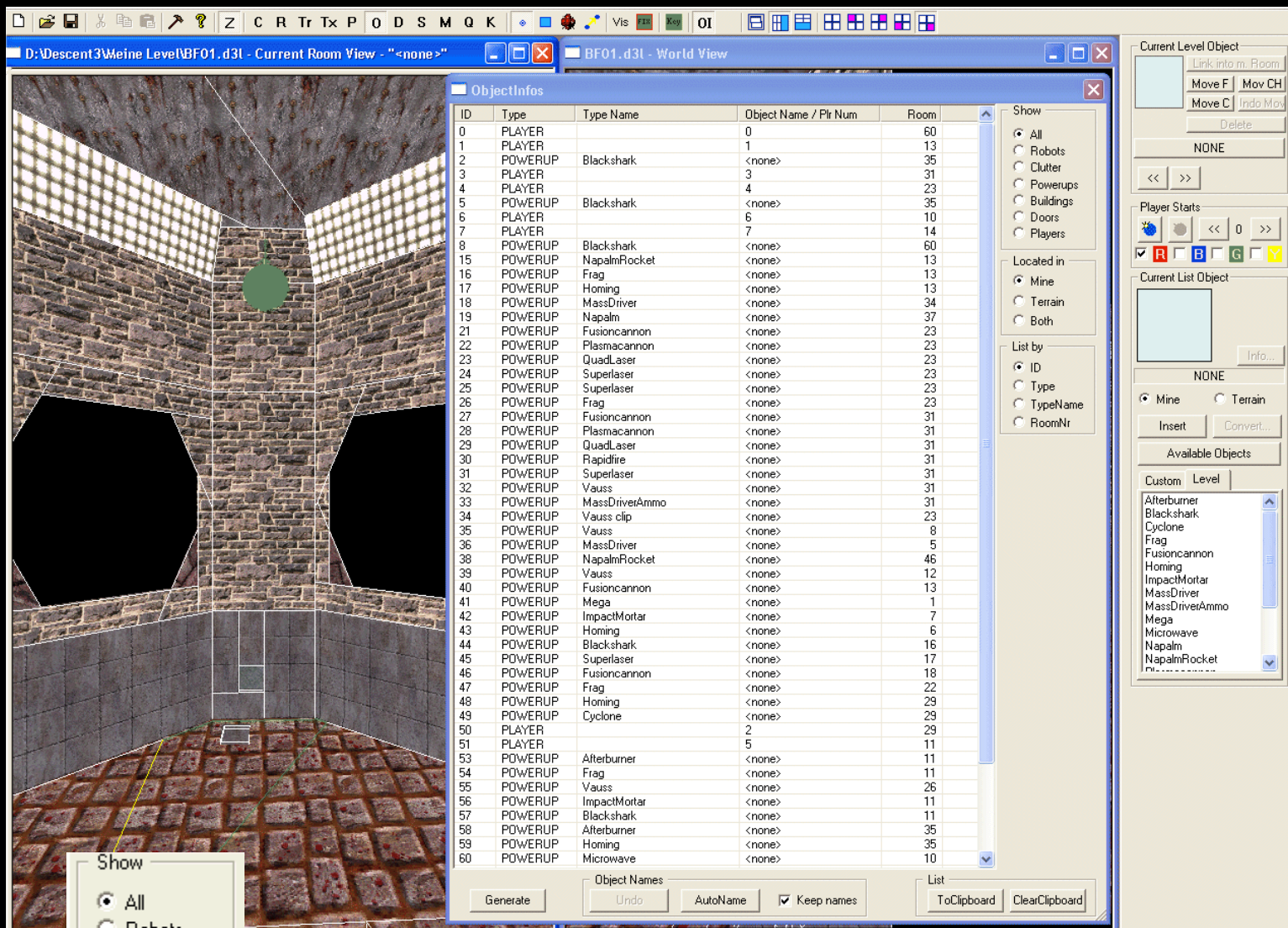
025 - Object Info Dialog (OI)

(LL)Atan

The OI dialog is a helpful tool with which you can organize your level objects. It is primarily intended to assist with creating SP levels, but it is also useful for getting an overview of the objects in an MP level. Let's load a small MP level and take a closer look at the tool.

To do this we click on the **OI** button in the tool bar:

OI



The dialog opens with a variety of information. Within the output window there are columns for:

ID	Type	Type Name	Object Name / Plr Num	Room
----	------	-----------	-----------------------	------

We have different options to display the object information in the window. In the example above, all objects within the mine are displayed consecutively by ID.

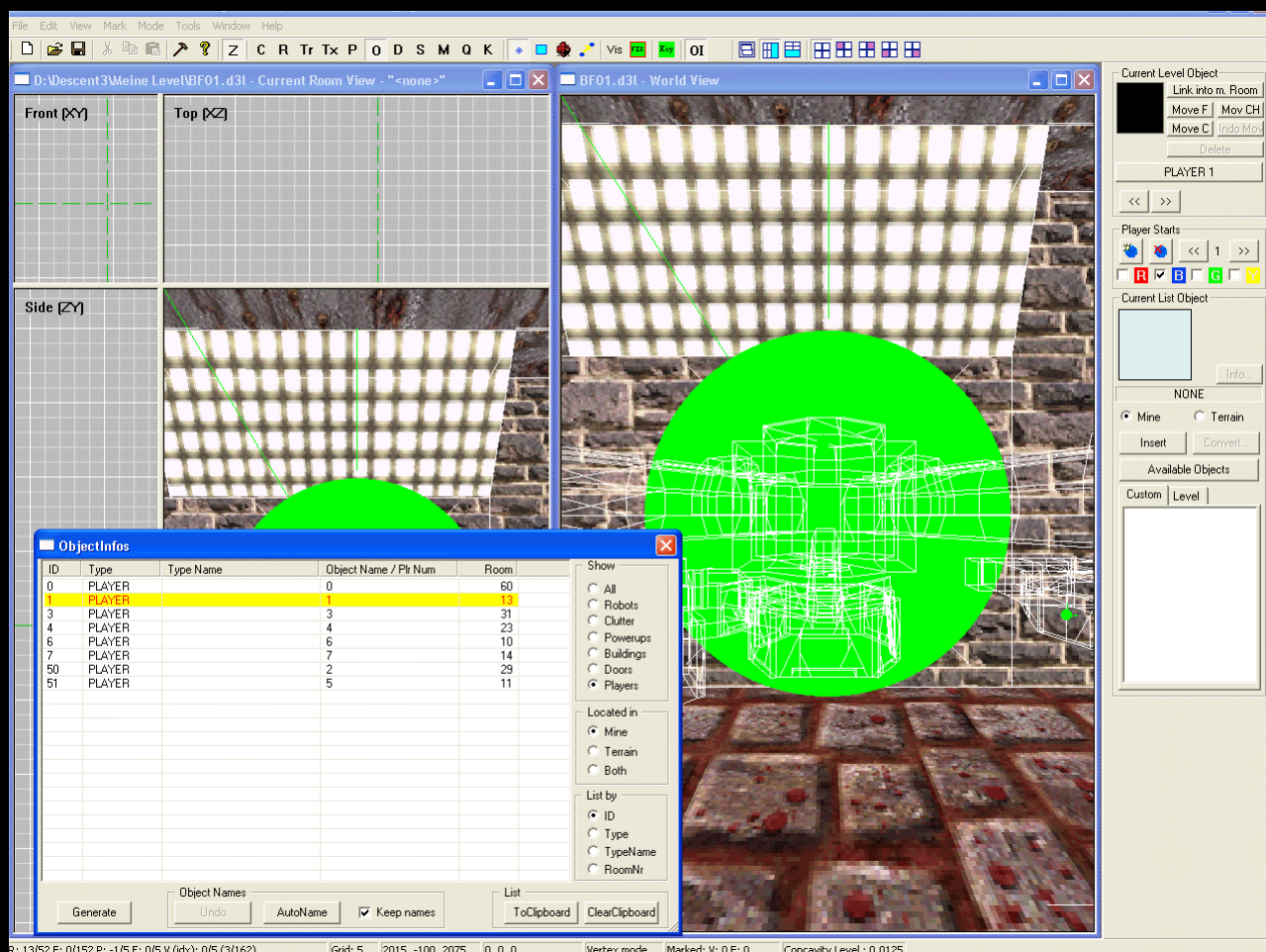
Clicking on Show provides a quick overview of the player objects used *Players*(Right).

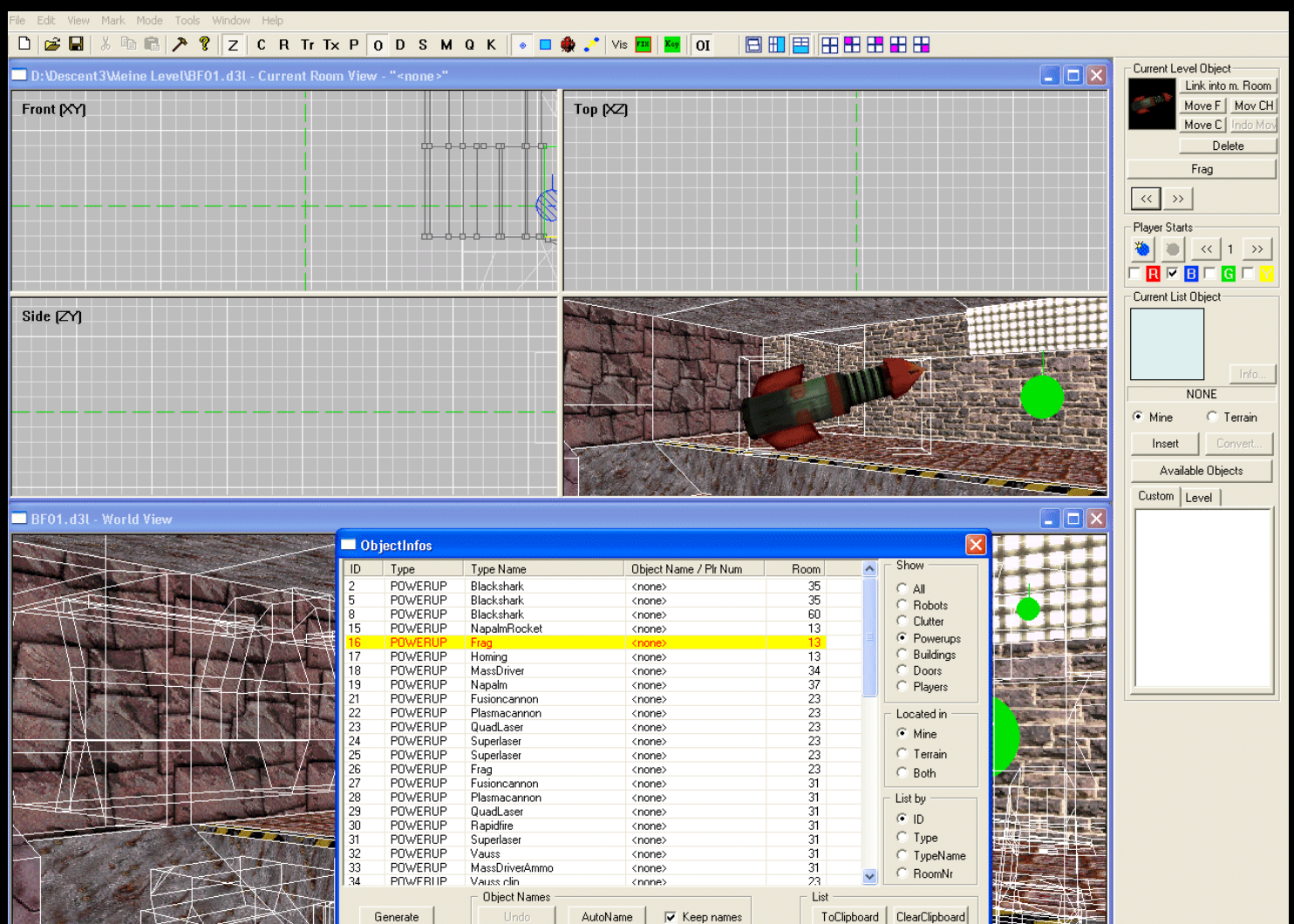
The OI dialog then only shows the player objects:

ObjectInfos						
ID	Type	Type Name	Object Name / Plr Num	Room	Show	
0	PLAYER		0	60	<input type="radio"/> All	
1	PLAYER		1	13	<input type="radio"/> Robots	
3	PLAYER		3	31	<input type="radio"/> Clutter	
4	PLAYER		4	23	<input type="radio"/> Powerups	
6	PLAYER		6	10	<input type="radio"/> Buildings	
7	PLAYER		7	14	<input type="radio"/> Doors	
50	PLAYER		2	29	<input checked="" type="radio"/> Players	
51	PLAYER		5	11		
					Located in	
					<input checked="" type="radio"/> Mine	
					<input type="radio"/> Terrain	
					<input type="radio"/> Both	
					List by	
					<input checked="" type="radio"/> ID	
					<input type="radio"/> Type	
					<input type="radio"/> TypeName	
					<input type="radio"/> RoomNr	

The order results from the ID, i.e. from the index with which D3Edit saves this object in the level. What is more interesting is the room number, which allows us to see which room contains which player object.

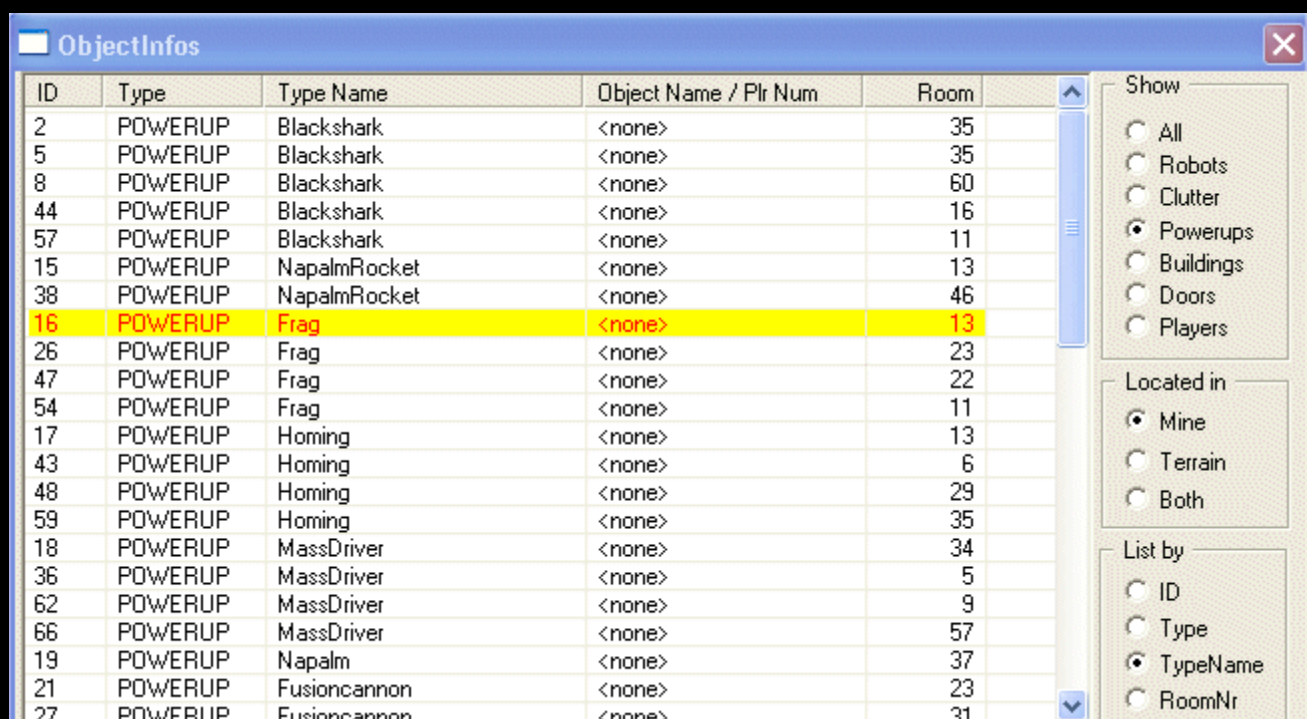
If we want to take a closer look at one of the objects, we click on the corresponding line. D3Edit immediately displays the selected object and in the main window with the object is highlighted in the dialog.





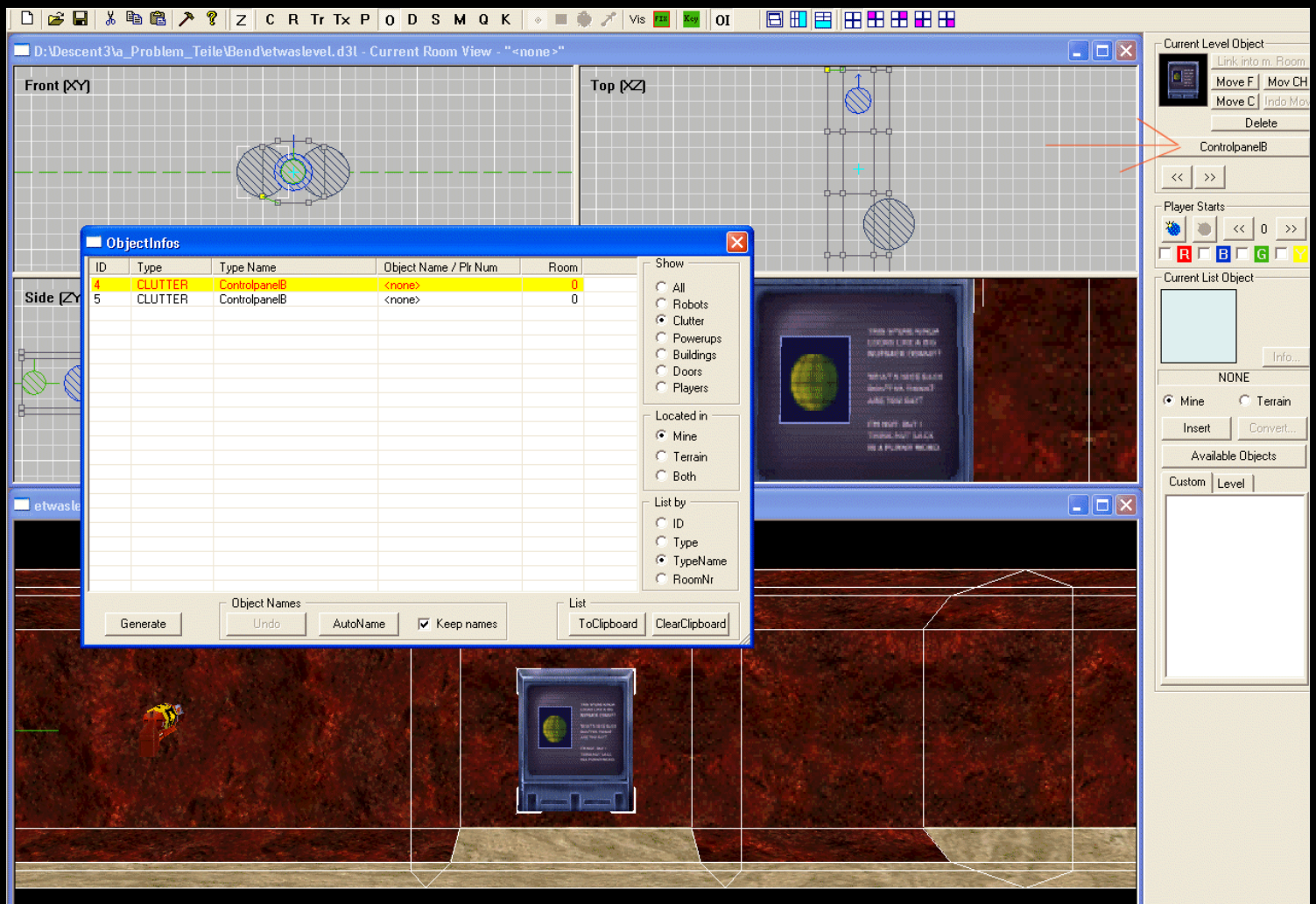
Now, for example, we select the powerups and then click on object number 16. D3Edit takes us to the room that contains the object and zooms directly on the question. On the right edge we see that the object bar is also changed accordingly. The current object is also displayed there immediately.

If we want to get an overview of the number of frags in the mine, we list the objects by 'Type Name'.



This way you can quickly recognize *how many* of these objects *where* are inserted and can immediately jump to the object with a mouse click.

An object must be named in order to address it via DALLAS. To enter a name there is a button in the object bar (see arrow). After pressing, an input window appears.



But we can also click on the column on the left *Object name* click. Obtained thereby we have an input window in the corresponding row and column.

Here we then enter an object name and confirm with return.

ObjectInfos				
ID	Type	Type Name	Object Name / Plr Num	Room
4	CLUTTER	ControlpanelB	<none>	0
5	CLUTTER	ControlpanelB	<none>	0

The new name is immediately displayed in the object bar.

ObjectInfos				
ID	Type	Type Name	Object Name / Plr Num	Room
4	CLUTTER	ControlpanelB	Überwachungspane_1	0
5	CLUTTER	ControlpanelB	<none>	0

When you enter something in the object bar, it updates automatically.



There is also automatic naming. Let's take a look at the powerup list for a level:

ObjectInfos

ID	Type	Type Name	Object Name / Plr Num	Room
2	POWERUP	Blackshark	<none>	35
5	POWERUP	Blackshark	<none>	35
8	POWERUP	Blackshark	<none>	60
44	POWERUP	Blackshark	<none>	16
57	POWERUP	Blackshark	<none>	11
15	POWERUP	NapalmRocket	<none>	13
38	POWERUP	NapalmRocket	<none>	46
16	POWERUP	Frag	<none>	13
26	POWERUP	Frag	<none>	23
47	POWERUP	Frag	<none>	22
54	POWERUP	Frag	<none>	11
17	POWERUP	Homing	<none>	13
43	POWERUP	Homing	<none>	6
48	POWERUP	Homing	<none>	29
59	POWERUP	Homing	<none>	35
18	POWERUP	MassDriver	<none>	34
36	POWERUP	MassDriver	<none>	5
62	POWERUP	MassDriver	<none>	9
66	POWERUP	MassDriver	<none>	57
19	POWERUP	Napalm	<none>	37
21	POWERUP	Fusioncannon	<none>	23
27	POWERUP	Fusioncannon	<none>	31
40	POWERUP	Fusioncannon	<none>	13
46	POWERUP	Fusioncannon	<none>	18
22	POWERUP	Plasmacannon	<none>	23
28	POWERUP	Plasmacannon	<none>	31
23	POWERUP	QuadLaser	<none>	23
29	POWERUP	QuadLaser	<none>	31
63	POWERUP	QuadLaser	<none>	19
64	POWERUP	QuadLaser	<none>	60
24	POWERUP	Superlaser	<none>	23
25	POWERUP	Superlaser	<none>	23
31	POWERUP	Superlaser	<none>	31
45	POWERUP	Superlaser	<none>	17
30	POWERUP	Rapidfire	<none>	31
32	POWERUP	Vauss	<none>	31
35	POWERUP	Vauss	<none>	8
39	POWERUP	Vauss	<none>	12
55	POWERUP	Vauss	<none>	26
33	POWERUP	MassDriverAmmo	<none>	31
34	POWERUP	Vauss clip	<none>	23
41	POWERUP	Mega	<none>	1
42	POWERUP	ImpactMortar	<none>	7
56	POWERUP	ImpactMortar	<none>	11
49	POWERUP	Cyclone	<none>	29
53	POWERUP	Afterburner	<none>	11
58	POWERUP	Afterburner	<none>	35
65	POWERUP	Afterburner	<none>	60
67	POWERUP	Afterburner	<none>	3
60	POWERUP	Microwave	<none>	10
61	POWERUP	Microwave	<none>	25

Show

☐ All

☐ Robots

☐ Clutter

☒ Powerups

☐ Buildings

☐ Doors

☐ Players

Located in

☒ Mine

☐ Terrain

☐ Both

List by

☐ ID

☐ Type

☒ TypeName

☐ RoomNr

Generate

Undo

AutoName

☒ Keep names

List

ToClipboard

ClearClipboard

The objects are all in their basic state<none>marked, so they do not contain a name. After clicking on the **Car name button** The objects of a type are automatically numbered consecutively. If objects have been named beforehand, you can use **Keep Names** determine whether the names are retained during automatic naming. With **Undo** the process can be reversed. The list can also be with **ToClipboard** saved to the clipboard for further processing. It is possible to use the clipboard **ClearClipboard** to delete.

ObjectInfos

ID	Type	Type Name	Object Name / Plr Num	Room
2	POWERUP	Blackshark	Powerup_0	35
5	POWERUP	Blackshark	Powerup_1	35
8	POWERUP	Blackshark	Powerup_2	60
44	POWERUP	Blackshark	Powerup_3	16
57	POWERUP	Blackshark	Powerup_4	11
15	POWERUP	NapalmRocket	Powerup_5	13
38	POWERUP	NapalmRocket	Powerup_6	46
16	POWERUP	Frag	Powerup_7	13
26	POWERUP	Frag	Powerup_8	23
47	POWERUP	Frag	Powerup_9	22
54	POWERUP	Frag	Powerup_10	11
17	POWERUP	Homing	Powerup_11	13
43	POWERUP	Homing	Powerup_12	6
48	POWERUP	Homing	Powerup_13	29
59	POWERUP	Homing	Powerup_14	35
18	POWERUP	MassDriver	Powerup_15	34
36	POWERUP	MassDriver	Powerup_16	5
62	POWERUP	MassDriver	Powerup_17	9
66	POWERUP	MassDriver	Powerup_18	57
19	POWERUP	Napalm	Powerup_19	37
21	POWERUP	Fusioncannon	Powerup_20	23
27	POWERUP	Fusioncannon	Powerup_21	31
40	POWERUP	Fusioncannon	Powerup_22	13
46	POWERUP	Fusioncannon	Powerup_23	18
22	POWERUP	Plasmacannon	Powerup_24	23
28	POWERUP	Plasmacannon	Powerup_25	31
23	POWERUP	QuadLaser	Powerup_26	23
29	POWERUP	QuadLaser	Powerup_27	31
63	POWERUP	QuadLaser	Powerup_28	19
64	POWERUP	QuadLaser	Powerup_29	60
24	POWERUP	Superlaser	Powerup_30	23
25	POWERUP	Superlaser	Powerup_31	23
31	POWERUP	Superlaser	Powerup_32	31
45	POWERUP	Superlaser	Powerup_33	17
30	POWERUP	Rapidfire	Powerup_34	31
32	POWERUP	Vauss	Powerup_35	31
35	POWERUP	Vauss	Powerup_36	8
39	POWERUP	Vauss	Powerup_37	12
55	POWERUP	Vauss	Powerup_38	26
33	POWERUP	MassDriverAmmo	Powerup_39	31
34	POWERUP	Vauss clip	Powerup_40	23
41	POWERUP	Mega	Powerup_41	1
42	POWERUP	ImpactMortar	Powerup_42	7
56	POWERUP	ImpactMortar	Powerup_43	11
49	POWERUP	Cyclone	Powerup_44	29
53	POWERUP	Afterburner	Powerup_45	11
58	POWERUP	Afterburner	Powerup_46	35
65	POWERUP	Afterburner	Powerup_47	60
67	POWERUP	Afterburner	Powerup_48	3
60	POWERUP	Microwave	Powerup_49	10
61	POWERUP	Microwave	Powerup_50	25

Show

☐ All

☐ Robots

☐ Clutter

☒ Powerups

☐ Buildings

☐ Doors

☐ Players

Located in

☒ Mine

☐ Terrain

☐ Both

List by

☐ ID

☐ Type

☒ TypeName

☐ RoomNr

Object Names

Generate

Undo

AutoName

☒ Keep names

List

ToClipboard

ClearClipboard

There is a special feature behind this **Generate** button. Anyone who has gone very deep into DALLAS programming will appreciate this treat. This output is also sent back to the clipboard and can then be pasted. Here is an example from the *Mission to Saturn Part II* -Script.

```
/*
$$ENUM MyRobots
0:"B_P2Gr2"
1:"B_P3Gr2"
2:"B1"
3:"B2"
4:"B3"
5:"Bot_32"
6:"Bot_33"
7:"Bot_34"
8:"Bot_35"
9:"B_P2Gr1"
10:"B4"
11:"CameraDoorOpen"
12:"Bot_51"
13:"Bot_52"
14:"Bot_55"
15:"Bot_56"
16:"B5"
17:"B6"
18:"B_P1Gr2"
19:"Bot_53"
20:"Bot_54"
21:"Bot_43"
22:"Bot_44"
23:"Bot_45"
24:"Bot_59"
25:"Bot_46"
26:"Bot_60"
27:"Bot_61"
28:"Bot_62"
29:"Bot_63"
30:"Bot_64"
31:"Bot_65"
32:"Bot_66"
33:"Bot_67"
34:"Bot_68"
35:"Bot_69"
36:"B8"
37:"B9"
38:"B10"
39:"B_P3Gr5"
40:"B_P1Gr5"
41:"B_P2Gr5"
42:"B_P1Gr4"
43:"B_P2Gr4"
44:"B_P3Gr4"
45:"B_P3Gr6"
46:"B19"
47:"B20"
48:"B21"
49:"B22"
50:"B23"
51:"B24"
52:"B25"
53:"B26"
54:"B27"
55:"B28"
56:"Bot_70"
57:"Welder1"
58:"Bot_36"
59:"Bot_37"
60:"Bot_71"
61:"B29"
62:"Bot_38"
63:"Bot_85"
64:"Bot_86"
65:"B11"
66:"B12"
67:"B13"
68:"B14"
69:"B15"
70:"B16"
71:"B17"
72:"B18"
73:"Bot_87"
74:"Bot_88"
75:"Bot_47"
76:"Bot_48"
77:"Bot_49"
78:"Bot_72"
79:"Cam Teleport1"
80:"B_P1Gr6"
81:"B_P2Gr6"
82:"B_P3Gr1"
83:"B_P1Gr3"
84:"B_P2Gr3"
85:"B_P3Gr3"
86:"B_P1Gr1"
87:"B7"
88:"Crash Plague"
$$END
*/
```

```
# define NUM_BOT_OBJECT_NAMES 89
char *Bot_Object_names[NUM_BOT_OBJECT_NAMES] =
{ "B_P2Gr2",
  "B_P3Gr2",
  "B1",
  "B2",
  "B3",
  "Bot_32",
  "Bot_33",
  "Bot_34",
  "Bot_35",
  "B_P2Gr1",
  "B4",
  "CameraDoorOpen",
  "Bot_51",
  "Bot_52",
  "Bot_55",
  "Bot_56",
  "B5",
  "B6",
  "B_P1Gr2",
  "Bot_53",
  "Bot_54",
  "Bot_43",
  "Bot_44",
  "Bot_45",
  "Bot_59",
  "Bot_46",
  "Bot_60",
  "Bot_61",
  "Bot_62",
  "Bot_63",
  "Bot_64",
  "Bot_65",
  "Bot_66",
  "Bot_67",
  "Bot_68",
  "Bot_69",
  "B8",
  "B9", "B10",
  "B_P3Gr5",
  "B_P1Gr5",
  "B_P2Gr5",
  "B_P1Gr4",
  "B_P2Gr4",
  "B_P3Gr4",
  "B_P3Gr6",
  "B19",
  "B20",
  "B21",
  "B22",
  "B23",
  "B24",
  "B25",
  "B26",
  "B27",
  "B28",
  "Bot_70",
  "Welder1",
  "Bot_36",
  "Bot_37",
  "Bot_71",
  "B29",
  "Bot_38",
  "Bot_85",
  "Bot_86",
  "B11",
  "B12",
  "B13",
  "B14",
  "B15",
  "B16",
  "B17",
  "B18",
  "Bot_87",
  "Bot_88",
  "Bot_47",
  "Bot_48",
  "Bot_49",
  "Bot_72",
  "Cam Teleport1",
  "B_P1Gr6",
  "B_P2Gr6",
  "B_P3Gr1",
  "B_P1Gr3",
  "B_P2Gr3",
  "B_P3Gr3",
  "B_P1Gr1",
  "B7",
  "Crash Plague",
};
```

[Back to Section B](#)

Section C–Advanced building

For those who are really serious about level building, it is recommended that you learn the angle sets of the triangle. Very helpful when latching, bending or generally calculating certain values you need. And it's not that complicated...

The Sirian method is a well-thought-out process for high-quality work.
Sirian has already built tons of levels, including for D2.

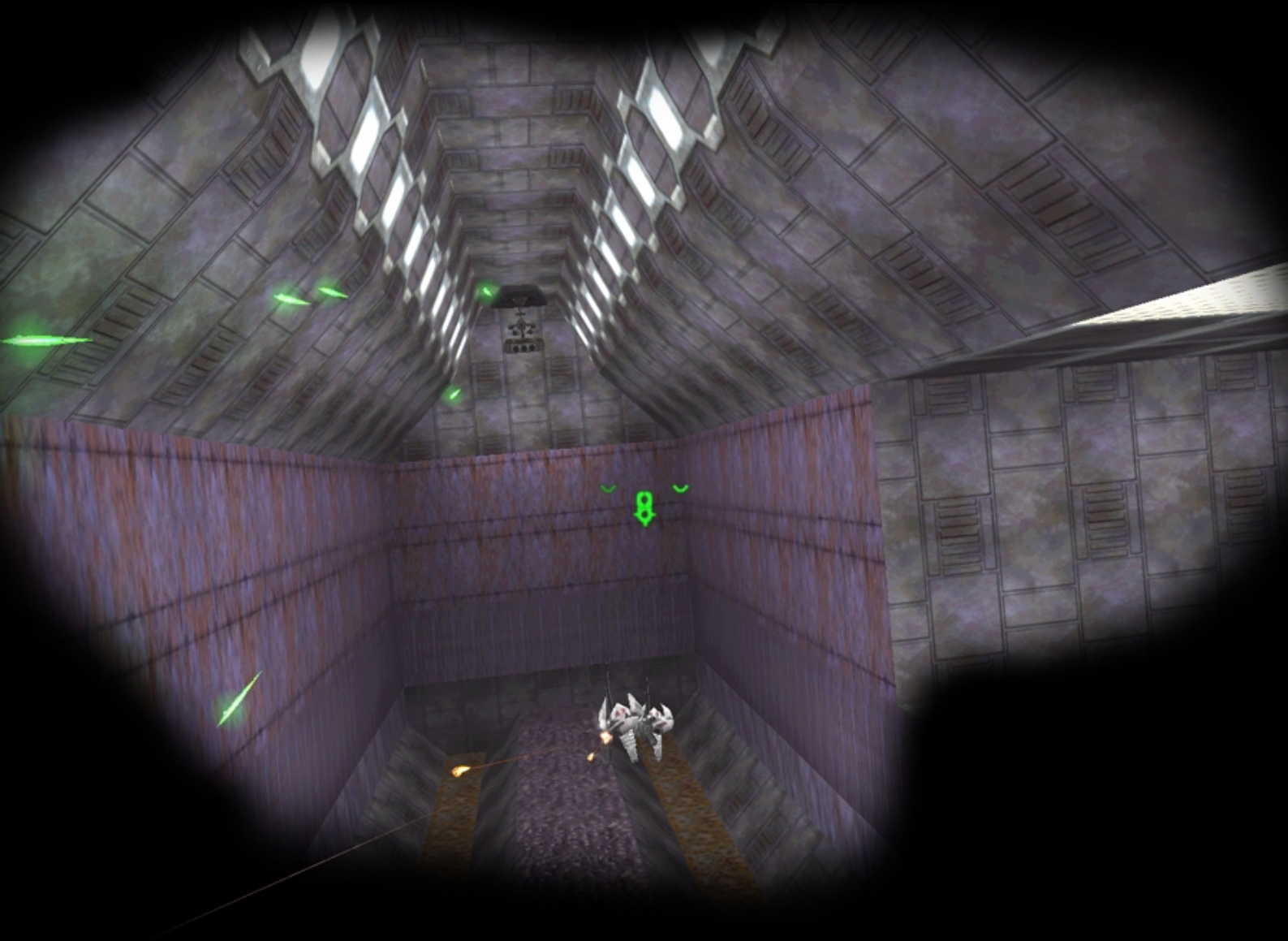
026	Construction of a high quality Shell	lesson 1	Sirian	107
027	Multiple layers	Lesson 2	Sirian	110
028	Multiple extrusion	Lesson 3	Sirian	114
029	Multiplanar extrusions combine	Lesson 4	Sirian	118
The three Tuts lie thematically between here and the previous section				
030	Share space	Explains the room division function	Atan	122
031	Vertex operations on F	Covers adding to and Removing verts from faces.	Hydra	125
032	Select segments	Use of the -Tool	Atan	126
The next five tutorials cover bending components in D3Edit.				
033	Bending – basics	The bending operation from the perspective of D3Edit	(LL)Dark	128
034	New Style Bend, v39	v39: the New Bend Dialogue	Ragil Ral	130
035	The Bend function	The bending operation from a practical point of view	Schplurg	133
036	Refurbished bend Function in v40	v40: Bend, Retreaded	Ragil Ral	135
037	Bend: Mathematics	Calculate the required values for this desired result	Papacat	137
038	Bend, objects included	Take objects with you during bending operations	Atan	139
039	Bend without bend	Elegant realization of bends without the bend function.	Fischlein	141
040	Lathe – deepening	Here the extrude and the lathe Function explained in more detail.	Hydra	142
041	Extrude – depression		Hydra	145
042	Extrude: Zero	Zero width extrusion	WillyP	151
043	Mirror Mirror...	How to put mirrors in a level as well their limitations	Ragil Ral	158
044	Space within space	An inner space connect	Fischlein	161
045	Make a sphere	Steps to create a sphere	WillyP	163
046	Complexity quite simply	Step by step to a very complex structure	Papacat	171
047	Primitives	Basic shapes - very simple	Ragil Ral	182
048	Create rock formations	Create Irregular Shapes: object	Hydra	188
049	Cave style tunnel	Create Irregular Shapes: Space	WillyP	190

The Sirian Method

This very in-depth tutorial was originally posted by Sirian - one of D3Edit's beta testers - on the 'D3Edit Forum'; it consists of four parts.

This explains how to build high-quality shells that can be easily processed further. It's an almost foolproof method once you get used to it and simplifies a lot of things.

You should have at least a little knowledge of the tools in D3Edit, otherwise you will have a hard time here, going through them several times won't do any harm.



Unfortunately I wasn't able to find anything else from this author...

026 - Building a high quality shell <>

Sirian

The first step to making a room is creating the shell. This is the conglomerate of faces that separates the 'inside' from everything else. The shell must be airtight... no double vertices, no gaps at all, and all edges fit snugly between the faces.

Now, if you're making a large room that's mostly rectangular, your shell can be a single face that you extrude. This is how I do almost all of my rooms. I usually make the floor, then extrude it so that the floor and ceiling are each a single face, or multiple faces for concave spaces.

My walls are divided, and for several reasons. Firstly, it is absolutely essential that you have some 'containment' for each face where you plan to attach another room and create a portal. If you append a room to your current room and the two faces don't match, the editor will 'break' one or both faces to allow appending without concave faces. This is actually very, very helpful, but more on that later. When it breaks the faces, the editor splits them into smaller faces and this adds vertices to ALL ADJUSTED faces which are 'hit' by the 'jumps'. If it happens to you that you have another portal face attached to the face you are currently attached to, it can be hit by a 'jump' and gain one or more additional verts, creating a 'mismatched portal', a very one very bad thing. So when planning your rooms, you need to plan WHERE you will have your portals, and as a general rule of thumb you will want to have a 'containment' around the planned portals so that any 'jumps' that occur when adding rooms are absorbed by 'buffer' faces, which you have between the portals.

For example, let's say you make a room with a cube-shaped shell. One face as a floor, one as a ceiling, one for each wall. If you create a portal in the north wall, in its upper left corner, and then later attach a room to the west wall, in the middle of the face... you will most likely have a 'jump' from the western attachment, which is the portal in the north wall and it'll throw you off your feet.

But what you can do instead is make the west wall three faces instead of just one. The 'middle' of the three faces is the place where you attach the one portal, then the 'buffer' is between that and the north wall, so that no jumps reach the north or south wall. And you don't get any mismatched portals that force you to delete and restructure entire rooms.

Trust me, I learned the hard way.

Additionally, creating split walls gives you a place where you can vary the wall textures to make the room look more interesting.

So let's go through the steps of building a shell using the Sirian method.

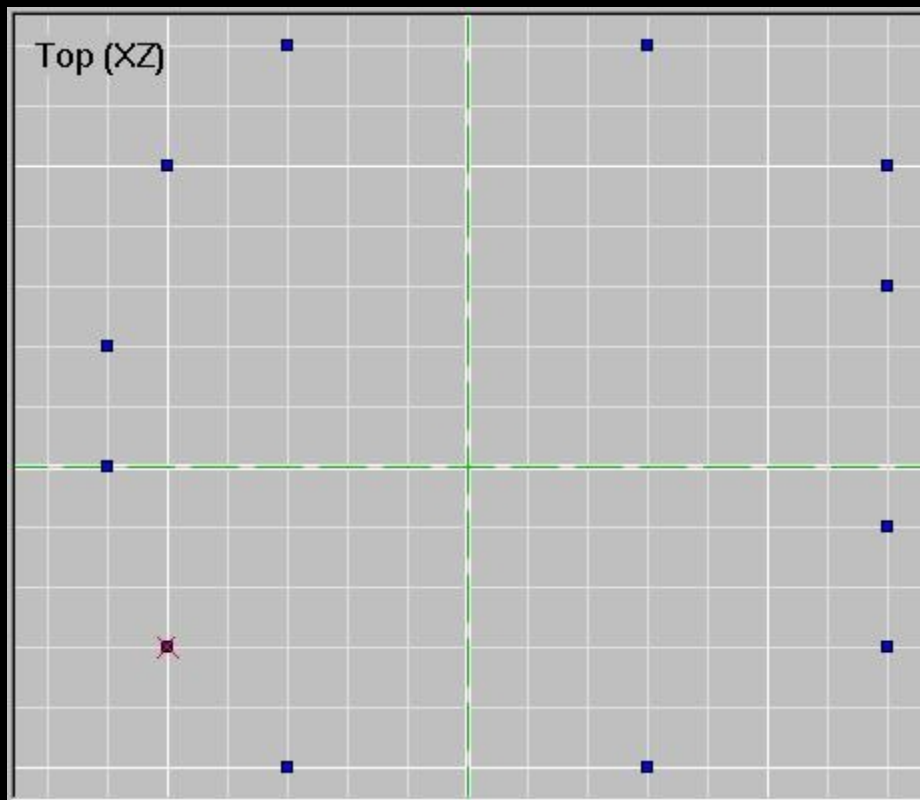
- 1) You have to go into vertex mode. The editor starts in this mode, but if you have previously edited something else you might be in a different mode.
- 2) Select the face you will extrude from. For me that's usually the floor, but occasionally a wall. We will use the floor in the example.
- 3) Select the appropriate 2d view (for the floor this would be the top view [XZ]).
- 4) Get an image of the shape of the floor in your mind's eye (it has to be non-concave - you can make concave shapes, but that involves multiple extrusion of multiple faces).

4a) I ***STRONGLY*** recommend that you use the 3d cursor (the red one) for all 2d operations. When you use the mouse and click, you lose control of the depth of where the cursor is. If you move the cursor using only the keyboard, you retain full control over its exact position in the room/level. To move the cursor, use **Ctrl cursor** key (or, IF your numlock is off, **Ctrl** and **2,4,6** and **8** on the number pad). The cursor will always stay on the grid, so you will have to change the grid size to get it where you want it. And in some cases you will need to switch between 2D views to control depth as well as height and width.

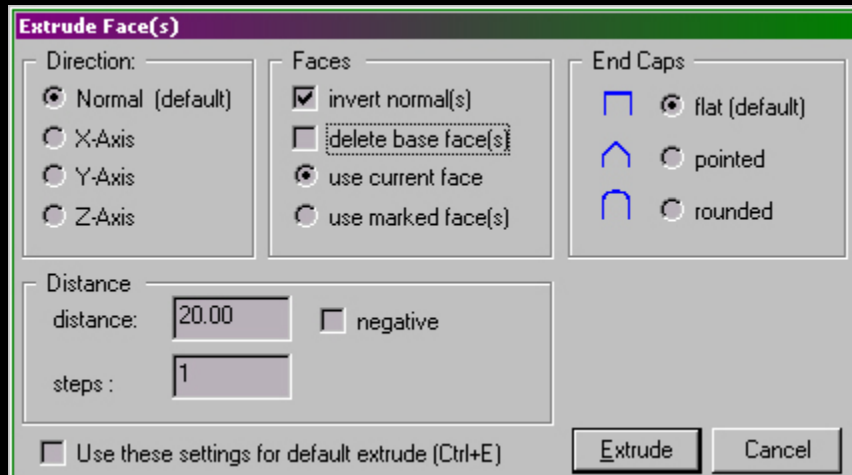
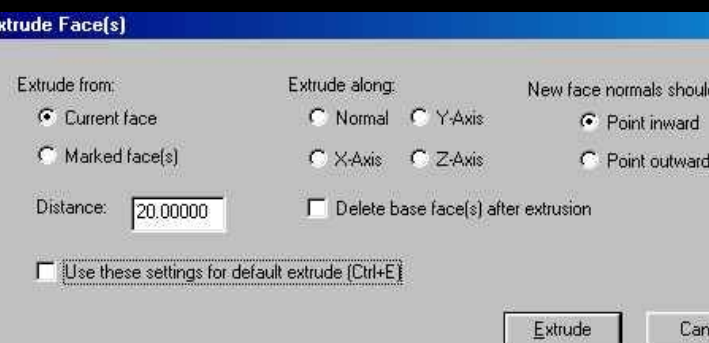
4b) So put the cursor where you want to place your first vert, then press **Into the** to put it. Move the cursor to the next location, **Into the**, move cursor, **Into the**, etc. This can involve changing the grid size multiple times if you want to be fancy. If you're making a floor, I recommend the verts counterclockwise so that the normal looks UP when the face is created. Gridsize in the picture is 10.

(Note: from D3Edit v39 it is the other way around)

5) After you have placed all the verts, you are ready to insert the ground face. Use **Shift Ins** to do that. You can texture the floor at this point if you want.

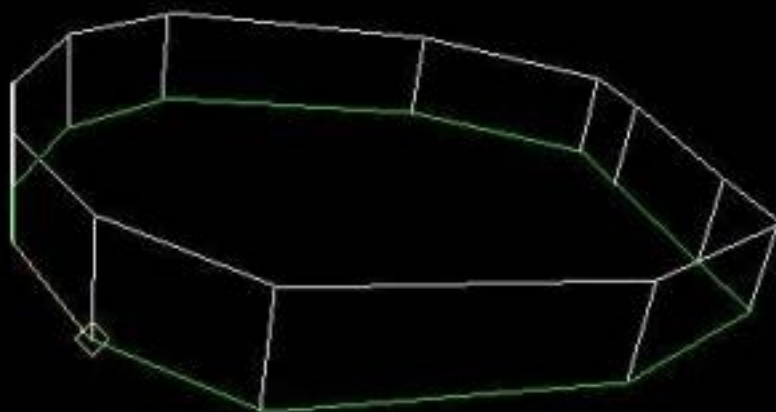


6) Now you are ready to extrude. Make sure your normal is facing the right direction, if not, go into face mode and flip the face first. Then extrude from the current face, pointing inward, with any height. In this example I used 20; remember that 20 is the size of all textures. A 20x20 face fits with a texture. 20x30 is then one and a half textures, and with some it looks bad. So you have to plan your textures with your room sizes and shapes or at least make sure that when you choose the textures (when you get there, if you don't plan ahead) they fit the room well, IF not all made of nice 20x20 faces (or multiples of which) is constructed.



Above: v39 dialog, right:
v40 Extrude dialog

6a) Note that in D3Edit 0.9 you will have to manually texture each face after extrusion. However, in newer versions automatically applies the current texture from the Texture Bar to all extruded or laminated faces, even if you create faces manually, so if you select the texture for the walls before extruding, they are ready and you only have to do the ceiling



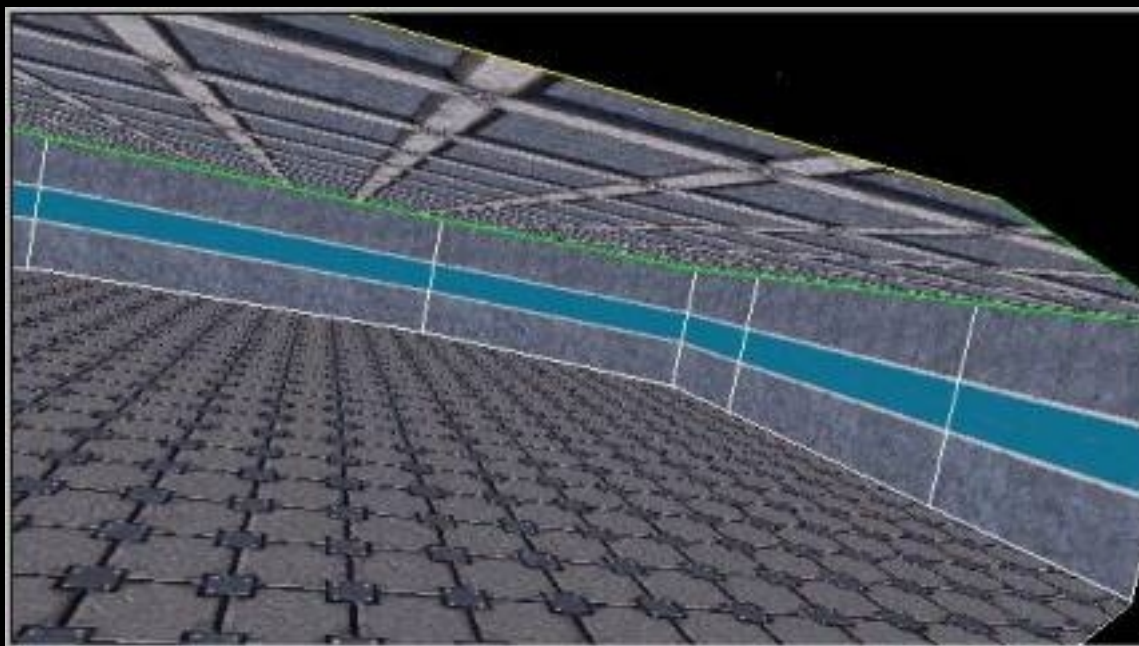
or perhaps texture some of the walls with an alternative texture change. Texturizing during

Moving forward is my method... I know some others who prefer to build the structure first and texture it later, but then it becomes a Sisyphian task. It becomes less annoying if you finish some or all of them as you progress.

Once you've extruded, you'll have a complete shell. If it's the size you wanted, you can start working on the things you're putting IN the room (lights, columns, stained glass windows, partition walls, inventory, whatever).

Note, you can have walls on the same level as the ones on the right here in the example room. This is not a concavity. I ALWAYS do this, for a number of reasons mainly involving texturing or portal surrounds.

Here is a view of the textured shell. Using the Sirian method, you can have a base shell ready and textured in a minute or two, without moving any of the verts or having to realign any shell textures later.



027 - Multiple Layers <>

Sirian

Now that you can build a basic shell, let's move on to something more complex: Layers. You can use layers for a variety of purposes, but again this mainly includes texturing and/or portal bordering. However, they can also include the ability to manipulate verts in your walls to change your shell and make funky shaped shells that aren't cube-like. And you can't do that with a simple extrusion because that would make concave faces. Currently, if you do it this way the editor tends to make some non-planar faces, but you can fix these with triangulation. The editor reports exactly which faces have a problem when you use them

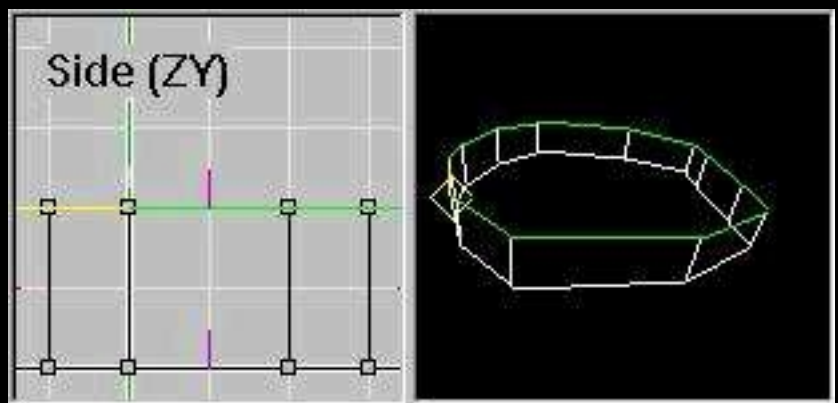
Verify-Function in File-Use menu.

We'll pick up where we left off in the previous lesson and move on to the example room we created there. It is 20 units high. Now let's bring it to 50 units in three levels.

1) The first thing you need to do after the first extrusion is to mark off the base face. Go into Face Mode and press U to uncheck everything. Next, select the ceiling face. You can do this in many ways, click in the top view until it is highlighted included, or using the 3d view, pan/zoom around until you can click on the face (its bottom in this case, the side that looks down on the rest of the room)

2) You need to mark the ceiling face, it will be the base face for your next extrusion. Then flip the face so that its normal faces up, like

in the picture on the right.

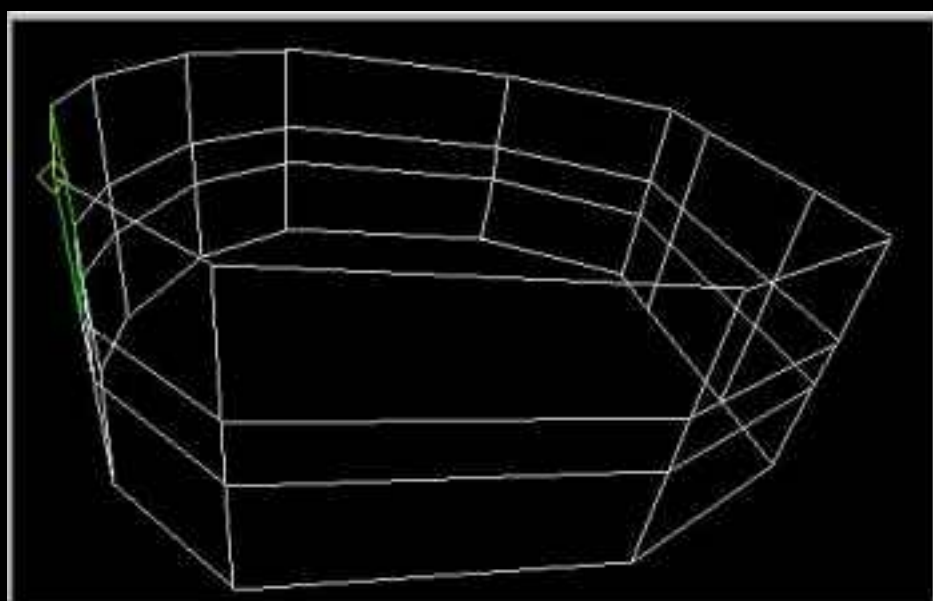


3) Now you are ready to extrude again. Do it and set the correct height for your next layer, then extrude. Note: You can use the box for Delete Base Face(s) after Extrusion tick. Otherwise you will have to delete them manually as they belong away.

4) Repeat 2) and 3) for each layer. In this example we're just doing one more layer. Note: Each layer is a multiplication of the total number of faces in your room. Two layers have TWICE as many shell faces as just one layer. Most levels i

ever made were seven, in a room with twelve faces per level, so that was 86 faces *only* for the shell (including base and ceiling). You have to keep an eye on your face crowd. If it gets too big, your space will be hard for low-end users, and might even be hard for some players make it unplayable.

Here is a picture of the finished shell.



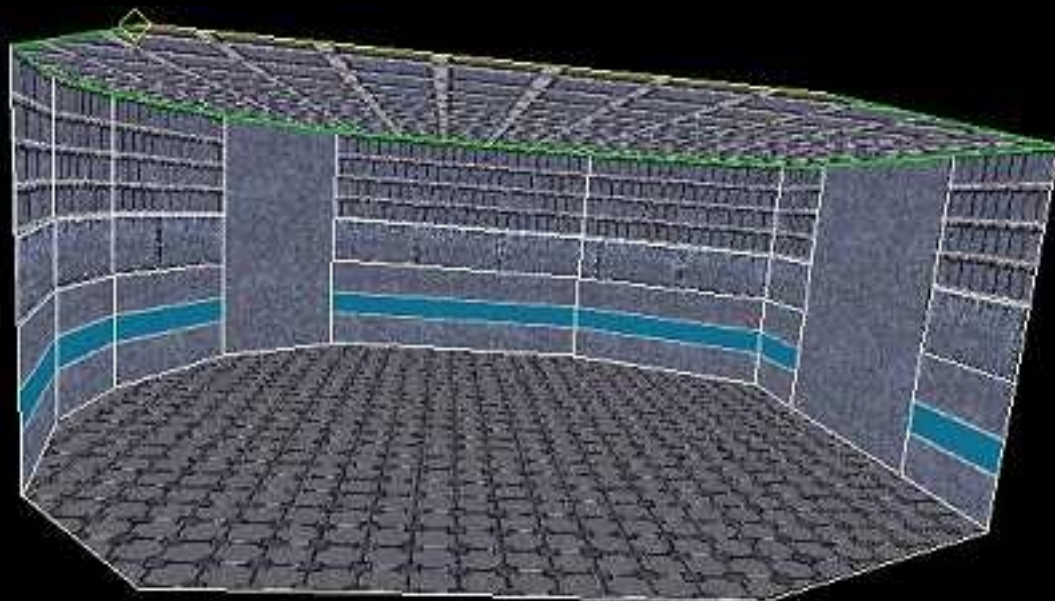
Now you have a room divided into levels. It's a little small, but could serve as a good waypoint between larger rooms, or as a side room. It's tall enough that you can fill it with a variety of things, including overhead lights, a raised platform, an altar (with a powerup on it), whatever you have.

You could now decide, on the one hand, that you want to have a few vertical stripes to break up the horizontal ones, and on the other hand, that you want to add another space, but not at the top or bottom of a wall, but in the middle. In any case, you can combine faces to achieve your goal.

Unions of faces will NEVER give you t-joints. This is because no verts are deleted and no edges are merged. The edges shared by both faces are deleted, but all others remain as they are, protecting the integrity of your shell. So you are completely free to combine all the faces you want, as long as this doesn't interfere with any other plans you may have.

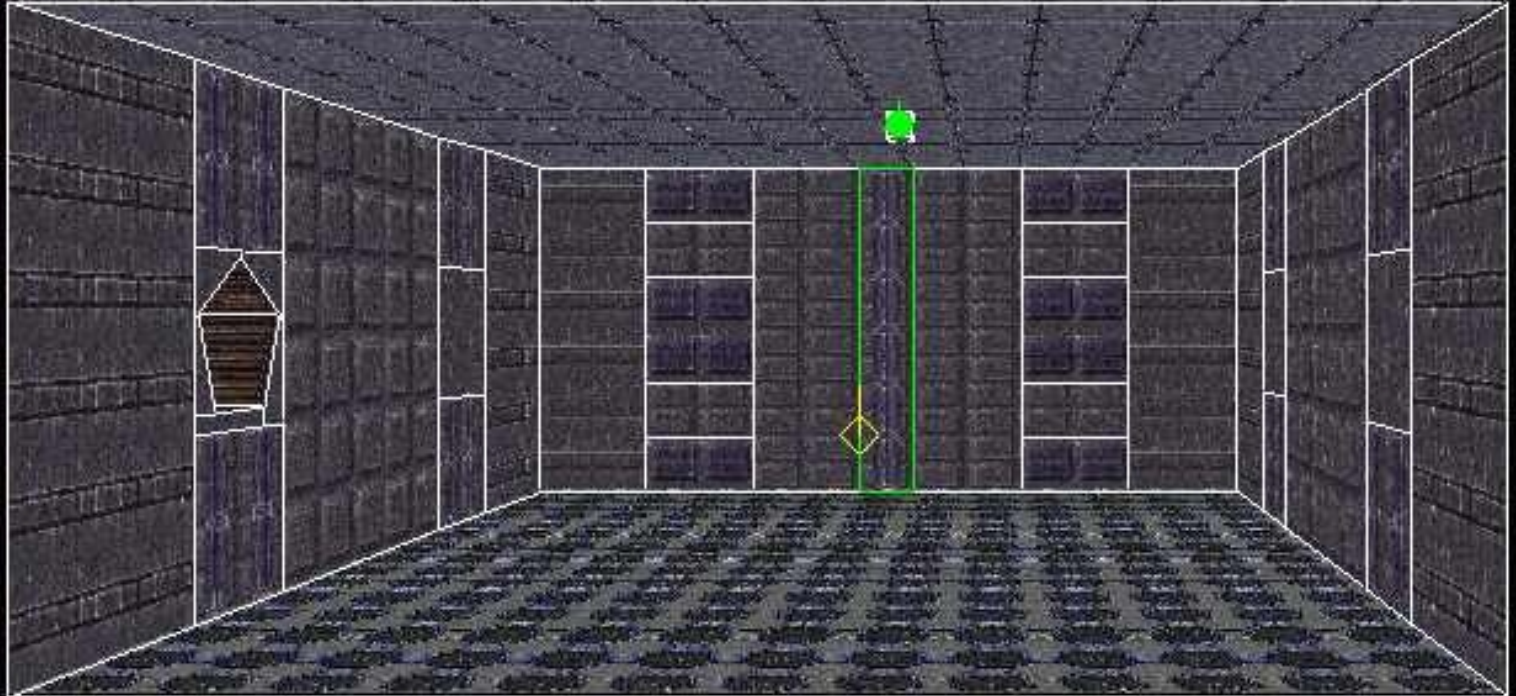
To merge faces, first be in Face Mode and make sure no faces are highlighted. Then select the face with the texture that the new face should have (if you don't want to keep one, it doesn't matter which face you choose). Highlight this face. select the other face, then use the Combine Faces button (under Face) from the Geometry Bar. The new face remains marked, so you can combine a whole series of faces quite quickly.

Here's a look at our example where three of the walls have been combined for portals. Notice that I do too textured differently. The room could now serve as a hub leading to three other rooms. Of course he still needs lights and maybe other sprucing up, but the shell is finished, textured and flawless.



In my Entropy level Siege Maximus, the main room located between the two fortresses was intended to have a whopping 16 portals connecting 16 different rooms. This includes four entropy bases, four sniper ports, four entrances to the red base and four to the blue one. To achieve this, I did an extrusion in five planes, and I had seven wall segments in the north and south walls and five segments in the east and west walls. That was 24 faces per layer and 12 faces for five layers in my space after I finished extruding. Well, I didn't need ALL the faces, I just needed the layers for the portal enclosure, so it was time to optimize.

I had four portals that went into one level, another four one level higher, four more in the level above, and I wanted a level above and below the portals. So I made my layers, and then started combining rows of faces where portals wouldn't go. Take a look at the shell of the room in the picture here. This screenie was created using the special 'Render Shell Faces' option in the settings dialog, with the non-shell faces ticked off and not displayed. All lights and fixtures are IN the room, but only the Shell is shown in this view.



Note that in walls where there will be no portal, I have combined all five levels into one face. And also a number of other faces. This allowed me to go from 122 faces down to 58 in the base shell by merging faces where I could without compromising the portal bezels.

Look at the pentagonal appendix in the west wall. This is an entropy base, and there is one on each of these 'midplane' faces in both the west and east walls. These are only four of the 16 portals, which explains why the east and west walls only have nine faces in them while the north and south walls have 15 each. Do you see the cracks caused by attaching the pentagon face of the Entropy Base to the larger, square face of the main room? If this entire west wall was just a face instead of that enclosure structure you see there, it would be impossible to attach the second portal with precision, since the cracks caused by the first portal would split the entire wall into a few pieces, introducing new verts and ALL Normally it would move around so I couldn't just do Place Room Attach room and be done. But by designing the wall with a border, then when I assemble the level, attaching all the rooms is a piece of cake, and I never get mismatched portals. Containment, containment, containment: it can save your level!

Now look again. Do you see the current face that I selected in the screenshot? It is in green while a small part is yellow. The yellow is the current edge of the current face, and is evidence of the layered Sirian method I used to make this space. This edge is the second level from the floor, and it mates with one edge of the adjacent face. Like I said, you might be better off making layers and then merging faces than trying to make a layer and then split some of the big faces into smaller pieces. And if you do it that way... you'll have a lot of T-joints to fix.

Also note that the current face in the screenie itself has a containment, left and right on both sides, so that it is buffered between the two rows where the portals to the four base entrances go (in levels two and four). This middle row is where the two sniper ports will be attached. That's why I needed seven rows in the north and south walls. Row one is a buffer, row two has two portals, in levels two and four, row three is a buffer, row four has two portals, which are small and rectangular and do not need to be contained vertically, but need horizontal containment, row five is a buffer, row six has two portals on levels two and four, and row seven is a buffer. Whew! There's a lot going on here but... I had success getting six portals on this one wall with no errors, and I had some nice texturing to do during the process.

Layered extrusion has its applications. This complex room has 16 portals, and it is just a box. However, I added a few bridges and fancy lights to it so that it doesn't look as bland as it does in this view, which shows ONLY the shell. But this was the best example I had of what you can do with layers and extrusion to make your shell. Because, realize I made this ENTIRE room without marking and moving any vert. I even did the fixtures entirely with extrude or lathe, or in this case various lights, manually inserting verts and inserting the faces to make a light, and then copy/pasting the others of its kind from the first.

You ***CANNOT*** have a great space without a great shell. Learn to make good shells, and you'll be halfway to making executioner levels that look as good as they (I hope) play.

The Sirian method relies on intelligent use of extrude and lathe to make your shells. You can do a lot this way, with minimal work, some in the area of texture alignment, so it moves along quickly and always looks sharp.

027-sirian2.rar

[Back to Section C](#)

028 - Multiple Extrusion <>

Sirian

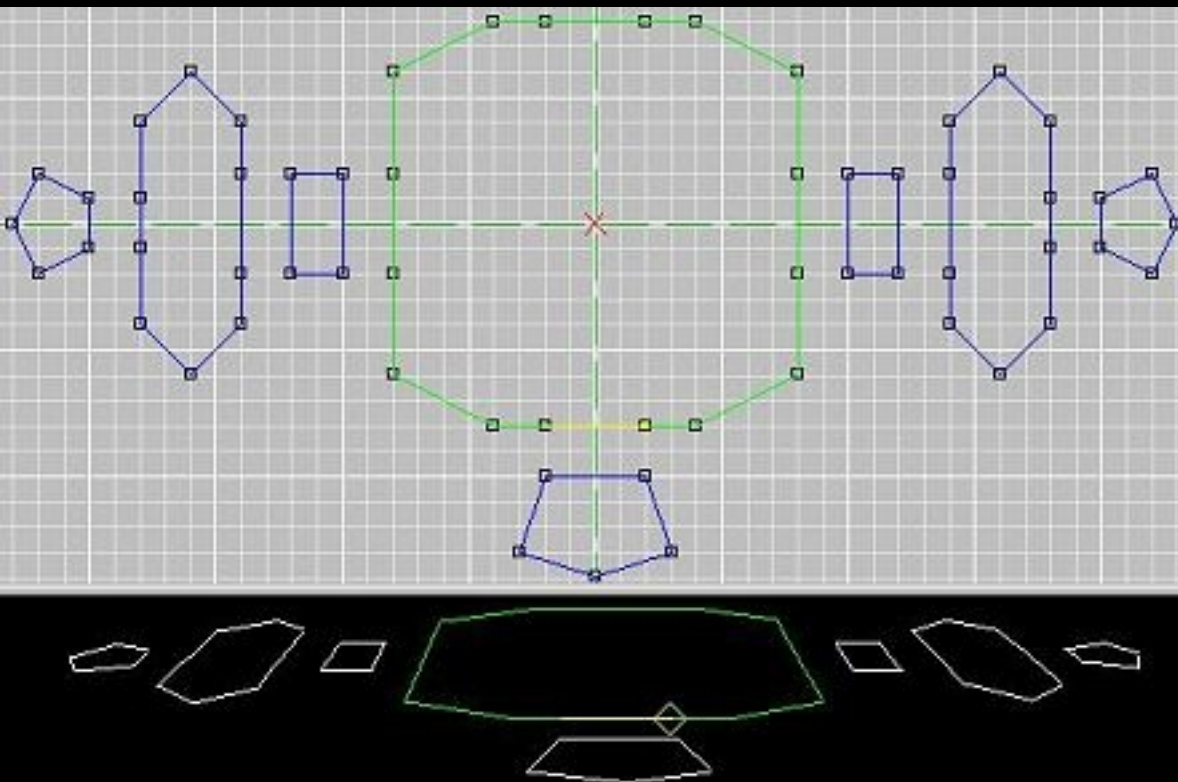
Sometimes you just can't make the shell you want from a single floor or wall face. This is a dangerous process if you don't know what you're doing, in the sense that you're creating T-joints left and right and filling your level with errors. You always want your shells to be airtight and bug-free to eliminate the risk of crashes or other issues during games.

So let's walk through the process of making a flawless shell with multiple extrusion. You must plan carefully so that you don't produce T-joints that leave holes in your shell. Look at this vertex pattern (right).

Now I make the other faces separately, **only** to show the parts and how to adjust the edges and verts

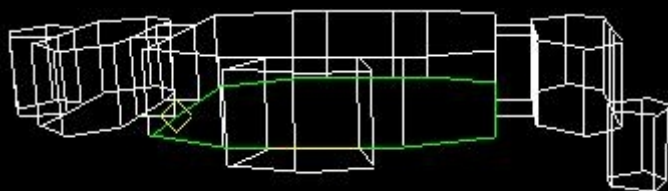
The reason you have to have so many faces is because you don't have any You can have concave faces. So you have to think about the shape you want, and

then the floor in so few Faces like possible break down, without a concave too have. Note also the yellow one edge on the current face: this edge MUST be exactly the edge of the Pentagon faces fit because this one together used edge is in the ground, and before it doesn't fit exactly do you have T's and a bad shell.



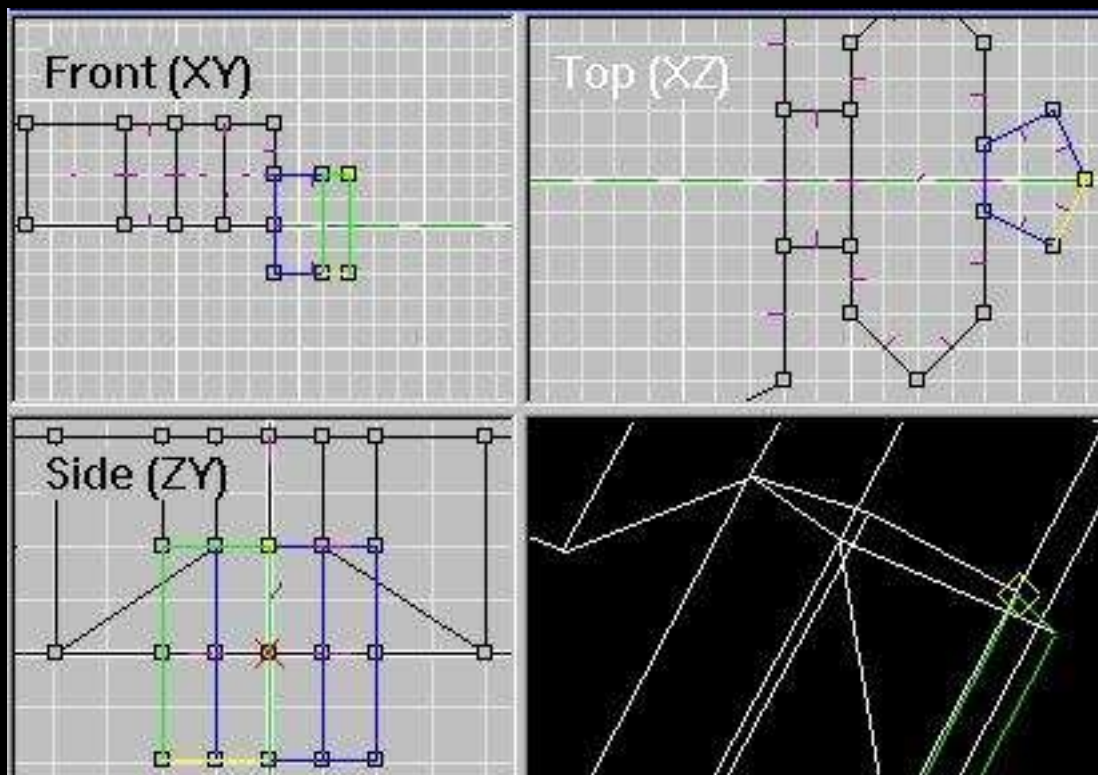
Now I'm going to move all these faces together to their correct positions and extrude them. (Do you see the grid? it's a 10 grid, not 5. You can work in the grids with the **Mouse wheel** zoom in and out or with **Shift mouse click** sway)

In current versions of D3Edit you can extrude from all faces at once by typing 'Extrude from: Marked Faces' in the Extrude dialog. Then bang, they are all extruded in one go, to the same length. If you're still running a 0.9 beta, you'll need to extrude them one at a time.



This is what it should look like after extruding. Do you see the block on the far right? It has been extruded down instead of up. Don't let this happen to you! Always make sure you do your ground verts counterclockwise (Note: clockwise at 0.1.39 beta 10) starts when you

Make floor faces so that your normals look up. And check before you extrude that no normal points downwards. If you do get one, you have to mark ALL the faces, mark the ones that are facing the wrong way, flipping, and mark the bottom faces again. Yum.

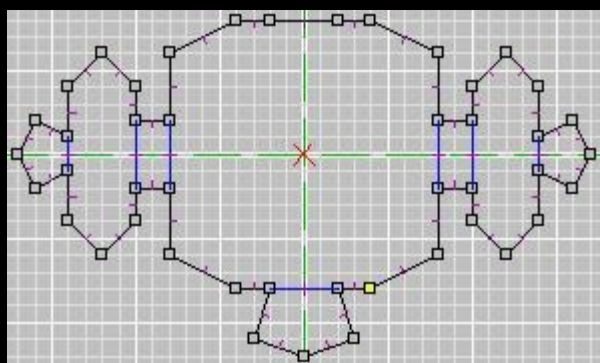
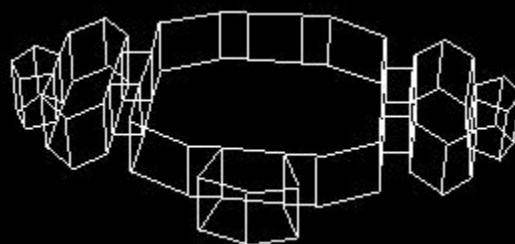


It's also a good idea not to connect the verts before the extrusion. Is to enough time once the shape fits. If you connect the verts, you can't use the blocks independently move from each other. you will be connected, and to move one creates distortions in others, like here:

In current editor You can versions now marked faces in Face Mode

move around. So the move COULD have solved the problem IF the verts hadn't been connected. So don't remove any extra vertices until the shell is finished, and THEN do that. And don't forget about it! You can create massive errors if you fail to connect the verts to seal these shared edges.

Now let's look at the room as it was intended: But we're not finished yet. There are some faces between the segments that need to be removed to open the thing up into a single area. You



The best way to do this is in the top view. First make sure you're in face mode, then unmark all the faces, and then use the mouse to mark the faces that need to go. To do this, draw a frame over the seam, one at a time. Once you have marked all the seams, it should look something like this (left):

Make sure you don't accidentally mark one of the bald shell faces. You just want to delete the seams between the extruded segments. Once everyone is marked, press **Remove** or choose **Delete** from the **Edit** Menu.

NOW is yours
Shell ready, and
You should
'RemoveExtra
Verts' make
to seal all the
edges.



And there you are:

A non-trivial looking, non-cube room made entirely with Extrude.

Texture however you want. Then your shell is complete and error-free, and you can start populating it with non-shell faces.

Another note:

This example space is **ONLY** an example. I actually wouldn't do this layout this way if I wanted to use it in a level. I would want this in three rooms as these wing areas on the sides would have no sight lines to most other rooms. They should be spaces for themselves. So this example room should actually be three rooms: the middle and the two wings. And that would also save you a little time because you just create a wing as your own room and use that room twice when you create your

.orf's composed around that.d31close. Well, that's not the main reason for breaking this room into three. The main reason is because it would be the best for the Vis Table and the game engine, and optimizes the performance of your level. But the fact that it would also require less work is a nice bonus.

Addendum - Texture Alignment

With the 0.9 Beta Editor, aligning textures is a tedious process. You will have to align the faces manually, one by one, if you need to re-align things after extruding. One thing you can do in this case is a simple step-by-step process to reduce the work:

- 1) Choose a face to align.
- 2) Click the middle button to reset the face to 'Default UV's' (this eliminates the throwing, twisting effect you get when you move the verts or when you end - if you don't want the textures to move with the verts - bend, you have to reset it afterwards.)
- 3) Align the texture, examine it in the 3d view in the textured view rather than wireframe. Note the U and V values.
- 4) Go to a similar face that needs to be aligned in exactly the same way, reset the UV's with the middle button, then enter the values from the original (that you wrote down). This face will now match the first one you made.
- 5) Repeat as often as necessary.

In the current editor there are all kinds of alignment tools you can use. Here's how to use them.

Align: Current Face or Marked Face(s): This is a switch. Align current only aligns the current face. But you can now use the process I mentioned above for the 0.9 in a quicker way: you mark each individual face that needs alignment, then switch to Align Marked, and when you align one, they're all aligned. Pretty, isn't it? And you can use Align Marked to reset an entire room in just a few moments: Mark all faces, Reset marked, unmark: Bang, you're done. No more wavy, dancing, bubbly, warped textures everywhere (If you like moving verts around a lot.)

Align Marked To Current: You can align a whole strip of faces at once (e.g. All walls of a level of a room: let's say you have a lot of non-20x20 faces and you want it to look professional...) So what you do is you Select a 'source' face, mark all the ones you want aligned with it, and then click on the source face so that it becomes current, then click Align Marked to Current. This aligns all the selected faces so everything looks smooth. All the marked faces must be adjacent to each other even though they share common edges.

Ctrl-click: You can manually realign textures, either in Textured Mode or Textured With Outline Mode (I personally prefer with Outline). You select the source face as your current, and then **Ctrl-click** on an adjacent face (which shares an edge with the current face) and the adjacent

Face texture is aligned to the current. The realigned face automatically becomes the current face, so you can walk around an entire room, or wall, or floor, or lathe object, or whatever, in a relatively brisk manner, always in view and seeing the changes in REAL TIME You move forward. THIS IS AN AWSOME FEATURE! Make sure you know how to use it, as its power is breathtaking. You can even cross rooms**Ctrl-click**en. Sadly, you can't do that between

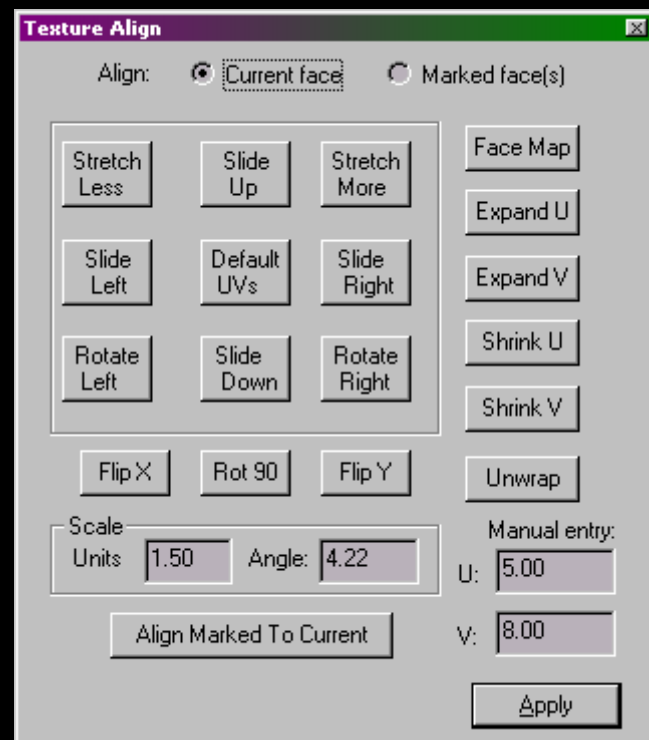
.orfS, but between rooms in a compiled.d3l...which allows you to create entire floors and ceilings and walls in your level. I tested this feature extensively in Divided We Fall, and Gwar commented after seeing it, *"This will set a whole new standard for texture alignment."*

The reason for this part of the exercise is simple: the floors and ceilings in the example room are misaligned. in 0.9 this is a nuisance, but in the current editor you only do a few**Ctrl-clicks** and it will look seamless.

Soon these tools will be in your hands.

028-sirian3.rar

Back to Section C



029 - Combine multiplanar extrusions <>

Sirian

Now we will venture into the land of T-joints and (I hope) emerge unscathed. Once you start extruding along different axes and connecting the results, you are predestined for T-joints and bad shells, unless you go to MAJOR pain to avoid them. And since you can't easily fix them in the 0.9 version of the editor, you'll then want to avoid them at all costs. In current versions you can add verts to faces and their edges, allowing you to set up T-joints, but this still takes work and a bit of experience, so I'll show you a way to do this without creating any T's.

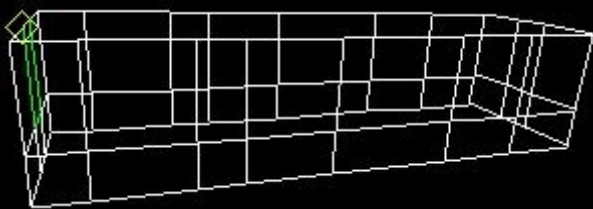
Let's start with the vert pattern you'll need for portal containments. We make a room with four portals attached: one in the west wall, two in the north wall and an angled extension to the south that leads to another portal.

We will approach combining the previous lessons as we are making a space in layers but with multiple extrusions coming from different directions. Start with the containment pattern in the 'primary' layer:



Well, the key to avoiding T-joints is to mirror your verts. If you only need one vert on one side, fine, but you have to place an opposite vert on the other side even if you don't need it. I want a portal between verts two and three as well as between verts four and five on the south wall. So technically I don't need verts four and five on the north wall, nor do I need verts two, three, six and seven on the south wall, but I had to mirror them on each wall. You'll see why in a moment.

Note that this is a 10 grid, and I've zoomed out to a wide view so the whole thing can be viewed. So every two grid points represent 20 units, which is the size of a texture bitmap. If you work in blocks of 20, you'll have an easy time texturing. If you start working off angles, you have to line up your textures or the level will look like crap.



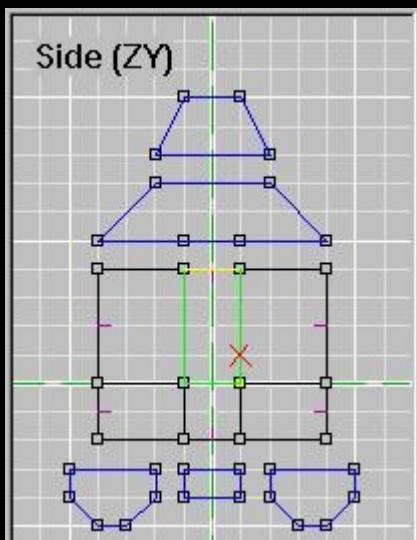
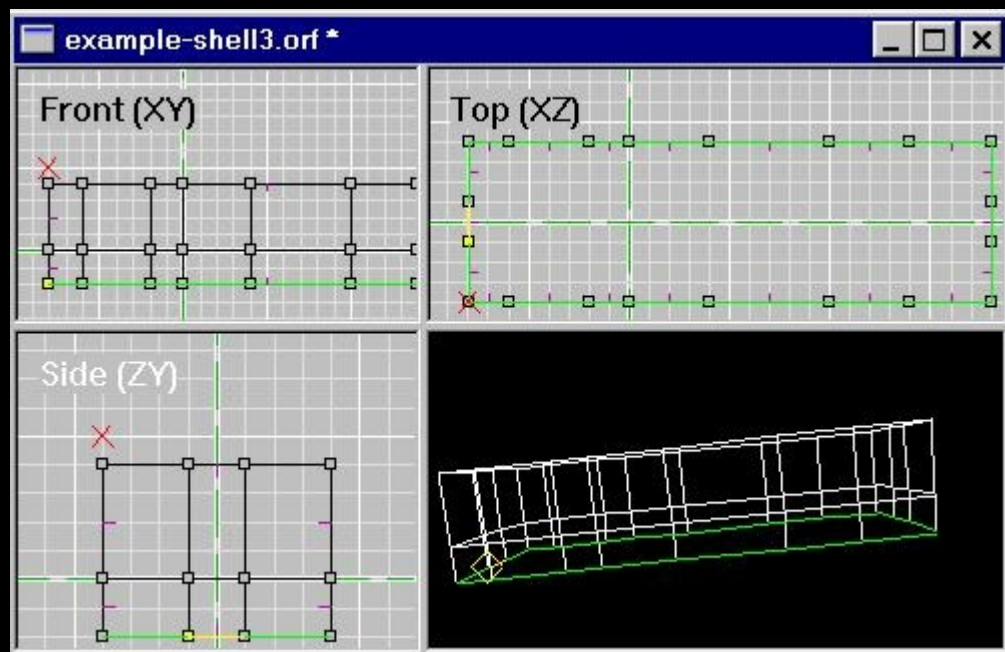
In this image I added the initial face, extruded 40 units up, flipped the bottom and extruded 20 units down to make a second layer underneath. After deleting the seams between layers I have the 'body' of the room finished, and for now the extrusions in the top view.

Now it's time to go into side view and prepare to extrude from this layer. At this point we MUST work with the coordinate cursor and a good knowledge of our whereabouts in all three dimensions.

This is an image where the cursor has been moved to the exact coordinates where I will begin placing new vertices. You have to place the cursor at the correct width and height in the side view, and then look in one of the other 2D views to see if the depth is correct. If you are NOT at the correct depth, click in one of these windows and move the cursor with it **Ctrl cursor** Keys. IMPORTANT NOTE: clicking in an active window *moves* the cursor. Clicking on an inactive window merely activates it. So you have to keep an eye on which of the 2d views you have active, if any. Trust me, if you don't already know, you'll soon find out more about the frustration of accidentally clicking in the wrong place at the wrong time and then having to manually reset the cursor.

Once the cursor is at the correct grid point in all three dimensions, you need the side view active, because that's where we will carry out our next step.

Insert Vert, Ctrl-Cursor to move the cursor to the next point, Vert insert, repeat. You know the drill. Only now you can't get away with cheating and using the mouse: you **have** to Use cursor as we



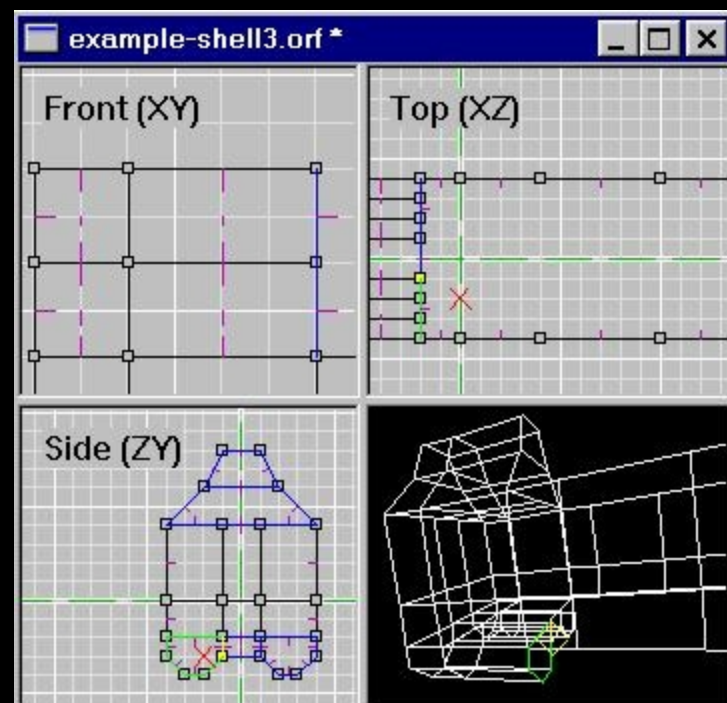
no longer work within the reference frame (moving the reference point for little things like this isn't really worth it in my opinion - it can cause more problems than it solves). The left image shows the completed face pattern. As you can see in the 3d view, they are aligned with the west wall at one end of the room. I pulled them apart so you can see each individual face. Normally you don't have that kind of distance between them.

Next you need to check their normals, make sure they are all facing the right direction, then it's time to do a layered multiple extrusion, combining what we did in the first two

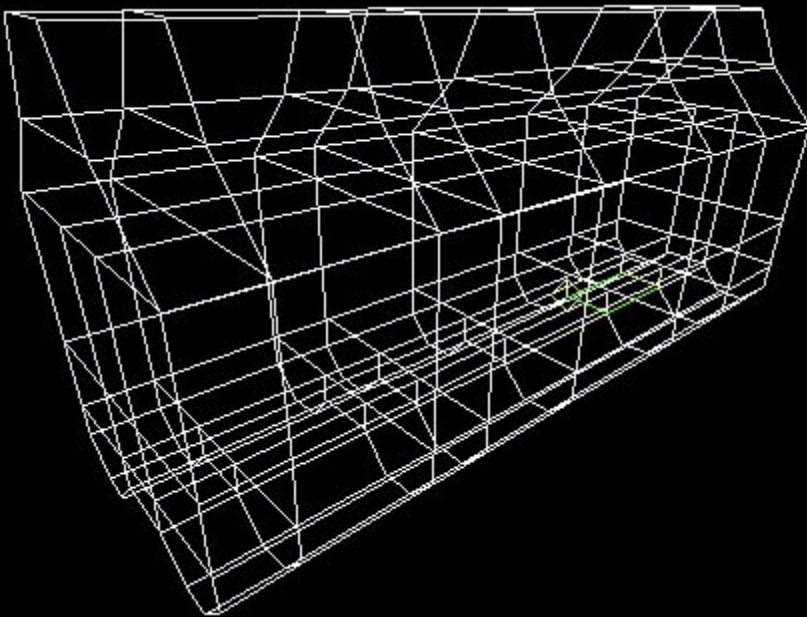
lessons covered. In a current editor version you can extrude all faces at once. In the old 0.9 version you have to do the extra work of extruding them individually, and in this case you **MAY** want to separate the faces, as I did here for illustrative purposes, to make it easier to see your progress.

Here you can see that I have extruded two layers. You **must** match the extrusion depth with the verts of the room 'body' walls have. That's the secret to avoiding T-joints: lots of planning. If you extrude behind a wall vert, you have a T. If you extrude to a length where there is no wall vert, then you have a T. If you extrude the length of the room in one go, you have a boatload of T's that can You can't afford it in the 0.9 version. Now they are

The actual steps in performing the extrusions are the same as I showed you in previous lessons. You mark off all the faces, then mark the ones you want to extrude, flip them if necessary, extrude the correct length, and repeat until you've marched through the length of the room. Note that you can simply select the entire face group in the top view, just click

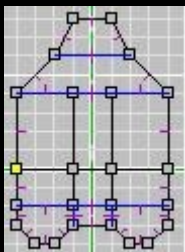
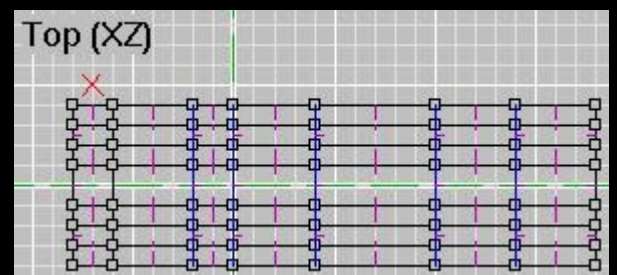


and pull, and with no risk of unwanted marks unless you get really, really sloppy. Also note that in the image I already have the next face group marked and ready to extrude, and they have already been flipped (see the normals?)



Here on the left you can see what it looks like when the extrusion is done. Now we need to go and delete all the seams, and we need to do this in two steps.

Here I have marked the seams which can be safely marked in the top view. There are some which are not marked, but I already have them before deleted. Time to delete these.

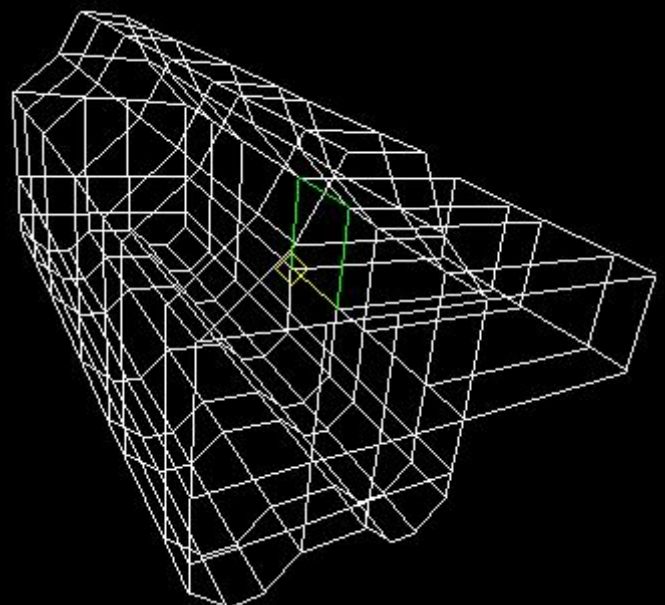


And here on the left you can see the remaining seams, which will be marked in the side view. Once these have been deleted, we have completed this stage of the process.

Now let's move on to the angled section I mentioned earlier. We will for this stage extrude from the front view. Anyway, this time we won't be making new faces to extrude from, but rather existing ones. Use faces that are part of the south wall. These faces need to be marked, then flipped, then extrusion can take place. Like this:

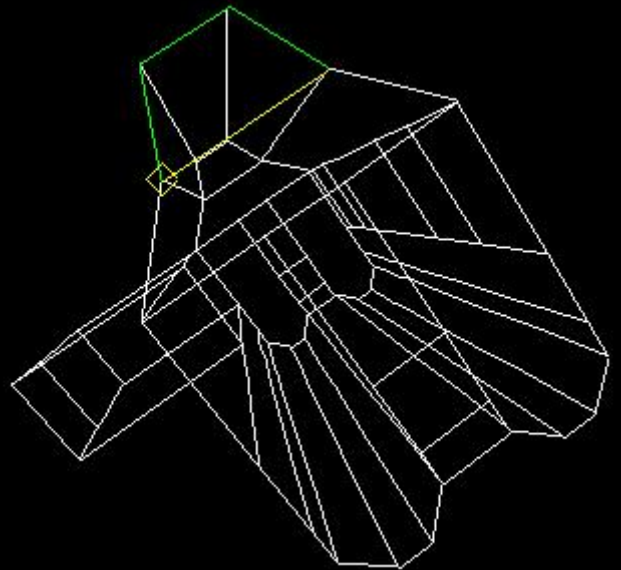
After extrusion you have to delete the seams. There aren't many, but you can switch to the textured view to make sure you haven't missed any.

Now we have a shell, and despite the fact that we have eleven different faces in three different directions, we have NO T-JOINTS AT ALL. This is the 'safe' way to perform complex operations using the Sirian method - in D3Edit 0.9 the only way at all. NOW that all shell faces have been created and all seam faces have been deleted from the interior NOW is the time to create the extra vertices remove so that these edges are fused on all these faces and your shell becomes airtight!

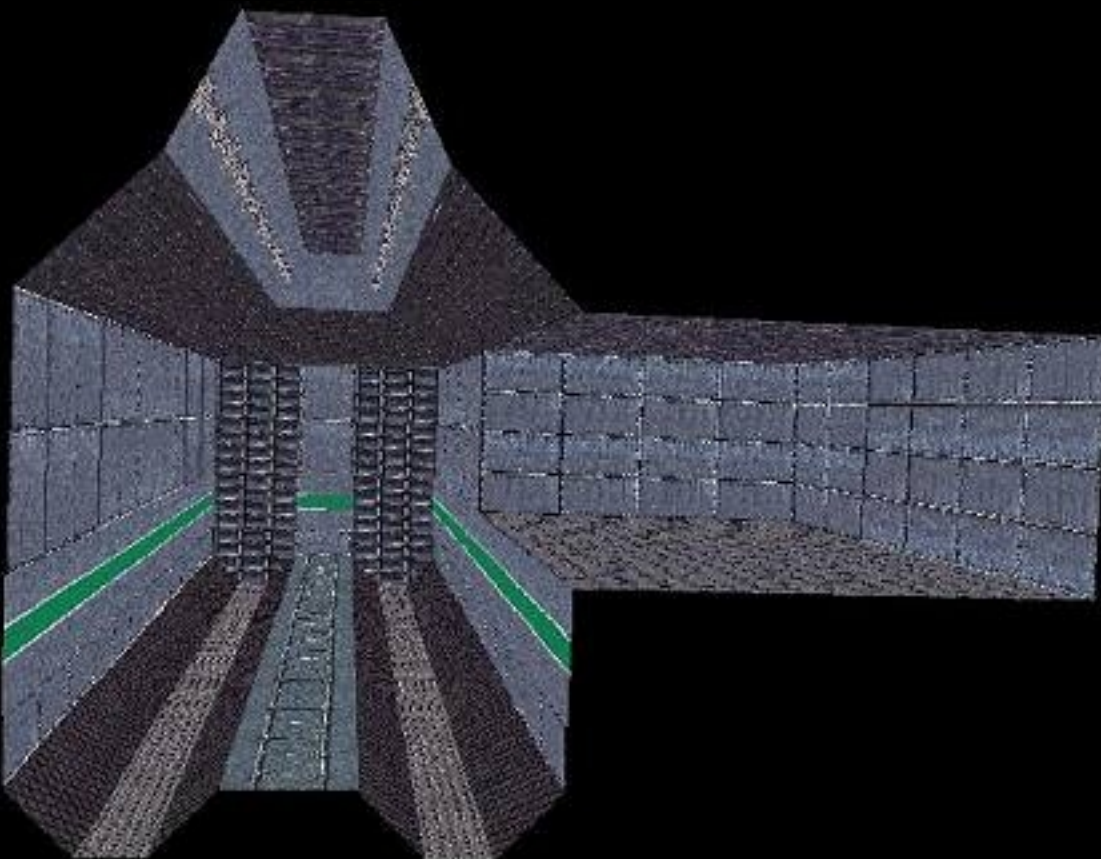


In fact, it is now necessary to merge the verts so that we can move on to the next step. And what is it? Choosing a texture scheme and then merging faces where possible to keep the polygon count for the room low and increase FPS.

The reason you should choose your texture scheme first is because it's a royal pain to have to split a face that you shouldn't have merged. Texture the room, then see which faces you can safely combine, paying attention to the textures as well as the portal borders. This is what I came up with after seven minutes of combining faces:



And in textured view:



Again, this is the 'safe' way to combine multiplanar extrusions, the only way to do it without creating T-joints.

030 - Share space

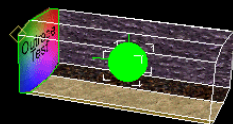
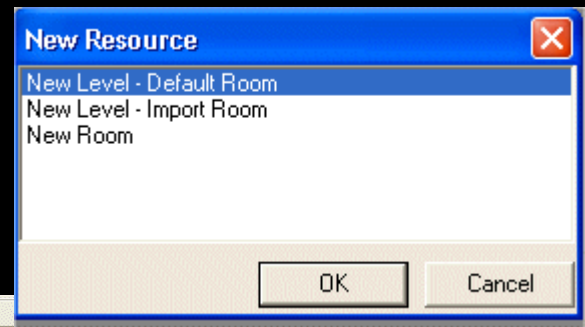
Atan

v1.0

Let's start the editor and create a room:

The result should look something like this:

We don't need to pay any further attention to the player object in the room.



(id): 0/8 (0/16) Grid: 10 0, 0, 0 0, 0, 0 Vertex mode Marked: V: 0 F: 0 Concavity Level: 0.0125

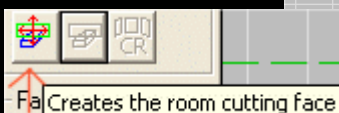
Another note:

This feature is not yet 100% polished and tested, so approach it carefully.

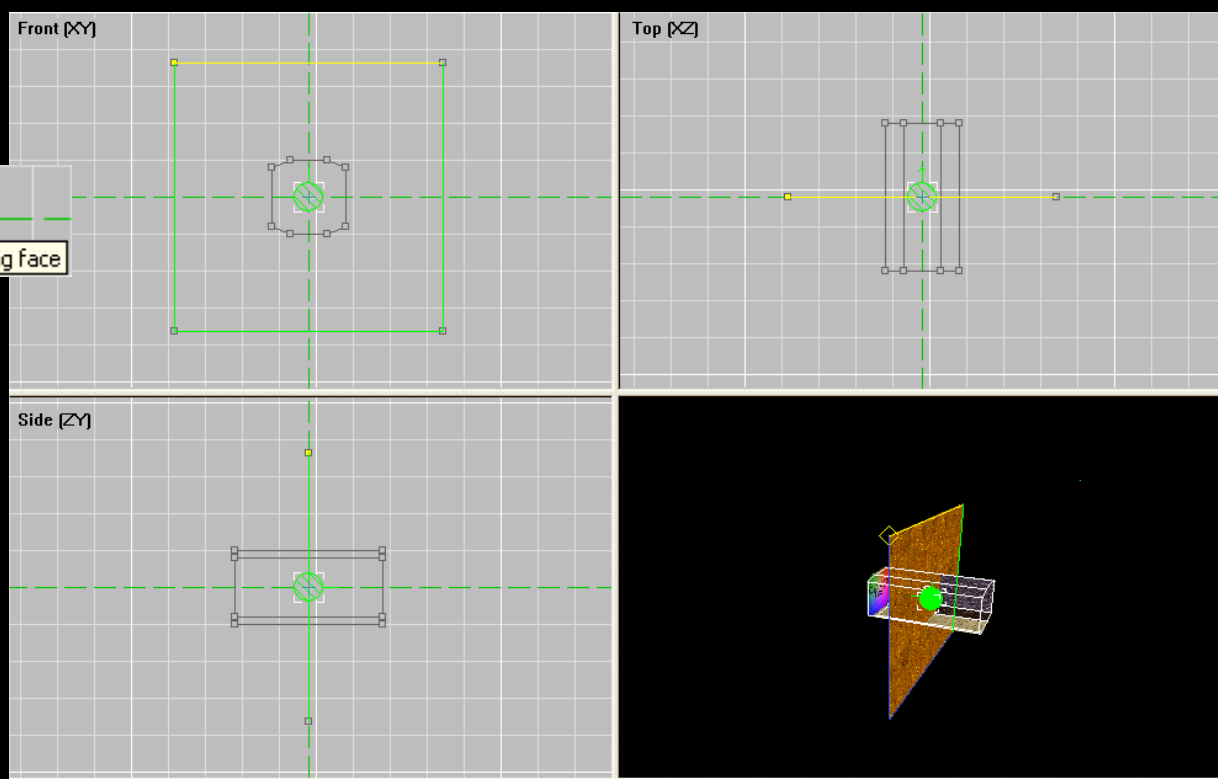
For complex rooms, even those with interior fittings, you should not have a face created, the results are not particularly great. It requires a lot of work, so try it yourself. Portals should also be used with caution treat.

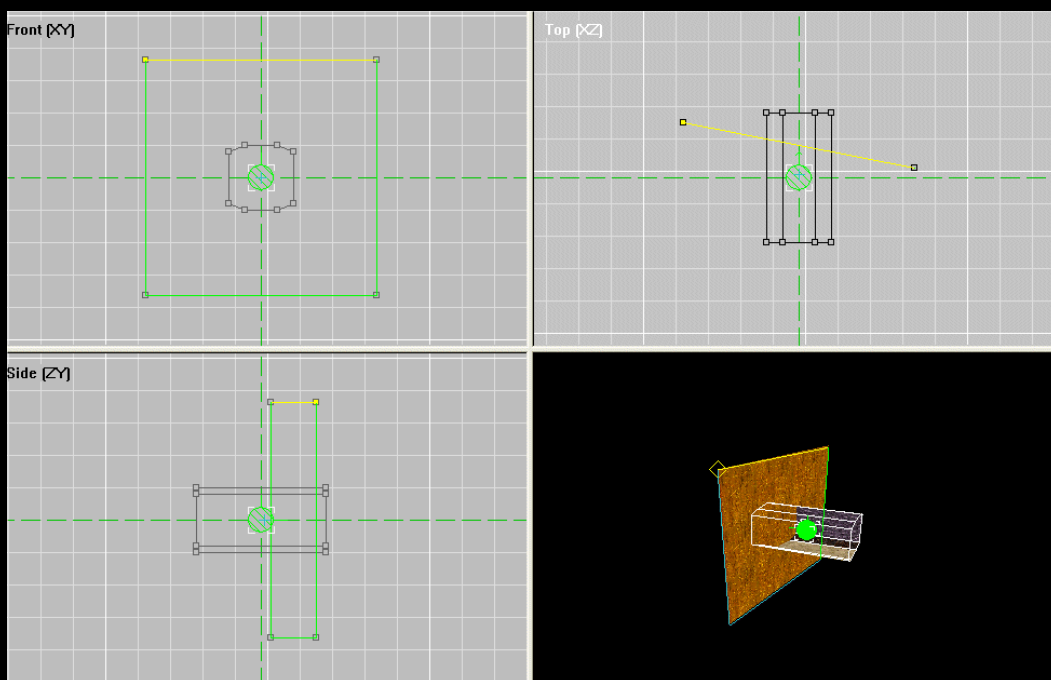
Process:

In the room
Press the bar
we these
Button:



This will
automatically
Cutting Face
(with the
Current texture)
matching
Space created.



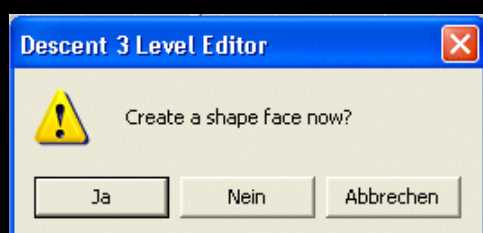


Positioning the Cutting Faces:

To do this, we use the when the grid is active corresponding Keyboard shortcuts that **Numpad**Keys. (e.g **2,4,6,8th, Ctrl+Alt+1** or **3**)

Cutting the space:

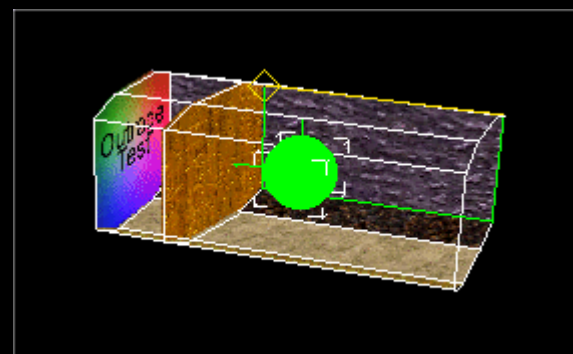
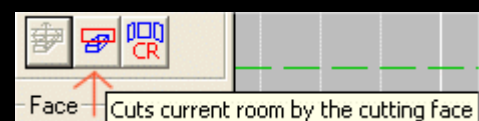
Drink in the Room Bar
we use this button:



A query will be made as to whether we are there want to insert a new face. If not desired, only the vertices will be added to the intersection points.

In complex rooms, inserting the face does not make sense, as a lot of rework would then be required.

We confirm with yes and get (right):



Now we could select all the required faces individually, mark them, copy them and paste them again as a new room.
But it works a little differently. To do this, we first mark the new face.

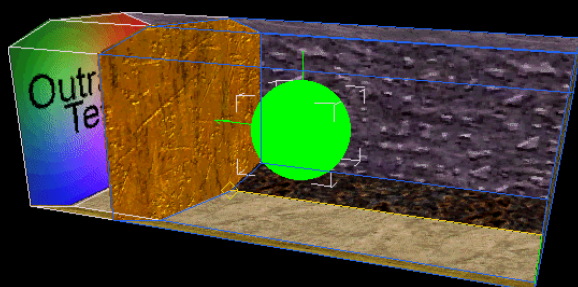
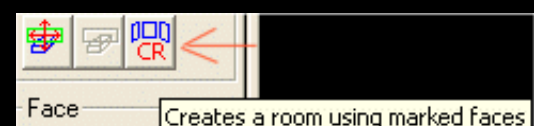
Then we use the 'Split marked Faces Vertices'-Function:

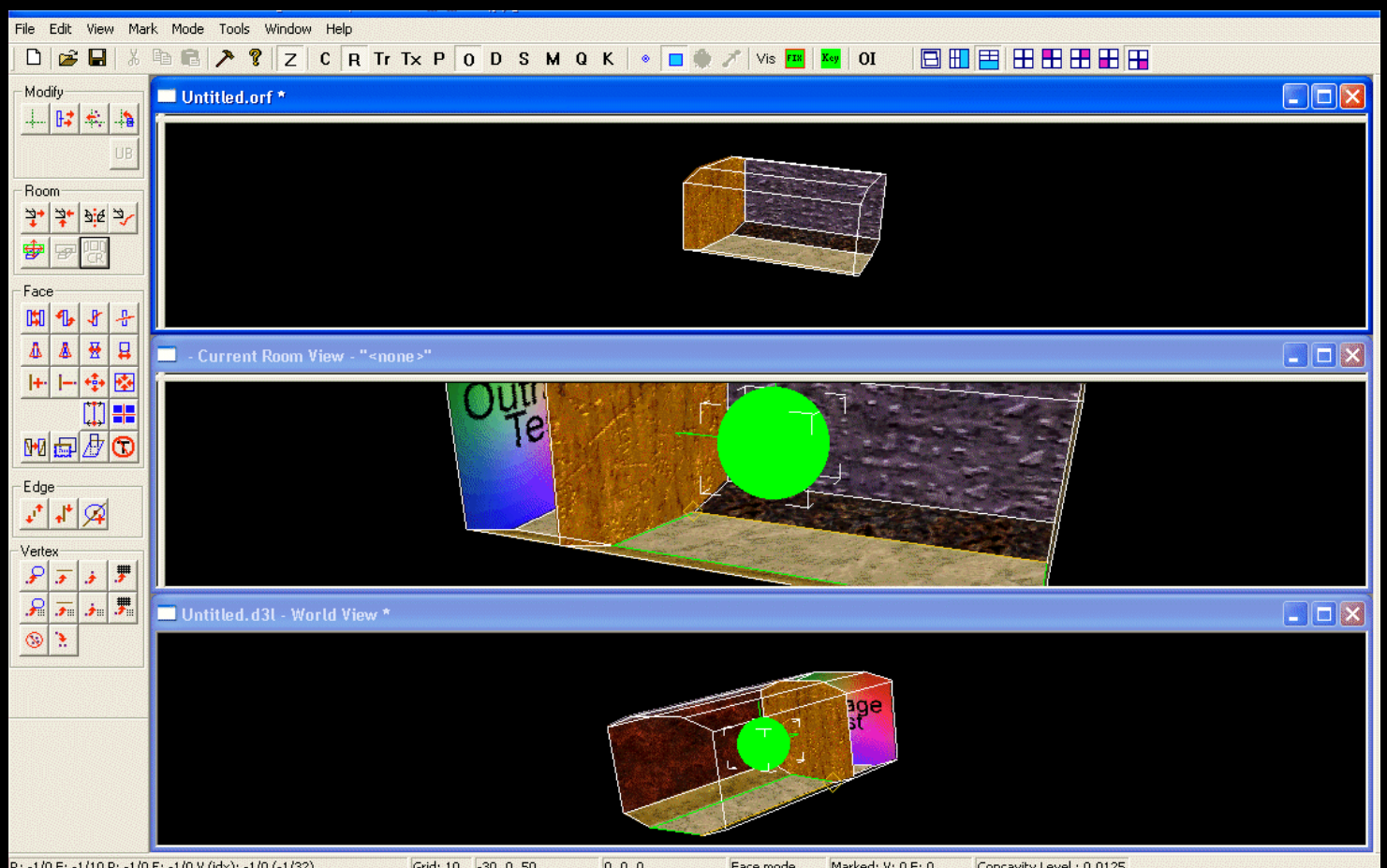
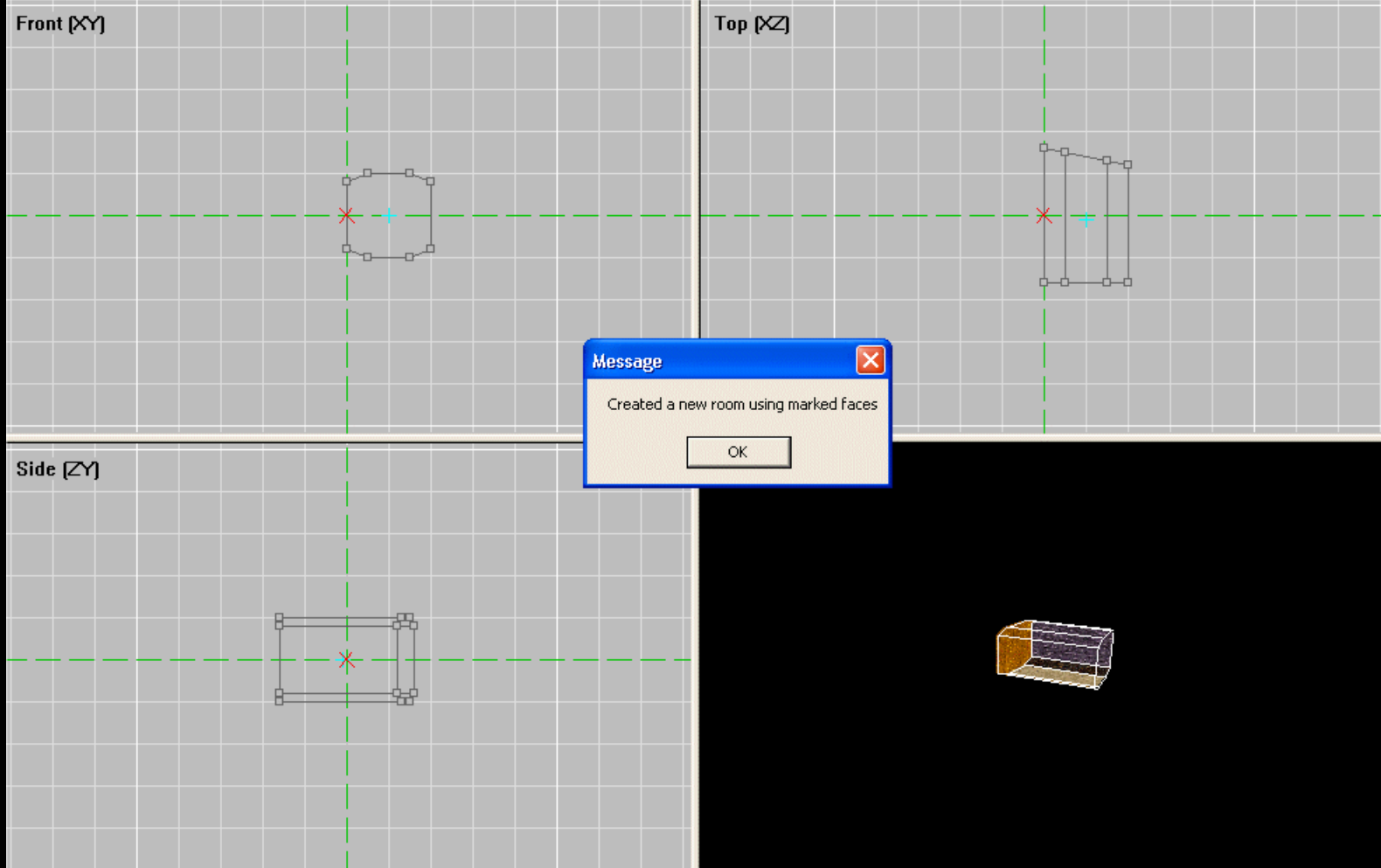


Then we have to select a Current Face within the desired area and Numpad **0** press. This means that all the required faces are marked in one go. (This way you can

Also very good at removing parts from existing structures and moving them, for example.)

Now press the CR button and we receive our new one
Space immediately in one own window:





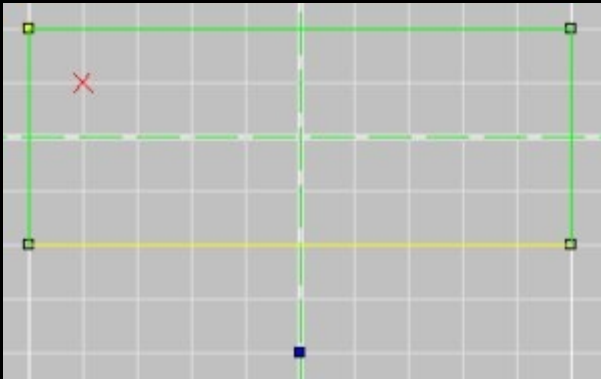
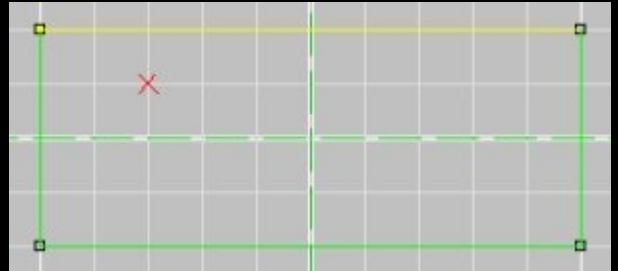
After separation, you can delete or move the copied part in the original.
Don't forget, you have to close the room shell again and delete duplicate vertices.

Back to Section C

031 - Vertex operations on faces

Hydra

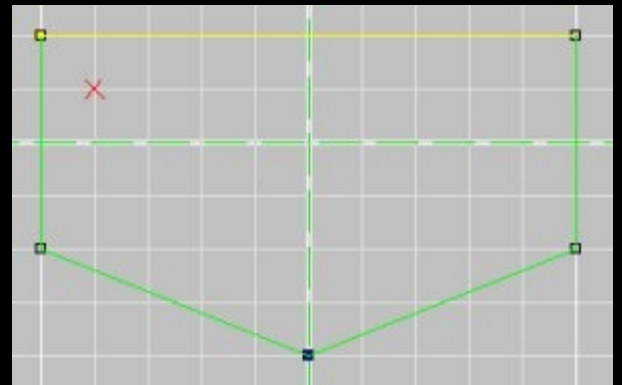
In some cases in D3Edit you will need to change the number of vertices of a face. This could be because you have a T-joint or just because you want an extra edge on that face, this is always a handy feature if you don't feel like redoing the face. On the right is an image with a face, I want to add an extravertex to this face.



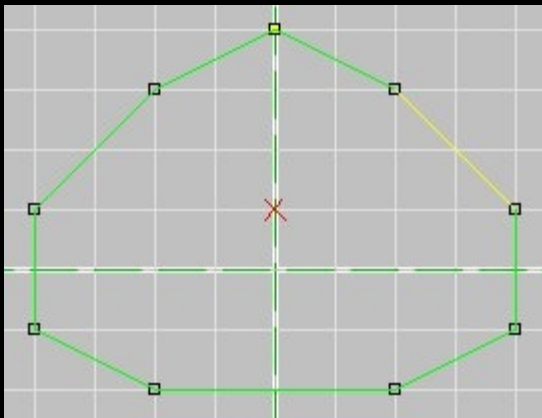
Now go into Vert Mode (Ctrl-R) and insert a vertex, moving it if necessary **wherever you want it on the face**. Then select the face where you want the vertex to go and select the edge (the yellow line in the outline of the selected face, use E to switch between them) of the face that you want to split. In the picture on the left I have selected the face and the edge that I plan to split. All I have to do now is press the 'Add Vert' button:



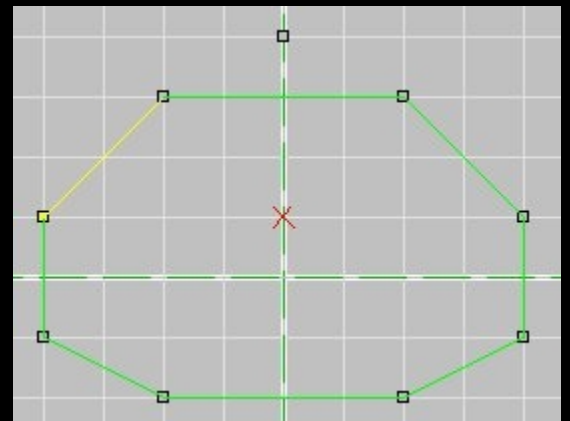
And you're done! Was that easy or what? Just don't forget to align the texture afterwards or it will be stretched out fun. This feature comes in very handy when trying to remove those devilish T-joints from the level.



A vert from one
Deleting Face is almost exactly the same. All you have to do is select the face whose vert you want to remove and the vertex you want to delete (go through the verts with the V key) and press the 'Remove Vert' button:



A confirmation dialog will pop up to make sure if you really want this, if you think everything fits click 'Yes' and the vert will be removed from the face. Don't forget to align the texture again. The vertex itself always remains untouched in case it is used by other faces, if you no longer want it, simply delete it.



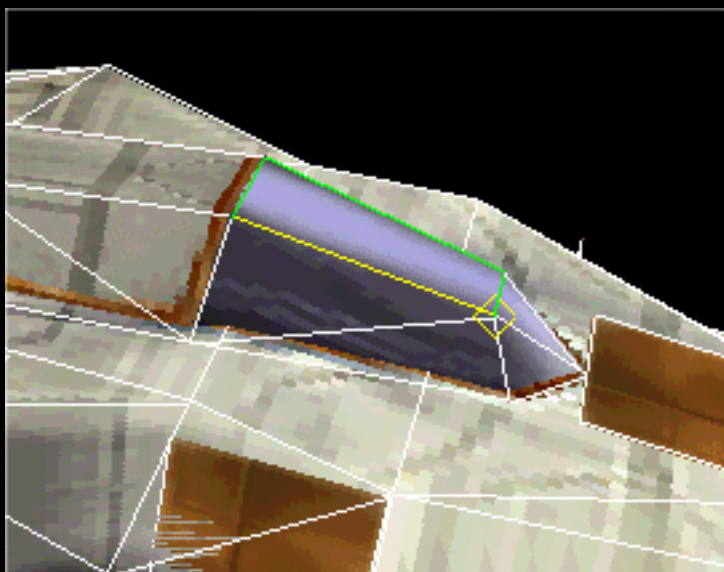
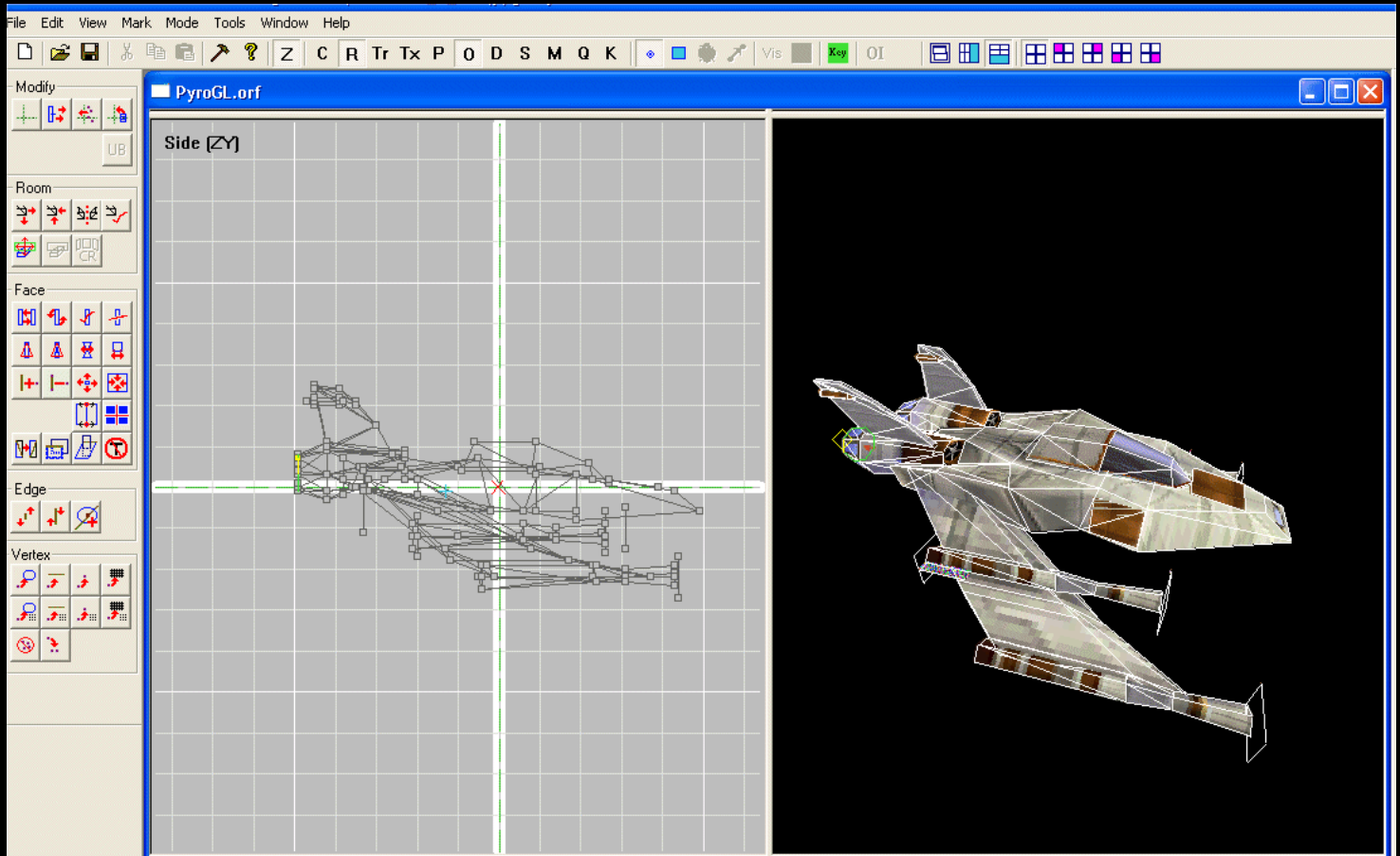
Back to Section C

032 - Select segments

(V1.0)

Atan

We have built a small spaceport into our level and want to park a few Pyros there. The idea: A Pyro should be placed with an open cockpit, for maintenance work etc. With the OOF editor we already have the Pyro from OOF into R converted and in D3Edit loaded. (If the OOF editor is not running, OOF2ORF also works).

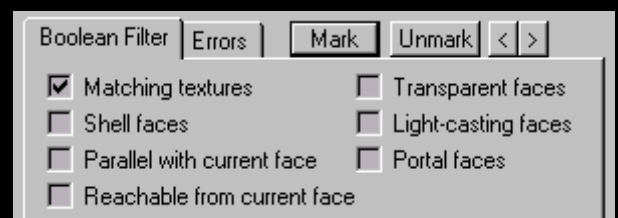


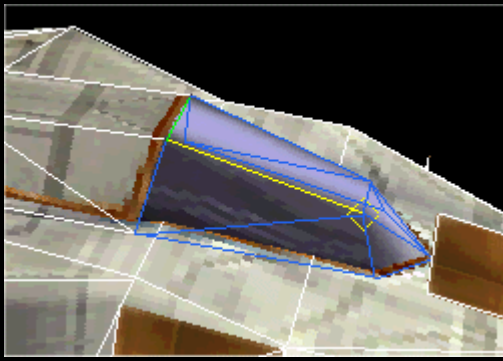
We'll call that **K**. Open the tool and select: Boolean Filter -> Matching Textures -> Mark:

With **tab** or **Ctrl-F** we switch to face mode via the tool bar:



We now select a cockpit face,

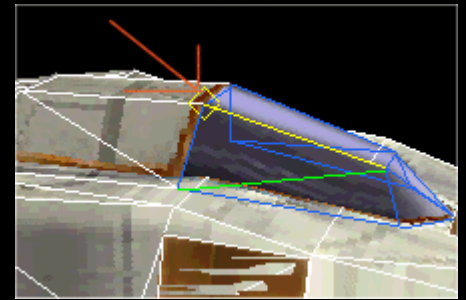




Now let's put this function to use:

The display doesn't change, but we have all the vertices of the marked faces from the

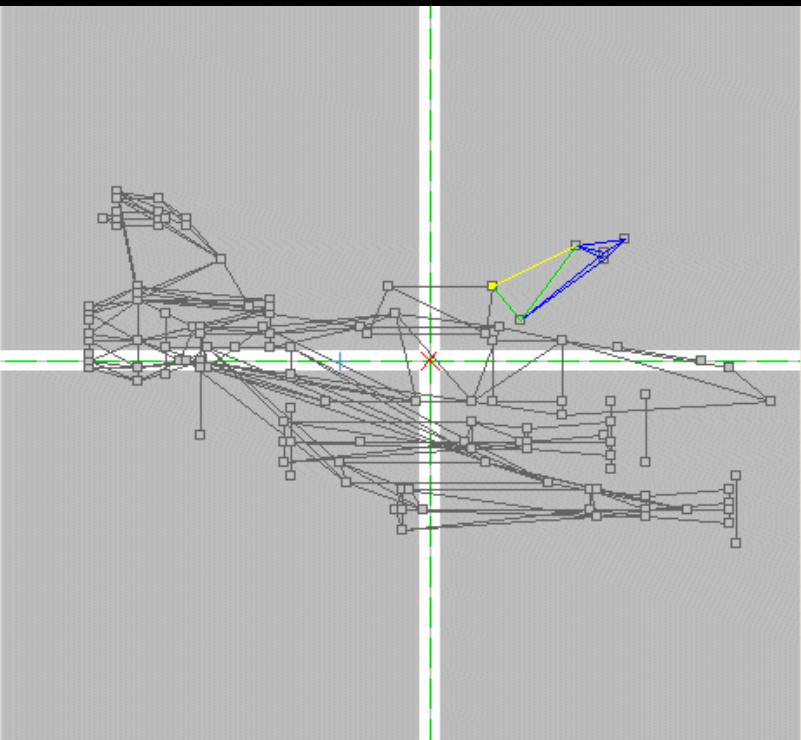
Structure solved. The faces have now been separated from the neighboring faces. Now we have to select the vertex around which we want to rotate the marked faces (right):



Ctrl-R Press to select vertex mode **g** go to the correct vertex

Ctrl-F Press to select face mode

With the side view and buttons activated **Ctrl+1** or **Ctrl+3** Let's rotate the faces into the desired position:



Maybe do a little interior work, add a ladder or simply close the holes.

Delete the unnecessary faces Glow, GP etc. if necessary, double vertices and our parked Pyro is ready.

The faces can also be quickly selected individually and then marked, allowing individual areas to be removed

However, processing will certainly be more complex than the method described above.

Back to Section C



033 - Bend – Basics

(LL)Dark

While this text is still valid, it refers to v39 and older.

A notice:

- It's best to only use the "top view window" (i.e. where the XZ axes can be seen)! There are always problems in the other windows.
- Always turn left or right - not up or down!

For simplicity's sake, instead of a space, I just used a line of verts. (the small **blue squares**)

In order to achieve an acceptable and reasonably predictable result, it is important to set the values for **Distance** and **Angle** to specify correctly.

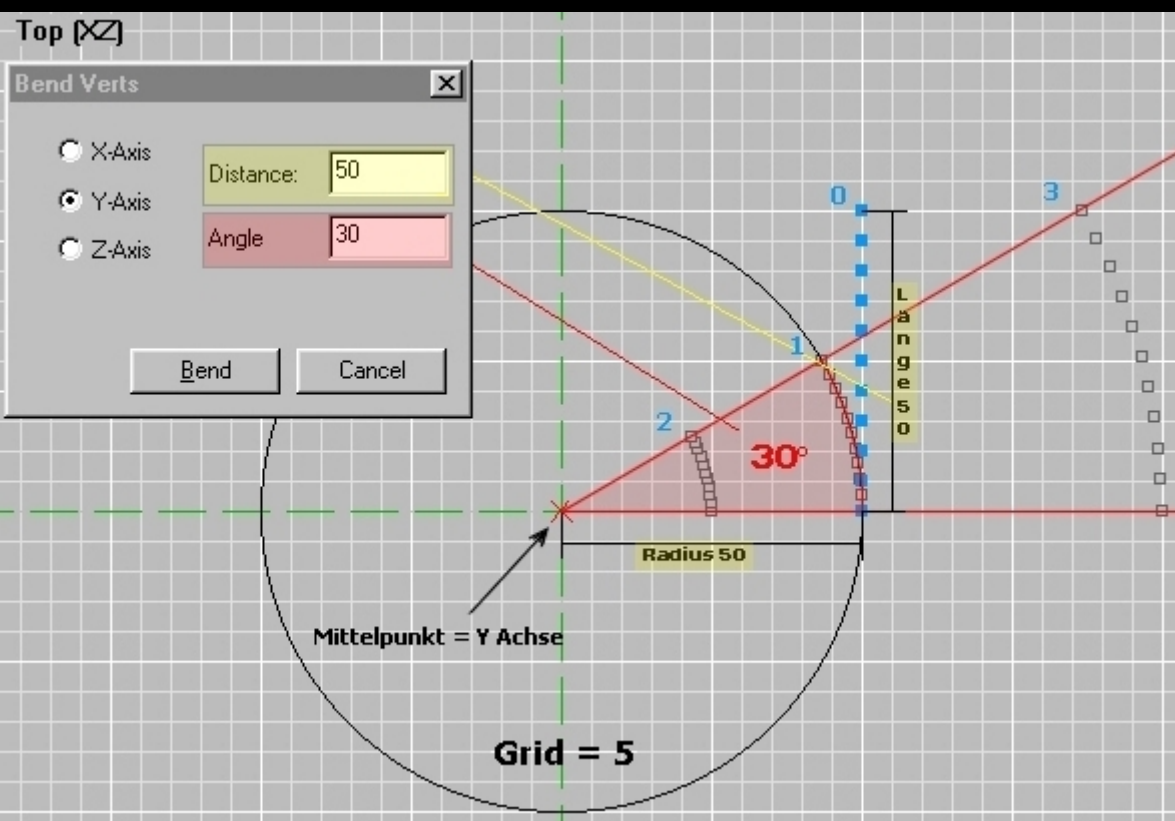
The **Distance** is the **radius** of an imaginary circle around the bend axis (here **Y**). It should be identical to that **length** of the room to be used. The values that belong to each other are shown in the picture **yellow** underlined.

Angle corresponds to the circular angle in degrees around which the space to be surrounded **Focus** should be bent. The values that belong to each other are shown in the picture **red** underlined.

The problem that most people have is that after bending, the length of the given space does not match the previously assumed dimensions. The final length depends on the distance (**radius**) the Verts to **Focus** away.

This can be clearly seen in the picture. **0** corresponds to the unspecified verts. All Verts (**1, 2** and **3**) were given with identical values (**50/30**).

However, each with a different **radius** from the **Focus** (the verts of **0** were moved to the appropriate positions). As a result, the **length** As the distance increases, the angle (**Angle**) always **30°** remains. If you move closer to it **Focus** approach, the verts are compressed accordingly. However, the curvature of the given space remains the same in all cases. (**30°**)



Is problematic however, the Prediction of the **Final length**. Is meant thus the route the ones in the picture **length** is designated. (basically taken so the height of the End of space over the z-axis - in the picture "Length" designated)

You give it **Angle** 90° so is this one **Final length**=the distance to **Focus** But this doesn't work at other angles!!!

It becomes even more problematic when you... **Distance** specifies a value that is smaller or larger than **length** of the room to be used. In that case the result will be **NOT!!!** the expected and set angle (**Angle**) are equivalent to.

Feel free to try it.

So in order to get reasonable results, you have to know beforehand how big should be the final curvature of the end space (= **Angle**) and you have to know how long the (uncurved) starting space is (= **Distance**).

To give the final space the desired one **length** (because you want to establish a connection to an existing neighboring room, for example) you "just" have to move the starting room to the correct distance from the bending axis (**Focus**) bring.

This distance can be calculated using the following formula

$$c = a / \sin(\alpha)$$

... because distance to the center = c = hypotenuse of the right-angled triangle, which at the same time represents the radius around point A (here the origin of the rotation) through point B. (the shorter leg is always a, point A and angle alpha are always opposite the respective side a)

- **c** is the desired distance from **>Focus<**
- **a** is the desired end length of the room (height of the given room end above the z-axis - labeled "length" in the picture)
- **alpha** is the angle (the red circle section in the picture)

in the picture for example **1** So: Distance to center = $25 / \sin(30^\circ) = 50$

The desired final length was 25. The distance to **Focus** must therefore be 50. Look at the picture - the values are correct.

Many thanks to Floyd for finding this formula for me.

To for a certain distance from **Focus** To calculate the final length of the room, you only need to change the formula a little.

In this case it would be: $c * \sin(\alpha) = a$

If all this was too much calculation and formulary for you, then at least make sure to enter the CORRECT values for Angle and Distance.

Then bending works too... 😊

(LL)Dark wishes you a lot of fun bending

Back to Section C

034 - New Style Bend, v39

Ragil Ral

For Bend with the v40 please read {no} - {bendv40} as the Bend function has been completely reworked.

Since v38 there is also the option to exit with the 'New Style' dialog. To switch on the New Style go to Settings->Bend->put the hook there New style set. The new Bend dialog looks like this:

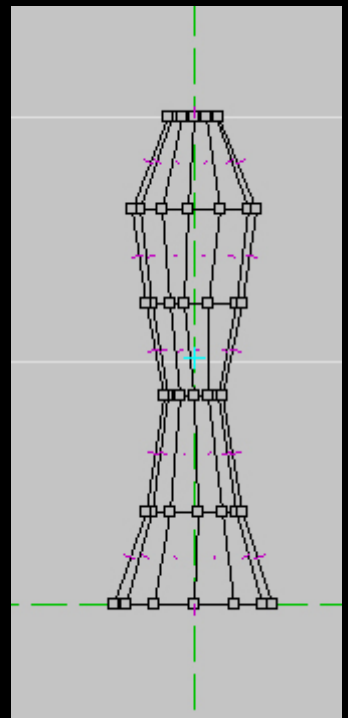


But what actually makes this thing different?

It takes away the calculations that Dark did earlier, at least for certain situations.

On the one hand, the length of the marked verts is determined, and on the other hand, the 'Width' (always depends on the 2d view used) averaged an (imaginary) center line drawn in, which is used for the length.

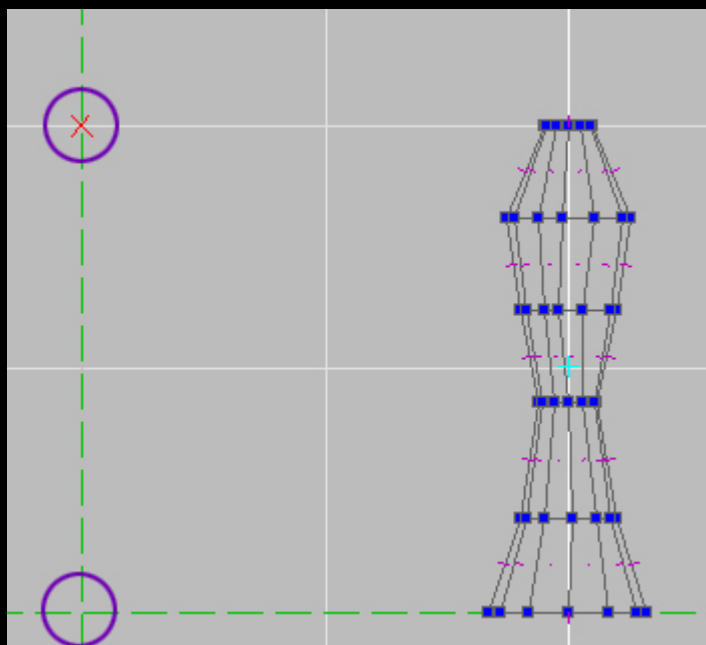
The construct on the right is given:



It is exactly 200 units high and has a diameter between approximately 25 and 66 units, but is centric.

The axis goes exactly through the middle (the center of reference is still up 0,0,0), I moved a few verts so you can see them. Bend will refer to this axis when referring to 'length'.

To use Bend, one of the 2d views must be activated; the example takes place in the front view. Furthermore, vertices must be marked (nona).



Bending is always counterclockwise. If you want to go in the other direction, switch to Distance the sign.

If you now call up the Bend dialog, the reference point and the cursor are moved to the points/values that the Bend dialog shows (circled in purple):

The figure is 200 high, so the center of reference is moved 200 and the cursor is placed on the target where the bend will land. You change Distance, the cursor moves to the corresponding target point.



The Bend dialog opens by default with the setting to bend 90° to the left and the final length...

we'll get to that later.

The dialogue on the left belongs to the figure from before:

As you can see, the axes can no longer be selected; they are determined by the view that was active when you called Bend.

Distance and Angle: They still have the same function as they had in the classic Bend. Only, Distance is: Here you determine the final length of your result,

where:

Ref Zero to Height causes the reference point or reference axis for bending to be set in such a way that the end product has the length (see above) that has the marking. If you take an angle of 180° or more, the...

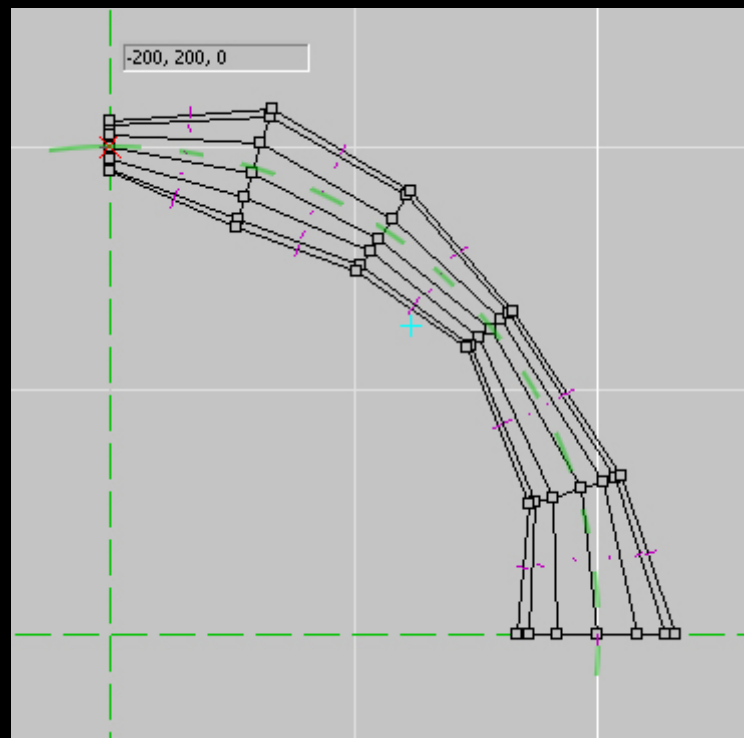
Length is simply aligned in the other direction (only in the front view).

The result looks like this:

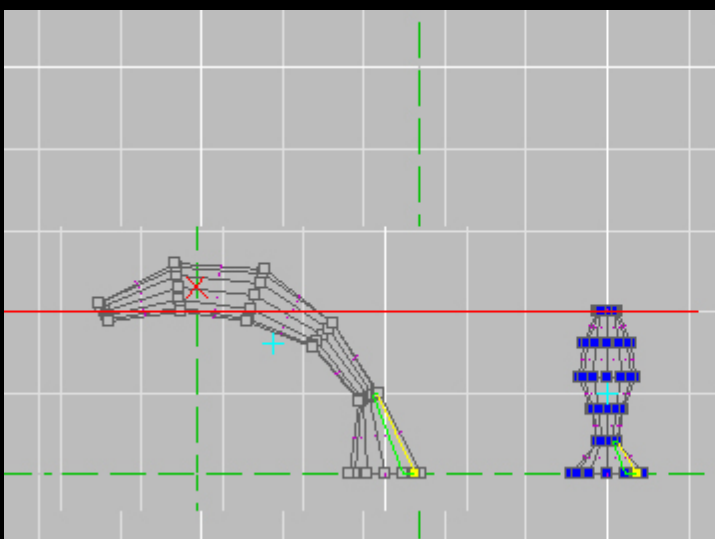
I drew the old axis as a green dashed circle and inserted the position of the cursor.

If you specify 62.185° as the angle, the piece will have exactly this curvature and the (imaginary) central axis will be the length of the original marking.

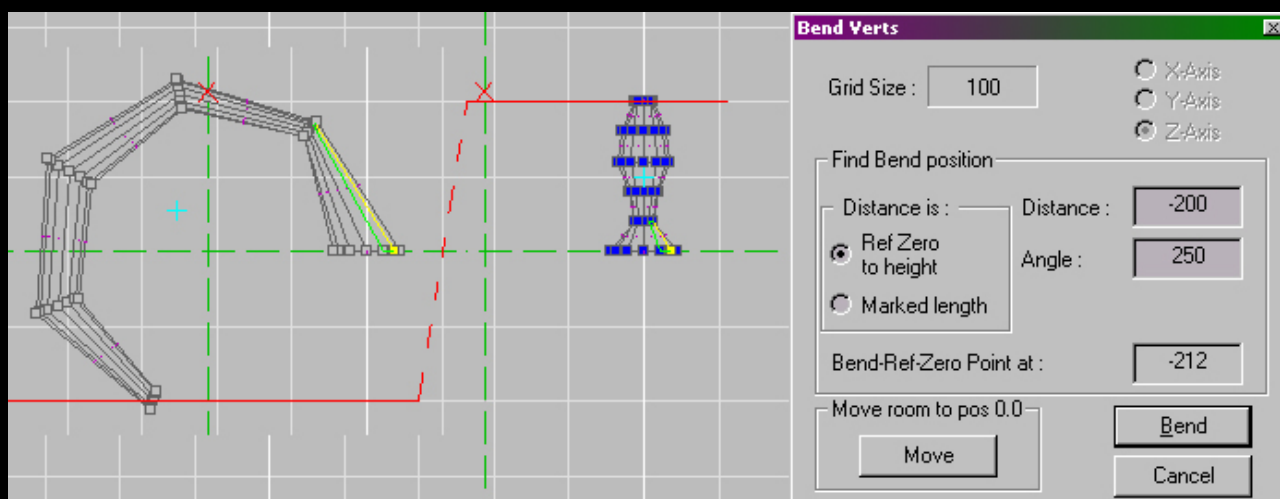
But if you want the outside or inside of the given verts to reach a certain length, you have to take the other dimension (width or diameter) into account and add this Distance factor in; Here you have to stick to Dark's comments again and adjust the values in the dialog box accordingly.



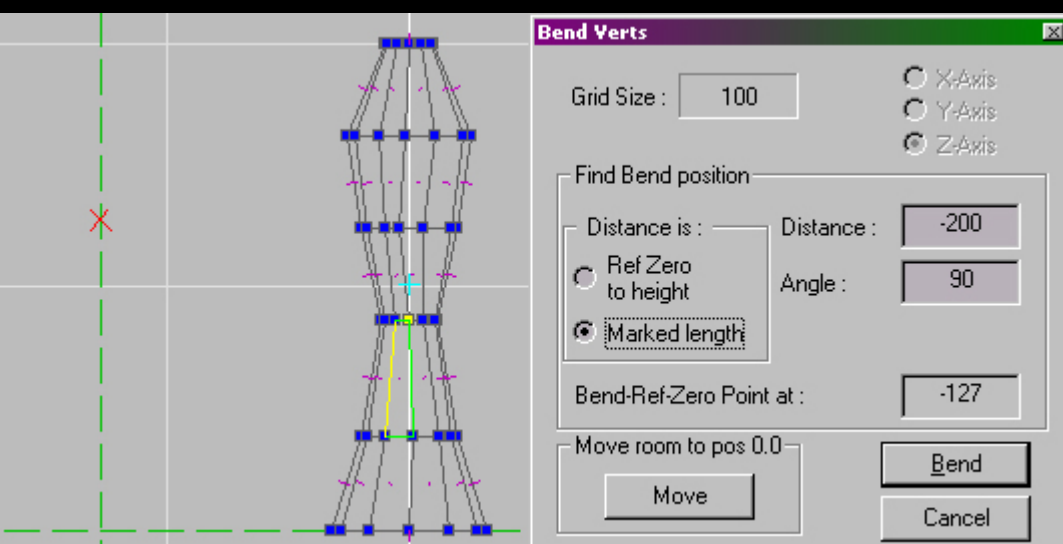
Angle greater than 90°: The red line represents Distance.



Angle greater than 180°: The red line represents again Distance, the 'folding' is indicated by the dashed line.



Marked length causes the length of the given object itself not to change, but rather that



it's just about that specified angle is bent (left):

Of course, this also means that the larger the angle, the smaller the construction becomes purr together.

Note Bend ref Zero Point at: he's right now no longer matches Distance.

The result:

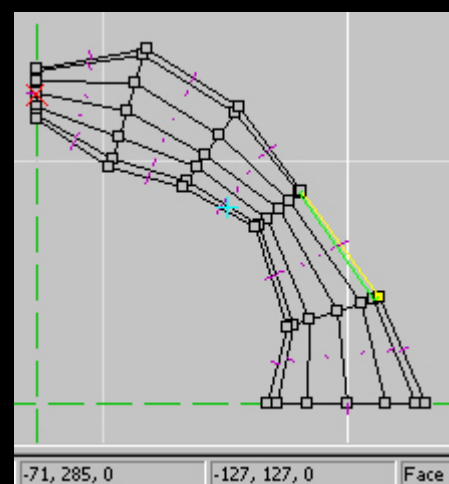
You can also see the position of the cursor at the bottom right of the image.

If you do the math, you get the effective length of the (imaginary) central axis to be 200.27 units, which is because New Style Bend works with integer values.

Furthermore, the 'offset' of the object from the coordinate origin is relevant; in this example, the center line of the object was congruent with the origin axis. If it is not exactly on it, the offset will be doubled Distance opened if it is positive.

If you take other values for Distance, then the achieved one corresponds Angle is no longer exactly what you would expect, I'm still experimenting here myself... but take the new editor version, the v40, stick to chapter 036 - Redesigned bend function in v40 and you'll save yourself a lot of experimenting.

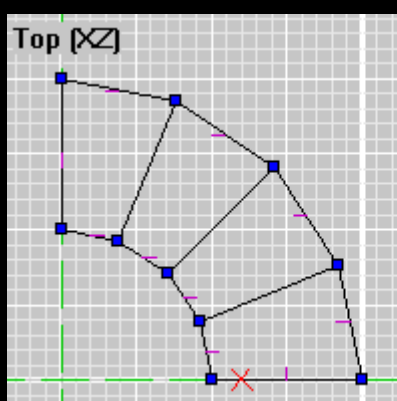
Back to Section C



035 - The Bend function <>

Schplurg

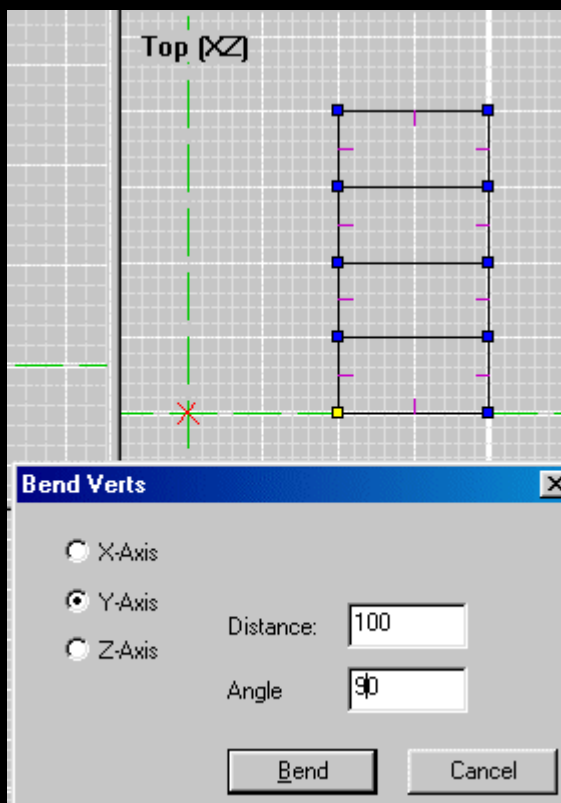
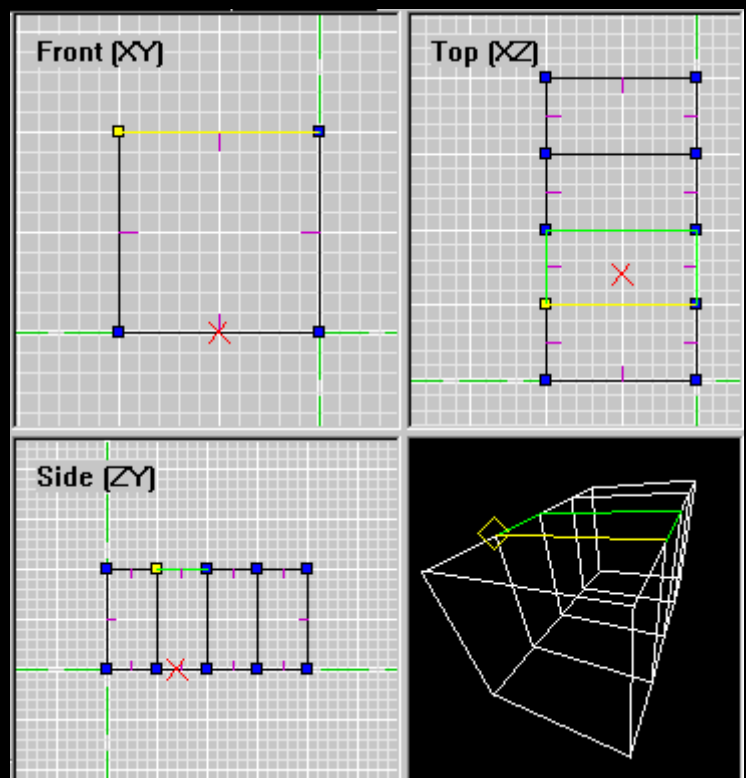
Perplexed? Please refer...




I recommend following these steps and using the exact specifications provided in this tutorial when building your tunnel. Once you're familiar with how this works, later experimentation will produce some very interesting results.

Our finished tunnel is bent 90 degrees to the left when you look at it from above like in the picture on the left.

Start by building a tunnel that is 100 units long and 50 units wide and 50 units high. To do this, start with a 50x50 face and extrude the tunnel along the Z axis in 4 segments, each 25 units long (right). The segments are necessary because the tunnel will bend with these connections.



Go to the **Vert Mode** and mark all verts. We will only focus our attention on that **Top view (XZ)** because we can see our bend best there. Place the reference point 50 units to the left and flush with the front surface. Use **Ctrl-click** to set the reference frame (the point where the two green lines intersect). This is the central point around which the tunnel will curve.

You should save the room at this point, although there is an un-bend, but better safe than sorry. Open the Bend Dialogue about the button in the geometry bar:  There is one Note on the justification for setting the reference point as we did (the green lines).

The values in the Bend Verts dialog can get quite tricky. I'll describe them briefly; At the end of this exercise, I'll discuss these features in more depth.

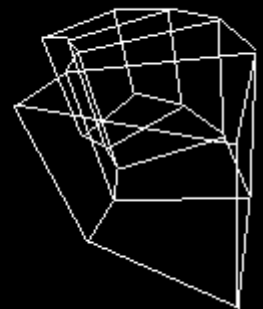
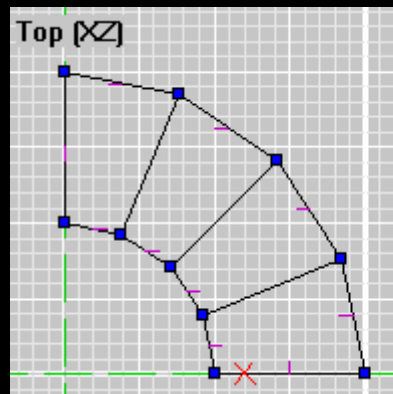
Axis is the axis pole around which the bend winds. In the previous image, X is horizontal and Z is vertical. You are therefore looking at the tunnel exactly from above along the Y-axis. This is the pole around which our tunnel will bend - the Y axis.

Distance should be set to the length of the section to be bent. Our tunnel is 100 units long... enter 100 here. Negative values can also be entered for other results.

Angle is the amount by which you want to bend the tunnel. For now use the safe and straight 90 degrees. :O)

Ok now, attention... take a deep breath and... (you saved it?) click Bend!

You should see something like this now. If it didn't work, relax, stretch, and then do the exercise again.



How it works:

There are some important keys when bending:

The Reference frame is the axis for the bend, like the hub of a wheel.

Distance is a big variable. Try the exercise above, but set the reference frame **right side** of the tunnel instead of the left. Enter the same values into the Bend dialog and see what happens... it will bend to the right, but also inside out!!! Now do the same thing, but this time set the distance **-200**... hey, it worked!

Angle also gives different results if you change the value.

Play with the settings using this simple tunnel and you've conquered a seemingly complicated aspect of D3Edit.

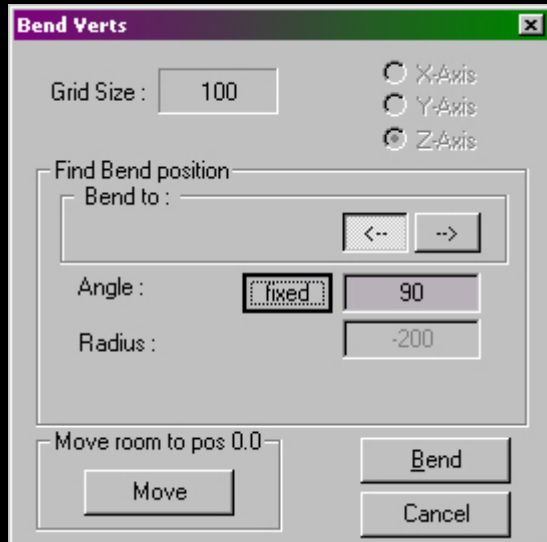
annotation: the following tut provides important information about the bend function and texture alignment on stretched surfaces. It is highly recommended that you work through them!

Back to Section C

036 - Redesigned Bend function in v40

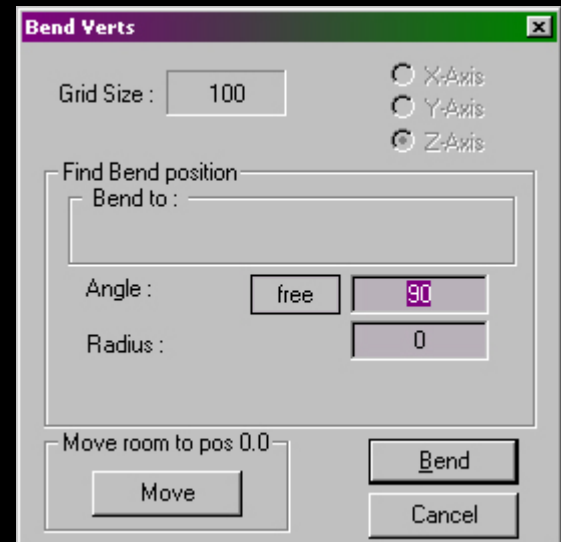
Ragil Ral

In the v40 the bend has been completely revised. There is no longer the option to end with 'old' or 'new' style, just a single dialog box. But Bend has now become simpler and more accessible. The dialogue now has two faces:



As with the new Style Bend, the axis selection is based on the view you were in when the function was called.

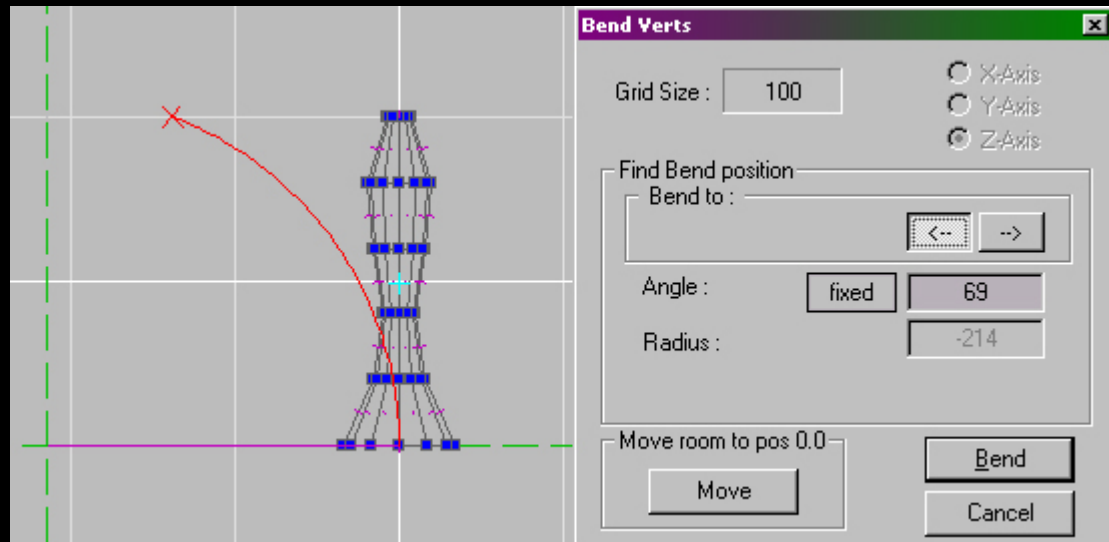
Note the button to the left of Angle.
This determines the bend mode.



Bend with fixed

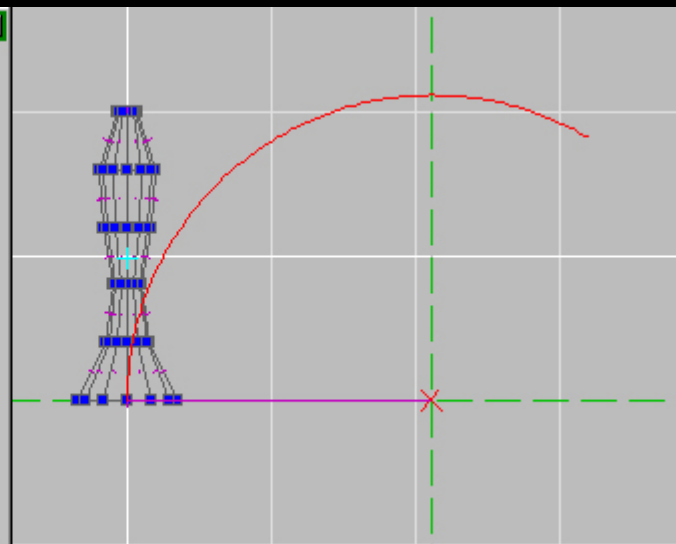
This setting behaves largely like new Style Bend. What is new, however, are the buttons with the arrows where you can specify the bending direction. As a special new treat, Atan has introduced something that you will soon appreciate, namely a preview of where the bend will land!

The red line shows you this. with the setting **fixed** the radius is calculated, angle can only be specified up to 90°. The Length of bend at (ahem) is always the marked length.



Bend with free

Now it may be the case that you need different angles and/or a different radius. In this case click on the button next to Angle so that **free** is inside.



Now you can
Follow her
heart's desire
and mood
Values
set:
angle
between 0
and 360,
radius
any.
Noticed
but that

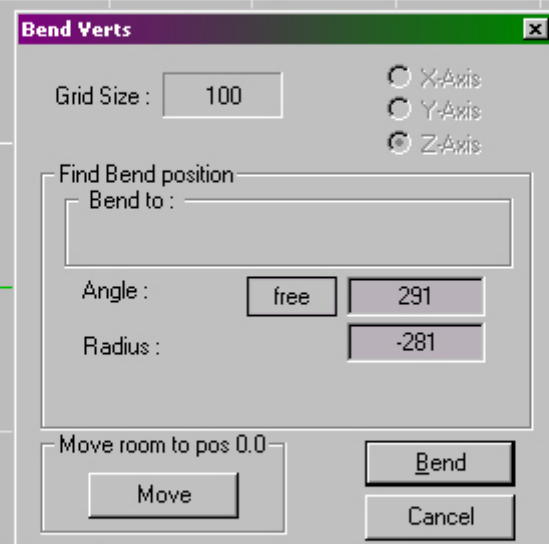
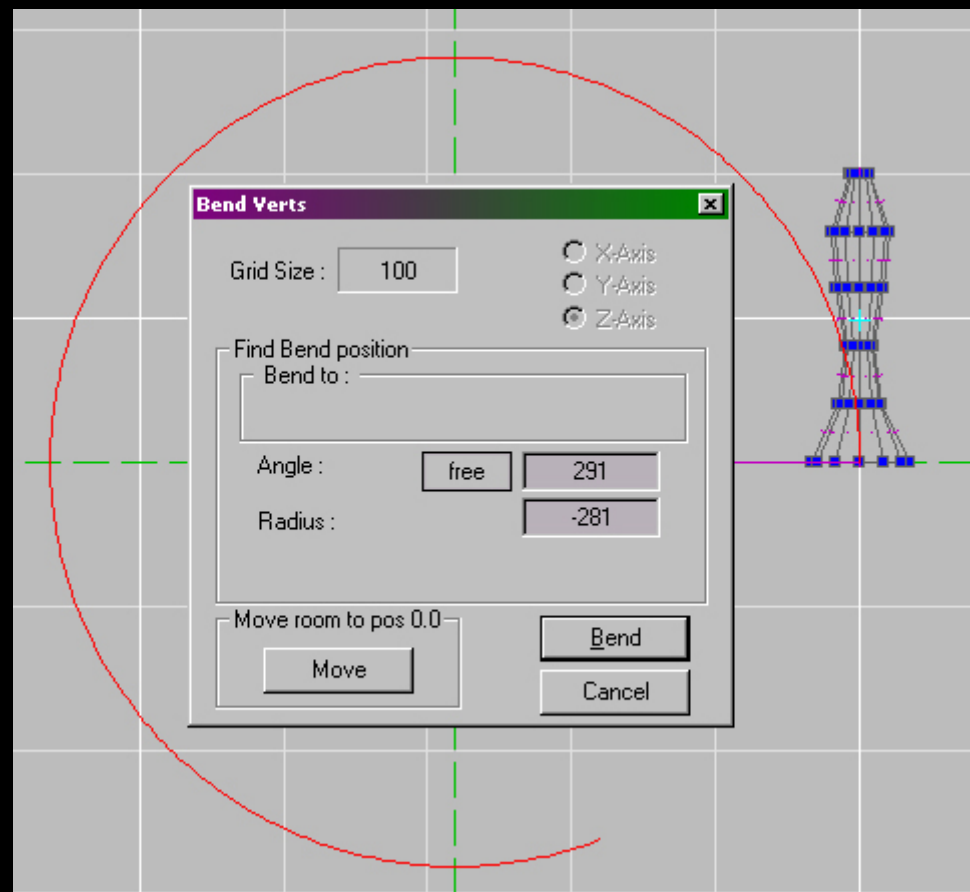
The length then changes accordingly, here you either stick to Dark's and Papacat's explanations about the mathematics behind bend to calculate the desired value, or you play around with the settings until the preview indicator ends up where you imagine it to be.

Basically you bend to the right, you change the direction again using the sign at radius.

Move room to pos 0.0

For both bend variants, **free** and **fixed**, this button causes the marked verts to be moved to position 0.0 (in the corresponding 2d view, of course), with the 'bottom' of the marked verts landing exactly on the horizontal axis and the angle to standard (Angle 90) is set. And if so

unlikely
didn't turn out as expected, there is still the UnBend button



Well, happy bending then!

Back to Section C

037 - Bend: Mathematics

Papacat

First, the two rules you have to work with in D3Edit:

1. Objects (rooms, columns, fortifications...) rotate with an increment of 11.25 degrees (360 degrees divided by 32) when you press 1 or 3 on the numpad.
(in newer editor versions this increment depends on the grid size. The increment listed here applies to grid size 10)
2. The Texture Alignment tool rotates textures by 2.8125 degree increments... that means 32 clicks will rotate the texture by 90 degrees.
(These increments also depend on the grid size set, but can be entered manually in the Align dialog.)

It's easy to remember: Objects rotate - start with 90... divided by 2 = 45... divided by 2 = 22.5... divided by 2 is 11.25. Use these increments or a combined sum of them to align objects or place them in proportion in space (See also [Increment list](#) in the Reisberg section.).

For textures, go two steps further... 11.25 divided by 2 = 5.625 divided by 2 = 2.8125. Use these increments or a combined sum of them when bending or rotating something to ensure that you can align the textures (especially floors and ceilings in tunnels).

Now first a refresher course in 7th level mathematics. After that, we'll go through the steps of bending a tunnel.

For now the obvious. 360 degrees in a circle. Here is a top view from D3Edit.

When you finish, you start in the first quadrant (QUAD 1). The object will bend/move counterclockwise around the axis. This will help you figure out what value you put in Angle you have to set in the Bend dialog.

Secondly, three formulas to find out how far you have to set the tunnel from the axis:

$\text{Circumference} = \text{Diameter} \times \pi$

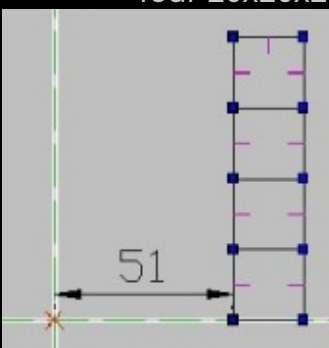
$\text{Pi Radius} = \text{Diameter} / 2$

The ratio of the tunnel length to the circumference is equal to the ratio of the bend angle to 360 degrees.

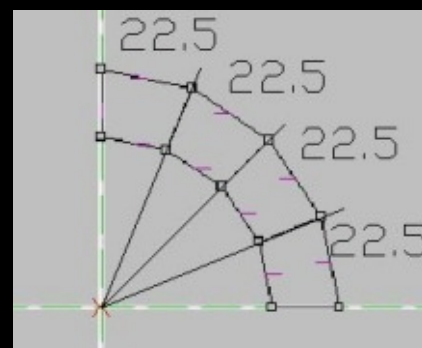
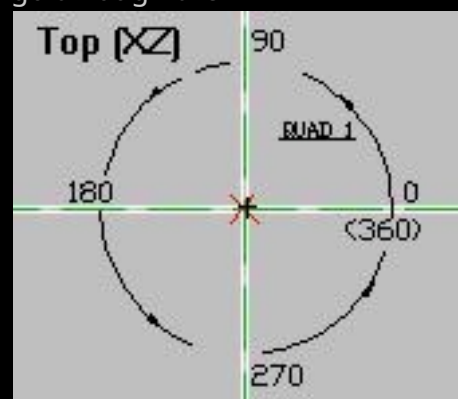
$\text{Tunnel length} / \text{circumference} = \text{bend angle} / 360$

The third helps to keep a certain part of the tunnel 'almost' the same size before you bend it. You can actually bend from any distance from the axis, at any angle, and D3Edit will squeeze or stretch the segments to fit the angle.

Let's make a tunnel and see what happens. We will bend an 80-unit tunnel made of four 20x20x20 segments at 90°.



I want to know if the segmented ceilings allow texture alignment before I waste time building. $90/4=22.5$... that's good for textures. It's also good for rotating, so I can rotate the tunnel into place if necessary (right).

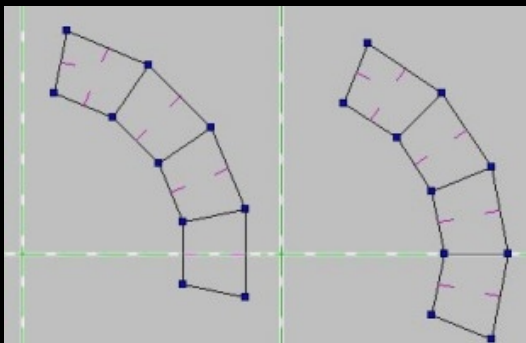
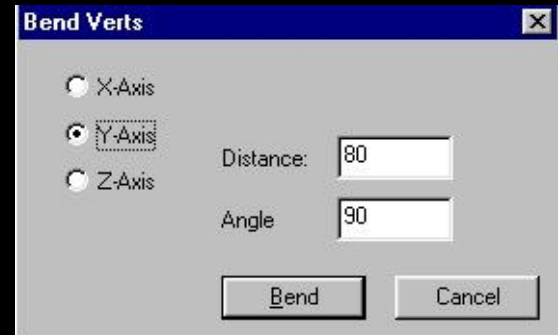


Now... I want the inner wall to stay 'almost' the same size, so how far away from the axis do I put it?

Figure out what part of the perimeter the tunnel will need. ($360^\circ/90=4$)
If the tunnel is 80 units long, what is the perimeter? 80 units x 4=320 units Now find the radius. 320 units / Pi = 101.9 diameter, 101.9 / 2 = 50.95 units radius.

Since the smallest unit in D3 is 1, we set the inner edge 51 units from the axis. Open the Bend dialog and set the variable as in the picture. Click **Bend**.

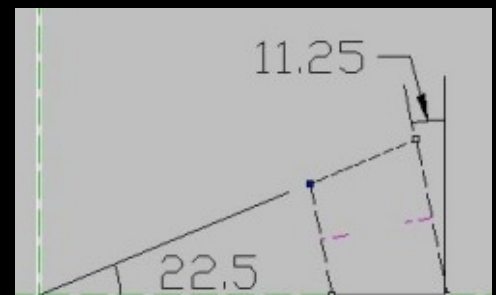
Since all the textures were either stretched or compressed when we created the tunnel, you must **Align** and set all textures to 'Default UV's'. But since we've made sure the segment angles are met, we have no problem rotating the ceiling and floor textures until they fit the walls.



Now watch what happens when you rotate the tunnel. Highlight all the verts and then press the **3** on the numpad once. The tunnel rotates 11.25° clockwise and the segment now sits across the X-axis. Rotate again and the seam between the segments is exactly on the X axis. Keep rotating and see how it behaves. This is great if you want to always put objects like brackets or columns in the same segment point, all oriented the same way... like all facing the center, or along the curve of the tunnel. Just place the object there

wherever you want it, then copy it. mark everything, rotate, and use Edit->Paste on top. keep rotating, Paste on top, Keep rotating...until you're done.

One final point. Remember what I said about the textures being at 2.8125°? That means whatever you want to align with. In the case of our tunnel, these are the inner and outer walls. If the seams are at 22.5° with our tunnel, the outer walls are at 11.25°... just cut them in half (picture on the right).



Take five minutes to learn and one to use and you'll become a master of bend and lathe tools.

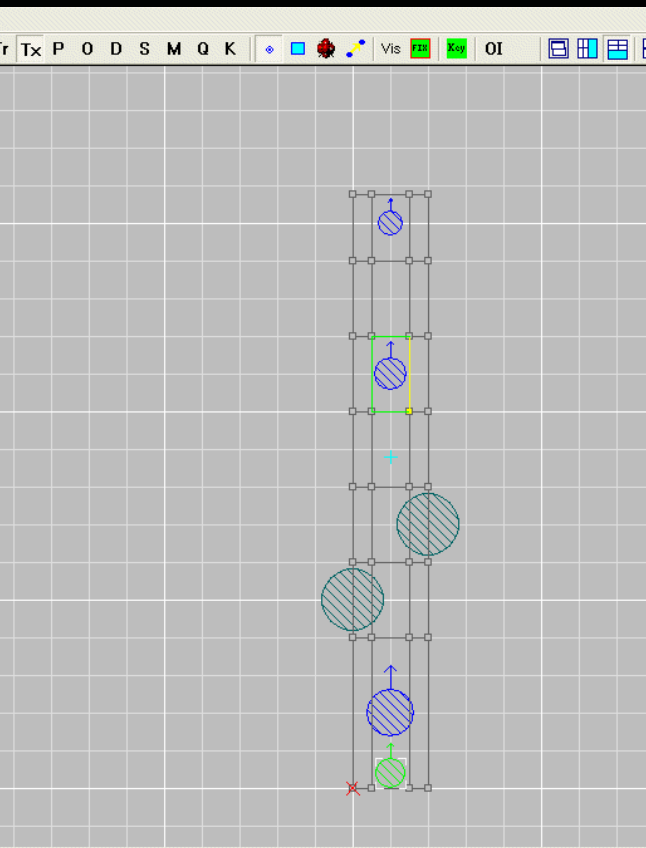
Back to Section C

038 - Bend, objects included

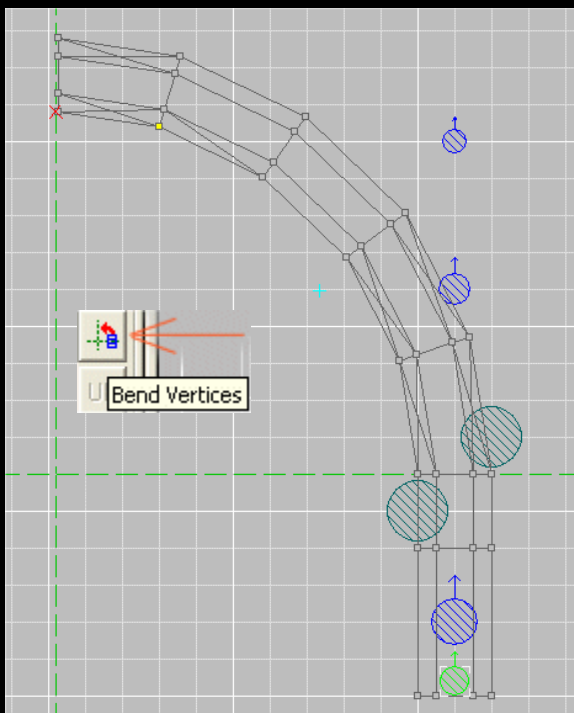
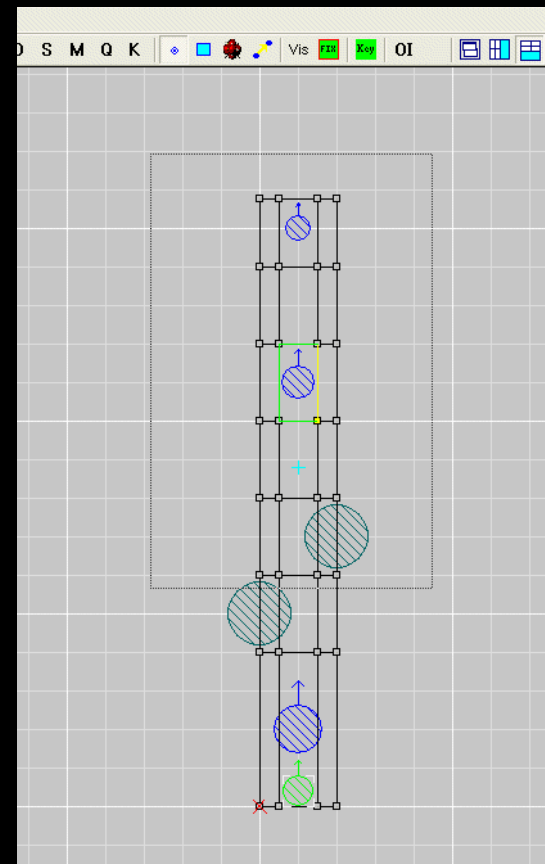
Atan

(V1.0)

itself. Either you have the twist forever a mystery. But it is still unknown that you can move objects immediately and with a little rework you will get a usable result.



Preferences. Unfortunately u see the objects all stay on their own

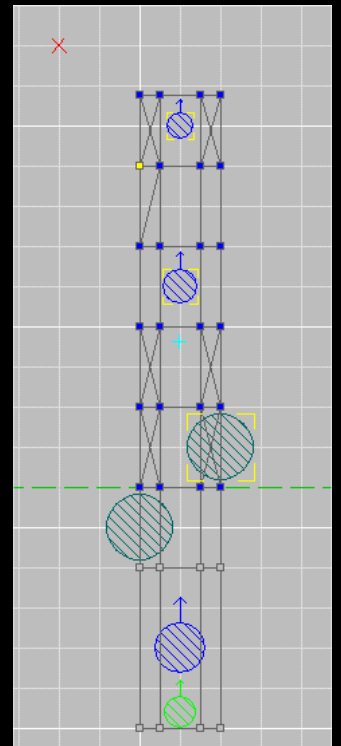
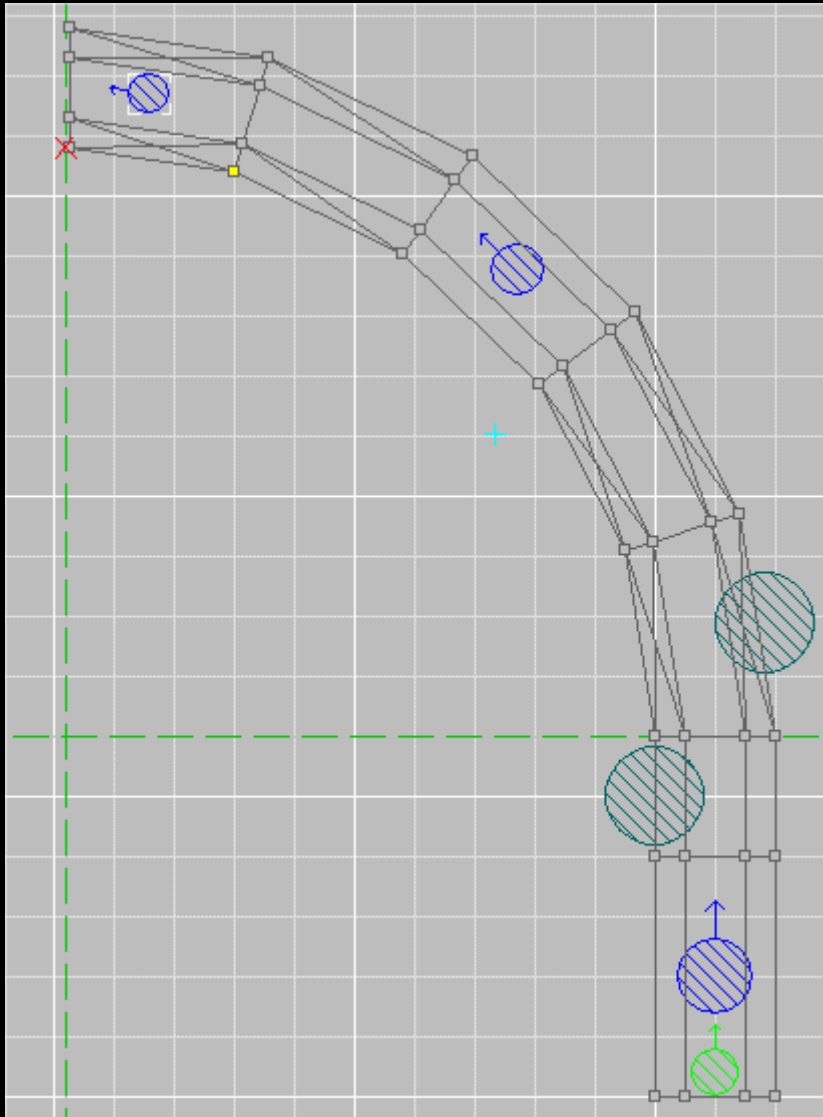


Extensive rework has now become necessary to position the objects. With the UNDO Bend button we bring the room back to its original state and also mark the objects before bending.

UB

To do this, we drag the mouse frame around the desired objects, beforehand **Ctrl-G** Go to object mode.

The yellow frames identify the highlighted objects.
We call up the Bend dialog again and bend our room with the given default settings.
The result looks like this:



A little more work and the matter is settled.

[Back to Section C](#)

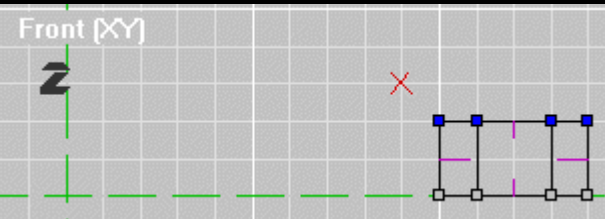
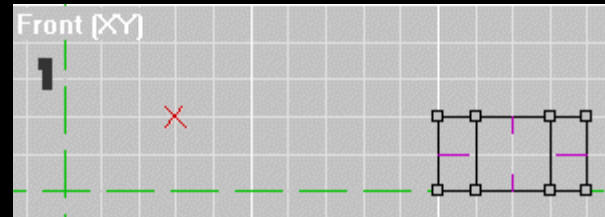
039 - Bend without Bend <>

Fischlein

(I have taken the liberty of editing this article.)

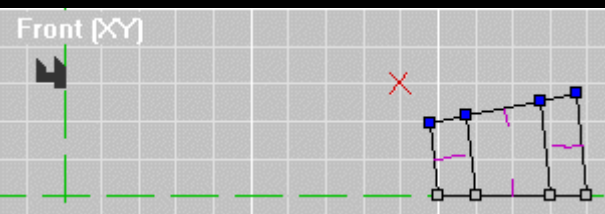
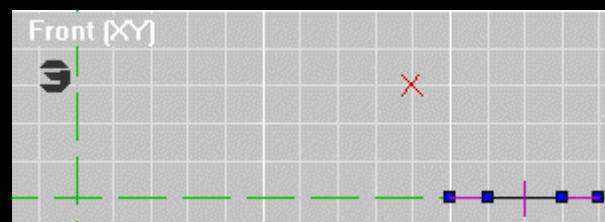
Anyone who wants to look at the space/bend can use the file. The grid size here is set to 10.

First I created a face with six verts and extruded it. How far doesn't really matter, just not too far. Note: Delete the top and bottom faces so you'll end up with less work!



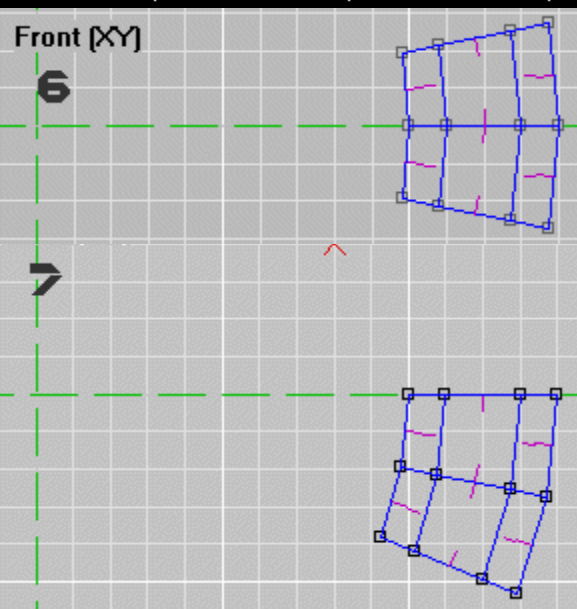
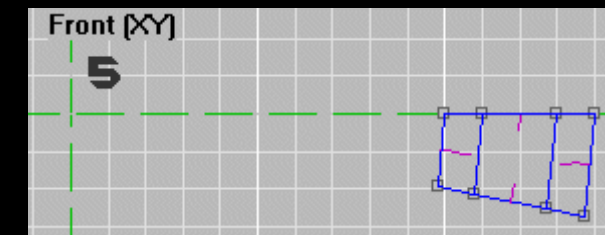
As you can see in picture 2, I marked the top verts, then I have the **2** on the NumBlock pressed twice (which pushes the vertices into one another, Figure 3).

Then I have **1** operated and as you can see in picture 4 it begins Arc.



I then started with the button **M** all verts Marked and I'm with you **Ctrl+F** Switched to face mode and then pressed the button again **M** pressed, in short I have marked all faces and verts to include everything **Ctrl+C** to

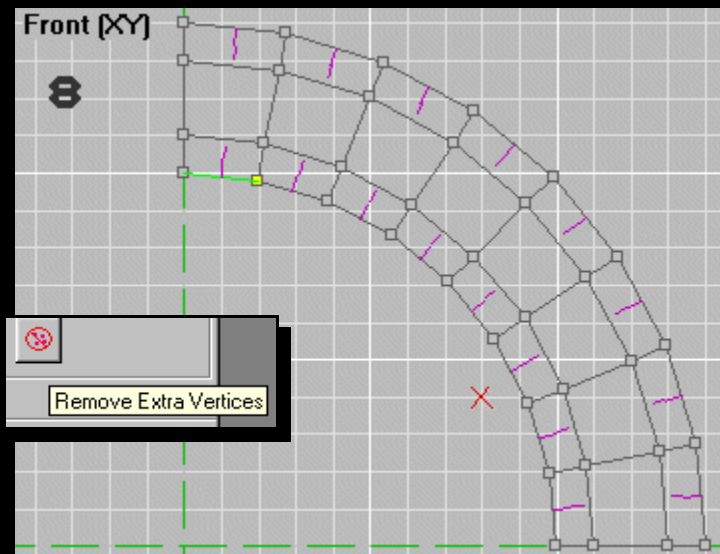
copy. Now I highlight all the faces and press the button **3** on the NumBlock. As you can see, the tunnel section has now turned downwards (picture 5). Then I have the buttons **Ctrl+Shift+V** pressed, which puts it on the clipboard again



inserted into the same position (=Edit->Paste on top, Picture 6). Then I press again the key **3**.

I repeat this until a tunnel with a 90° bend

was created. The result sees then look like in picture 8: But the tunnel is not yet complete! We still have to the Extra Verts remove, one



Click on the button in the picture on the right and you're done. If you want to insert the room into your levels, you now have to insert the faces at the two ends again. To do this, first mark the verts at one end, switch to face mode and press

the key **I**nsert, same thing again at the other end, done. The big advantage of this method is that it will always fit perfectly and the textures will also be aligned.

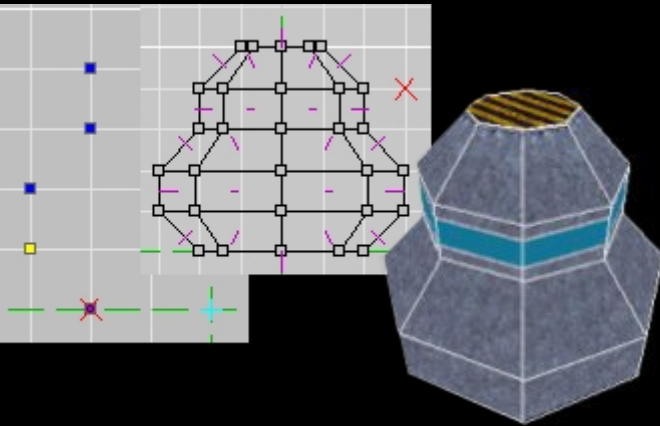
Danger: Different buttons apply to other 2D views; If you do all of this in the top view, for example **1** and **3** reversed. It's time to try things out! Back to Section C

040 - Lathe – deepening

Hydra

(has been revised)

overview



Lathe is a nice feature to have available. Although I don't use it very often in my levels, it can make really nice effects. This feature is perfect if you need a well-rounded object or space in your level.

The lathe function is quickly explained. After placing a set of vertices along an axis, go to the Lathe dialog and enter a few options.

After you have confirmed this, the verts will be rotated/copied around a set axis. Their application works well if you need a nice circular corridor or just a few large, even-looking pillars.

The biggest problem with this function is that it can create a lot of faces. If you make an object too detailed, you may have created 200 faces with just one lathe operation. You have to be careful with this when you make rooms. Again, the best thing about this feature is that it doesn't create bad shells or T-joints in your room, saving you time.

The Lathe Dialogue

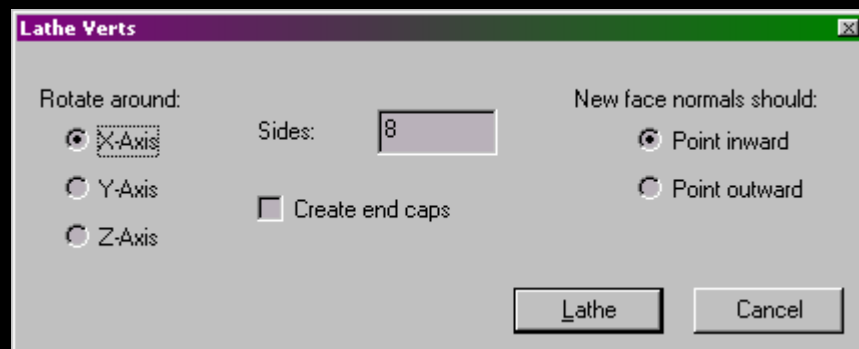
Rotate around: [X, Y, Z]

Rotates/copies the selected verts around the [X, Y, Z] axis.

New Face Normals should:

Point inward–All normals of the new faces look into the room. This is used to create space shells generate.

Point outward–All normals look outwards, or away from each other. This is mostly used for pillars or tubes.



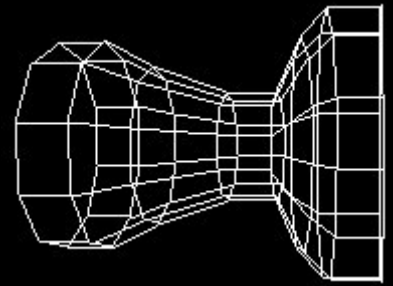
Sides:How often the verts are rotate-copied. For example, if you enter 4, a square object will be created if you enter 8 an octagonal one.

Create End Caps:Additionally creates faces at the top and bottom of your object.

Side note: I make these tutorials very free-form, mainly because here you not only gain skills in building rooms, but they allow you to experiment in an environment of your own creation.

Application 1: External

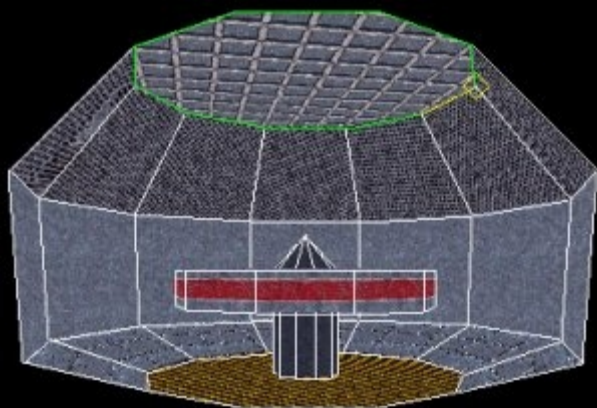
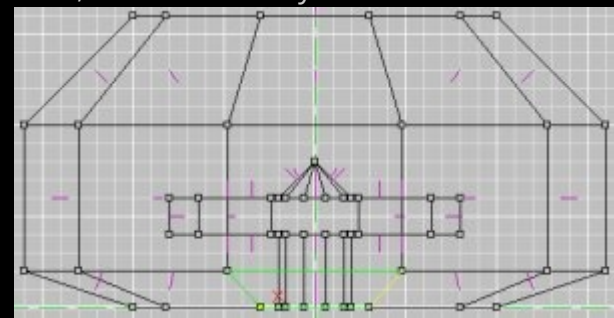
Just like with extrusion, when lathing large areas, advance planning is the most important thing. You should know what textures you will use, where you will use them, where other rooms will be attached, and maybe even where pillars or other things will go. This is a lot to think about, which is why I usually try to get all my ideas down on paper before I start.



There are basically two types of rooms you can create with Lathe. You can make a large round shaped room, or you can make a donut shaped corridor. I'm going to make a big room; mainly because it's not recommended to do corridors with Lathe as the entire corridor is one room, which can create a little extra lag as well as lower FPS. I started my room by simply placing the verts where I wanted them; note, the shape they outline does not have to be convex.

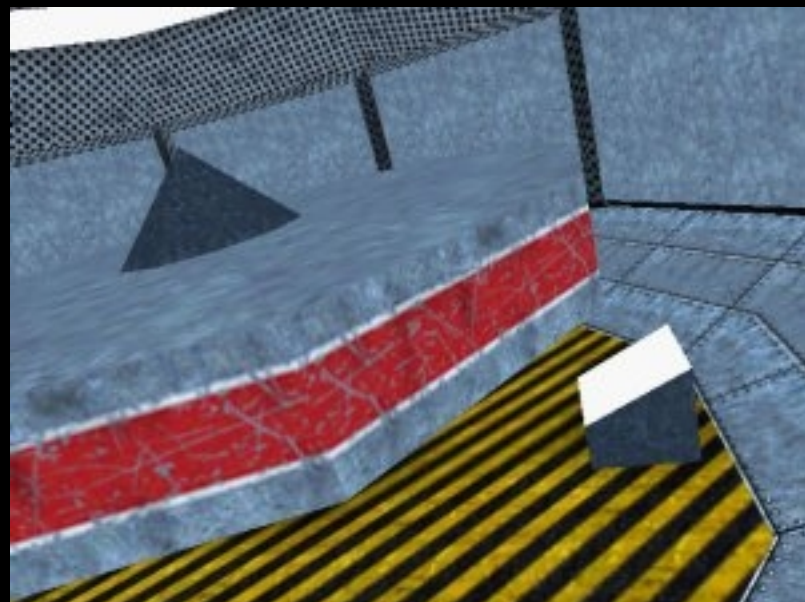
I latched these verts around the axis, and immediately I had a few bad normals. I had to go through each face and pick out the 'bad ones' and delete what isn't

took a long time. I examined what I had left and made sure it was latched correctly and that the normals were facing the direction I wanted. (comes afterward)



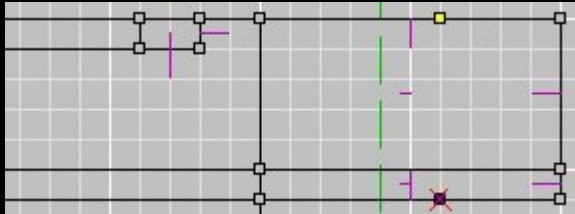
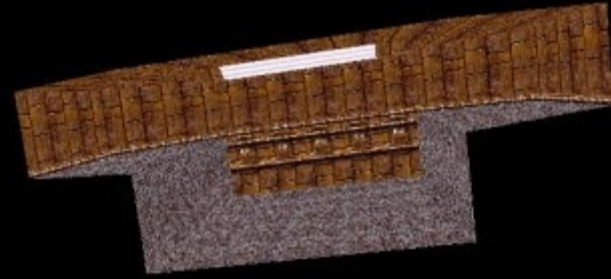
After that I added textures and made a few tweaks to some verts that had been moved a unit or two out of place by the lathing.

I added some lighting (a bit too much actually) and a few other things. This level would be far too small to have a proper dogfight, but it would be a good one
Connecting room between two other rooms.



Application 2: Internal

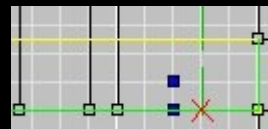
In this lesson I'll give you a few examples of how to use the Lathe function to create indoor objects, such as pillars. I will use the space I have for my Extrude tutorials (*comes later!*) because I like its structure.



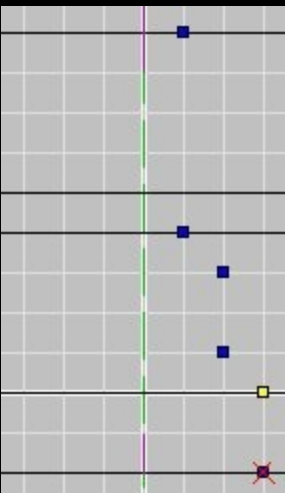
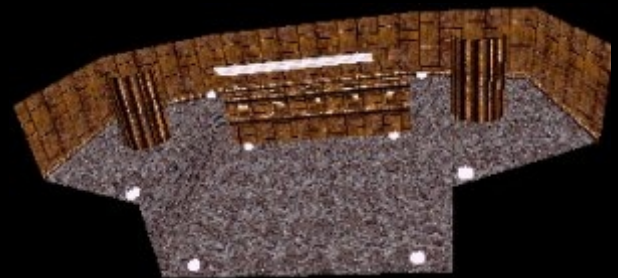
First, I want to make a few pillars along the edges, which will allow players to dodge fire and evade homing weapons more easily. You want your rooms playable and good looking. Note that I moved the reference point (the green lines that define the axes). Don't forget that Lathe always goes AROUND the axle, so you have to position the axle there,

where you want the center of your lathe to be. You can reposition the reference point into a 2d view by Ctrl-clicking.

After latheing a pillar I have the same thing Page made. Then I distributed some lights around the room using lattices like I did with the pillars, only much smaller. The 2d view shows the lights before, the 3d view below after I've lathed them all in the room.



on the other



Afterwards I realized that this level needed a reactor-like object. They look really cool in a level and they're pretty easy to make. I did the same thing as with the pillars, except I arranged the verts more staggered, so not in a line. I then gelathed them and textured the result.

And there is the result in D3. As you can see, even a few simple lathings can make a room look excellent. Although, like extruding, you need to make sure that the level itself is easy to navigate so that players don't end up in the walls during a firefight get stuck. (*We all know that...?*)

Back to Section C



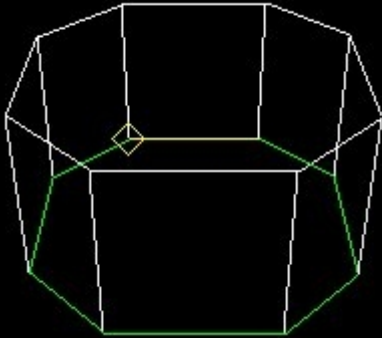
041 - Extrude – depression

Hydra

(has been revised)

Overview

Extrude is not only the most basic of all three geometric actions, but it is also the one I use the most. This feature is not only perfect for quickly creating columns and pillars, but it also efficiently creates a 'sealed' room shell with a few simple ones Clicks.



The Extrude function is quickly explained. All it does is copy a selected face and paste a second one a certain distance away, then create faces 'all around' the sides, sealing them. The image shows an extruded octagon. You'll see that she makes an exact copy of the selected face and then adds the 'sides', making it a space.

Do you have a look like this first? function, you will not only be able to make shells of simple rooms, but also shells of complex rooms like the one on the right. And best of all, the feature won't cause you any T-joints or bad shells in your rooms.



The Extrude Dialog (v39)

Extrude from:
Current Face–Extrudes from the selected (green/yellow) face.

Marked Face(s)–Extruded from all marked (blue) faces.

Extrude along:
Normal–This is the setting I actually use, it extrudes in the direction where the face's normal points.

[X, Y, Z] axis–Forces D3Edit

to do this, put the face in the [X, Y,

Z] axis, useful for extruding straight away from a slanted face.

New Face Normals should:

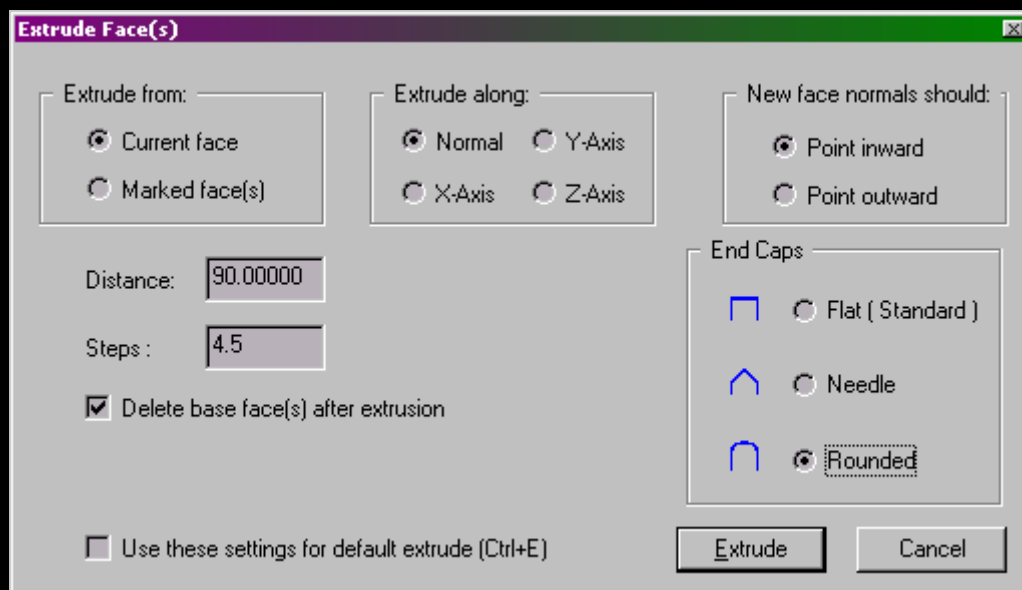
Point inward–All normals of the new faces look into the room. This is used to create space shells.

Point outward–All normals look outwards, or away from each other. This is mostly used for pillars or tubes.

Distance–Tells D3Edit how far away the second face should be positioned.

Steps–how often the extrusion should be carried out. Every step goes beyond the fullDistance.

Delete base Face(s) after extrusion–This will delete the faces you extruded away from (or in other words, the currently highlighted/selected faces). This is helpful if you are doing multiple extrusions or the base face you start with is not visible.



Use these settings for default extrude—This sets your current settings as a sort of 'default'. Then you just have to **Ctrl-E** Press and it will be loosely extruded without even displaying the dialog.

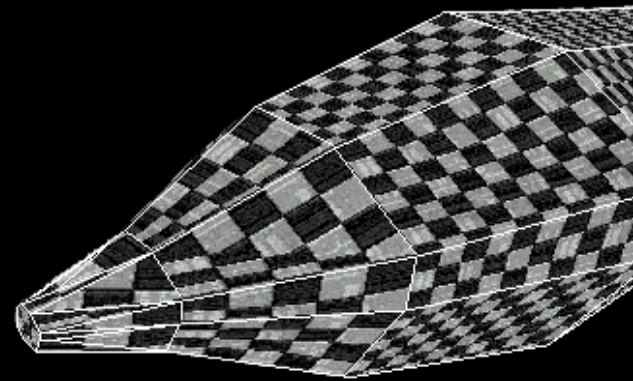
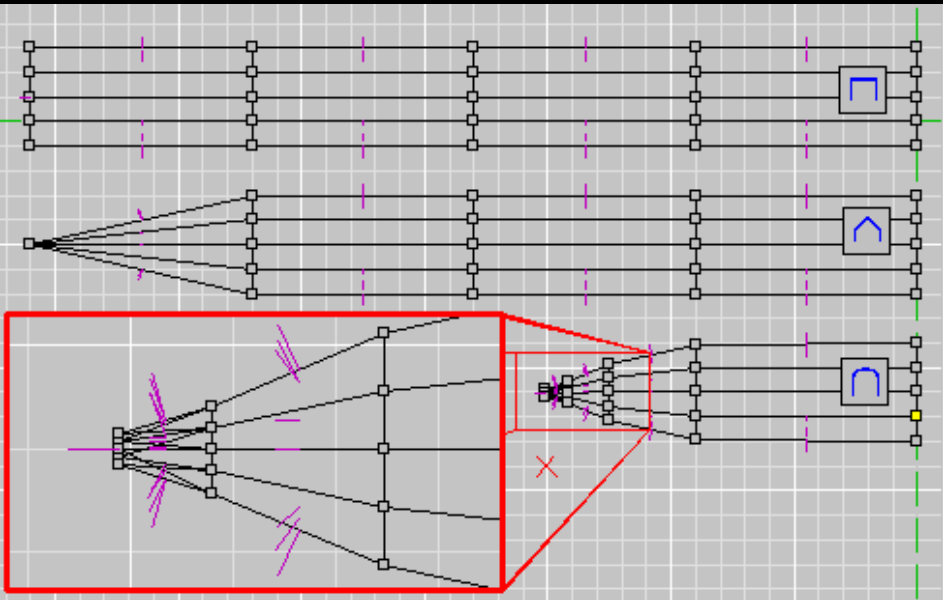
End Caps:

Flat—Roughly the same as the end caps in the lathe dialog. The extrusion result is completed with a face.

Needle—In the last extrusion step, you do not extrude onto a face, but onto a vertex.

Rounded—Hmm... interesting, see for yourself.

The settings used correspond to the dialog shown above.



Side note: I make these tutorials very free-form, mainly because here you not only gain skills in building rooms, but they allow you to experiment in an environment of your own creation.

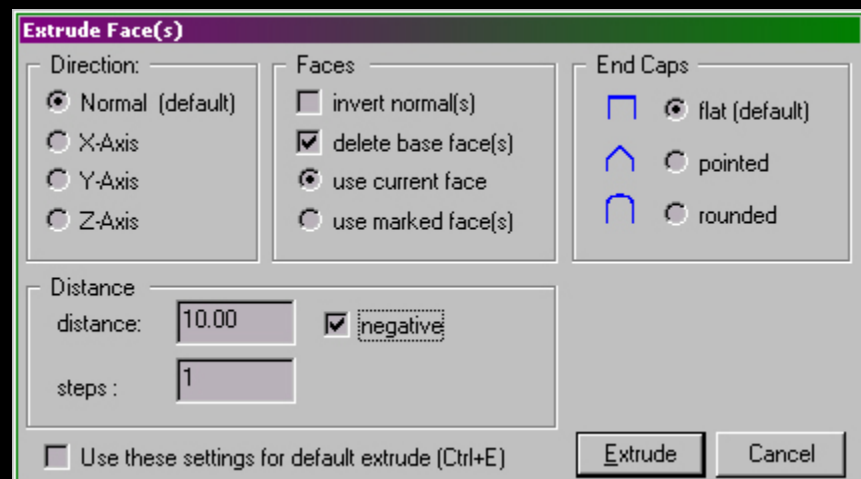
Update for v40:

The extrusion has been revised. The new dialog looks like this:

Direction and End Caps remains the same, has been changed Distance and Faces.

By default, the normals of new faces now always look in the same direction as the one(s).

Base face(s).



At Distance is now negative hookable;

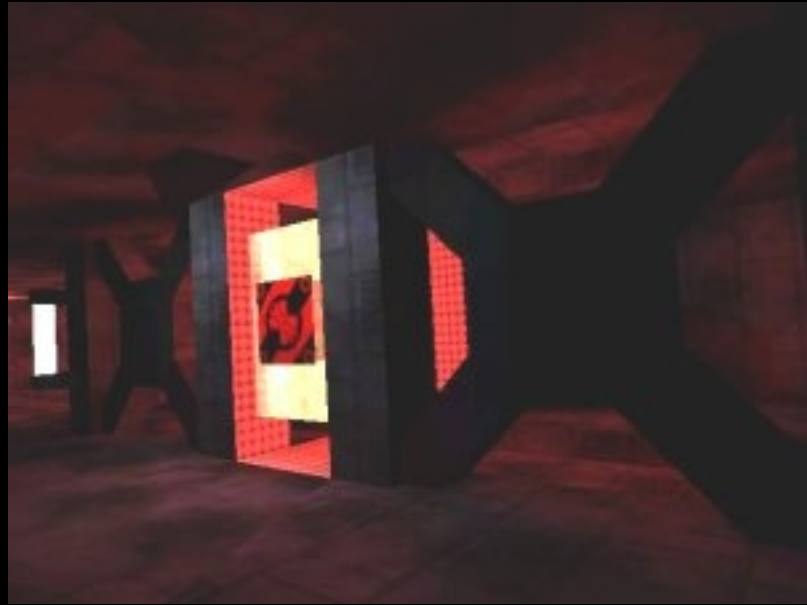
this setting causes extrusion against the face normal(s) of the base face(s). Under Faces there is still the setting invert normal(s), this flips the normals relative to the base face.

The new extrusion also feels a little different, but is more intuitive; Just play around a bit and think it through.

Application 1: External

One of the most important things when creating a room needs to be planned in advance. This is very difficult for me because I prefer to improvise while I'm building rather than sitting down and recording everything in my head, or rather on a piece of paper. I'm not saying that you have to have your entire level completely planned out, with all the pipes and pillars, but that you have to have a rough idea of how you're going to texture and where attached rooms are going to be. I show the picture on the right as an example of poor planning. I didn't plan for this, and it took me almost three days just to get this one simple room to function correctly. Therefore I guess that this picture is just me

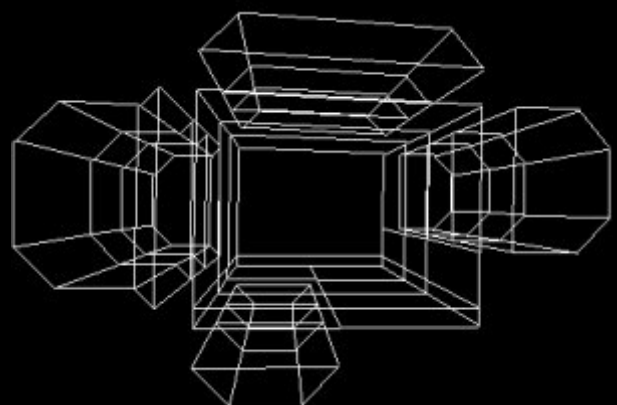
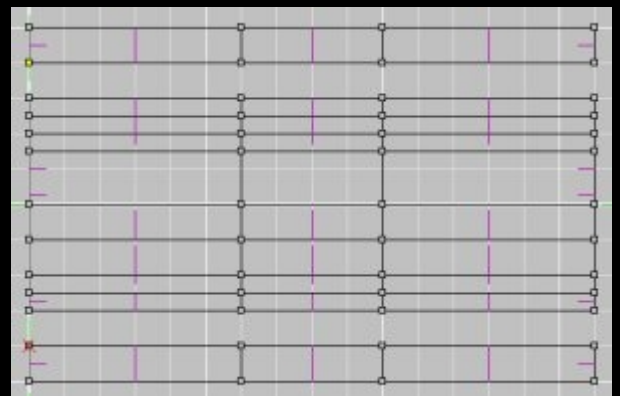
means something 😊

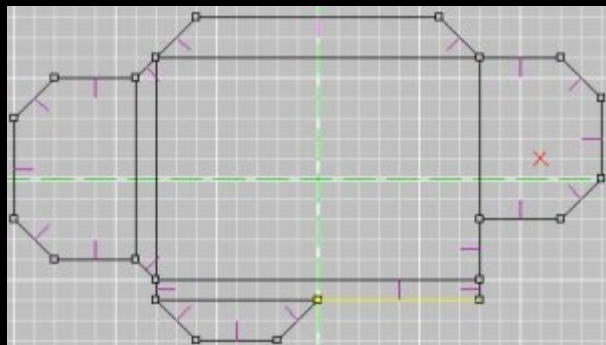


Once you have an idea of what your room will look like, you can start extruding parts of it. I want my room to be a large, complex looking corridor. I already have the room put together in my head (and on a piece of paper, I forgot what it's supposed to look like, so I start making the 'parts' of the room. The left picture is the front view (XY) of what I have up to now have.

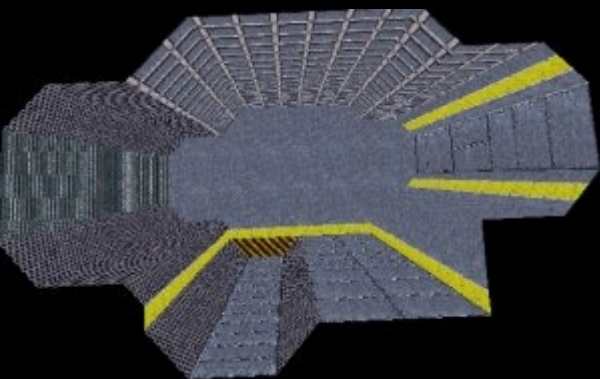
Then I marked all the faces and extruded them once 60, then 40 and then again 60 units. Note that I added vertices to some faces that I don't really need. These extra faces that this creates are actually used for extra textures, which helps make your space a little more realistic. This is where planning ahead comes into play, because you need to know not only where the portals of the other rooms will be, but how big those portals will be. Doing this prematurely can get you in a lot of trouble with T-

Spare joints and bad shells. The image on the right is the top (XZ) and 3d wireframe view of it.

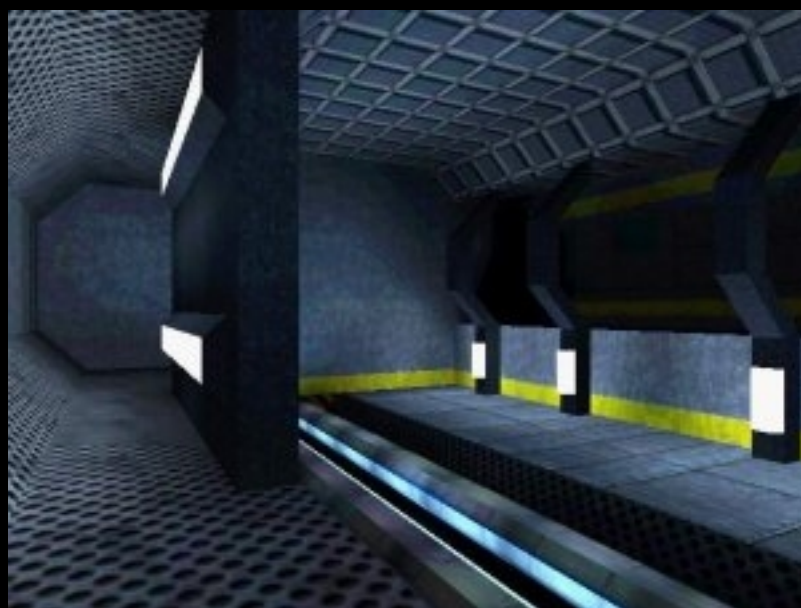




After I did that, I took all of the separated sections (I keep them separate as it allows me to see them more easily) and pushed them back together. Then I deleted the faces and removed all the extra verts, thus 'sealing' the level. Then I added a little creative texturing, which I really like. The top image (left) is the front view again, where all segments are united into a whole (note that the normals, which are usually blocking walls to other segments, have been deleted), the bottom is a textured view of the room like he is until now.

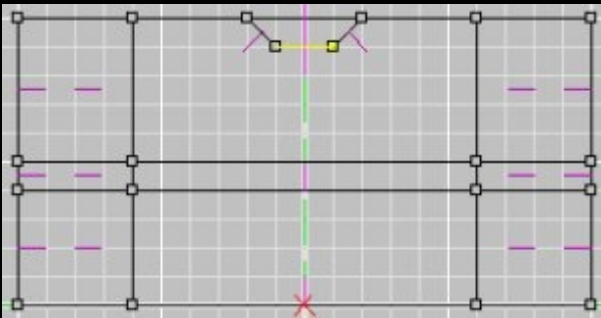
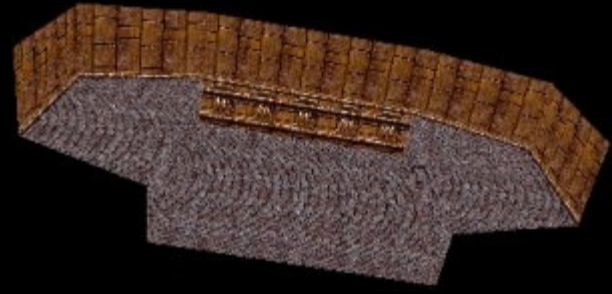


Then you're done with the shell, or 'frame', of the level. Next I will show you how to make piers and others
Adds obstacles to give your level a realistic touch. I finished the room with some pillars and lights and did a screenie in D3, I think it looks pretty good.

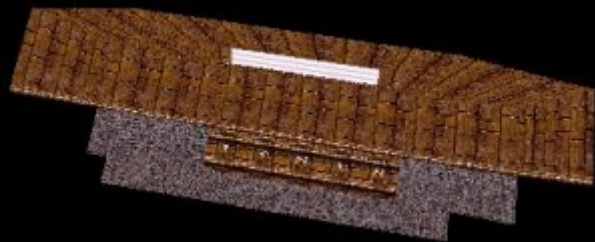


Application 2: Internal

Extruding pillars and various 'effect' objects like pipes is one of my favorite things about creating a level. Because I know I already have a 'sealed' shell and I don't have to mess with it anymore, I can now just be creative and make the rooms look nice myself. In this exercise we will make a few simple objects to spice up the space to the right.

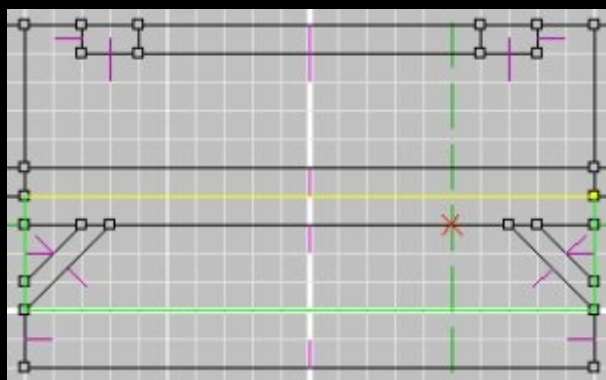
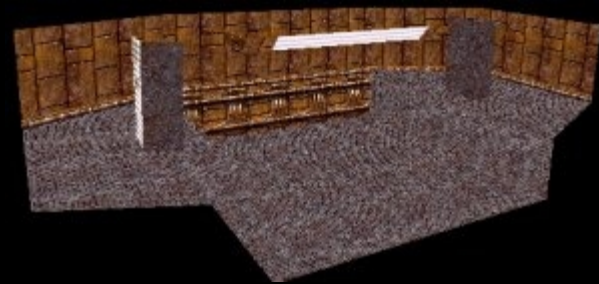
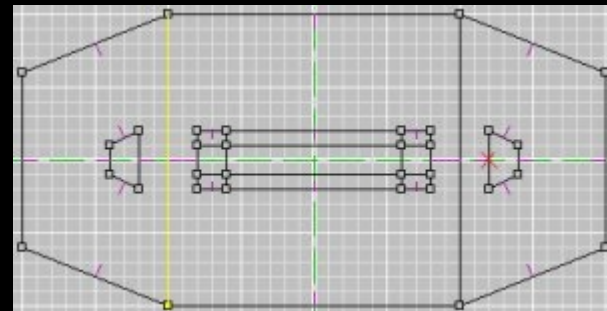


The first thing I decided to do was add my lighting to the ceiling. I created a small shape that I wanted the light to look like and extruded that face through the room three times, once by 10 units, then by 60, then again by 10 units. I did this because then I could make the light look like it was being carried through part of the structure. There are two pictures on the left, the first one is the shape I used (in the side view, ZY) and below that the 3d view.



About the lighting
I have to finish it

Two more lights were added on each side; these should not only illuminate parts of the room that cannot be reached by the large central light, but also the portals to the next rooms. This was just a simple extrude and texturing. These pillars are also large enough for a player to hide behind them and take cover from weapon fire (and avoid homing weapons). The top image shows the top view (XZ) and below again the 3d view.



On a whim I decided to add a bridge across the gap. It would be cool to be able to dodge under the bridge, but the current depth of the gap is not great enough; I extruded another 10 units away from the bottom, which gave me enough room to do some work. I started by making the bridge itself, which is more or less a cube. I then added some supports by creating them in the front view and extruding them to the side. The left picture shows the front view (the sides of the room are cut off) and the 3d view.

And if we look at it in D3, it looks pretty good. Although it's a bit dark, that's the effect I like in my levels. I like places where you can hide deep in the shadows. However, when making extrusion effects like this, you must remember to give players ample space to maneuver and dodge incoming weapon fire. Although extruding like this can make your level look more realistic, you also want players to have fun in the game.



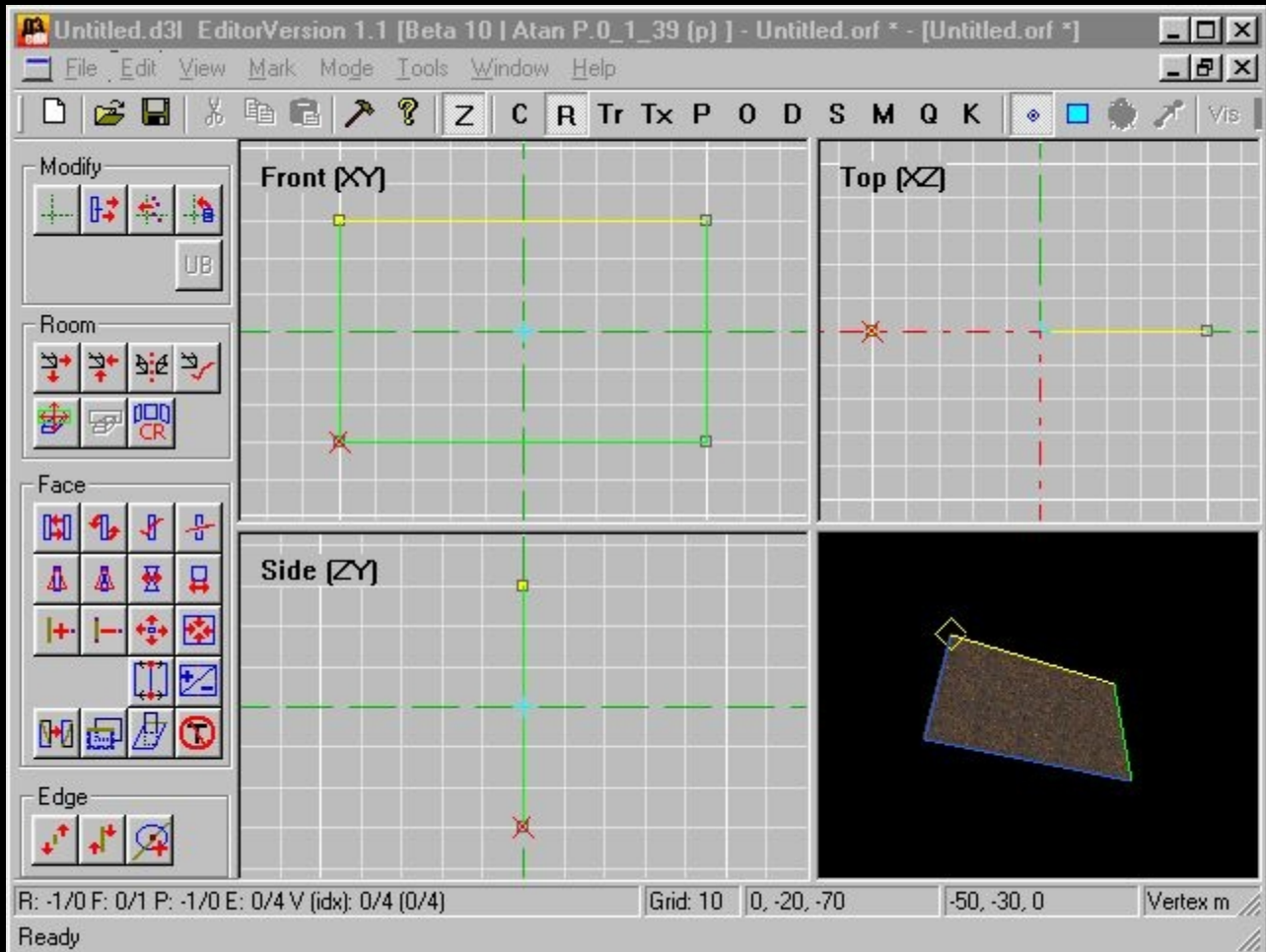
[Back to Section C](#)

042 - Extrude: Zero

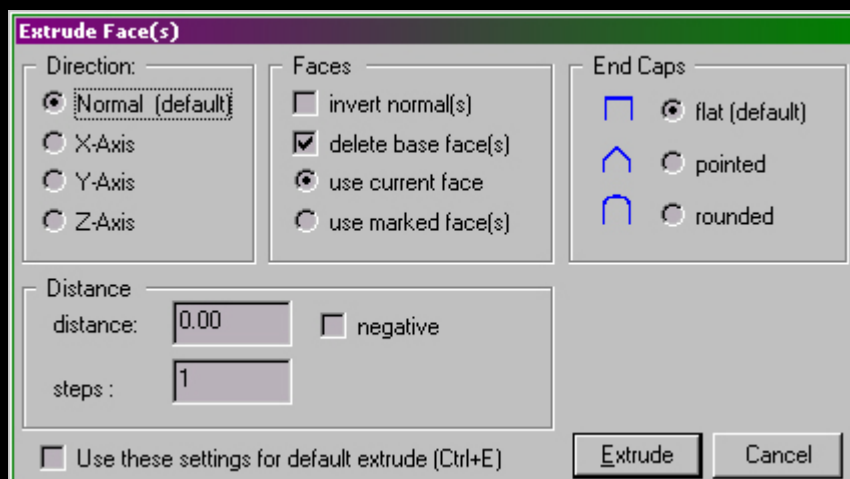
WillyP

(From Descentpedia, <http://www.prepare4descent.net/descentiapedia/DescentiaPedia>)

While using D3Edit, currently v39, I noticed a small quirk that should be taken into account. This also affects v38/2, but not v37. It's not a big deal as there is a simple workaround... since a picture is worth 1000 words, I'll describe it visually:

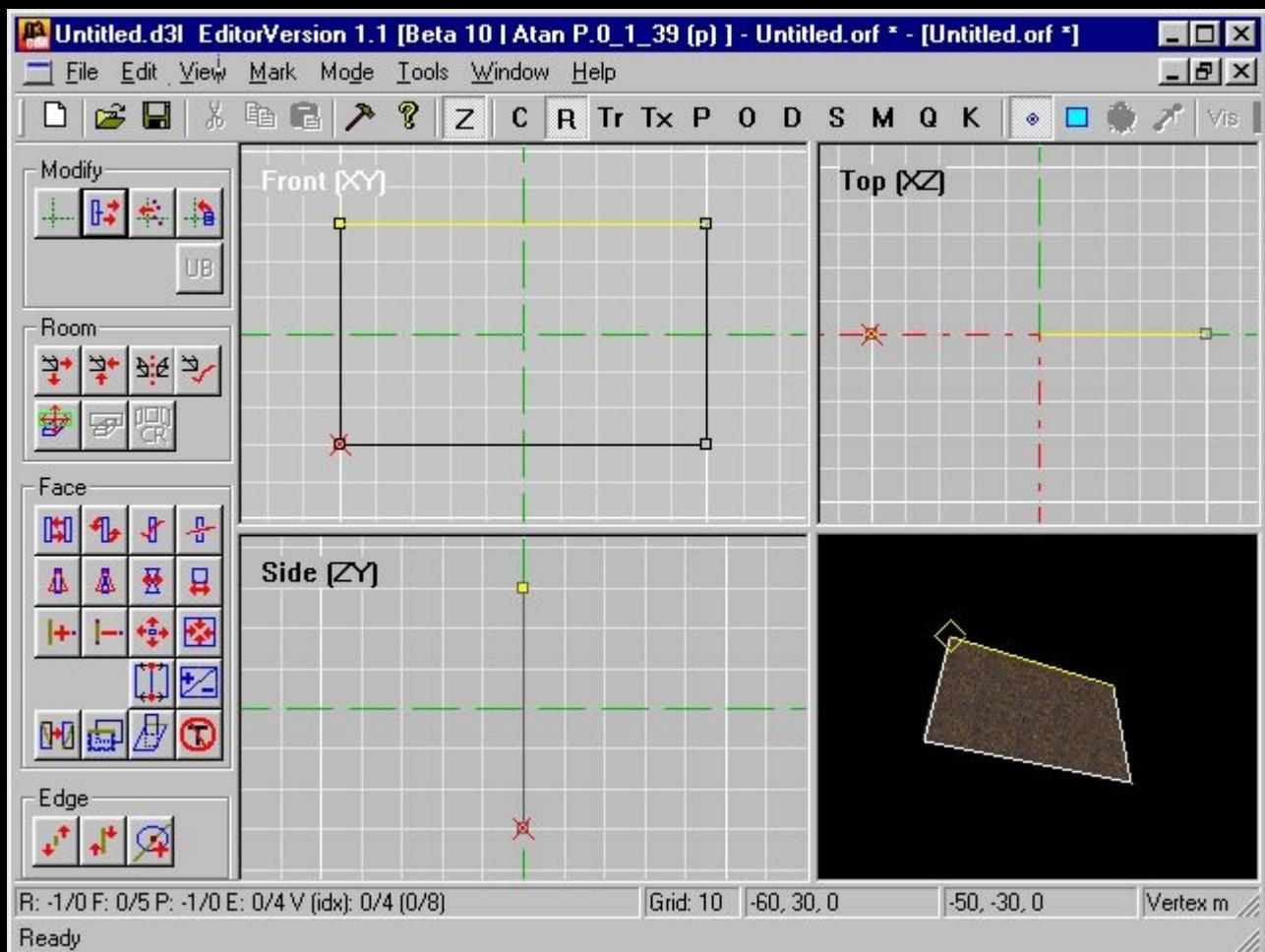


A minimum of one face. Could be part of a room or level.

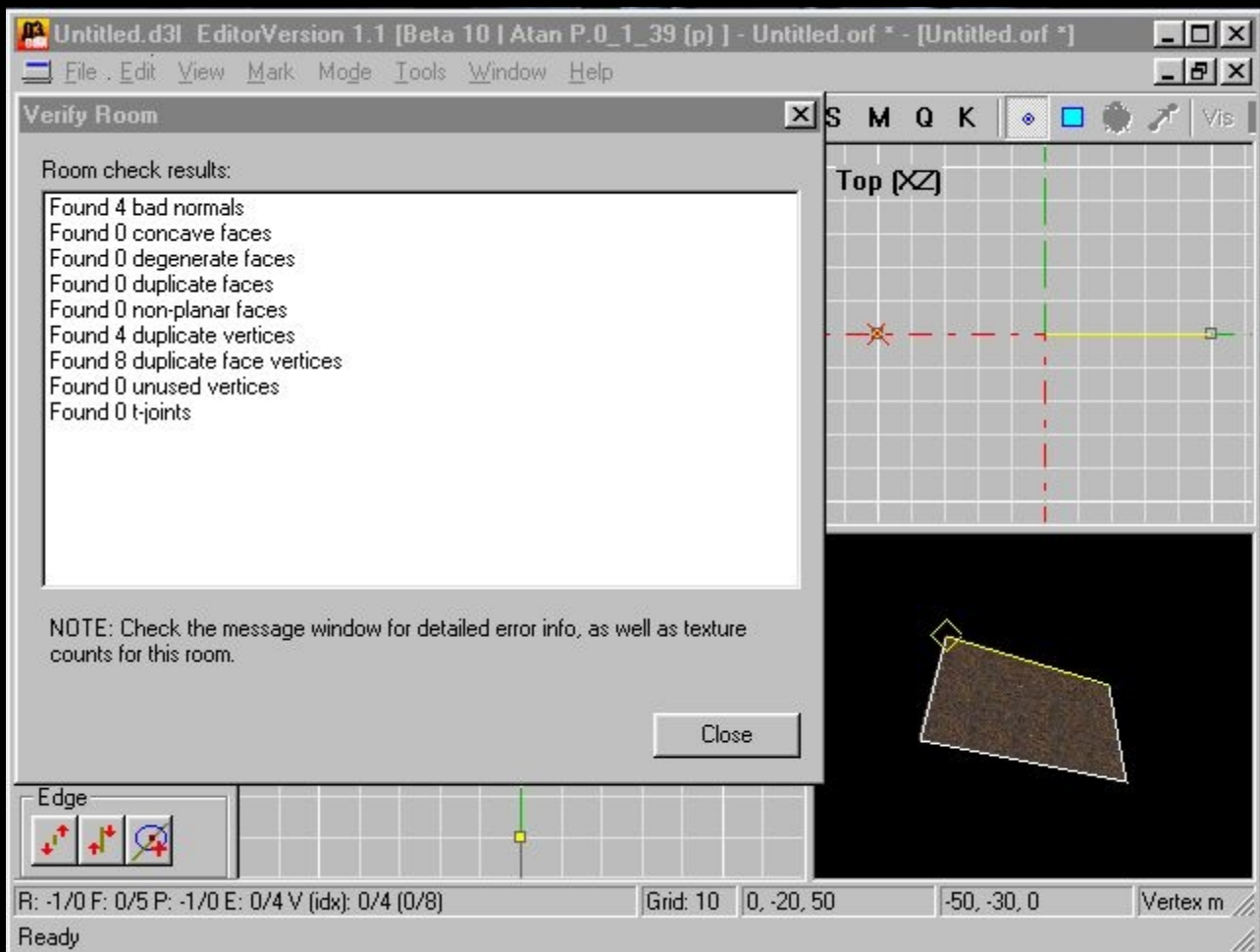




Open the Extrude dialog and set to 0 units, 1 step, delete base, flat.



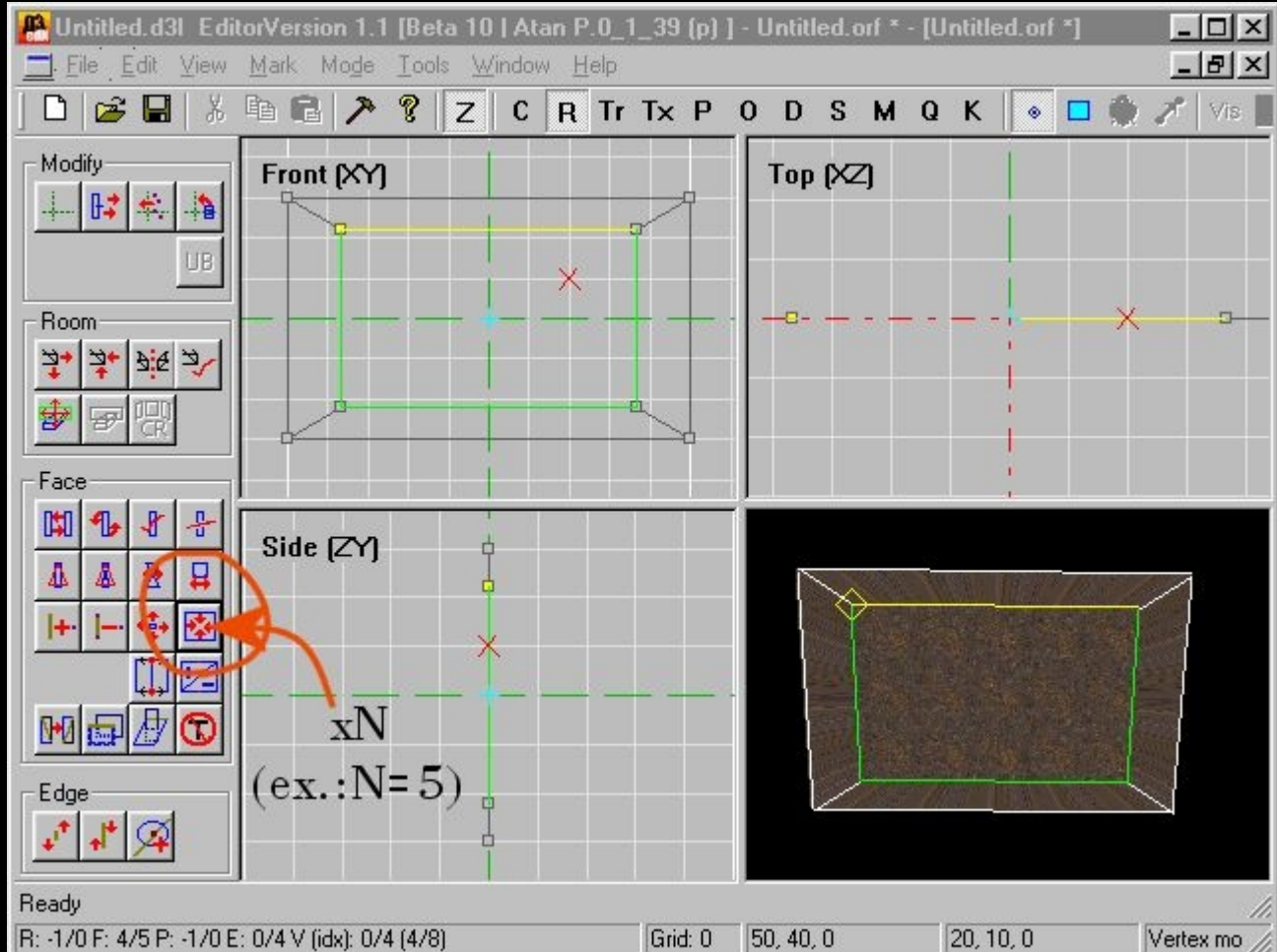
The face doesn't seem to have changed, but we now have five faces and eight verts.



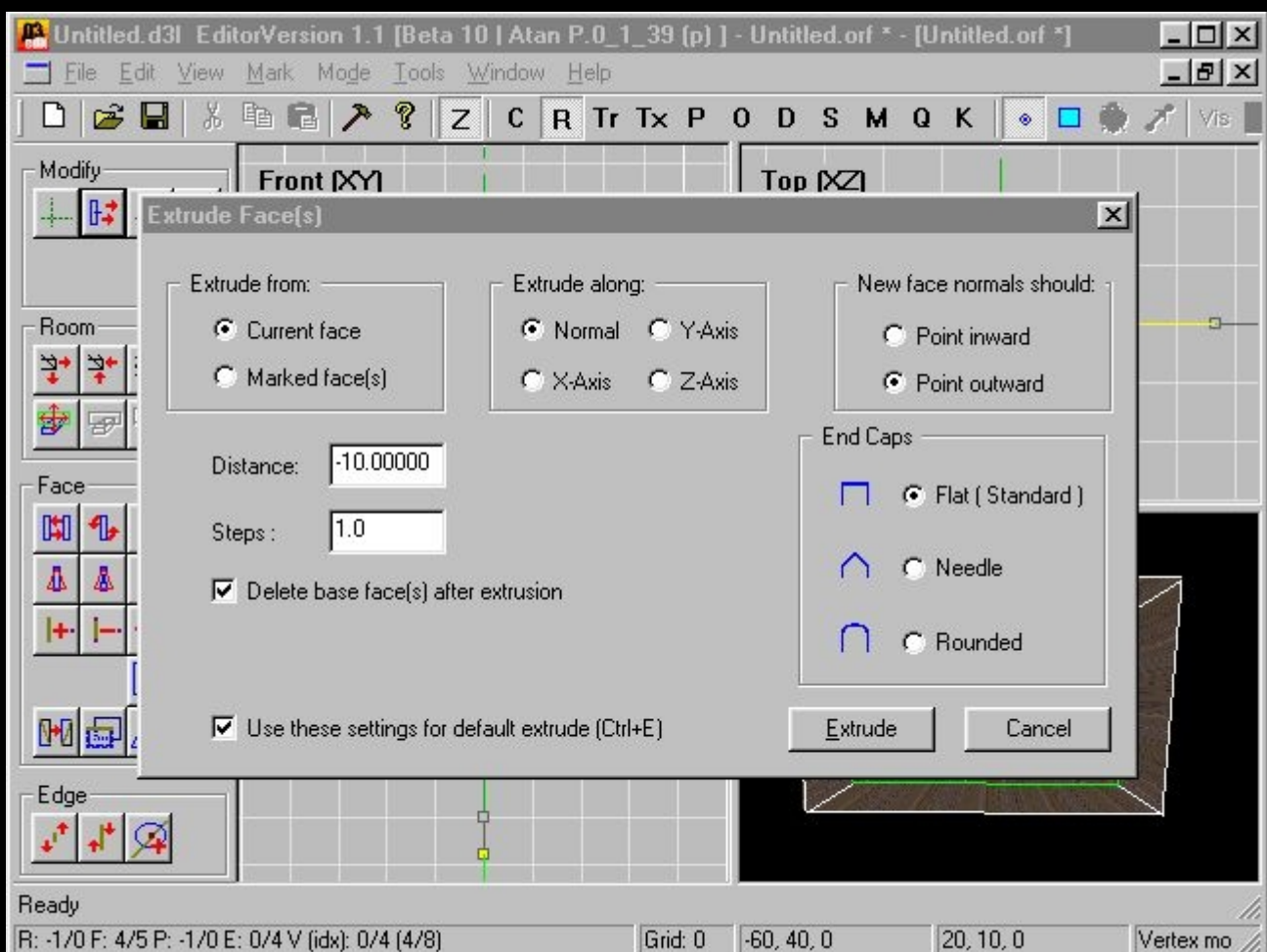
Ignore the errors for now...



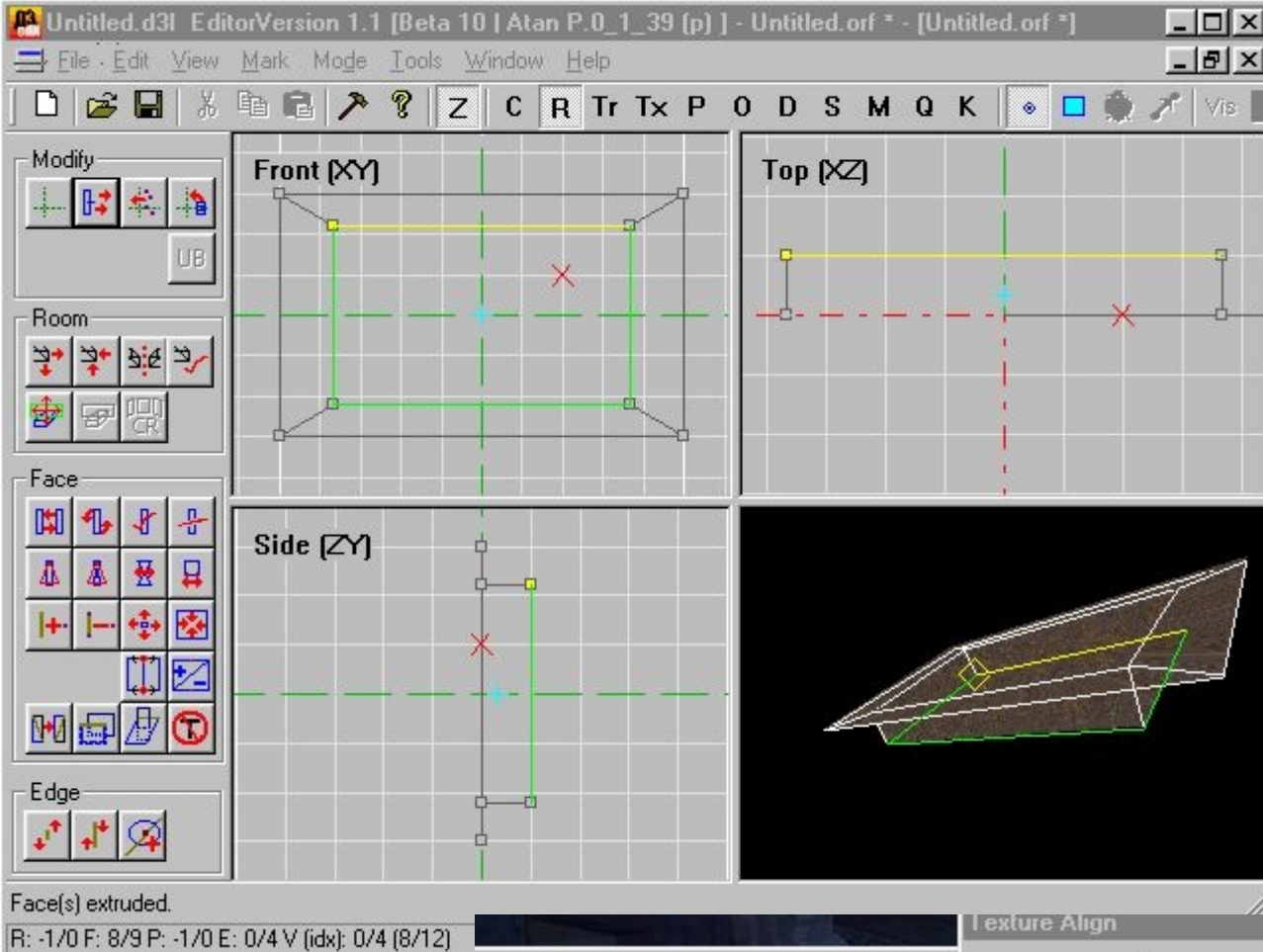
DO NOT execute Remove Extra Verts



Make the face current and shrink it. (in this example five times). As you can see, the texture on four faces is extremely stretched (as they should be, considering they were stretched from 0). In v37 and earlier, three of the faces would have been split.

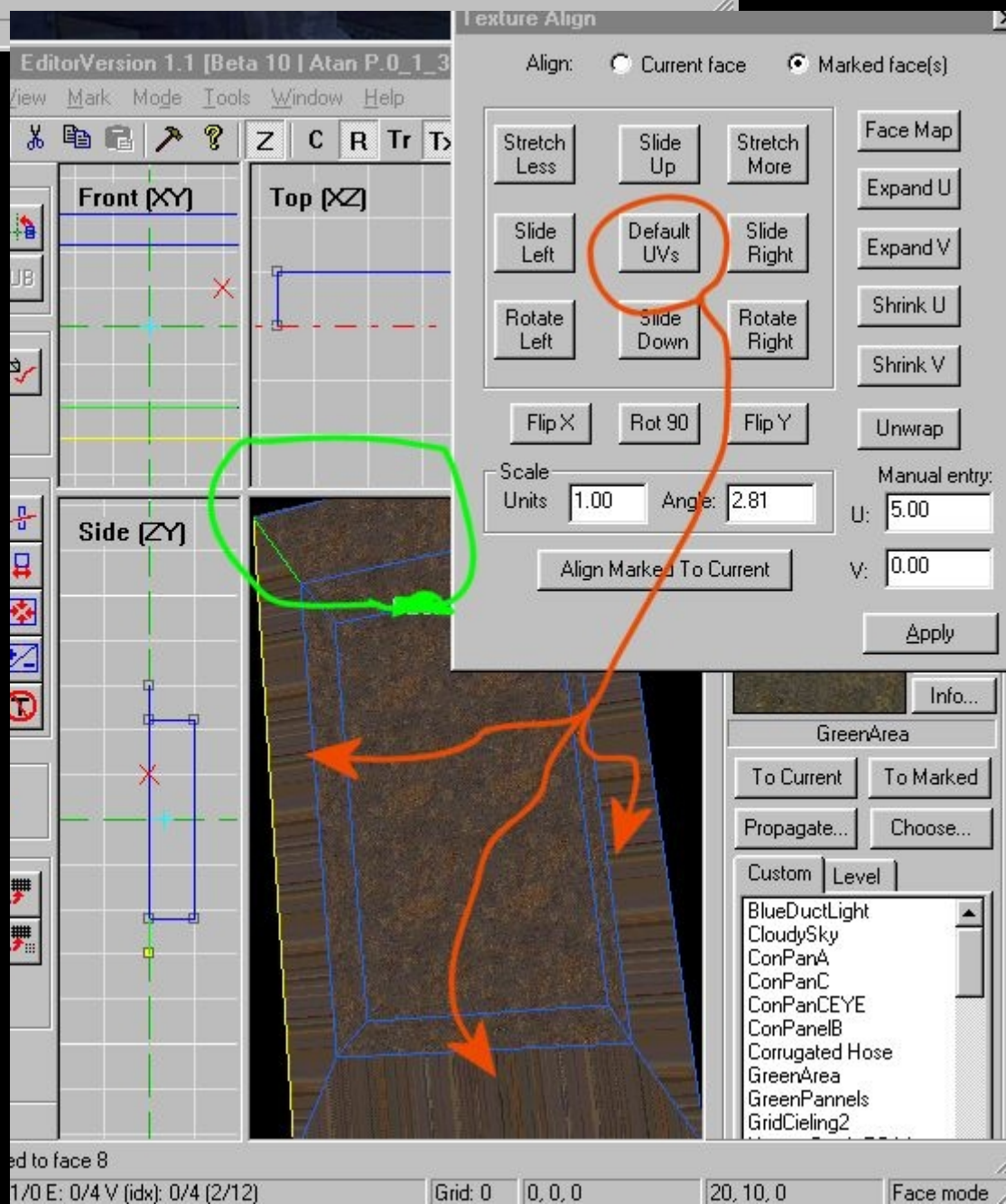


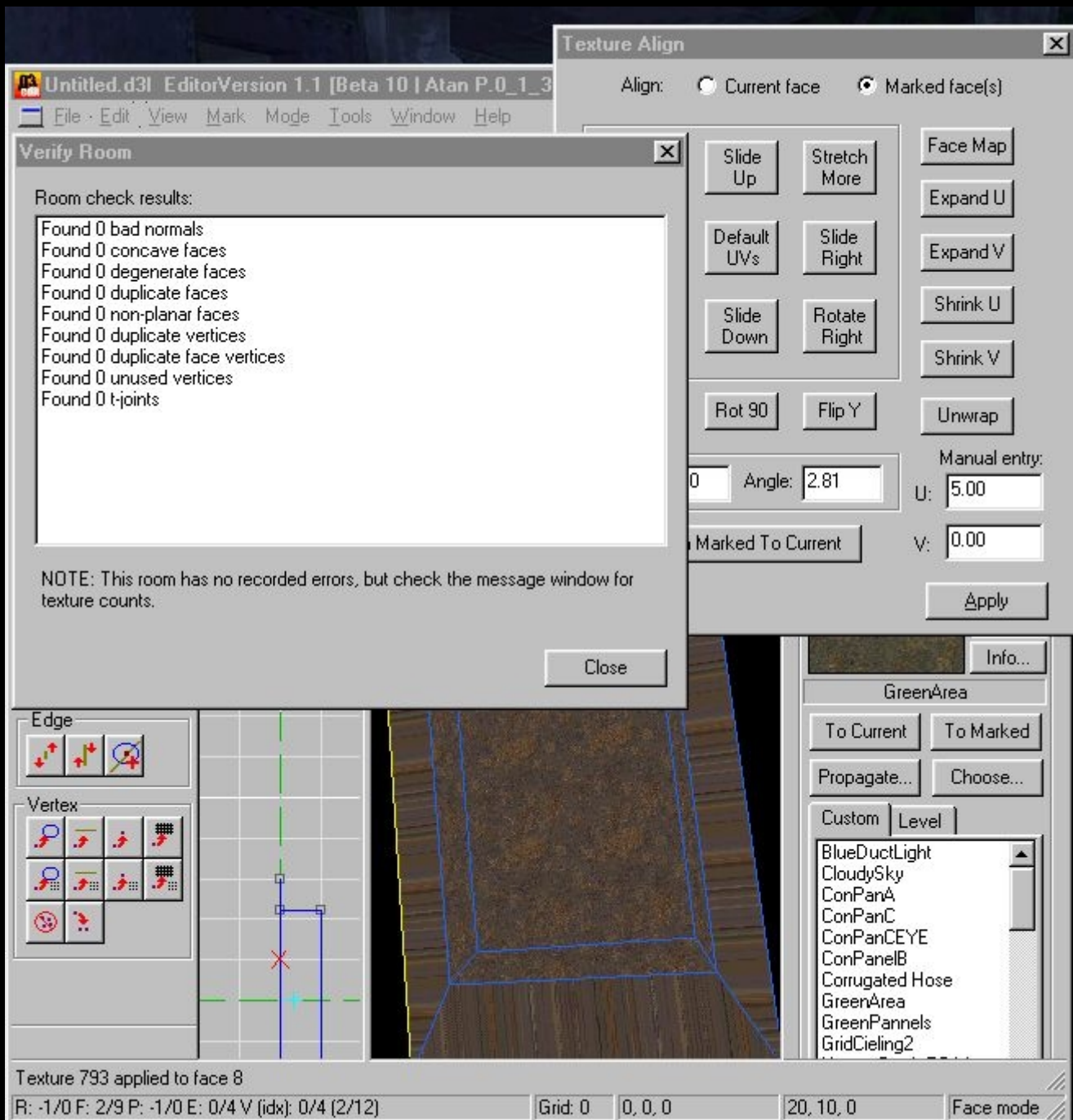
Just to show *Why* if I wanted something like that, I extruded the central face...



... approximately
SO.

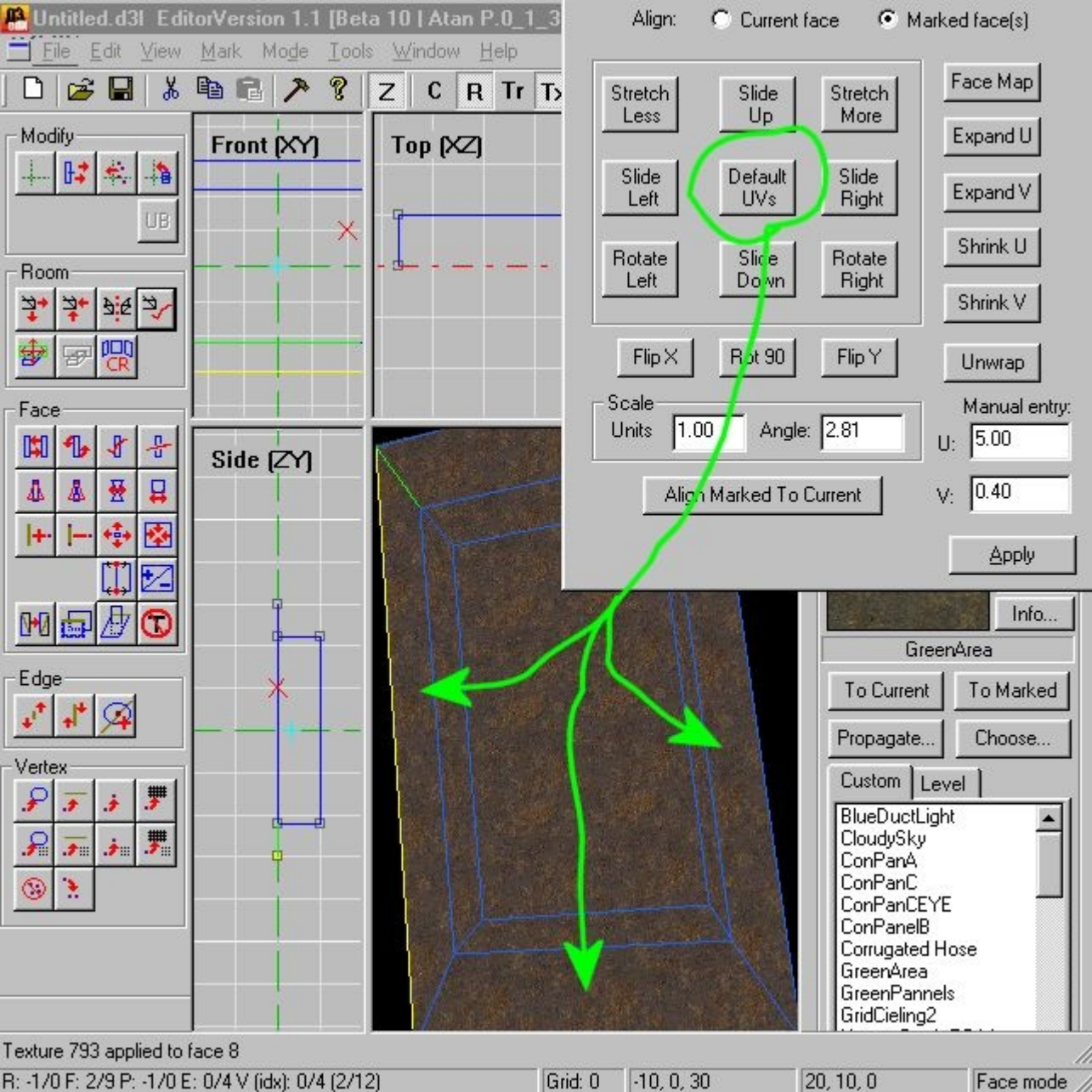
This would work in v37 and earlier mark and open them UV's to set. In v3 v39, (no longer in de for whatever reason, an not yourself. The texture *has* changed, but how we would like to have every conceivable Tried to reset the tray defaults only found one...





...Verifyto carry out.

Now reportedVerifyno more errors, and more importantly...



I can now this U/V on default set.

Example in the game:

Back to Section C

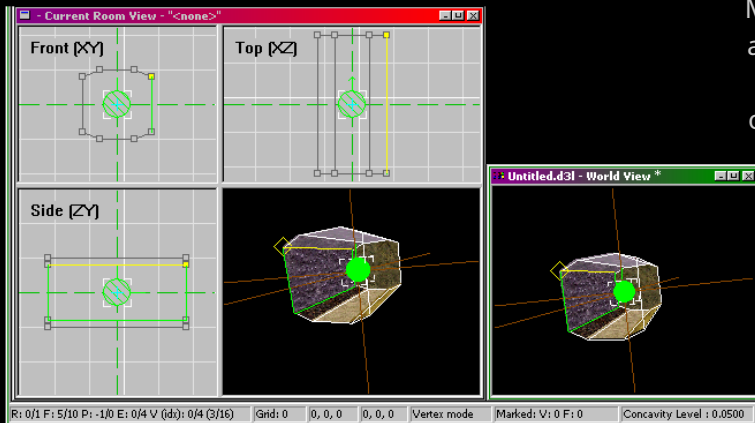


043 - Mirror, mirror...<>

Ragil Ral

One of the innovations in D3 compared to its predecessors are the mirrors. Used primarily as a design element in single player missions, they also have their appeal in multiplayer. Have you ever accidentally shot into a mirror instead of...? 😊

However, mirrors have serious disadvantages. On the one hand, the reflections have to be calculated, which comes at the expense of performance; For these reasons, mirrors are frowned upon in today's (2007/8) multiplayer levels. On the other hand, mirrors are also subject to certain restrictions, for example only one face per room can be a mirror. Unfortunately, unfortunately, multiple reflections cannot be realized either, D3 only calculates the first reflection and from then on displays the mirror texture as a normal texture; You'll see soon. It's a real shame, but I imagine the D3 engine would otherwise freak out if you let it calculate two mirrors facing each other... but couldn't you have limited that to 10 or 20? Well, there's nothing you can do. 😊 Another restriction is that a maximum of 31 faces visible at the same time can have a mirror texture, otherwise D3 will cause a befitting crash. Mirrors also don't work properly in the outside world; partly it's reflective, mostly the face is transparent.



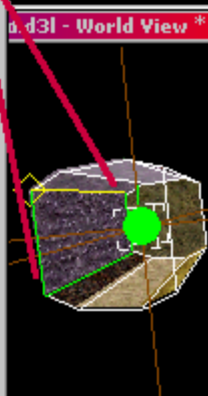
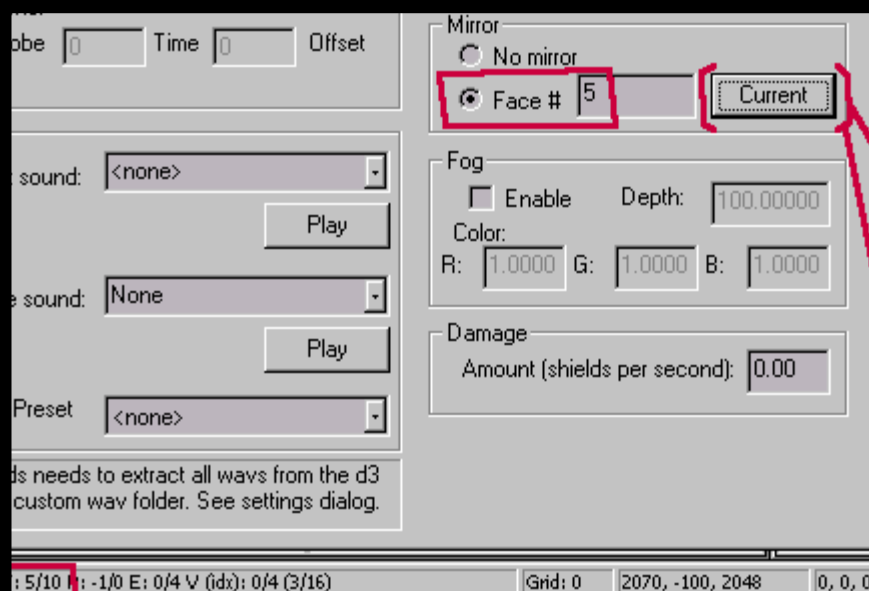
Mirror a thing that is assigned to a room n let's say a room. I have the one here because it is completely sufficient for the demonstration, it should be the one who goes through this and

Plus it's faster. SoFile->New->New Level - Default Room and now there is a level with a player starting point and everything. Go to Room View, then switch to Face Mode if necessary (**Ctrl-F** or **tab**).

Another restriction applies here; because mirrors are only possible if the used

Texture has the 'ALPHA' flag, meaning it is marked as translucent. The value of Alpha determines how reflective the texture will be. So give a face a texture like this, I used the 'Dirty Glass' (under Mine Textures) here, which has the alpha flag. Since mirrors are a feature of space, investigate Window->Room Properties. There you now have the option to define a face as a mirror. One press of the button **Current** enters the # of the current face, but of course you can also specify a different one. It's always about the face number *regarding the spaces*, see picture below right. So define one as a mirror and save the level first. However, the editor doesn't show you anything, the face is simply displayed with the texture, but since v40 you can see the mirror property, see further down. Oh, and no lights

forget because otherwise
Calculate is dark
So, grab the Lev, jump into
your pyrotechnics and
throw yourself in front of
the mirror.
I find it interesting that
the pyro in the
Rear view to the front
look when you have a
mirror behind you



Now the proof: We will add the room to this room

attach, so to speak

reflect. In addition

save it first

space as separate

File (in the Current

Room View). Opens

the room and places it in such a way that the mirror faces are

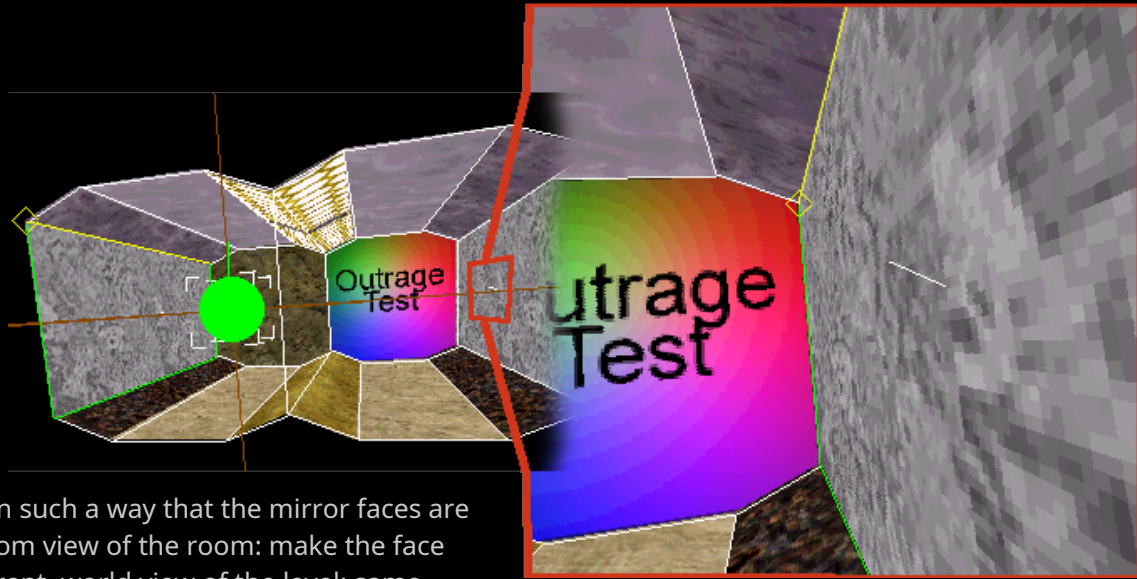
opposite each other (room view of the room: make the face

opposite the mirror current, world view of the level: same

make face current, Room->Place Room At Current Room, Room->Snap placed Room->Attach Room,

you know 😊 But then you still have to go there **Room Properties** of the attached room

Define the mirror, because it will not be saved with the room.



Lights calculate the level.mn3-ize, test, marvel. The screenshot shows that both surfaces reflect, but only once. The reflection shows the face behind the pyro, the 'Dirty Glass' texture, which is transparent (alpha flag!) and therefore lets us look out of the level, whereas in the back view this face reflects normally and the face in the main view also reflects the dirty glass texture. So you see exactly the same thing in the front and back views. Mirrored, so to speak.

SCORE: 0

►GUIDEBOT

040 mirror.rar

'Expelled' Mirror textures:

Sol BuildMirror1
Sol BuildMirror2
Sol BuildMirror3
MirrorTexture
C-SS Mirror
MirrorTexture2
Sol-Mirror-Breakable1
P-FacFloorMirror
P-She Floor Mirror

Mirrorable:

CloakSphere
Dirty glass
GreenMysterySurFace
hoard
InvisibleTrainBarrier
MarsForcedfield
MeshGlass1
mysterynoise
MysteryQuestion
PondRipple1
Sol-Buildwindows12
Sol-Glass-Breakable2
Waterfall1
GratedTest02x
... all with the ALPHA
flag



AFTERBURNER: 100%

SHIELDS: 098

ENERGY: 097

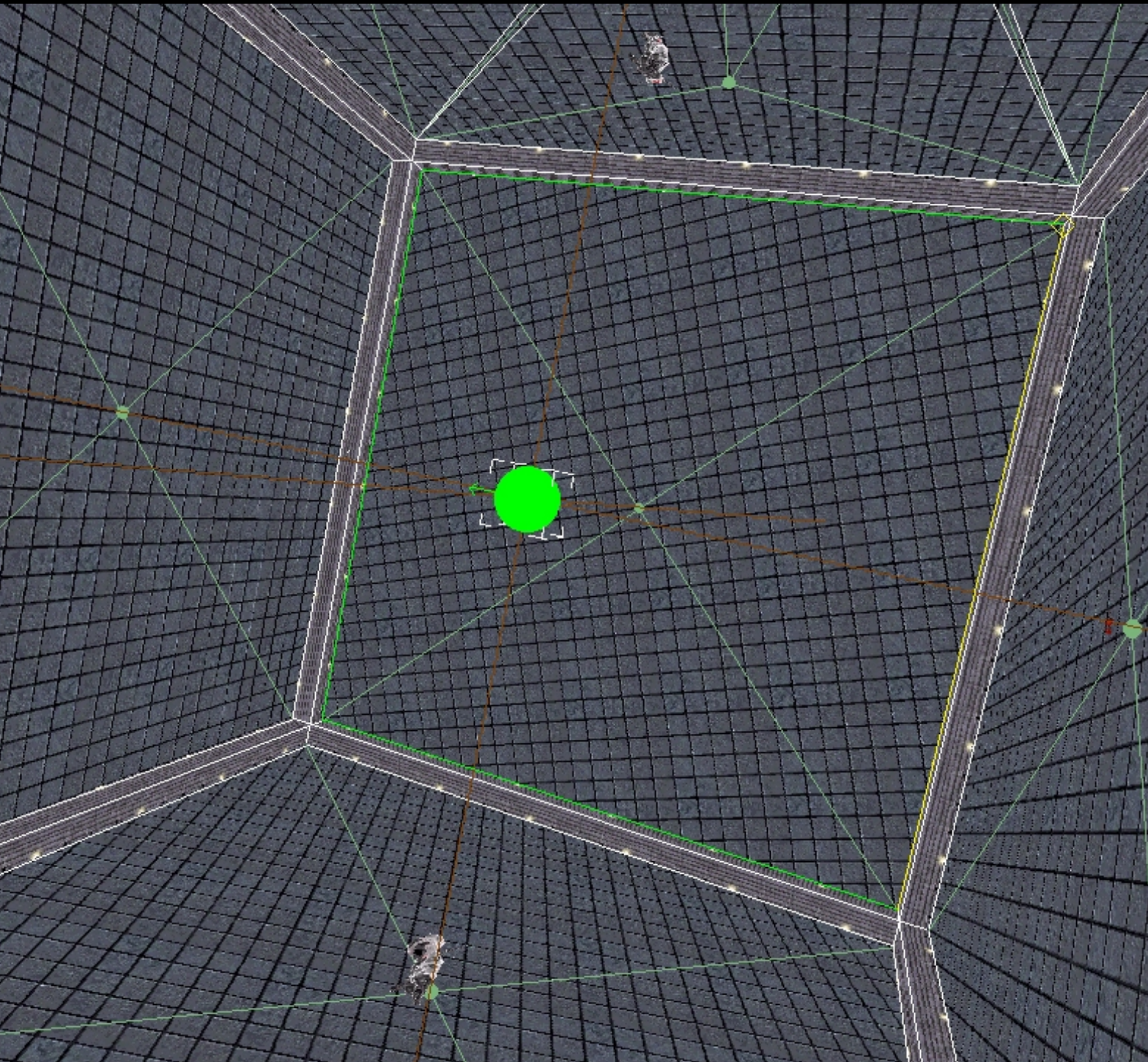


If you want a larger surface that consists of several faces to reflect, do the following: First make sure that they are all plane-parallel. Then give everyone the same texture and assign the mirror property to one of the faces. They will all reflect, but only the mirror surface will be opaque; So if you build something behind it, it will be visible through the other - also reflective - faces.

If the polygons are slightly inclined to the mirror surface, it still reflects, but it looks a bit, well, strange... try it yourself.

Update for v40

You can now see mirrors:



Similar to Trigger, only with the color **pale green**.

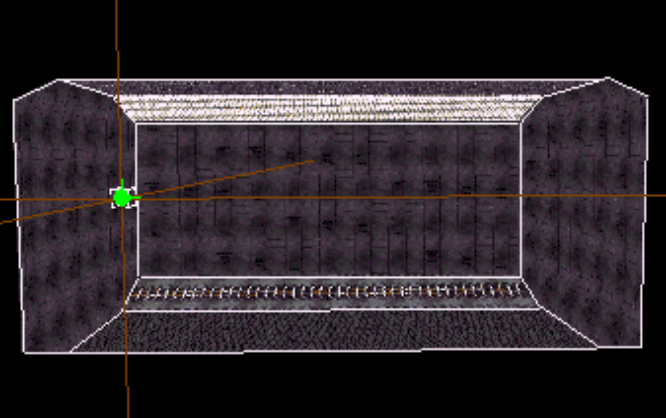
[Back to Section C](#)

044 - Room within a room <>

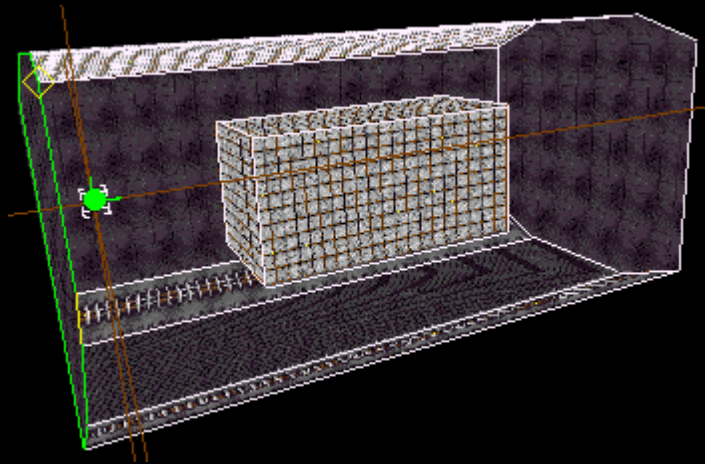
Fischlein

Open D3Edit and select File/New... "New Level - Default Room. Enlarges the room with the help of the **Expand Room** Function, approx. 35 times on the button. The room now has a size of approximately 100x100 units. Now texture it as desired.

Now press the buttons **Ctrl+G** To switch to object mode, click on the green player starting point and move (NumBlock: **2,4,6** and **8th**) this to one of the sides in the room.



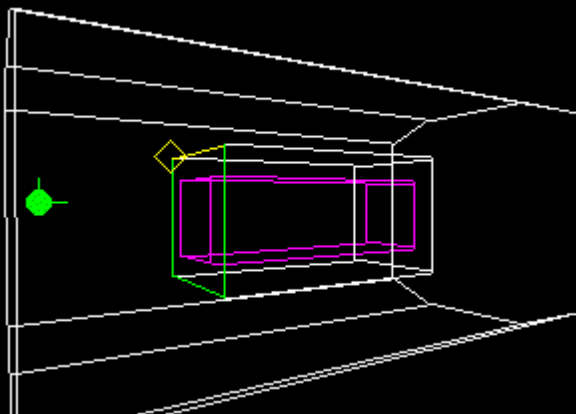
Now switch to Room View. I'm creating a simple room with eight corners of size L=100, W=50 and H=50, each units. (creates it next to the actual room) Once you have done this, mark the room in which you select it with the mouse (click and drag, first go into face mode). Now move it to the middle of the actual room (NumBlock: **2,4,6** and **8th**)



Ok, now choose from the menu File/New... New Room. We create it in a size of L=100, W=30 and H=30. This time the faces should point inwards. Saves the room as temp.orf and closes it.

Now open it again. I don't know why but you have to close it first and then open it again so that it works.

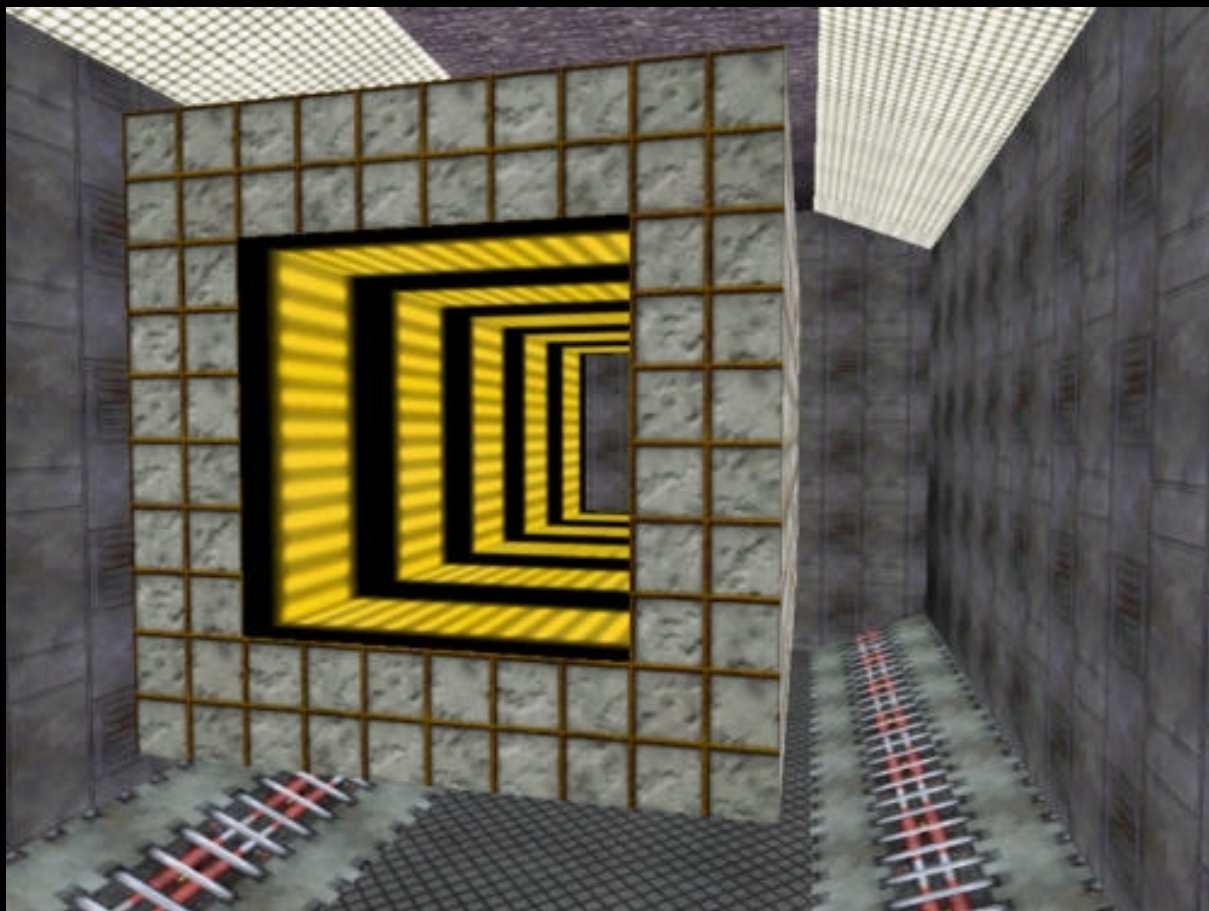
Now switch to World View and do one of the Side faces to the current face. (in the inner Room) Opens the Room menu and selects Place Room at Current Room (picture on the left). Now back into RoomMenu and selects Attach room.



Now click on the opposite face in the inner space and then the button **M** to mark the space. Now click on the opposite face of the outer room, go back to the Room menu and select joinRoom. In the following dialog box you will be asked whether you want to center the space in the middle of the face, we click on No.

Click on the inner room and go to the menu Windows, Room Properties and puts a check mark Refueling.

You now have an energy tanker in the room.

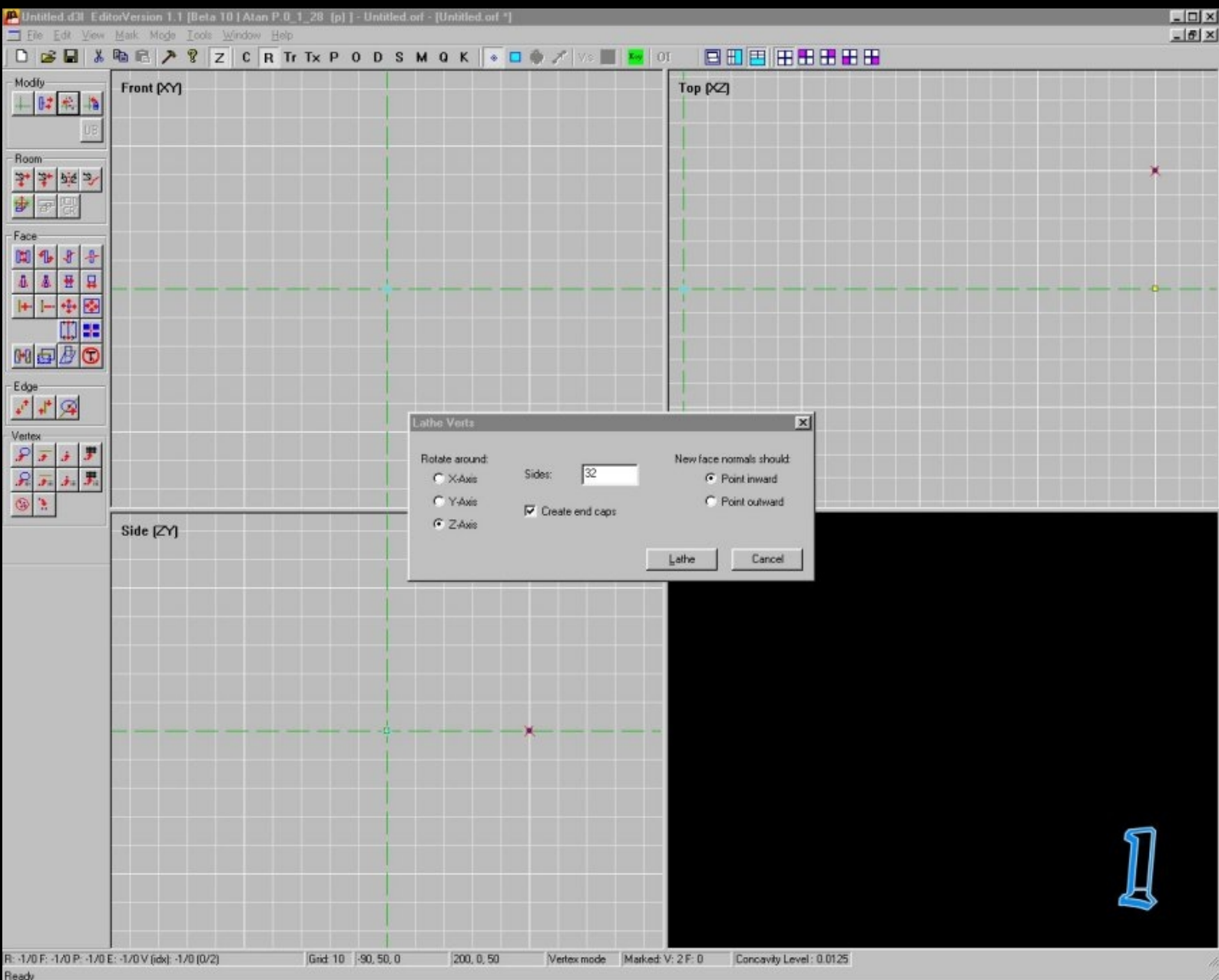


[Back to Section C](#)

045 - Make a sphere

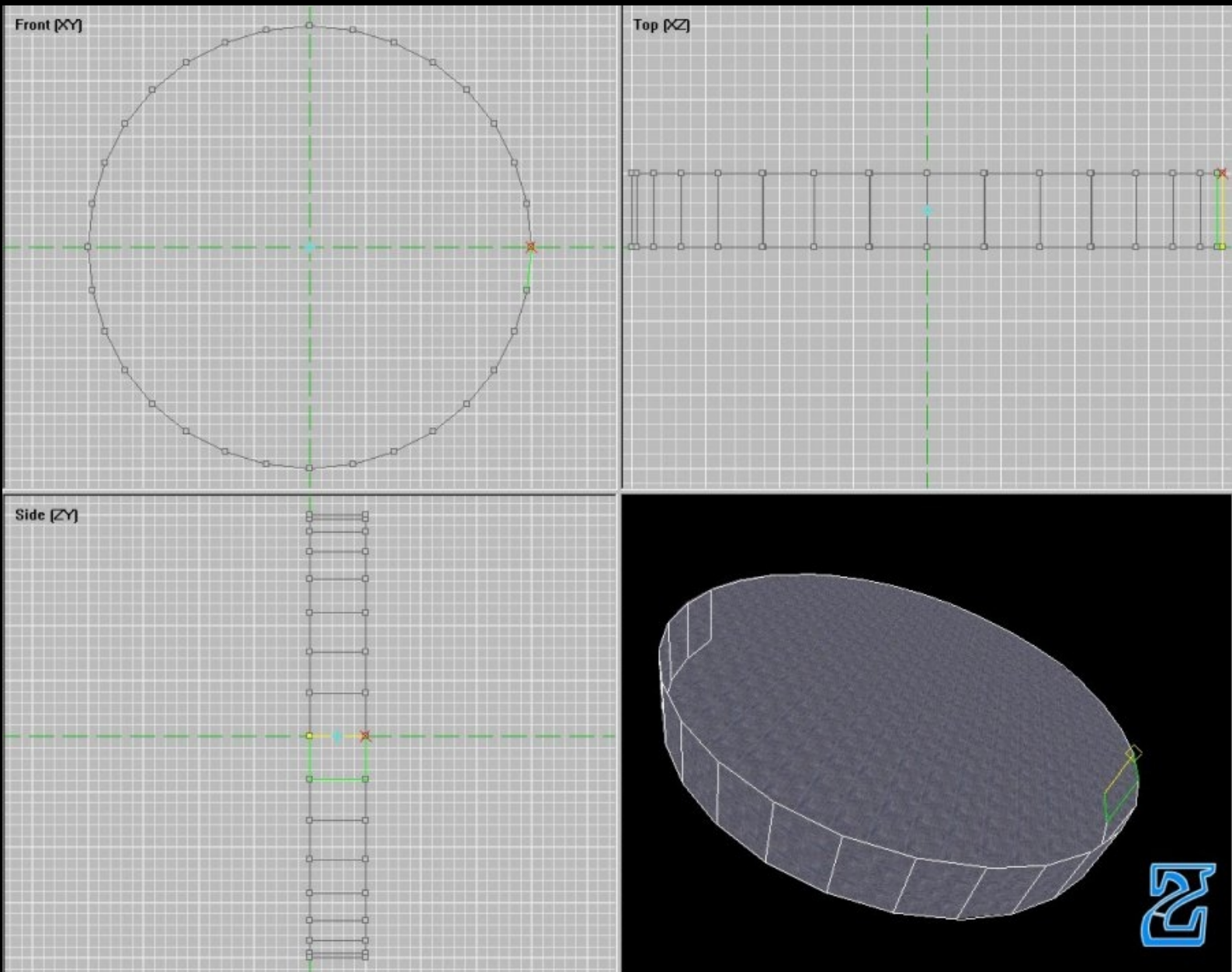
WillyP

Place vertices

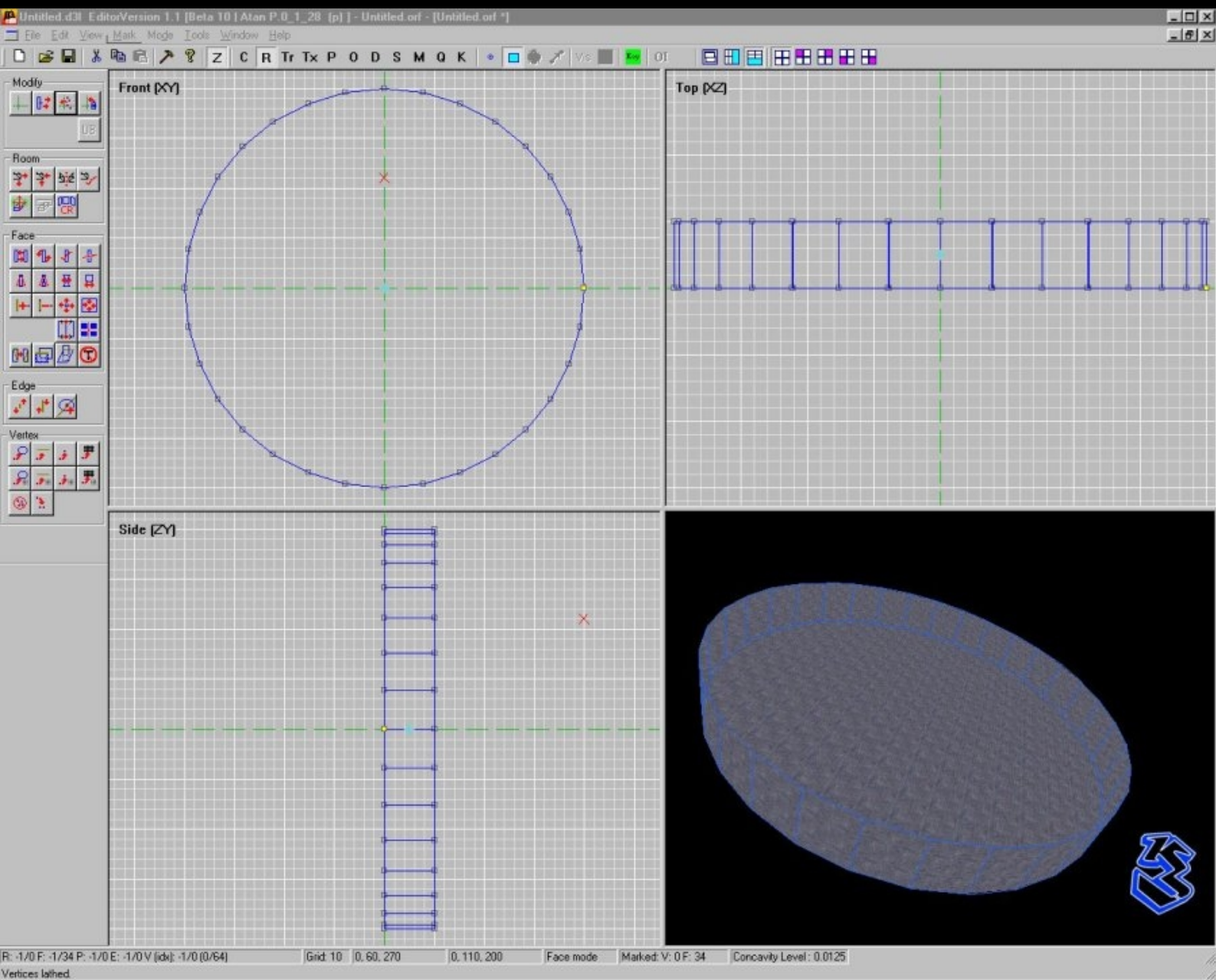


Lathe needs at least two verts for it to execute.

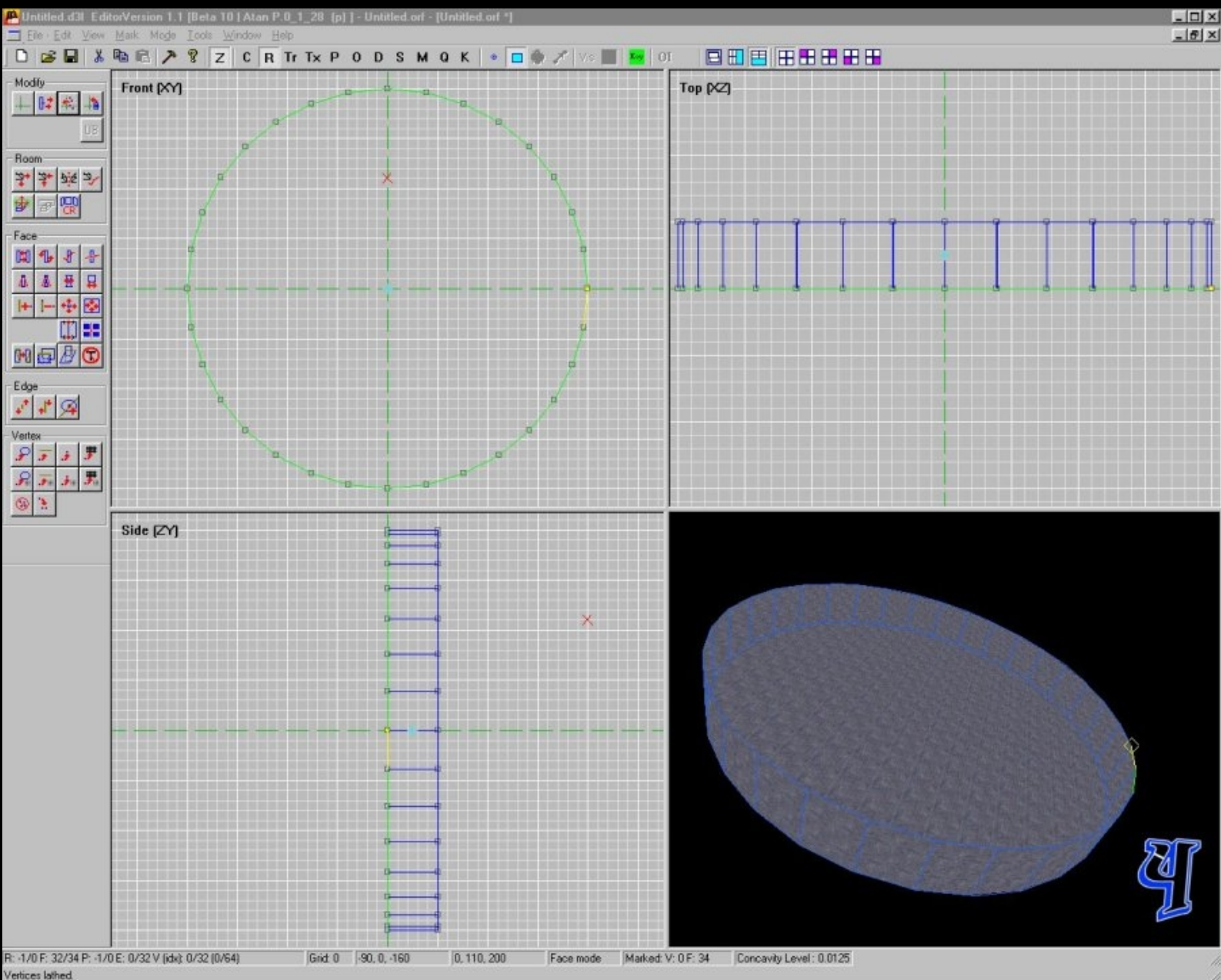
Cylinder lathen



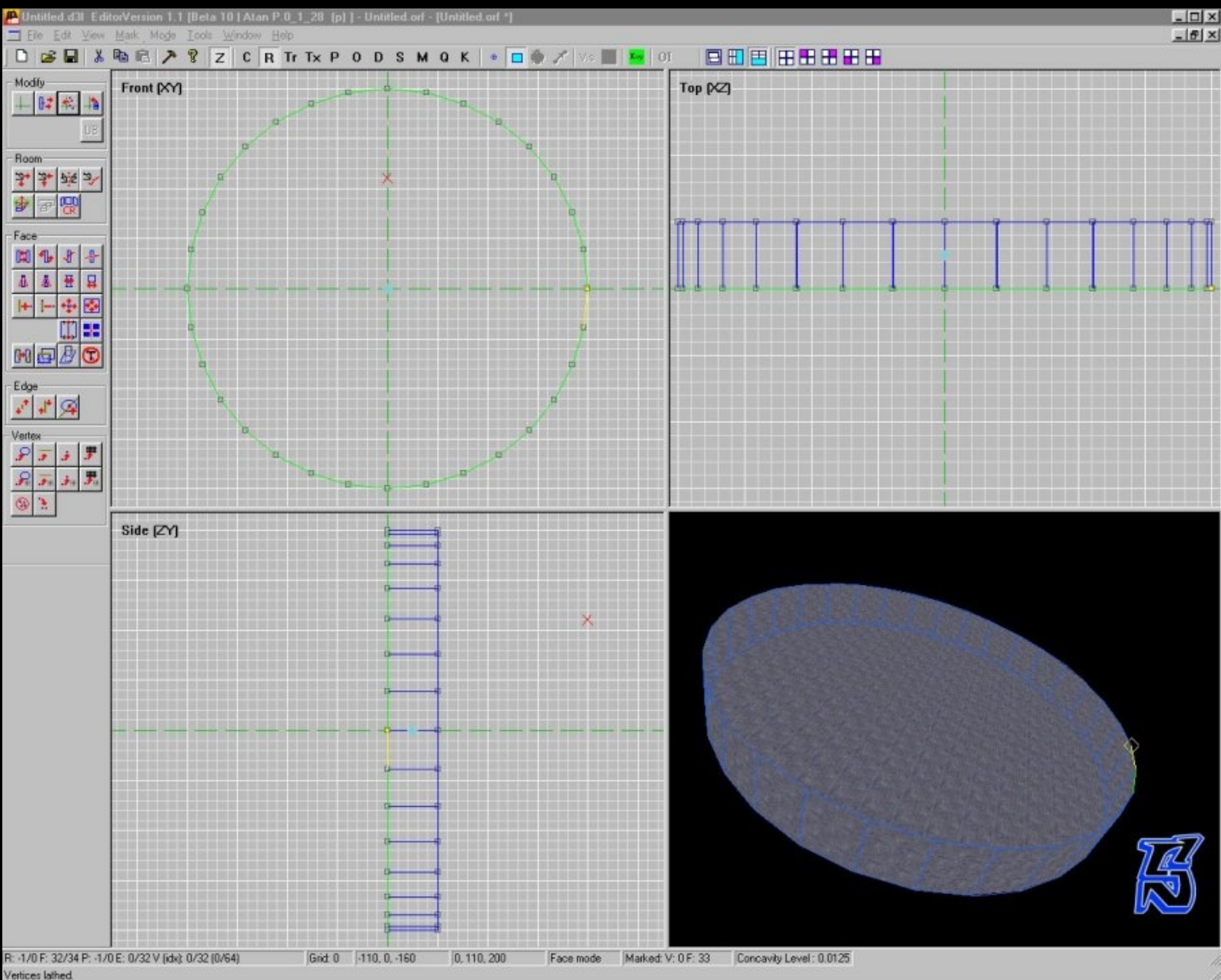
Mark all faces



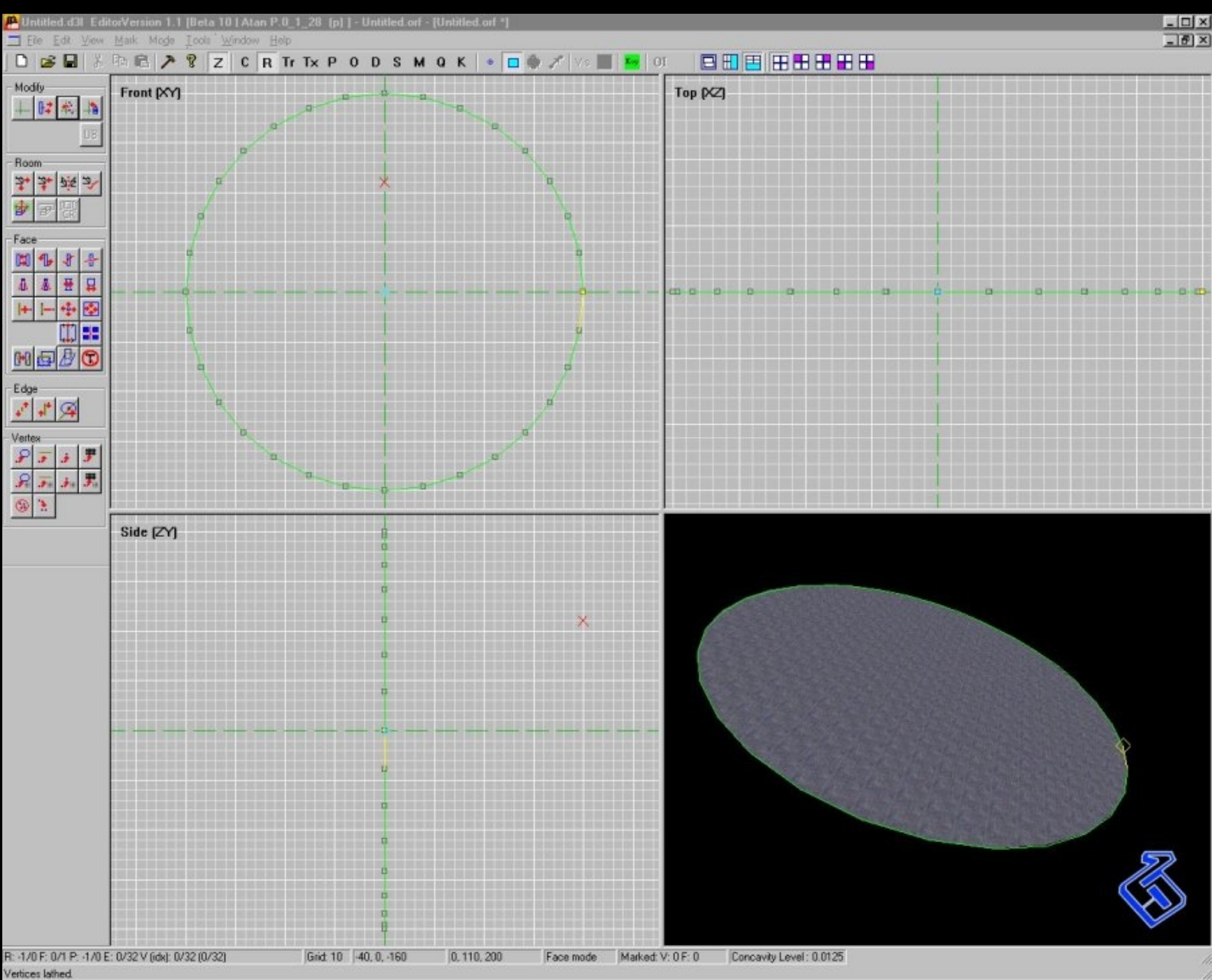
Select Face



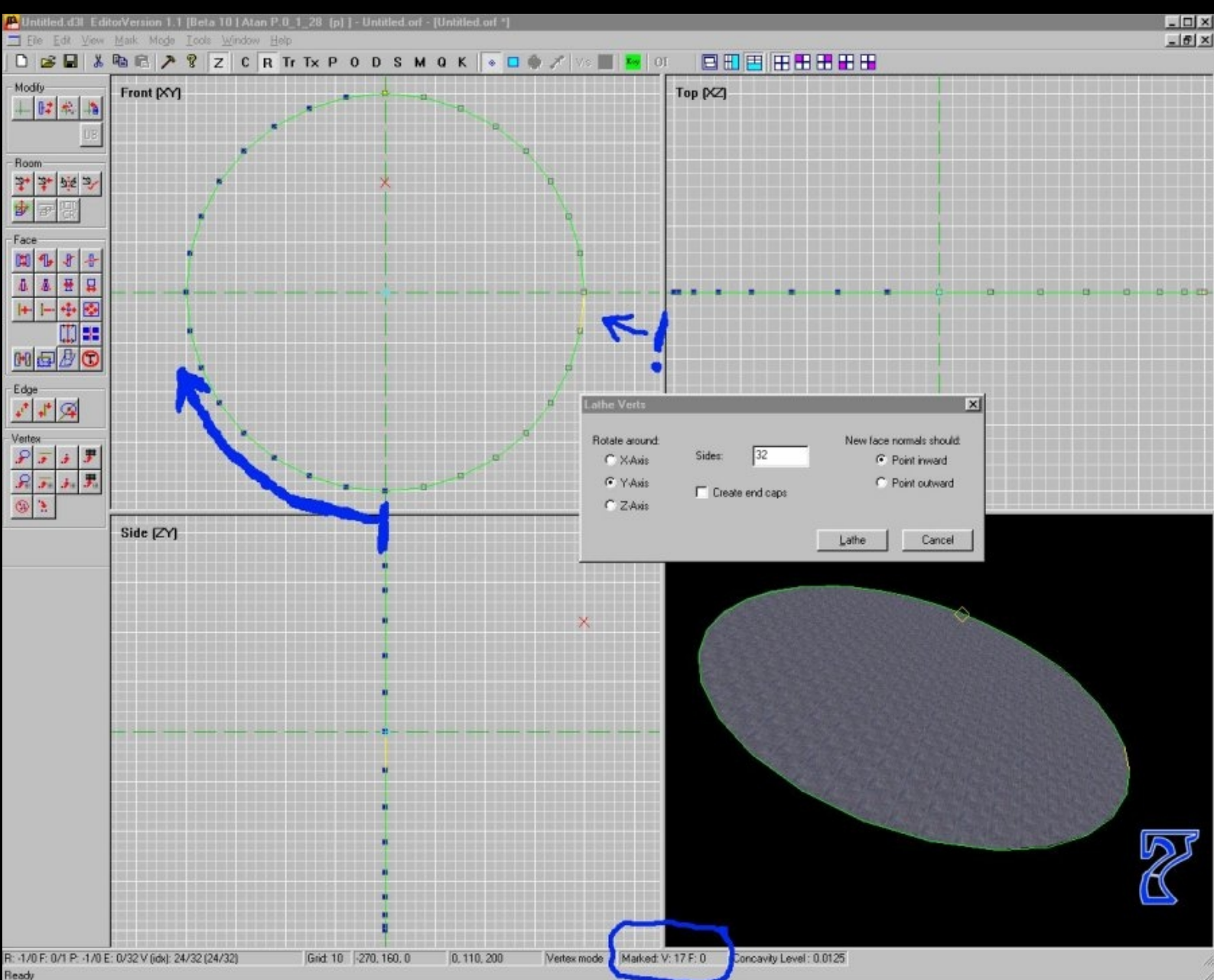
Mark Current Face



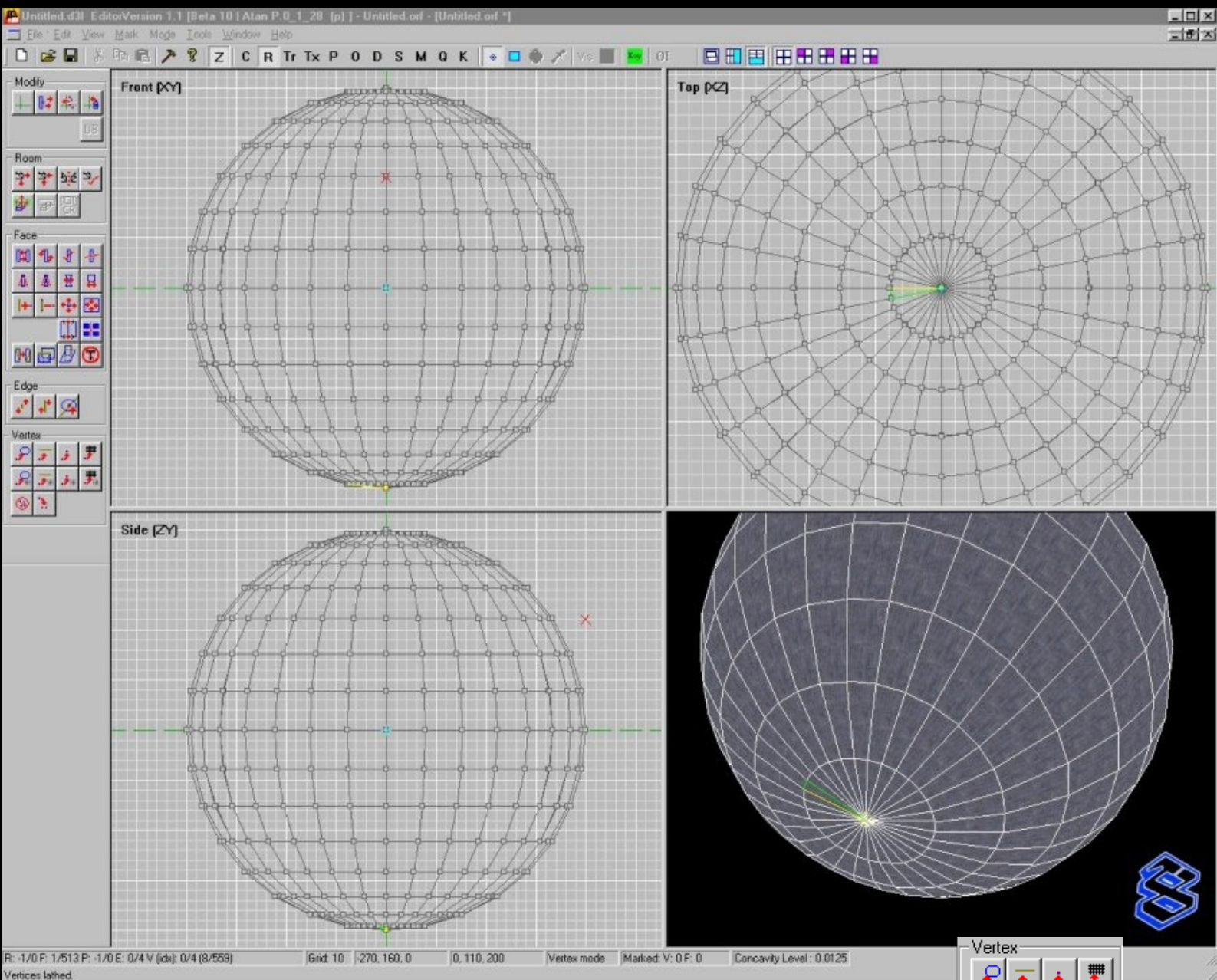
Delete marked faces



Mark vertices one after the other in order



Vertices Lathen...



...Verify and Remove Extra Vertices.

Back to Section C

Verify Room

Room check results:

Found 0 bad normals
Found 0 concave faces
Found 0 degenerate faces
Found 0 duplicate faces
Found 0 non-planar faces
Found 0 duplicate vertices
Found 0 duplicate face vertices
Found 0 unused vertices
Found 0 t-joints



NOTE: This room has no recorded errors, but check the message window for texture counts.

Close

046 - Complexity made easy <>

Papacat

In the next few pages I'll walk through building a room using Lathe, Extrude, Rotate and Copy/Paste.

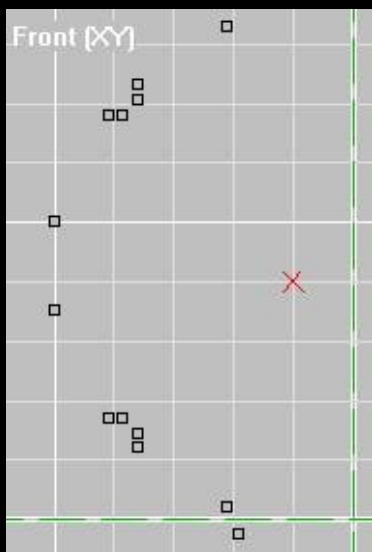
You'll see these tools in action and a few techniques on how to use them to easily create complex structures.

I'm going to recreate a room from Vortex1 so you have an idea of what we're going to try.

The idea is to learn the old 'the right tool for the right job' concept. As with everything, there is more than one way. These are techniques that I have found to be effective.

We will have the Vortex1 room with the windows at the viewing ports on either side.

I have one too.zip included. OpenSctut01.orf, there the verts are laid out and ready to go lathen.

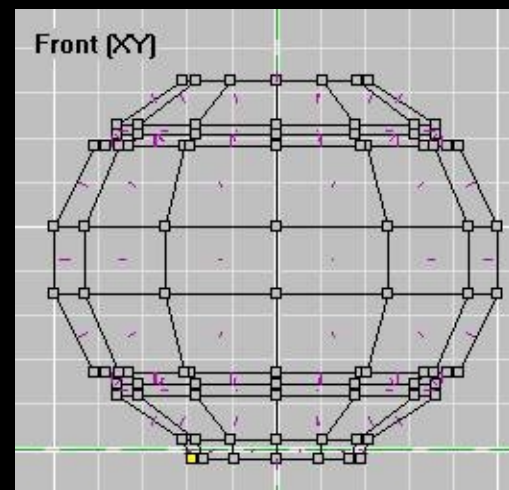


On the left I have laid out the verts, which I will lathe to the sphere. I also incorporated the tubes. Don't forget to always insert verts consecutively. You can't stuff verts into the middle of a chain; the lathe will be a mess. I went from bottom to top.

Mark all the verts, set 12 sides, normals pointing outward and lathe around the Y axis. Save your space.

Remember, there is no undo so save often and sometimes with a new name.

You can always delete extras if you are happy with the results.

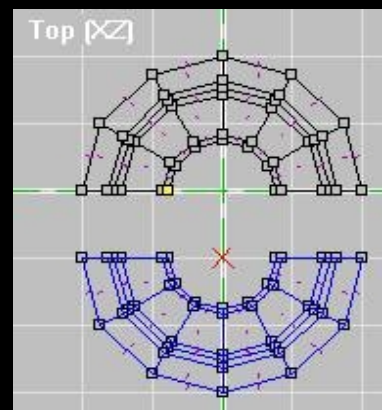
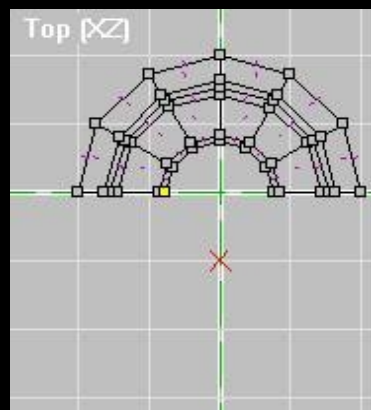
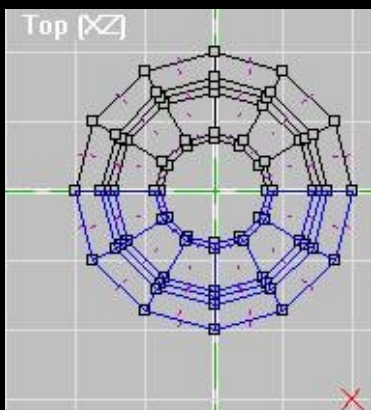


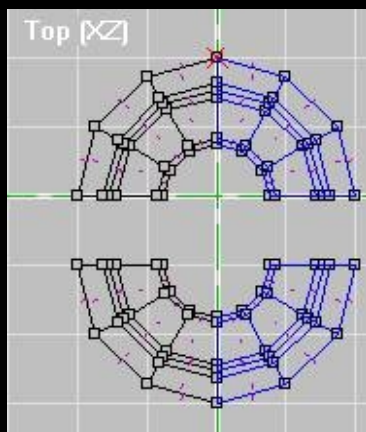
Now we will Copy(**Ctrl-C**) and Paste on top(**Ctrl-Shift-V**) to divide this into quarters. First make the top view active, then set the grid to 50. Go to face mode.

First, mark the entire bottom half by drawing a frame over it. Then copy that.

Mark them again and delete them.

Then use Paste on Top. While the inserted share is still highlighted, press the cursor down to move it 50 units.

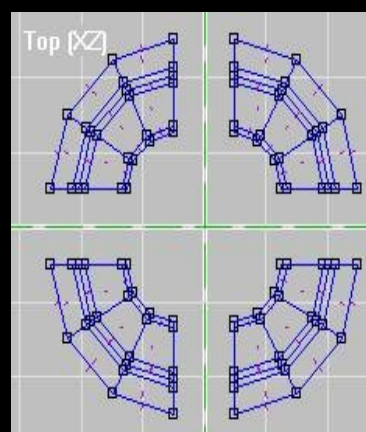




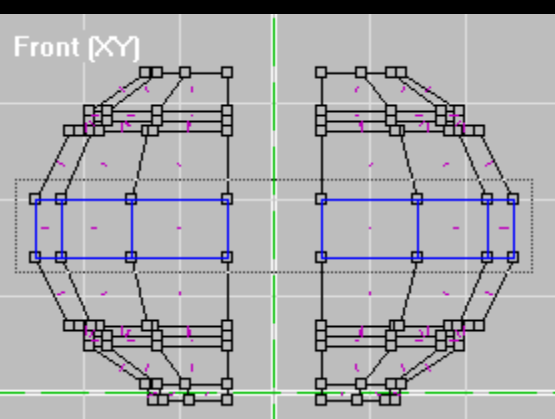
This is a good technique for segmenting rooms. Now do the same vertically.

Select one half (left), then copy it. Mark again, delete, paste on top and move 50 to the right.

I want to be centered at zero, so set the grid to 5, select all (M) and move 25 units left and up. It should look like the one on the right.



Before we get too far, let's slap on some textures. While all faces are marked do



using them from **To Marked** to 'LWwall'. Who here

If you've been paying attention, you'll see that I've had the faces the wrong way up until now.

Oooooops... Mark and flip your faces when your room also the wrong way round is.

Deselect everything (U) and draw a frame to mark the central faces. Make it 'LWPend'. Demark.

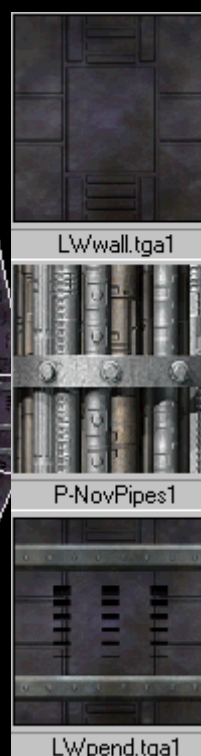
Mark the faces as shown on the left in the picture below.

Make her 'P-

NovPipes1'. The texture

was the wrong way around in mine. Check yours and follow up if necessary **Align**, put it on **Marked** faces and **Red90**. It should then look like the one on the right.

We'll align the textures later. I just wanted something to do

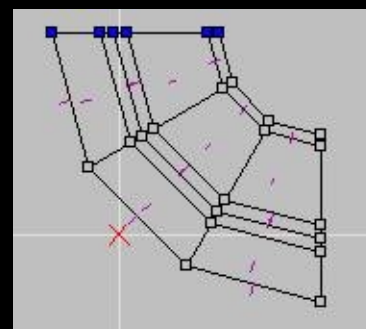


to get a little depth.

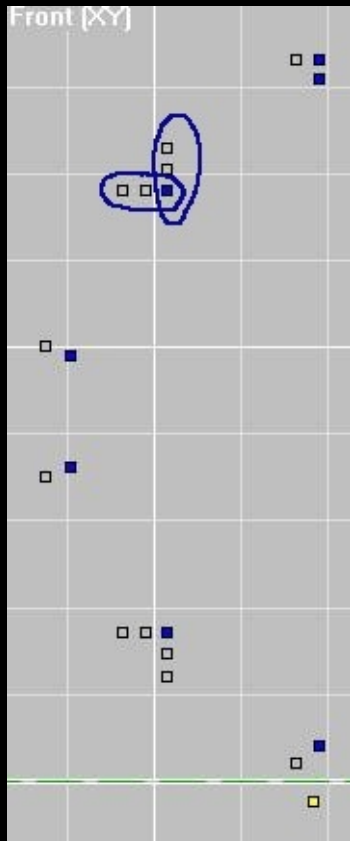
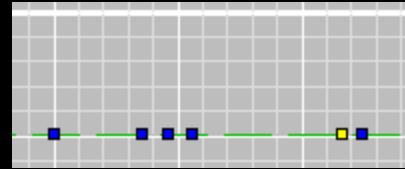
Now we will make the metal frames for the sphere. We will use Copy, Paste on top and Extrude. By copying parts of a room and using them as templates to build other parts, we have an easier time when it comes time to bring them together. Current file at this time is [Sctut02.orf](#).

First make the top view active and change the grid to 100. Select the verts like on the right and copy them. Choose Paste on top, then move the new verts 100 units to the left. By using Paste on top I know that the part will go exactly into place when it's finished. A nice sized grid keeps it out of the way of movement to work. Then move your top view so that the new verts are visible.

Navigation tip: Press that **Shift** button, click in the top view, hold the mouse button at the bottom and **DRAW** the window.



Here on the right are the new verts. We're now going to add some verts to make faces so we need the reference frame on the same layer. Set the grid to 5 and move the reference frame to the line by **Ctrl**press while clicking on the line with the mouse.



Make the front view active and move to the verts (**Shift-drag**).

On the left you see unmarked verts. That's where we'll start. Set the grid to 1 and set the verts you see marked here. Instead of using the mouse, move your cursor (that **X**) with **Ctrl** cursor keys. This makes it easier to be accurate.

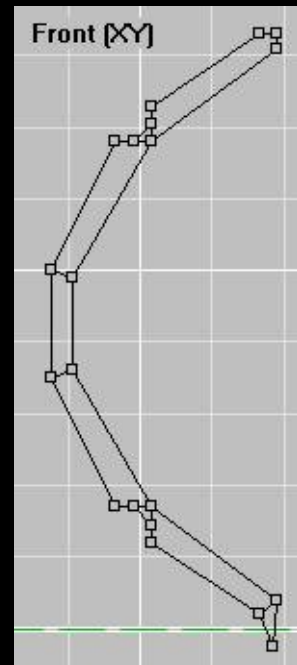
I've circled the verts on the left to warn you. Make sure they are in a straight line so you don't get concave faces.

Deselect all verts before you start inserting faces. I recommend not marking more than four at a time. This allows you to **Twist Face** use when the face is twisted instead **Swap Verts**. Then connect all the faces you need to connect.

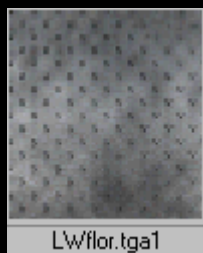
Also watch the top view to make sure the normals are all pointing in the same direction - up.



When you're done, it should look like the example on the right.



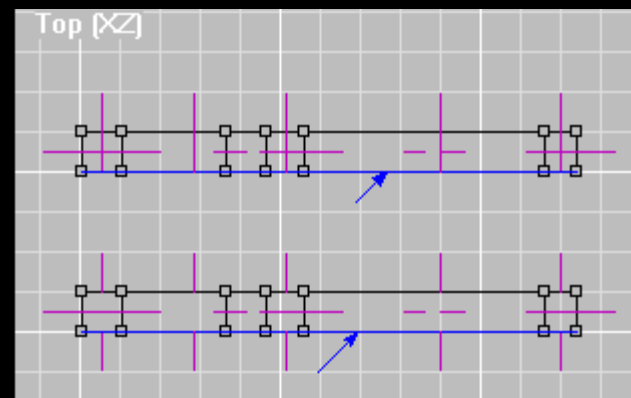
Time to **EXTRUDE**. Before we do that, keep in mind that all new faces will be textured the same as the parent face(s). Select the faces and change them to LWFlor.



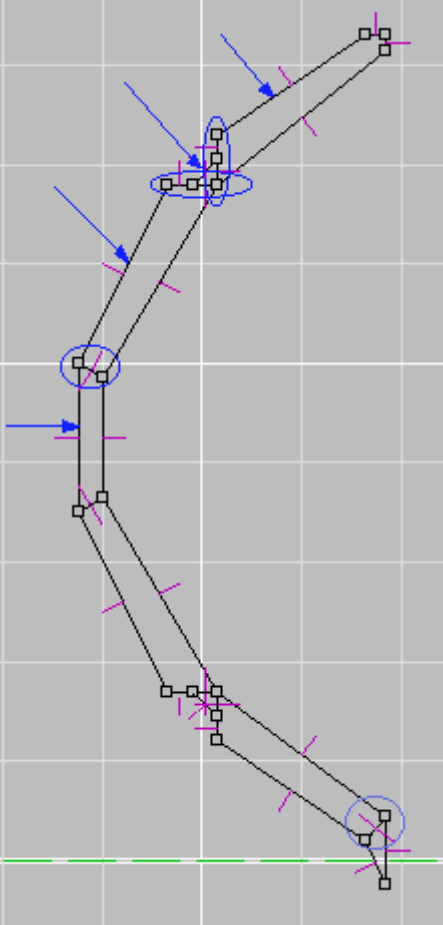
You've made sure that all the normals are facing the same direction - up. Mark all new faces, choose **extrude** and put it on: Extrude from: Marked Face(s), Extrude along: Normals, 5 units, Point outward. It is important that you make sure that Delete base face is NOT ticked. then extrude.

In the top view, your metal frame should look like mine here on the right. As you can see, the parent faces are still checking above. They are still marked from extrusion, so select the flip tool so they are facing down.

Making a multiface extrusion does two things that you need to be careful of. One is extra verts and the other is double faces. This is because D3Edit still treats each face individually when you extrude. This means that a side face is created for each seam between the original faces. If the original faces were on the same plane, the two new side faces will also be on the same plane but have their individual verts there.



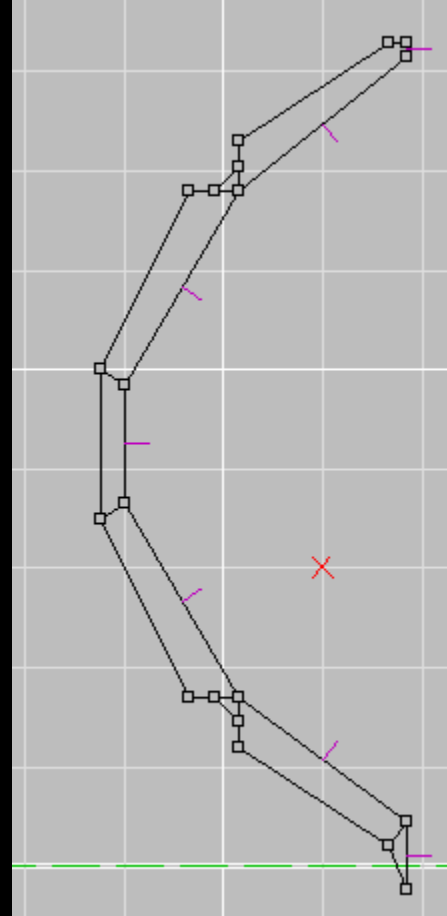
So before you continue: **Remove Extra Verts**.



On the left I have circled a few examples of double faces that were created during multiple extrusion. Mark and delete these.

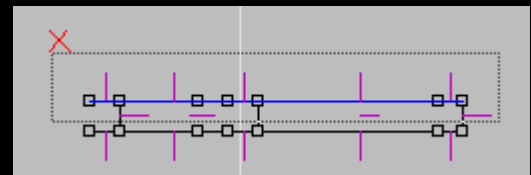
The arrows point to some faces that will be outside the shell. Delete all of these faces. When you're done, the piece should look like the one on the right. Save!

A frame is ready. Now about the space between the frames. We will make it so that there are no tubes in between. We're going to copy what will be the inside faces of the frames, change where the tubes go, extrude and then delete the faces that we don't need. It may seem like we're not making any progress here very quickly, but once we're done, things will come together quickly and easily.



The current status is [Sctut03.orf](#).

In the top view, change the grid to 100, select the faces as shown on the right. Copy, paste on top, and move 100 units to the left.



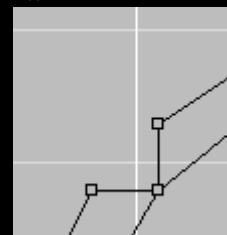
Now we will eliminate the tubular shape. Go to the front view and delete the small faces from the top and bottom two Tubular shapes.



Now remove those Verts from both Areas like shown..




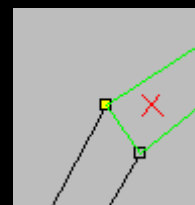
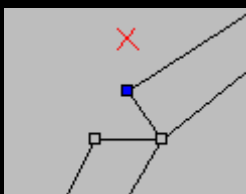
Don't forget that when using **Remove Vert from Current Face** the vert is still there, it's just taken out of the edge. Use **RemoveExtraVerts** to delete them..



Highlight one of the back verts and move it halfway to the opposite one as shown below. Since everything is still on the grid, it would be as simple as moving the other vert there.

You'll learn that here **Snap Marked Verts to Current Vert** tool to use. Do it other Face and the target vertex current.

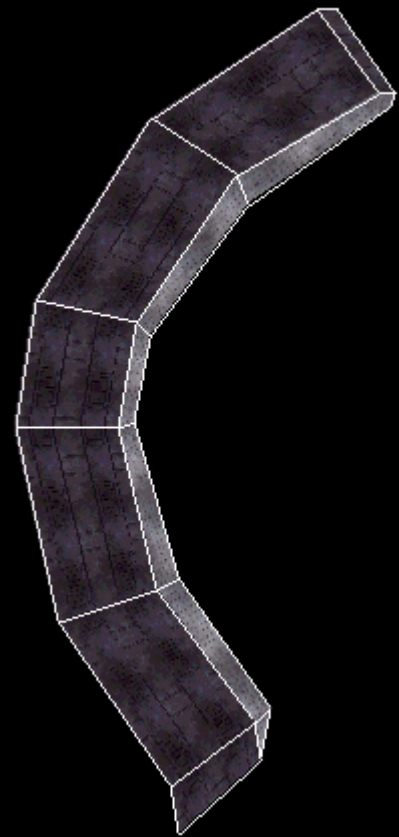
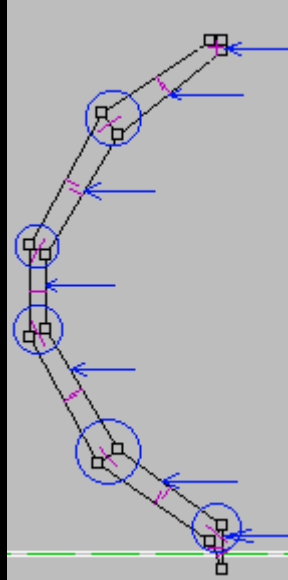
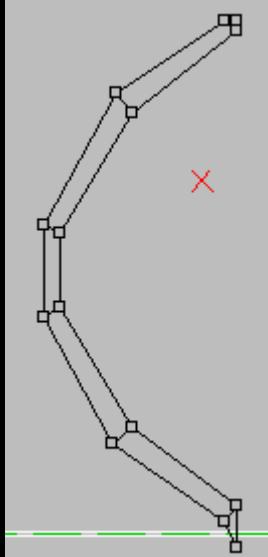
Then choose  (**Snap Marked Verts to Current**) and produce a result like below. The tool moves the vert, but you still have to do Remove Extra Verts so that the edges are connected. Do this in both places.



The new face group should look like the one below. Select them and extrude (in the top view) 40 units upwards, Normals pointing inward.

Mark and delete the faces that I circled and marked with arrows. Mark the wall faces and texture them with LWwall.

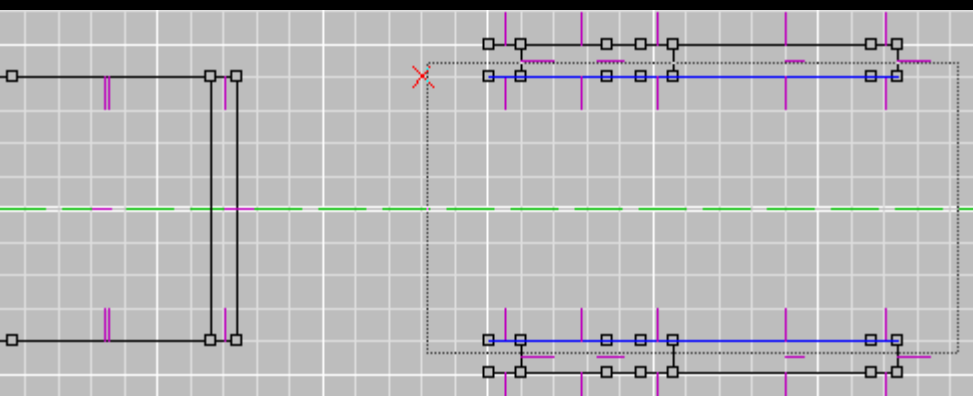
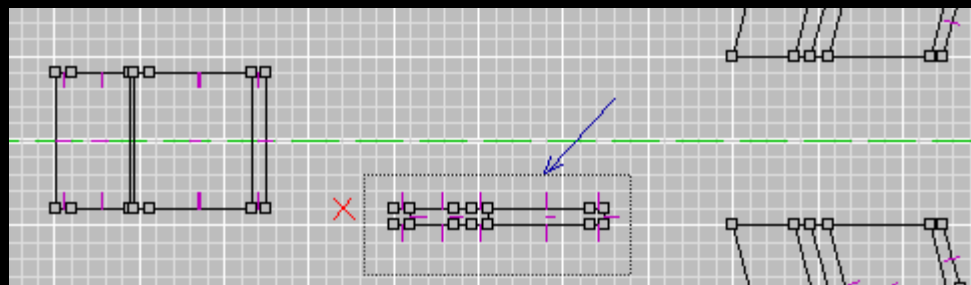
It should then look something like this:



The current status is now Sctut04.orf.

On the right you can see where we are now

Now mark the faces of the frame part as shown on the right and copy that.



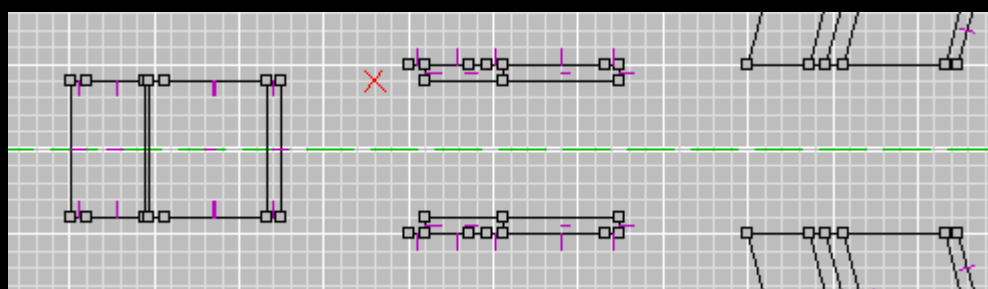
Paste on top (**Ctrl-Shift-V**), then move up 45 units.

Unmark everything.

Select the inner faces and delete them (left)

Your room should now look like this (right)

Now we're ready to put it all together.

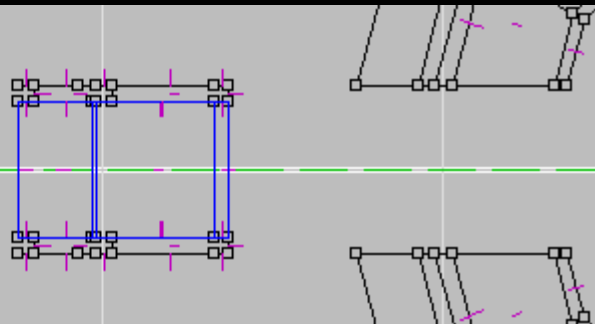
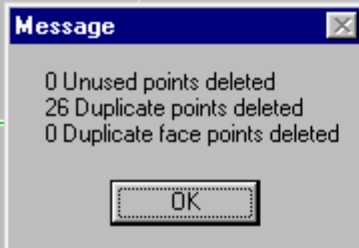
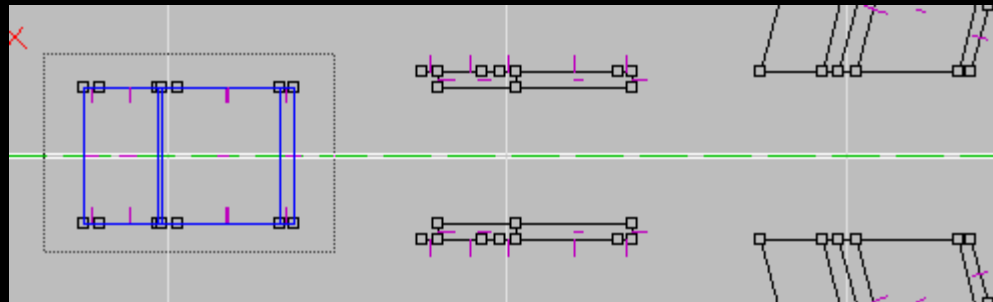


The current status is now Sctut05.orf.

One way to make sure the pieces come together correctly is to check for duplicate verts. When they meet you get duplicates. If not, then not. Press Remove Extra Vertices so we know we're starting clean.

First, activate the top view and set the grid to 100.

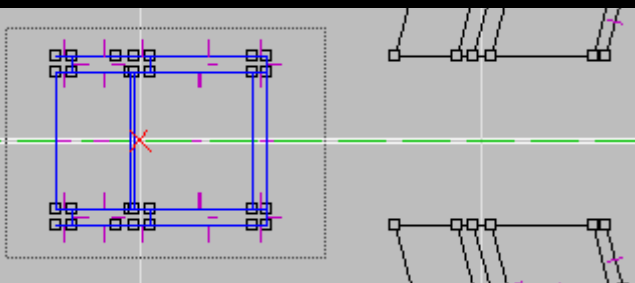
Then select the part of faces shown on the right and move them 100 to the right (one movement).



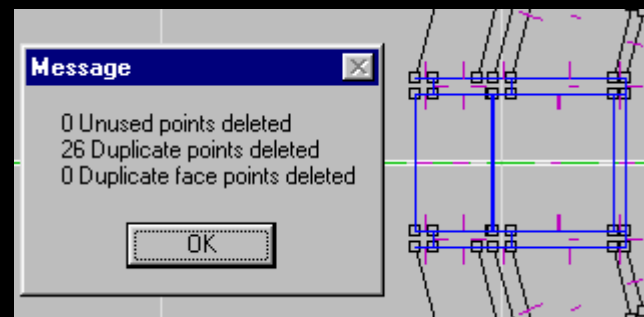
Then **Remove extra verts**. As you can see on the left, all verts met. By using copy in the previous steps we have

ensures that the verts are laid out exactly the same. By the use of

After pasting on top and then moving it around, we made sure everything stayed aligned and went together easily.

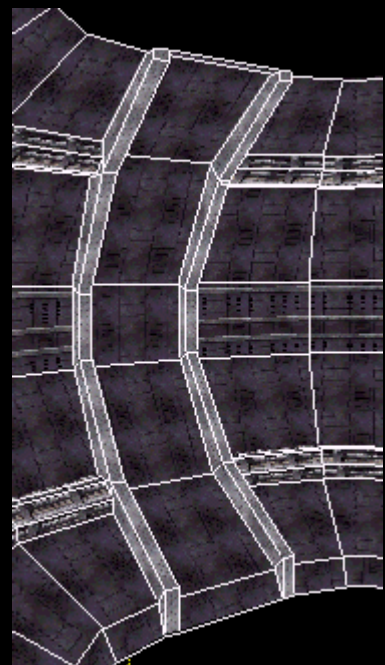


Mark the part Faces like left shown and move them 100 units to the right (one Movement). Then click **Remove extra verts**.



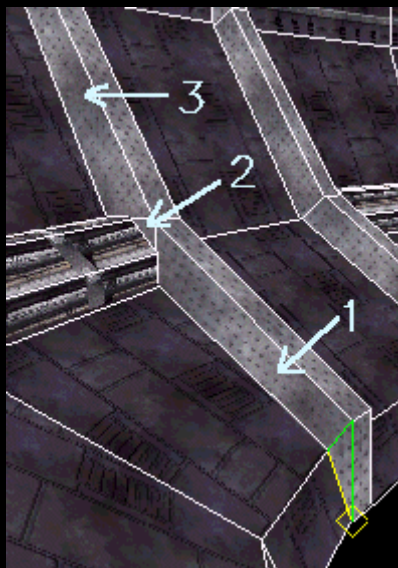
On the right we see the result so far. Looks good. Now let's align the textures. We don't want to copy things and then have to do four times as much.

While the entire new part is marked as shown in the images of the 2d views above, go to Align in the Texture bar, set it to marked Faces and choose Default UV's Now we are ready to align textures.



The quickest, easiest way to align textures is this **Control click method**. This is done in the 3D window that on Texture with outline setis. Start by doing a face current as shown on the right. Then hold down Ctrl and click a face that shares an edge with the current. This face aligns itself with the current and becomes the current itself.

As a side note, you can change textures by holding Shift and clicking a face.



This applies the current texture in the Texture palette to the face you click.

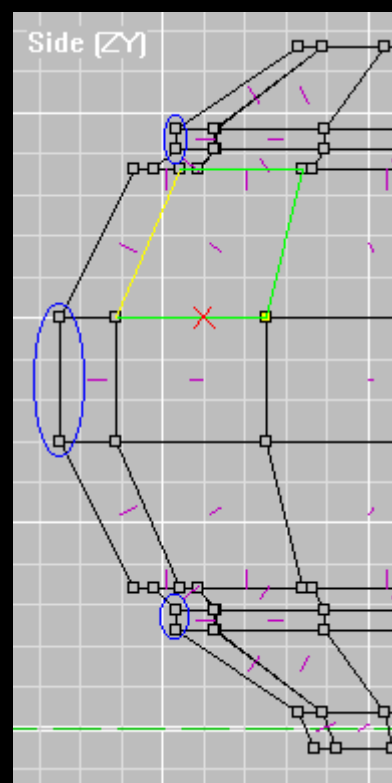
Just hold down Ctrl and go from face to face as labeled on the left.

Make both frames and the walls between them. Rotate the view so you can reach all the faces you need to. Don't start with the round part or the tubes yet.

Lathen has an interesting property related to texture alignment. Any faces that are parallel to the axis around which you lathe are nice and perpendicular to each other (curled to the right). This makes it possible to align them with each other. You can also align faces that are in a vertical row parallel to the axis. Once a face is at an angle to the axis, you can't align it side to side (well, you can, but you'll get weird results).

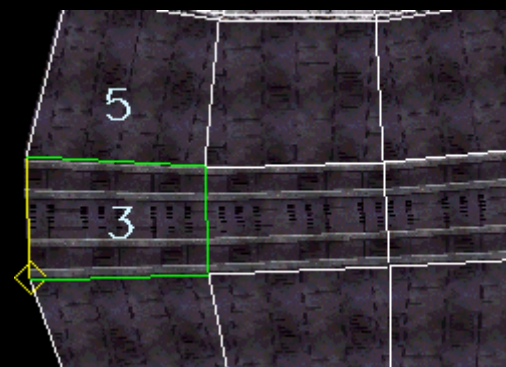
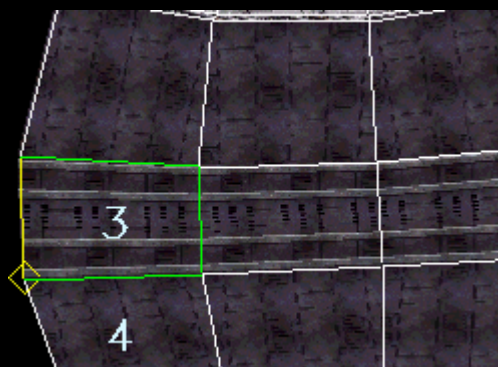
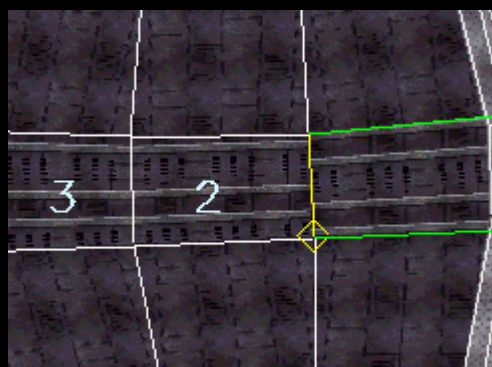
This makes round walls and tubes tricky to align. You really have to think about the order in which you click them. For our sphere, you'll want to align the central faces first, then work up and down from the central faces.

We will now focus on a quarter of the sphere.

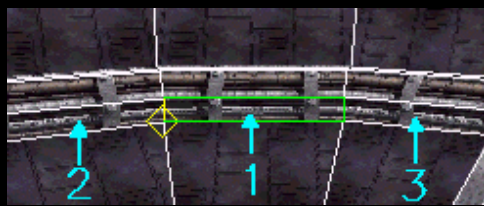


First do that face current, then align the face under 3. that is shown in the picture and use the align box to align it like this. Then Ctrl-click to align faces 2 and 3.

Power Face 3 again and align Face 5. Do the other two rows starting from the central faces work.



We make the tubes similarly to the above. Select a row of faces, align them along the length of the tube, then go back and work around the rows of segments, always starting from the same row. In our case we will work from the center outwards. We still know from the picture (blue circles) where the faces are that can be aligned from side to side.



Make 1 current, **Ctrl-click** 2
 Make 1 current, **Ctrl-click** 3
 Then up:
 Do 3 Current, align 4, then 5 Do 1
 Current, align 6, then 7 Do 2
 Current, align 8, then 9



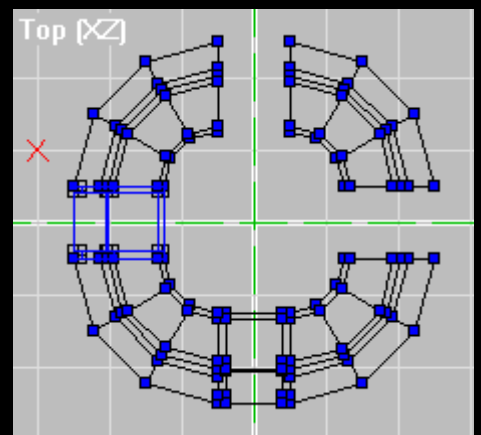
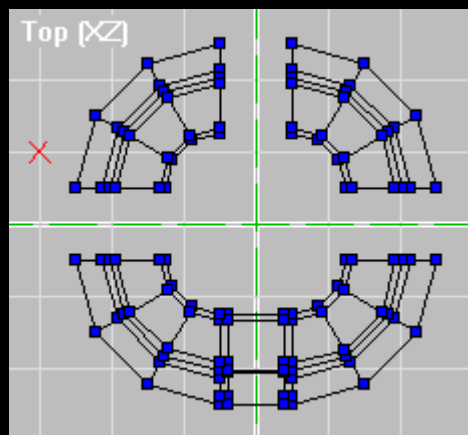
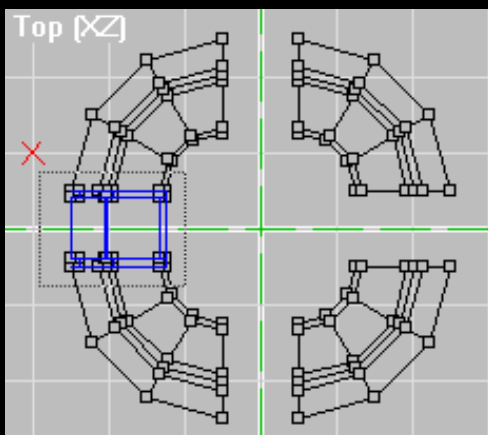
Oh, and before I forget, go to the wall segment between the frames and align that too. Current the middle face, work your way down, current the middle face again, then continue upwards.

Ok, now let's get the room ready; current status is now `Sctut06.orf`. There are two ways we can do this.

In the top view, select the
 Faces as shown below.

Ctrl-R(->vertex mode)
M(mark all verts)
 Eight times **3**(Rotate)

Ctrl-F(->face mode)
Ctrl-Shift-V(paste on top)



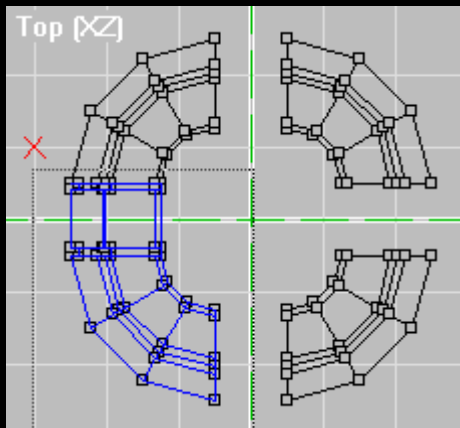
Keep going through the cycle **Ctrl-R, M**, Rotate (**3**), **Ctrl-F, Ctrl-Shift-V** until it is finished. If you have that, do it **Remove Extra Verts**.

Remember, once you copy something, it stays on the clipboard until you copy something else. That's why we can copy once and paste as many times as we need.

If we use this method, we would still have to go back and texture/align the three round sections that we didn't do before. But since I 1) hate aligning textures and 2) I'm lazy, I'll show you an easier way.

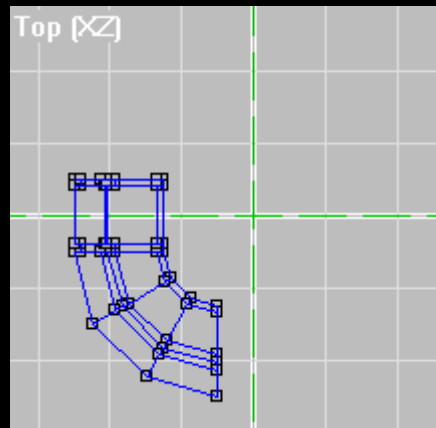
In the top view, select the Faces as shown below.

Ctrl-C(copy)



M(mark all faces)
delete all faces.

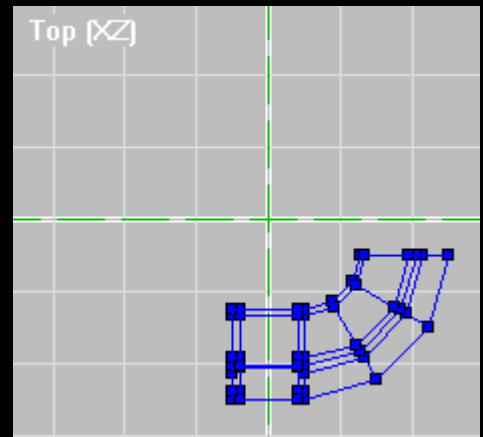
Ctrl-Shift-V(paste on top)



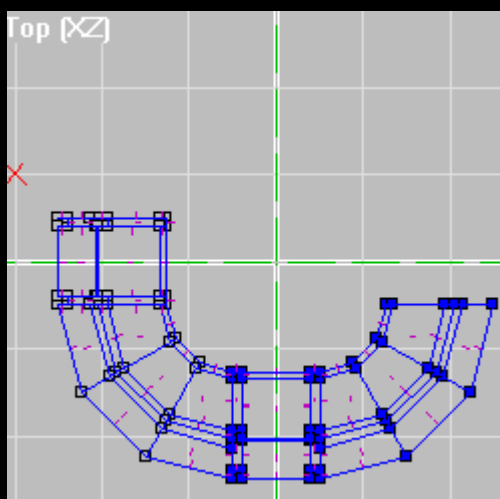
Ctrl-R(->vertex mode)

M(mark all verts)

Eight times**3**(Rotate)



Ctrl-F(->face mode)
Ctrl-Shift-V(paste on top)

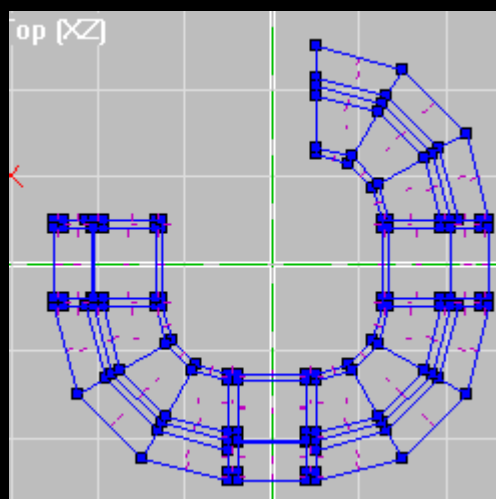


Ctrl-R(->vertex mode)
M(mark all verts)

Eight times**3**(Rotate)

Ctrl-F(->face mode)

Ctrl-Shift-V(paste on top)

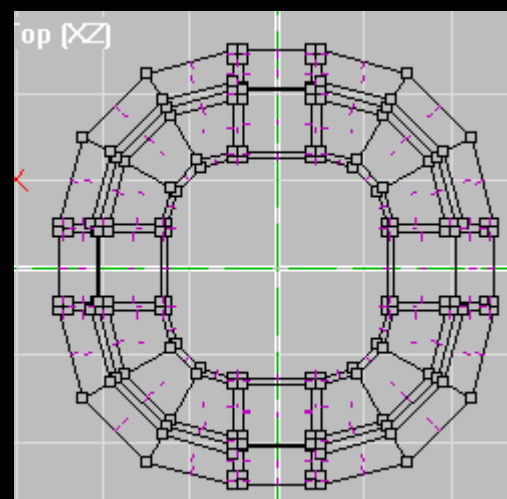


Ctrl-R(->vertex mode)
M(mark all verts)

Eight times**3**(Rotate)

Ctrl-F(->face mode)

Ctrl-Shift-V(paste on top)

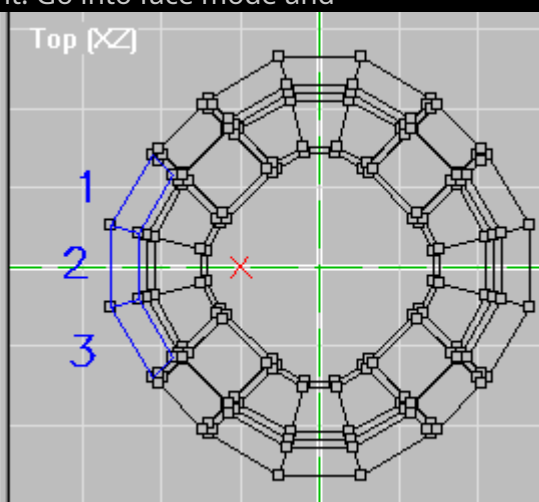
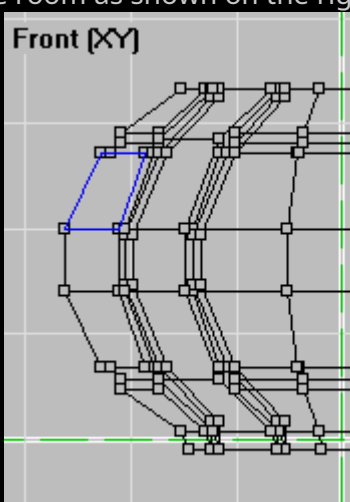


Remove Extra Verts and save. Current status is now Sctut07.orf.

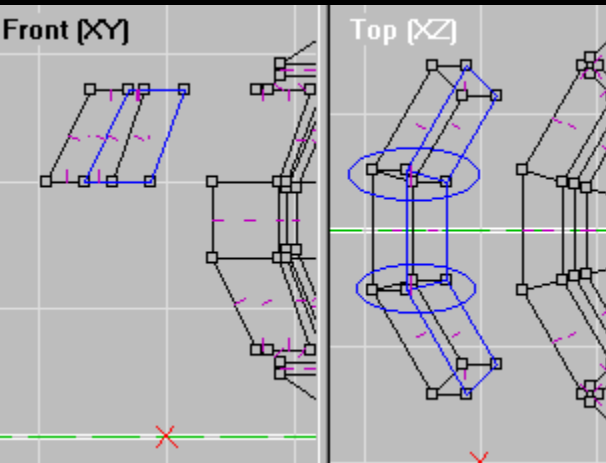
All that remains to do now is to cover the window projections and floor or ceiling.

First, the window projections:

Highlight all verts and rotate the entire room as shown on the right. Go into face mode and mark the three faces shown. In this case it is best to do the face current and then with the **Space**-Bar to mark. Copy, select the same three and delete them. We set them Window projections to the right side too, so delete these three faces too.

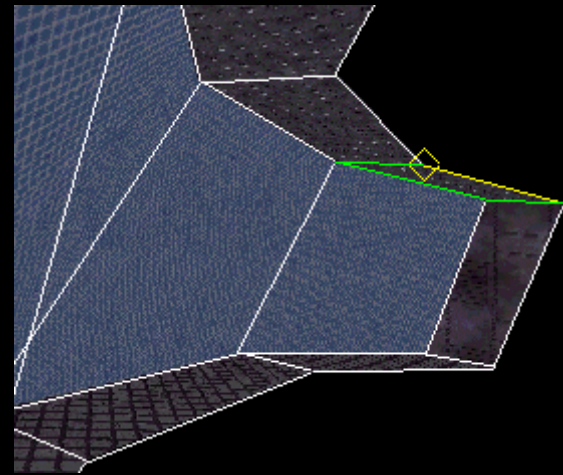


Set the grid to 50, **Ctrl-Shift-V**, then move the faces 50 units to the left to work on it.
 Select the three faces and extrude them along the X axis by -15 units (=15 units to the left). **Remove Extra Verts**.



Delete the three faces that we used to extrude and the two that I circled. Set in the 3D window

onTexture with outline, and use **Shift-click** to texture the new faces so that it looks like the one on the right. Don't forget to set the default UV's and the textures to align.



Now select and copy the faces our new section.

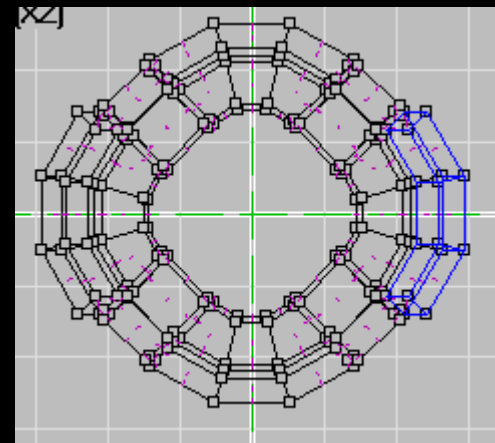
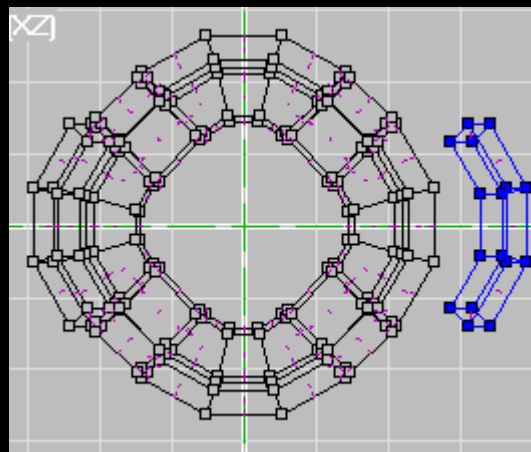
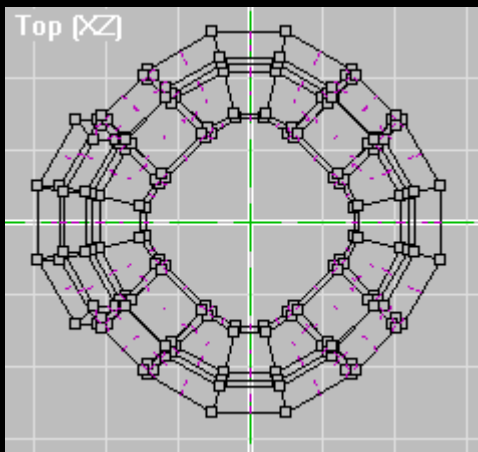
Highlight them again and move them to the sphere (50 units)

Ctrl-Shift-V(Paste on top) Go into vert mode and the verts the section you just inserted to mark.

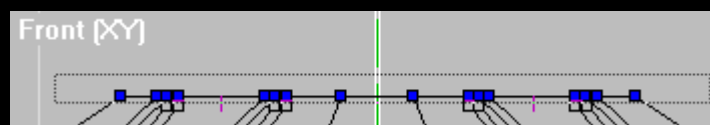
Rotate them to the other side.

Move them to the sphere (50 units to the left)

Remove Extra Verts
Save

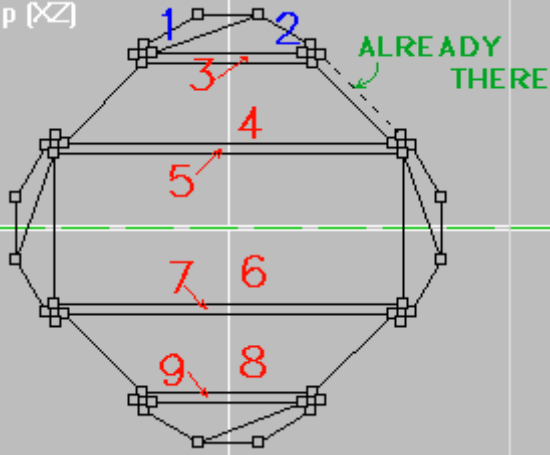


Ok, let's cover the floor and ceiling and we're done.

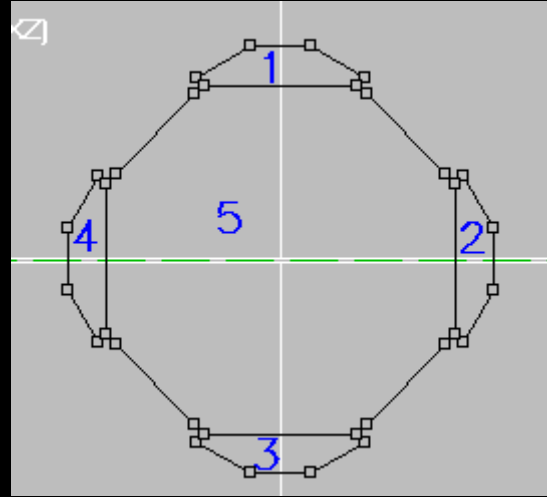


Make the front view active and set the grid to 100. Highlight the verts as shown here. **Ctrl-C, Ctrl-Shift-V**, and then move the new verts 300 units to the right to work on them.

Top [XZ]



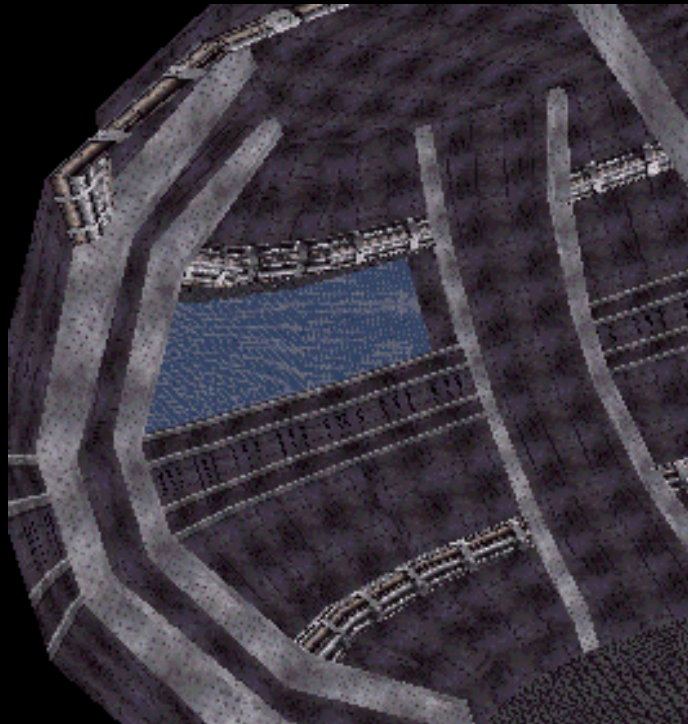
Mark four verts each and place the verts as shown on the left. Watch the front view around. Make sure all normals are facing down. Merge faces 1 and 2, then 3 and 4. Complete the merging so that you end up with five faces (right).



Highlight the five faces and move them 300 units to the left. **Remove Extra Verts.**

Do the floor in the same way. Texture and align the new faces.

Save.....You're done. Compare with right.



it

Sctut08.orf is the finished room. It may have seemed like a long time to get here step by step but now try going through the tutorial again. I bet you'll literally fly over it.

You learned how to use lathe, extrude, copy, rotate, paste on top and some other tools to easily build an interesting structure quickly and entertainingly.

I hope these techniques help you have more time to be creative.

Back to Section C

047 - Primitives

Ragil Ral

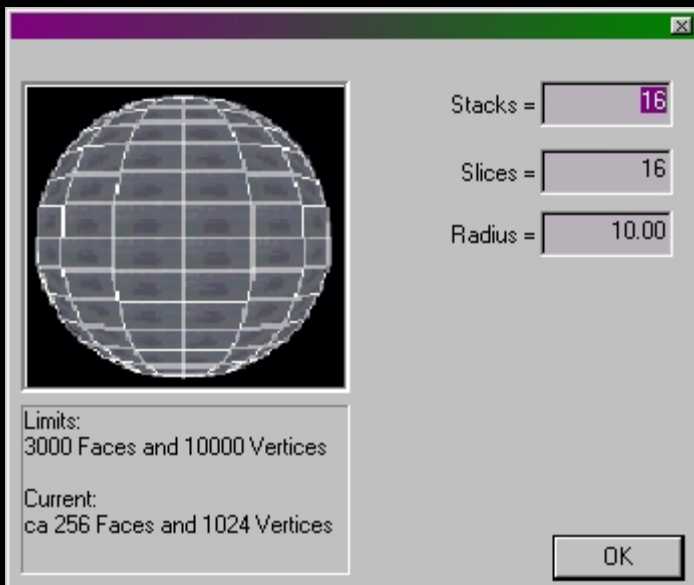
Since v40 there is something that makes the work much easier - namely the ability to create basic shapes on the fly. The dialogue for File->New has been revised and now offers many more options:

The individual characters and their settings will be described here below.

There are a number of settings for each character, most of which are actually self-explanatory. If the figure is created as a new room, the construction is located exactly in the middle of the room.



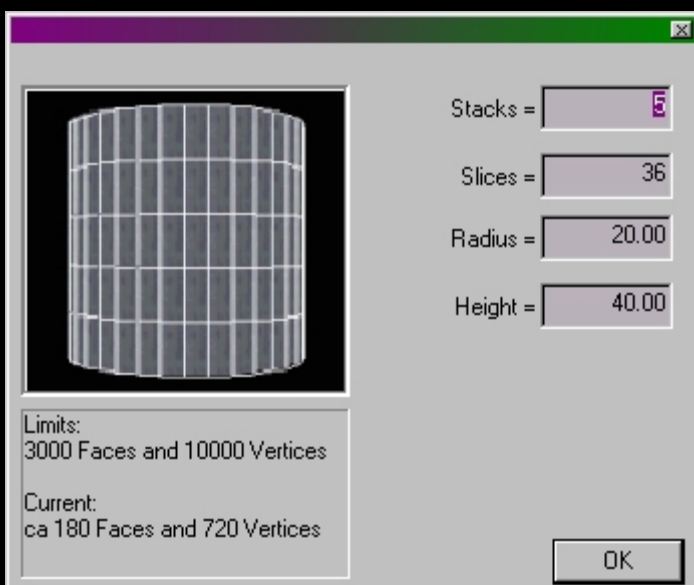
Sphere



Creates a sphere.

Stacks: How many layers the ball should have
Slices: How many segments (corners) should the ball have
radius:...

Cylinder

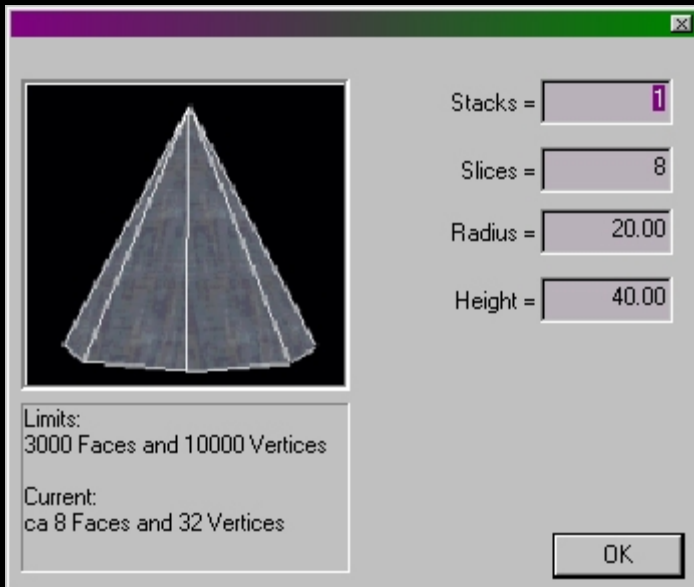


Creates a...(drum roll)...cylinder!

Stacks: How many disks should the cylinder have
Slices: How 'round' should the cylinder be? . . so like Sphere.

radius: Radius, and
Height: Height of the cylinder.

Cone



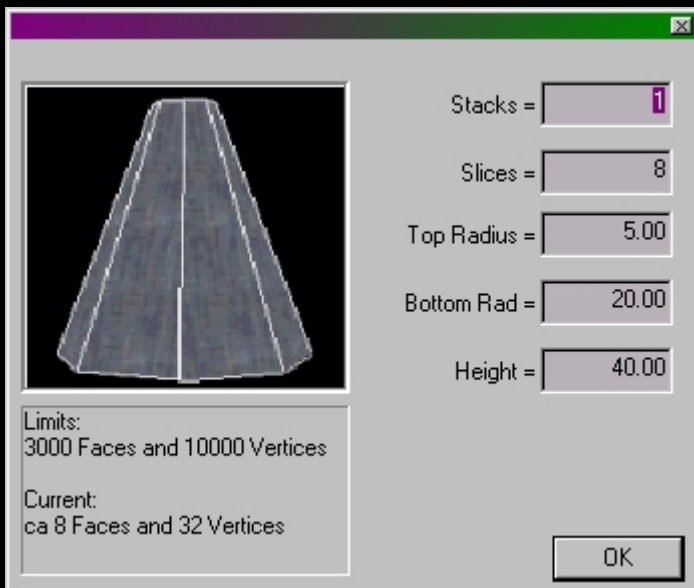
Creates a cone.

Stacks: How many slices

Slices: how many pages

radius and Height: the desired dimensions

Flat Cone



Creates a cone frustum.

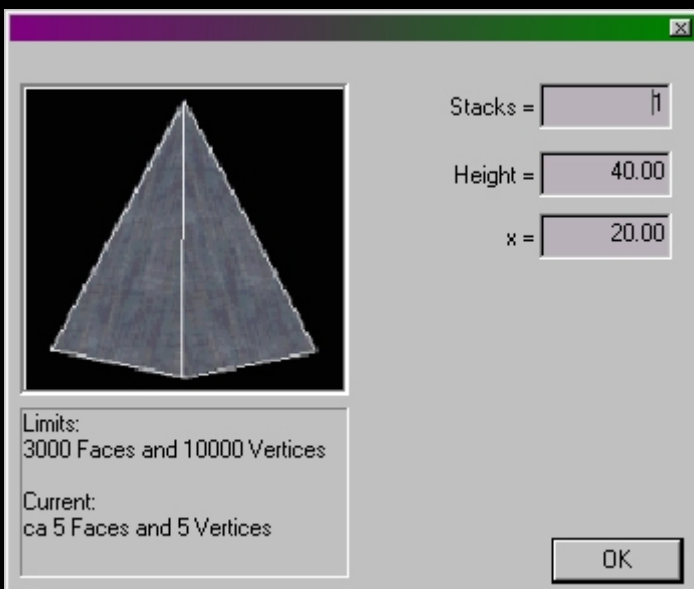
Stacks and Slices: like Cone,

Great radius: Radius of the upper deck face,

Bottom wheel: Radius of the ground face

Height:...

Pyramid



Creates a four-sided pyramid.

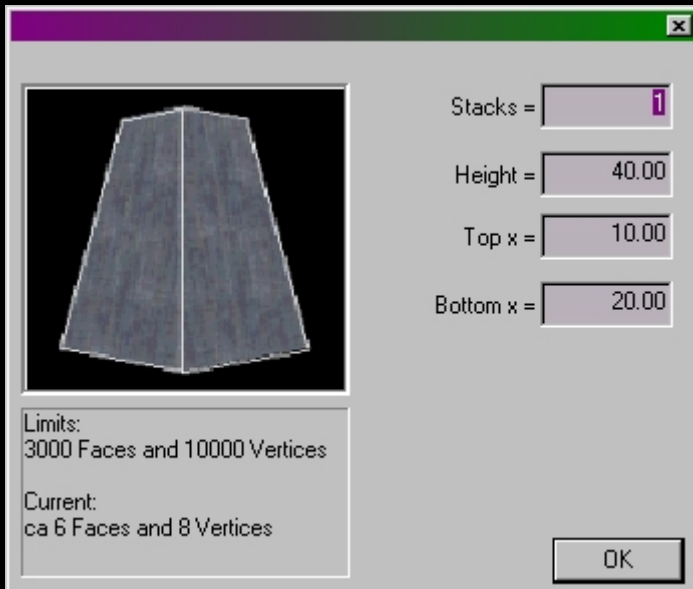
Stacks: as usual, the number of 'slices'

Height: how high should it be?

x: Side length of the base surface.

This primitive has the advantage that, in contrast to Cone (with Slices=4) the side length, not half the diagonal (=radius) is specified.

Flat Pyramid



Creates a truncated pyramid, similar to Flat Cone.

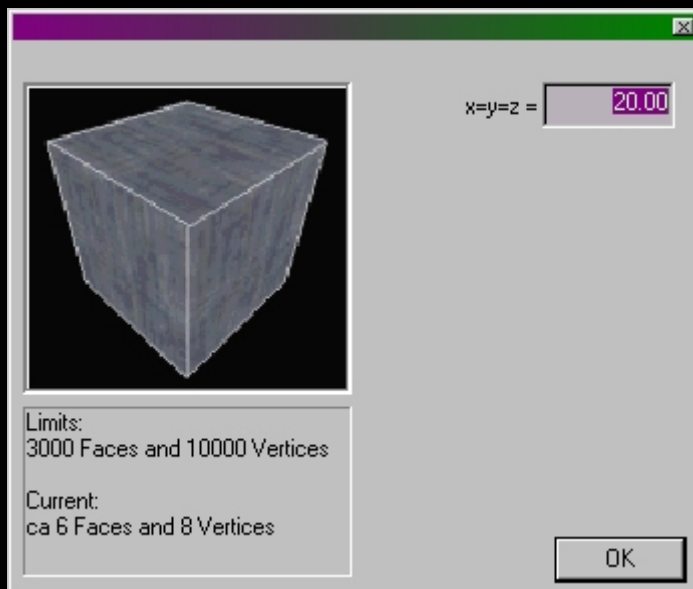
Stacks:...

Height:...

Top x: Side length of the deck face

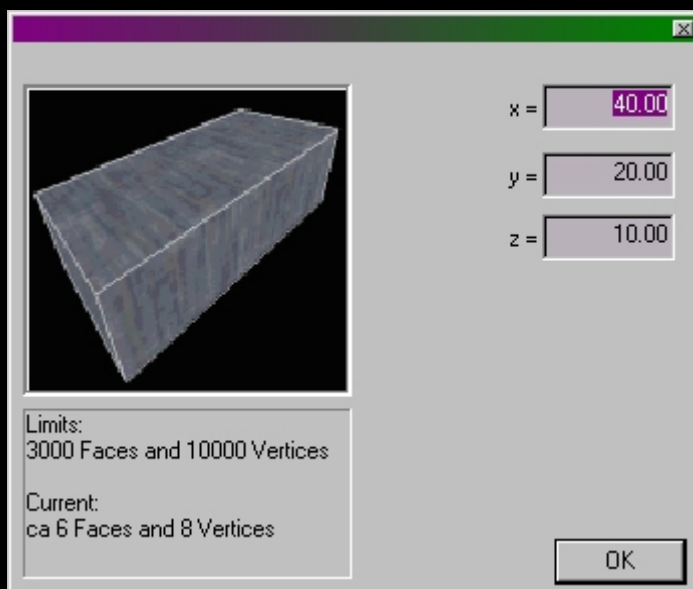
Bottom x: Side length of the bottom face.

Cube



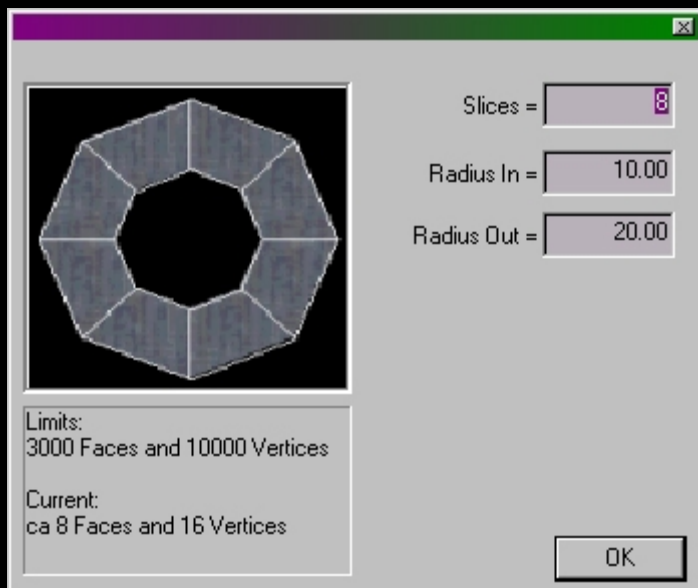
If you're in a hurry - This creates a cube with the specified edge length.

box



Creates a box with the specified dimensions.

ring



This creates a ring of faces.

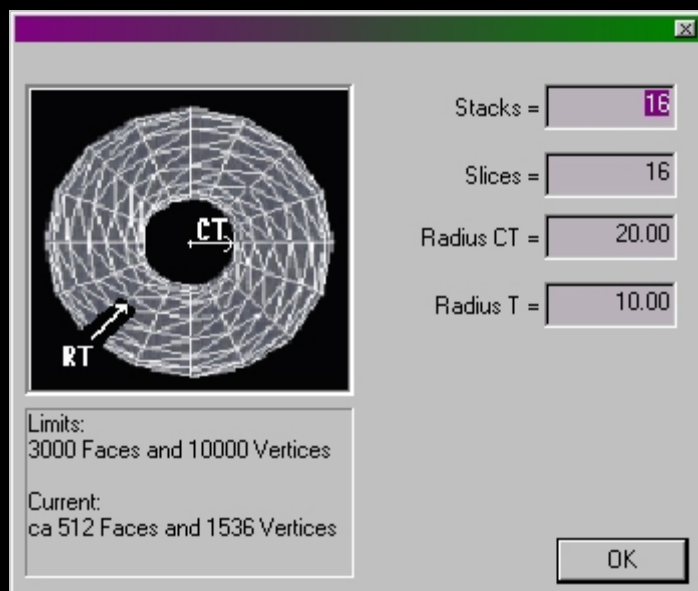
Slices: how many areas there should be

Radius In: Inner radius

Radius Out: Outer radius

Very practical for floor surfaces that need to be extruded away after the basic shape has been worked on.

torus



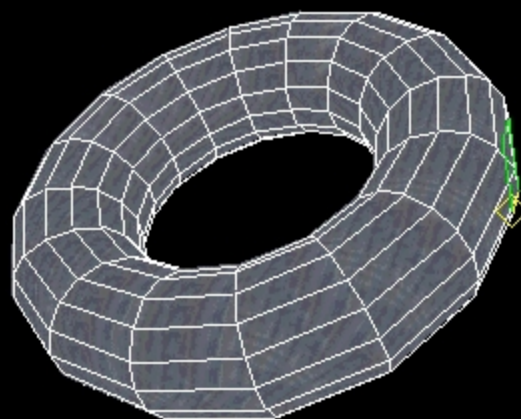
This one is a little more special: a torus is created, the famous donut shape.

Stacks: How many segments

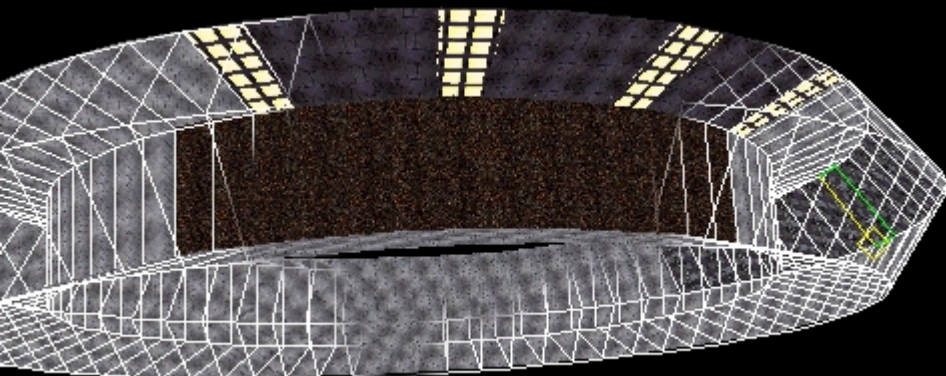
Slices: like 'round'

Radius CT: inner radius of the shape,

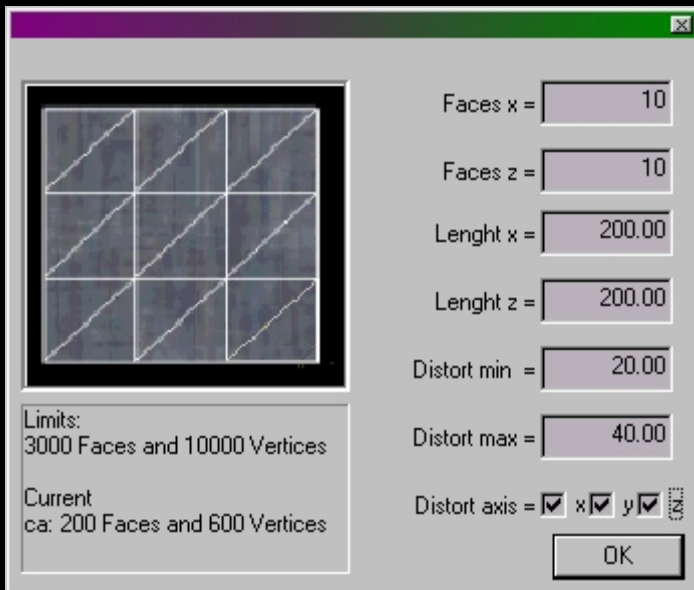
Radius T: Radius of the 'tube' itself.



A torus with stacks=64 and slices=5: a perfectly symmetrical pentagonal tour in five seconds



Grid



This creates a triangulated surface.

Faces x: How many edges should be divided into in direction x,

Faces e.g:toward z

Length x and Length e.g: the dimensions in the respective direction

And now here it comes: There is the possibility to have the verts 'randomly distributed' in certain directions.

Distort min: Minimal,

Distort max: Maximum displacement value.

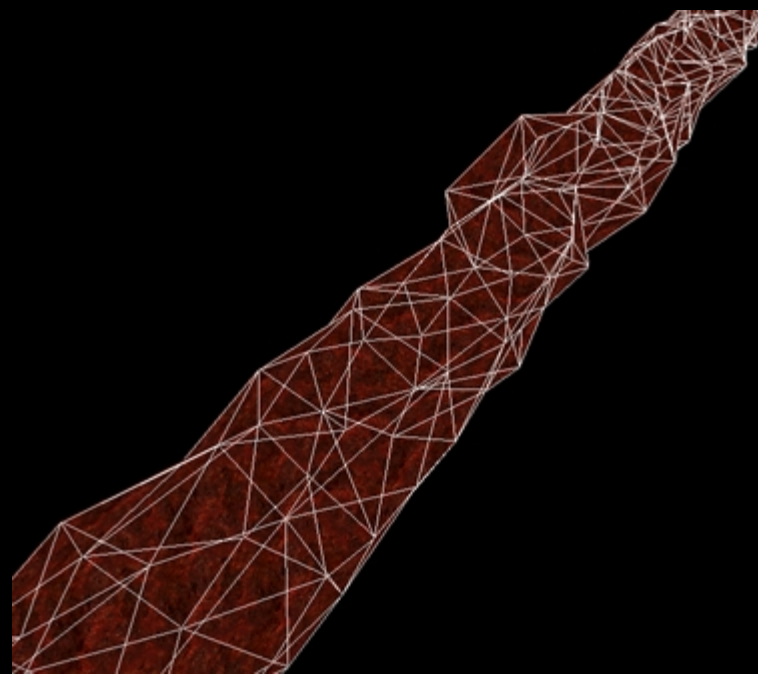
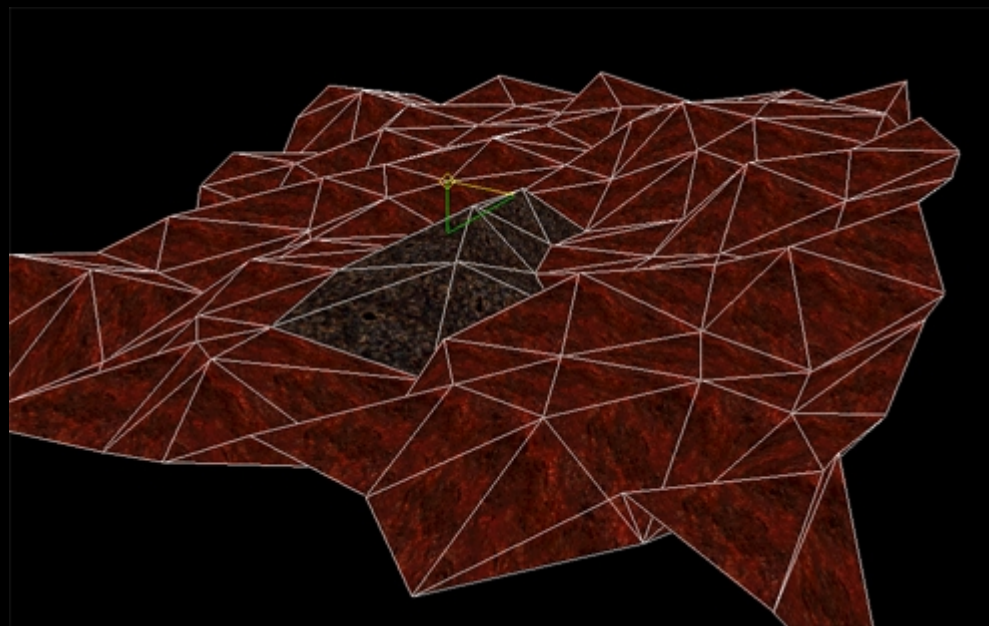
Distort axis: Here you specify in which directions

the verts are to be distributed; this is freely selectable.

The values at the top of the screenshot are not the default values like the others. The values produce something like the following figure:

This makes it child's play to create irregular shapes. You just start with something like that on the right, bend it and shape it - forge it, so to speak, and then sew it together.

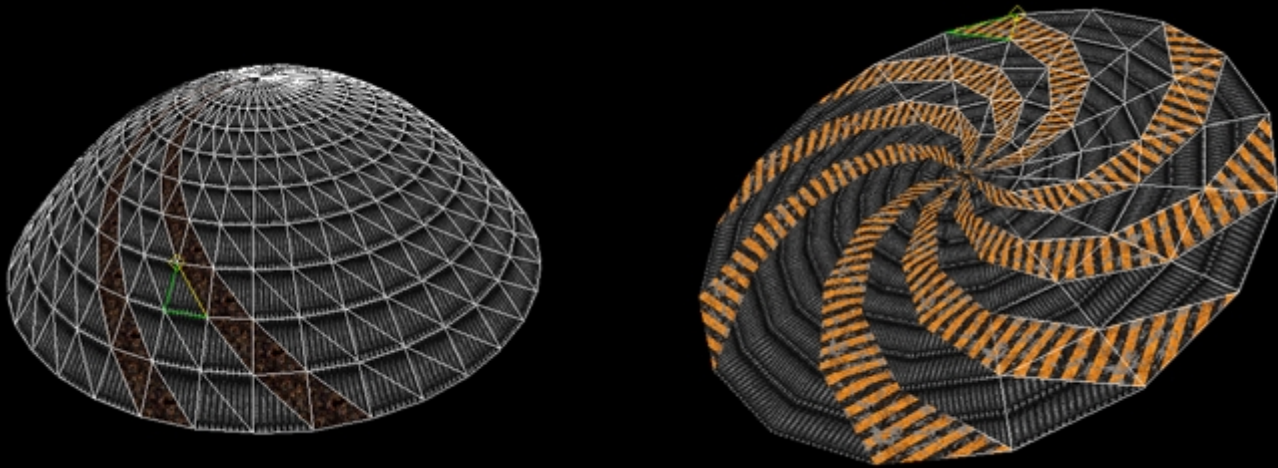
Here is a view of the inside and outside of a corridor that I created in this way:



It always works the same way, you create a 'basic grid', you then 'forge' it into shape (by twisting, moving or bending) and close it. The shape becomes with increasing values for Distort min and Max of course more and more resolved, here you have to start paying attention to the position of individual faces. It is also advisable to combine faces in regions that are not particularly noticeable by making them disappear with Snap-To-Vert; otherwise you will reach high face counts very quickly.

Such functions are usually only found in large, professional 3D programs!

But you can also do something like this:



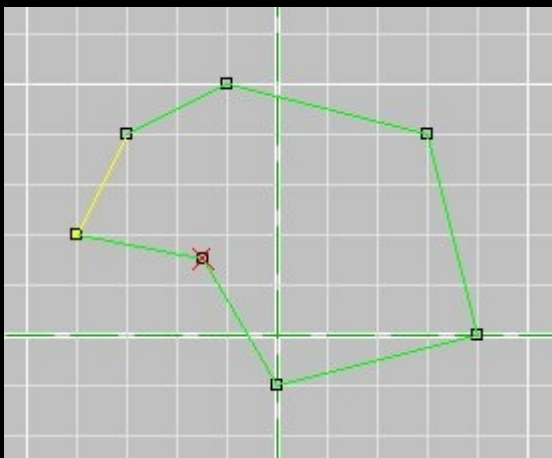
These figures were created from grids that had no distortion. Then I handed it over and pulled up the dome on the left one.

[Back to Section C](#)

048 - Create rock formations

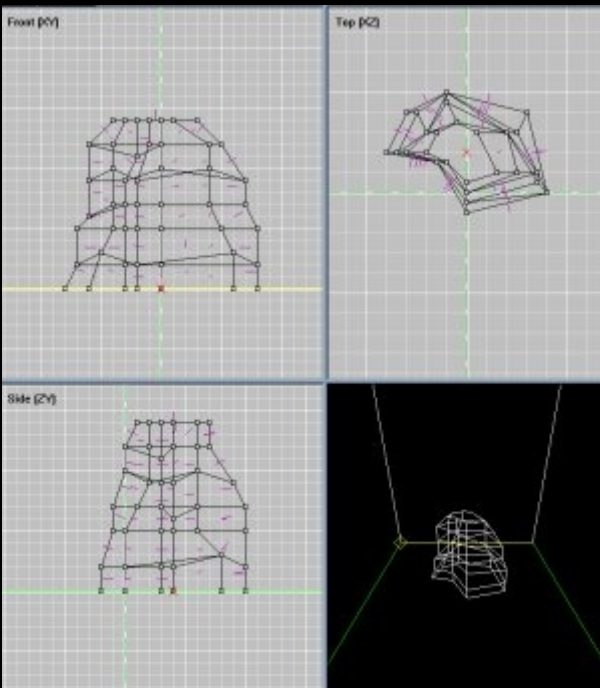
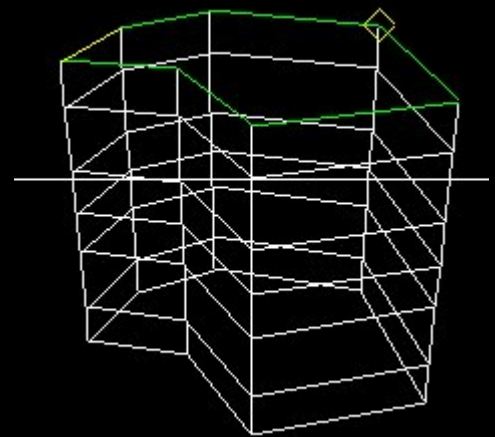
Hydra

Rock formations can look really cool when done properly, and there's a quick and easy way to do it. But just a warning, depending on how big the rock is and how much detail you've put into it, they can have a lot of polygons.



First of all, you should have a rough idea of what you want the rock to look like when it's finished. Like how it's curved, whether it's tilted or whatever. Then make the basic outline of the rock (like in the picture on the left) on the floor (or ceiling). This face does not have to be convex as it will soon be deleted.

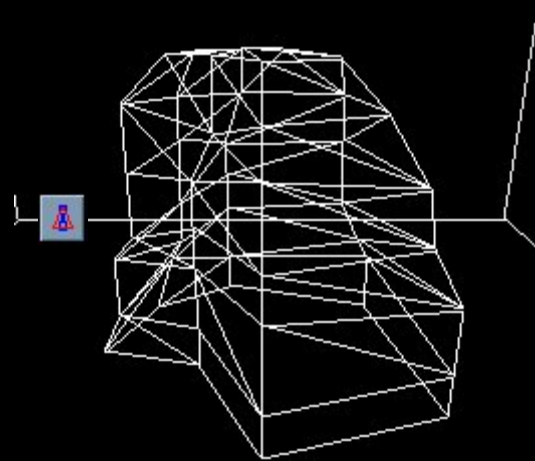
If you have a face that you like, extrude it. I usually do extrusions between 10 and 15. Staggering the extrusions makes the rock look more natural. Stop extruding when you reach a height that suits you. If you forget while extruding Delete base Facetick, go back and delete all of these faces. Don't forget to keep the top one though.



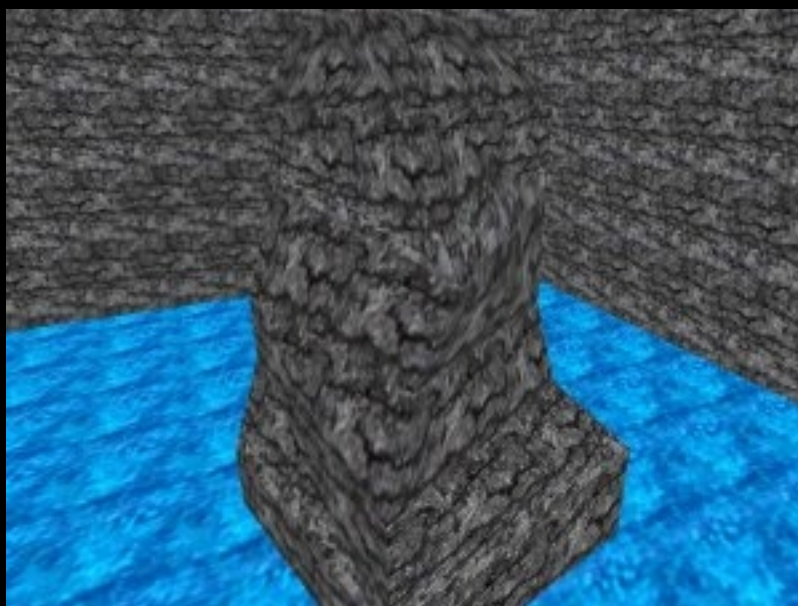
Now for the fun part. Set the grids to 5 units in all three orthogonal views and then start moving your verts around randomly. Don't move them too much in one direction (move one or two vertically too) maybe just one or two jumps most of the time. You should generally move the verts inward, toward the center of the rock. Afterwards you should see something like this on the left in D3Edit.

Now I'm sure you've all realized that this property is chock full of errors, we'll fix it right away. Go into face mode and mark your entire object. Then click the button **Triangulate non-planar faces**.

Wow look at all these new faces, select your object again and click on the button again. Repeat until D3Edit says all faces are planar. It should take four or five times. Since your faces are already marked, you could texture them straight away.



Then look at the rock in D3. Sometimes the verts got moved into bad positions and flattened a few faces, sending you back to the .orf-File and either tweak the rock or start over. Either way, this technique is a quick and easy way to make cool looking rock formations.



[Back to Section C](#)

049 - Tunnel in Cave style

WillyP

Edited by Fischlein

This is no longer for complete beginners. What has been treated so far should now fit well!

Make a room

First you will need to create a room. The first polygon (face) you make should be the size of the entrance portal. *The largest ship, the Phoenix, has a radius of 8.018578 units, so the polygon needs to be more than twice that size (20 is a good minimum).* The more verts you use, the more polygons you will get, and the more detailed the walls will be. As you can see from the picture, I created an octagon but added an extra vert to each side. For the same reason, use fairly short extrusions (30 are visible in the image).

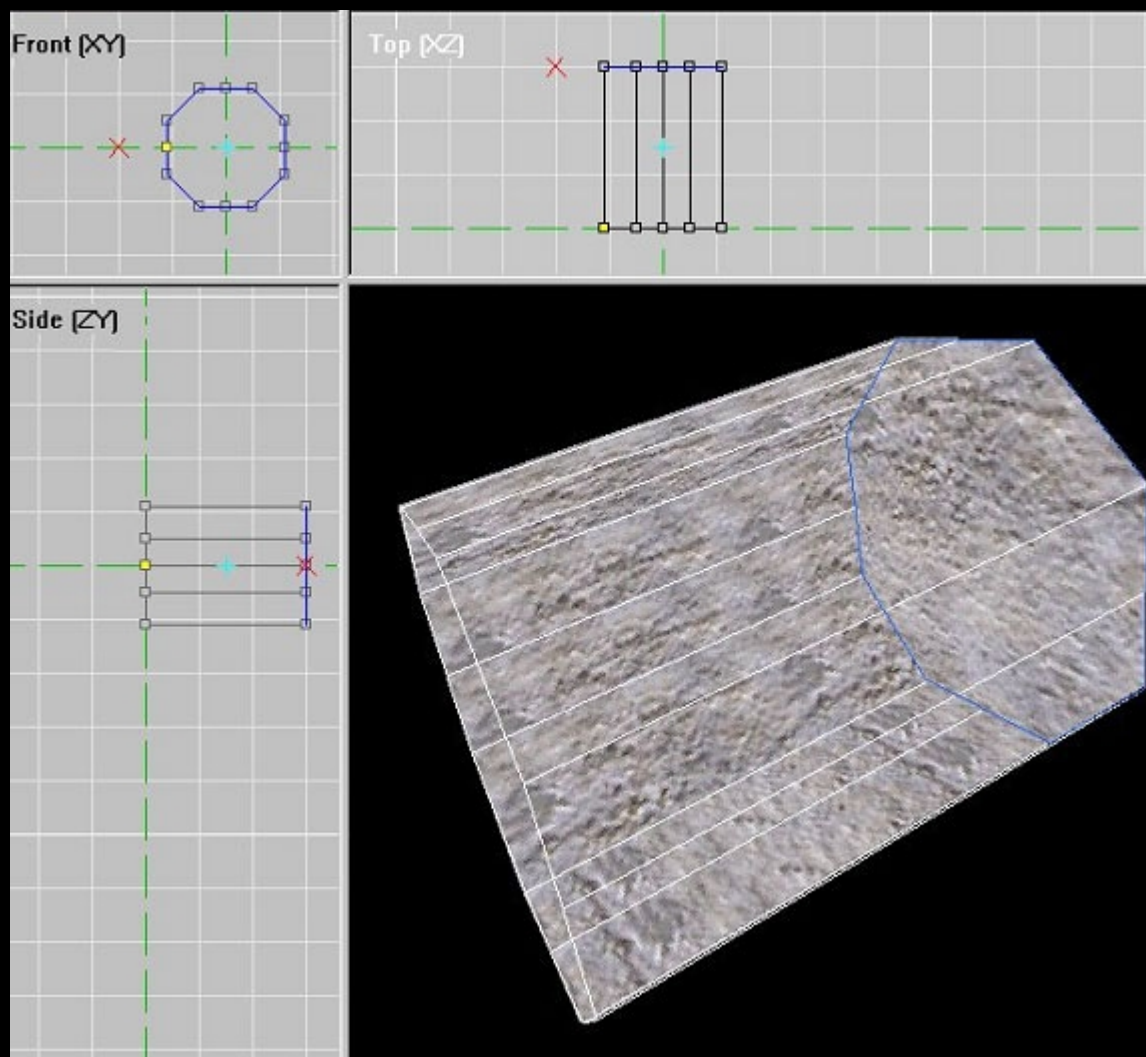
Another option... my favorite... is, in an existing one Space a face select that one will become a portal. Then, in World View, press **Into the** and presto, you have a new space! You just have to be careful that you do not use the portal in any way distorted, changed.

A tip:

Most Instructions

(at least the ones that do I have read) instruct you, one Polygon to mark that Normal to turn around and then extrude. I'll skip those

Select and rotate routine by using a negative number in the extrusion dialog. If you are extruding from the first polygon of a new space, use a positive number and choose **Delete Base Face** away. Second extrusion, enter a negative number, select "Point outward" for the polygon normals (for an interior; the negative number flips them) and select the **Delete Base Face** option. For later extrusions, select a polygon and press **Ctrl+E**, then extrude by the same value.



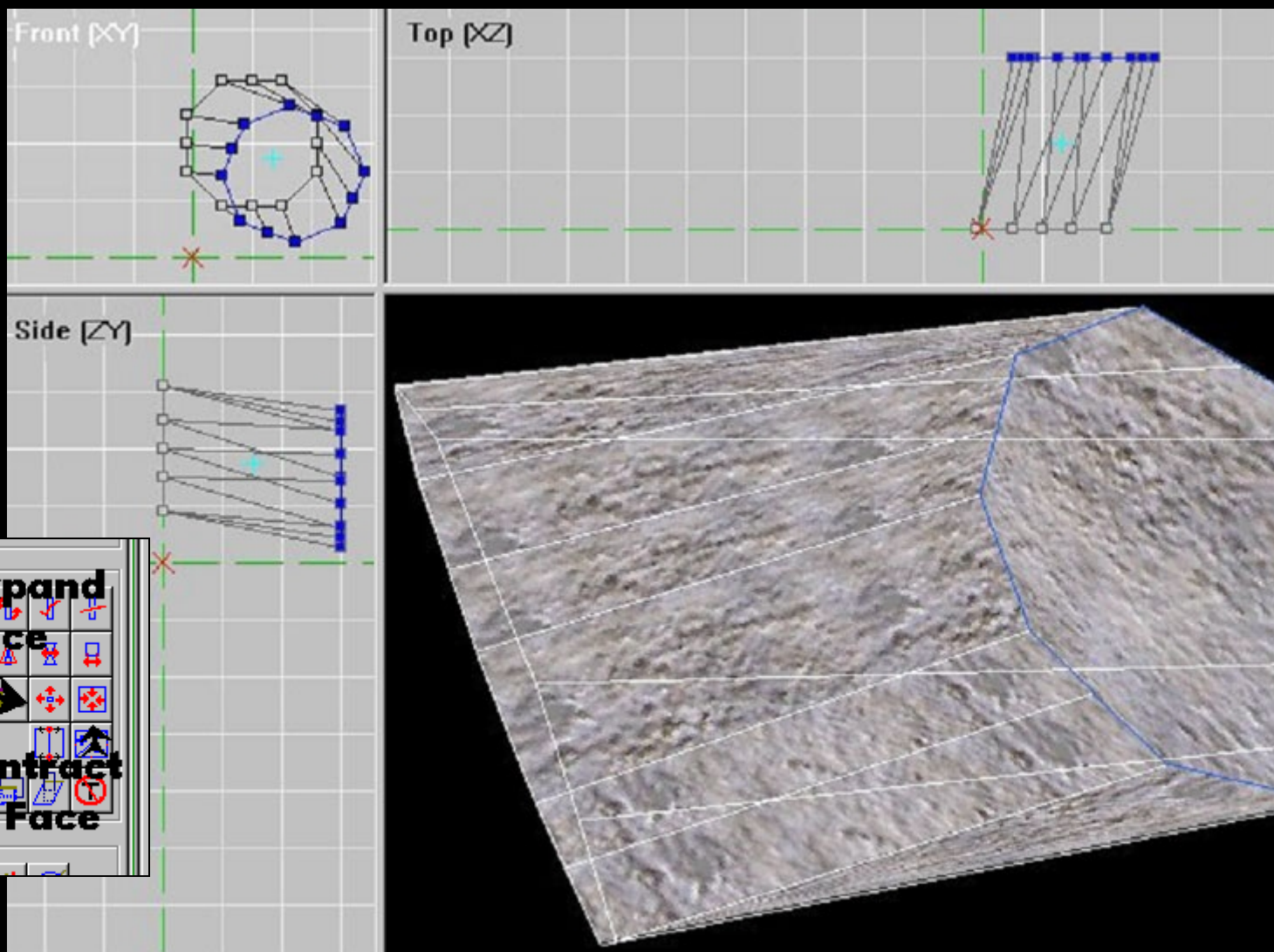
Move a polygon

Note the marked polygon (blue border) in the previous image. We want to move the whole polygon without distorting it. Press **W** button and scroll down to "Modify items... Moving marked items..." and try out the various options listed to move the highlighted polygon (you saved the room beforehand... right?) Note that up and down are relative to the window you are in and in the 3D view will not work.

A few things that don't come straight out of the box: The points (Verts) or the polygons (Faces) can be marked and moved, depending on which mode you are in. Points and polygons are not objects, i.e. "rotate object" does not work with them. "Twist" will rotate what is selected around the intersection of the green reference lines. Move these with you **Ctrl-click**. Use all four views to see the impact of what you do.

To keep moving!

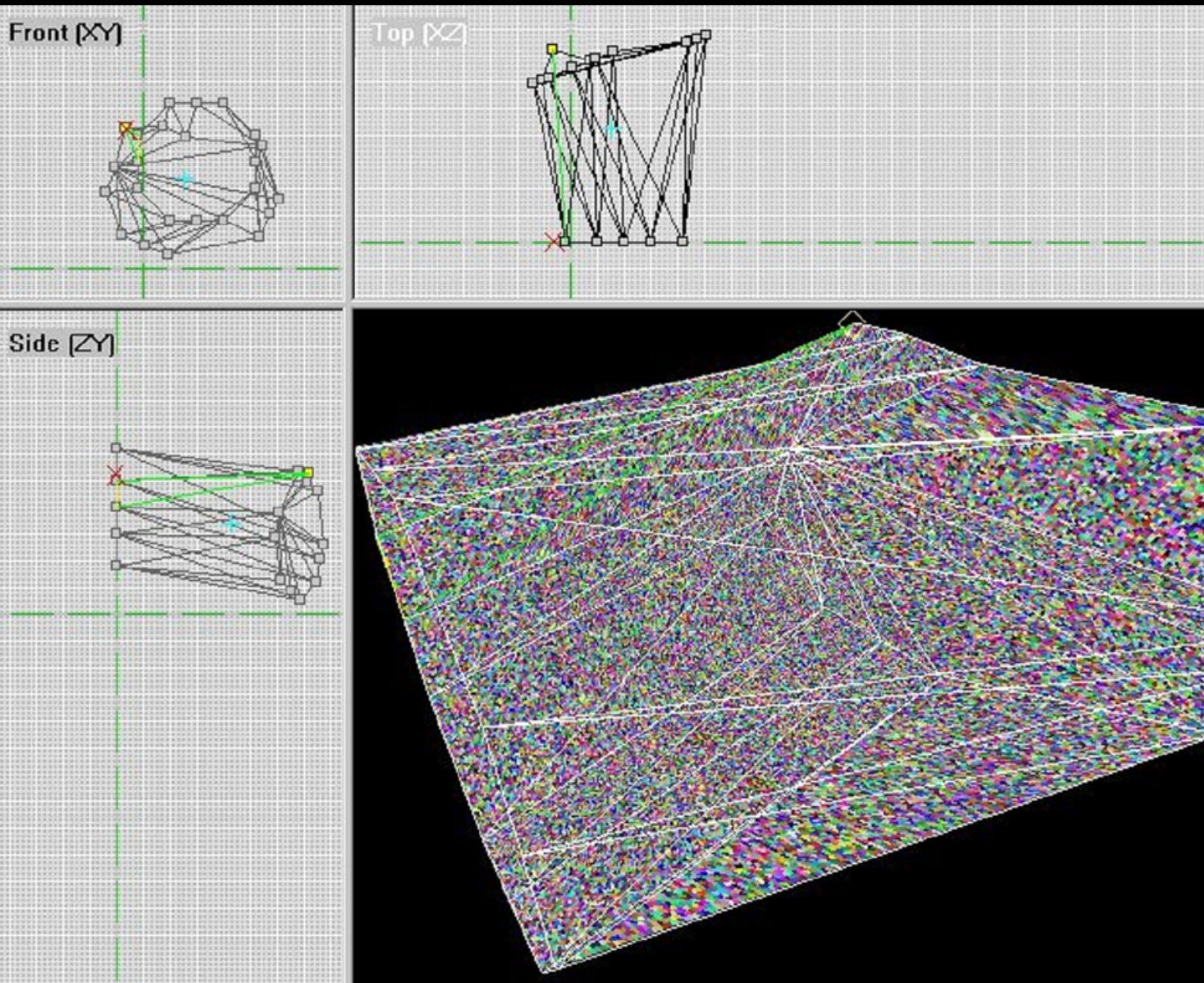
In this image I have slightly rotated the marked points in the front view. Now my two octagons are shifted and slightly twisted. Now don't worry about non-planar polygons, except for the one you're going to continue extruding from. Use that **Expand** and **Contract** Buttons to change the size of the polygon. Look at the marked points in the top view and the side view on the right, they are all on the same grid. If not, move it to a line to make the polygon flat again so you can extrude from it again. The polygon shouldn't be concave either. If you can draw a line between two points that lies outside the edge of the polygon, you have a concave polygon. Fix this by moving points in the front view.





Face Planar Check

Here (below) I moved the polygon a little more. But I also drew a point from the row with the polygon. I then have the modify button **Face planar check** used after all polygons have been marked. This fixed all non-planar polygons by cutting them into triangles. Since the polygon we want to extrude from is not flat, it has also been broken down into triangles. The largest of these is too small to fly through, so there's no point in extruding from it. I'm going to go back to my previous backup which I made before I pulled the point out of line. Also note that the texture switches to the "Default Texture" when polygons are decomposed in this way.



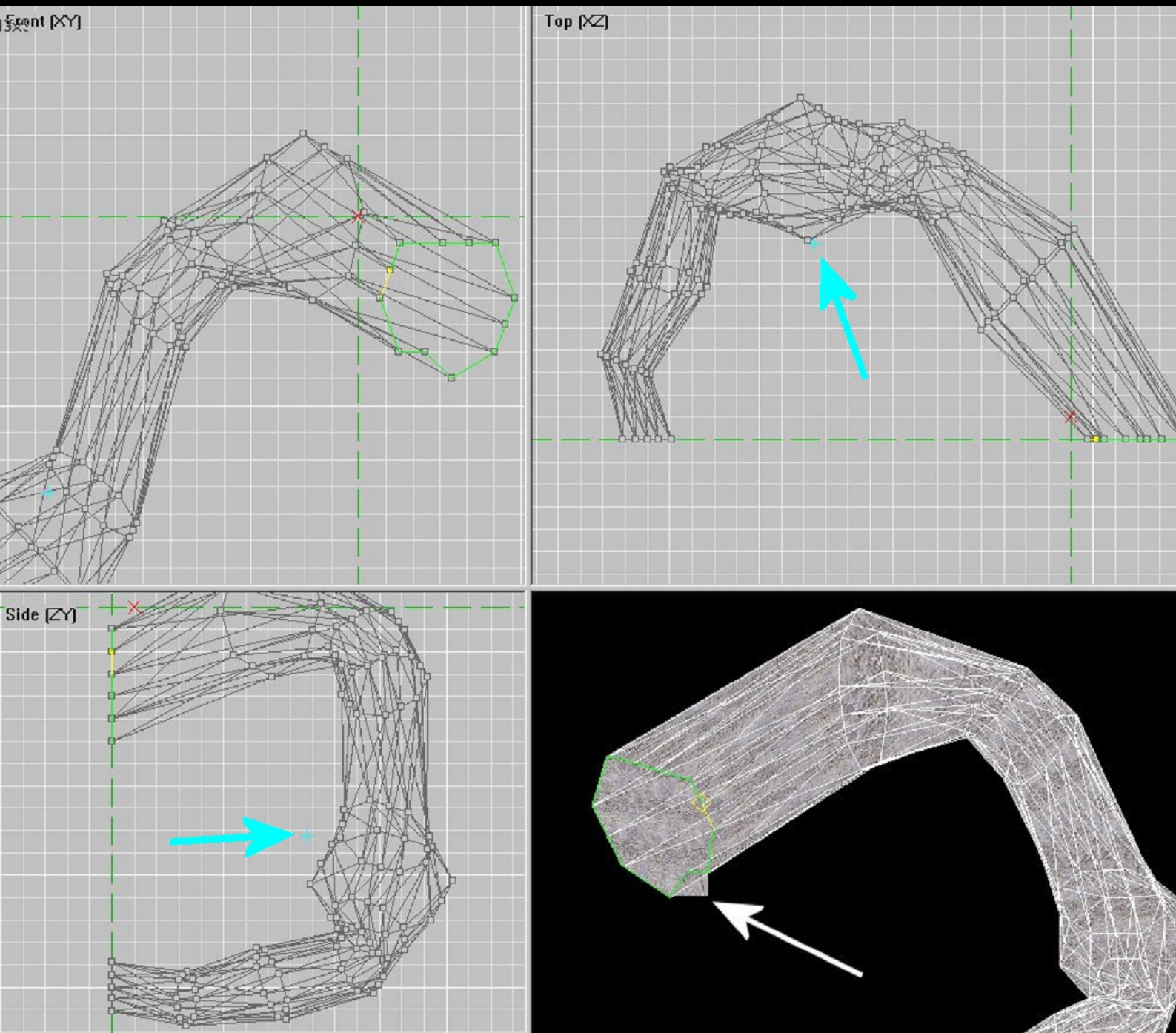
Are we already there?

If you look at the image below, you can see that I changed the angle and size of the polygon, extruded it by -30 units and repeated this a few times. Then I marked the points of the last extrusion and placed them on the same grid line in the side view, with **Snap to grid**. Now I have a few errors, the last polygon is concave due to the orientation of the points (white arrow in the 3D view). This is easy to fix, I'll just move a few points in the front view until it's right. The **turquoise** Arrows in the top and side views point to a crosshair of the same color. This is the space center, D3 uses it for calculating sound propagation, missile lock-up and AI movements, among other things.



Snap marked to grid.

It is essential that the (geometric) center lies within the space. Its position is calculated as the center of all corner points of the room.



We could fix this by taking a handful of points and moving them to the left in the side view. You would also need to move some down in the top view. And you would have to move it further than you think, since the center of the space moves even a little. Another method would be to insert a sub-object into the space, at a location where the verts would pull the center back into the space. For now we will accept that the room has an error.

We still have a little more work to do on this room. If you were to fly through this space, you would clearly see that it is a series of octagons. And although these are in random positions, they are still recognizable as octagons, disrupting the effect of a natural cave. The next step will remedy this.

Grab a handful of verts

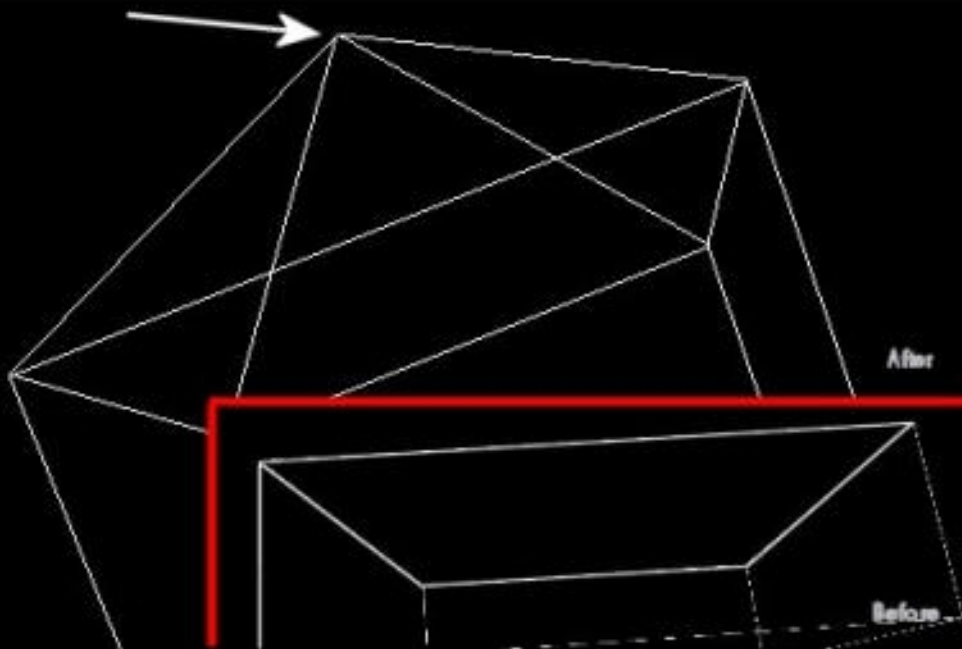


Just drag one or more points, select them, move them around, this way and that, use all three 2D views. Remove the mark again, repeat... you know what I mean.

Verify the room!

We now have a pretty nice looking cave, but also a few problems... *Non-planar faces*

For one thing, most faces are non-planar. The easiest way would be to mark all faces (Errors, Non-planar faces->mark), and then with that **Face planar check** to split. Another way would be to extrude them using the Needle setting. Needle gives you a lot more faces to play around with. It splits the polygon by putting a vert in the middle of it. The polygon is broken down into triangles, one for each edge of the polygon. The inserted one is then inserted



Point (arrow) around the amount along the set direction (X,Y,Z or Normal) entered for the extrusion. If the tunnel offers enough space, you can move the new needle vert up or down

pull down to stalactites

to recreate. (Mnemonics to each other note which ones point where -> <http://de.wikipedia.org/wiki/Tropfstein>)

If I **Lathe** or **extrude** To construct a large space, I will first go through all the polygons and **extrude**, Needle Press, change the distance here and there. Then I go back and

"Extrude Needle" some of the

resulting polygons here and there, then I push and slide points as necessary to get a random look, with rock ledges dragged into space, rock crevices pushed out, broken rocks sticking out of the ground. And down from the ceiling.

Room Center not in the room

Well, it is quite clear that - in the picture with the turquoise arrows - the center of space is not in space.

If you are sure that the room center is inside the room, but the verification says it is not, then update to the latest D3-Edit version first. Older versions incorrectly stated that the center was not in the room when working on a single room. Are you still getting this message? With this style of a room it is very hard to tell by looking at the 2D views whether it is really like that or not.

One way to check this is to import the room into a new level. In the first room the **player** -Start in the center of the room. You can go to the current room view and clearly see in the 3D view whether the starting point is in the room or not.

Portals not in line of sight

You should be able to draw a line between the portals without leaving the room. There is a BNode on each portal. These lead the AI from portal to portal, the AI then tries to get from one portal to the other. Because of the U-shape of the room, she can't do that here. The bots have some leeway in this regard, as they move around a bit when they get stuck.

We've all seen Guidebot get stuck in a wall, trying in vain to get through the wall while trying to guide you to your destination. Homing weapons have a slightly harder life, as they commit 'suicide' on the same wall while your opponent is just around the corner in the next room. (see the [Tut about BNodes](#).)

texturing

Texturing the cave tunnel is quite easy, mark all the faces **M** and then you can apply a rock texture to all faces with one click. Or, if you have added sub-objects, you can press 0 on the number pad in face mode and in the 3D view. This only highlights the faces that are connected to the current. In the alignment dialog select Marked Faces and press Default UV's.

Since this room is a jumble of rocks, it's more realistic to not align the textures with each other.

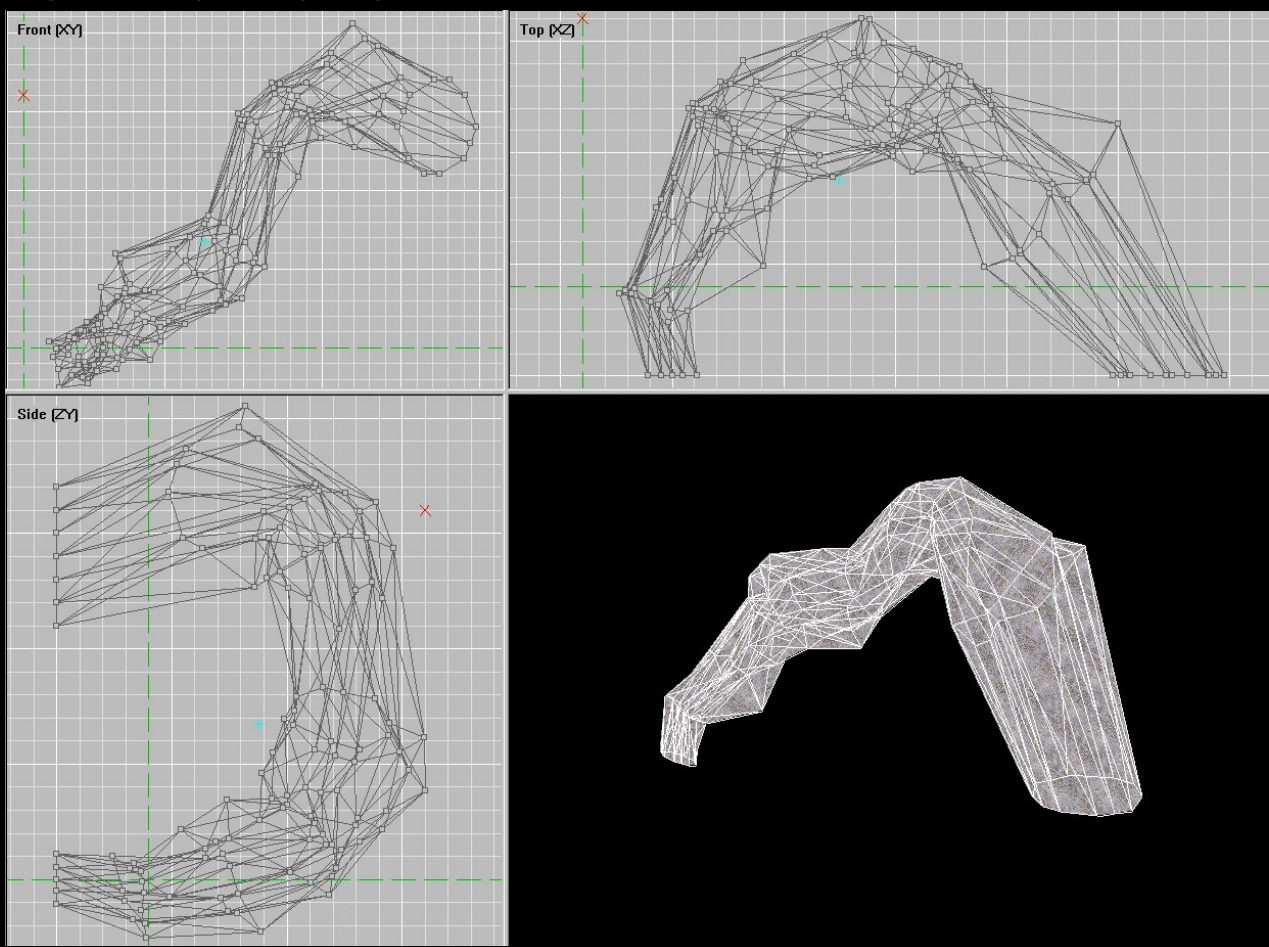
lighting

The easiest thing would be to just light the room with ambient light (ambient light), but that wouldn't be the most realistic. Some other options for lighting would be crystals growing from the rocks, sinks with lava, lamps, and of course combinations of everything.

Making lava puddles is quite easy. Just drag down a few verts, make the cave floor a little lower and change the textures to lava. This is very bright, so it doesn't need much, and it's something you would expect in a cave. At least in a Descent cave, anyway



This is the completed tunnel. I will add a lamp to get the room center into the room. The room needs light anyway. I could use ambient light, but I like the effect of transitioning from dark to light areas, point lighting allows that.




Screenshot from within the game:



Going from well-lit areas to dark ones can be very scary!

[Back to Section C](#)

Section D-Debug

By now I think everyone will have had their own experience with construction errors or their (perhaps even successful?) mination.

Thanks to the new editor versions, finding and repairing errors has thankfully become much easier.

Although this is not about building something; However, start up the editor and recreate the errors - it gives you a feeling of how the editor behaves, how the errors behave, what the whole thing looks like, etc. etc... The editor could then crash, but I I think that doesn't matter



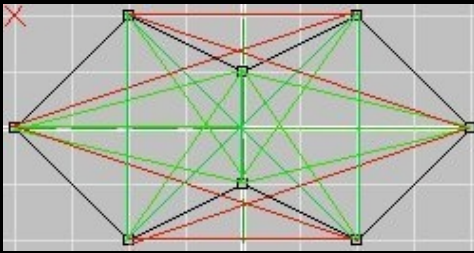
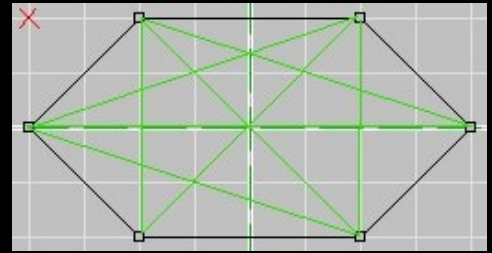
Fix basic errors				
050	Concave faces	Repairing concave faces	Hydra	198
051	Non-Planar Faces	Getting rid of uneven polygons	Fischlein	199
Repairing T-Joints on Foot				
052	T joint	Fixing T-Joints: Instructions	Fischlein	201
053	T-Joints: Second version	T-joints fix in screenshots	The Edge	202
Error checking and handling routines.				
054	Verify your level	Explained briefly and succinctly	Kyouryuu	206
055	Verify your mine	with examples	WillyP	211

[Toc](#)

050 - Concave Faces

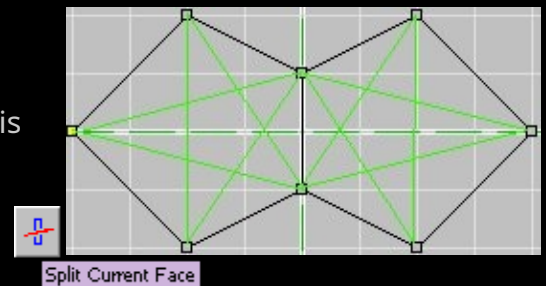
Hydra

Every face in Descent 3 must be convex; If there is one that is not convex (= concave), the game treats it strangely. In the right image is an example of a convex face. I have drawn lines between all vertices, note that none of the lines go outside the face.



As I said before, if you put a line between all the verts, none should be outside. If you have one outside (red lines), you have a concave face. like this one on the left. Fortunately, the problem is fairly easy to fix.

This is one way to solve the problem. In D3Edit you are allowed to split faces. I selected the middle two verts and clicked the button **Split current face**. On the right is the result of this, now I have two convex faces, which D3 will handle correctly.





[Back to Section D](#)

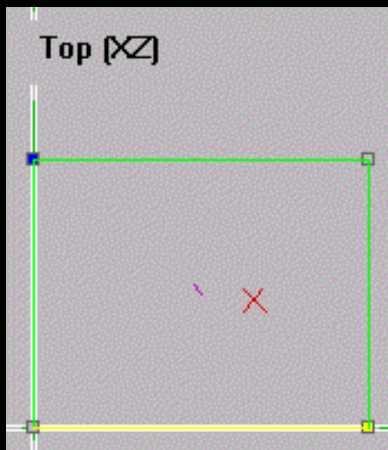
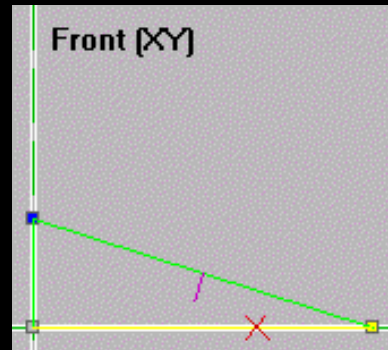
051 - Non-Planar Faces

Fischlein

(revised)

For this topic you still need the Windows Editor if you are working with an older D3Edit version; in the newer ones there is the button , which  so issues reports about processes during construction.

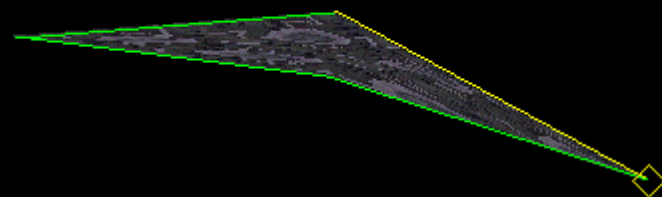
I have prepared a few pictures here that show a non-planar face.



Imagine a surface lying on the same level. You move one of these four points and the others not, then you have a non-planar face.



There are now two options for this

with four verts. These verts should



Find and correct faces.

Solution 1: Go to the room where you have Non-Planar Faces. Now switch to face mode and mark all faces. Once you've done that, go to the

Geometry Bar and clicks this button:  If there  Non-Planar Face in the room

then click in the following dialog box **Yes**. If you have several non-planar faces in the room, you have to repeat this until no more non-planar faces are displayed in Verify Room.

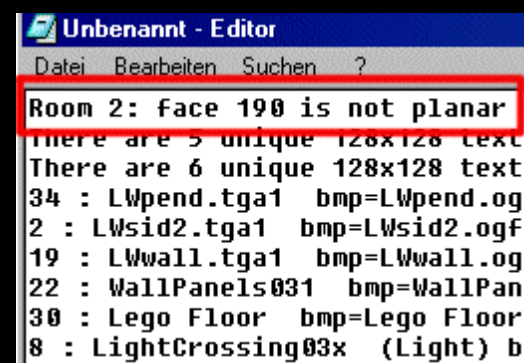
(Important !!! All faces must be marked.)

Solution 2: This route is a little more extensive. First switch to World View and last then run Verify Mine. (picture on the left)

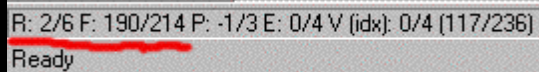


Now click on the button **Close**. Now open the Windows Editor and select Edit, Paste from the menu. What you see now should look something like the picture on the right.

What does the part outlined in red tell us? The 190th face of room 2 is non-planar. We now go to the World View view in D3Edit.



Pay attention to the status line (picture below). Now press the button **R** to switch to Room 2.



(R:2/6 tells us room 2 of 6

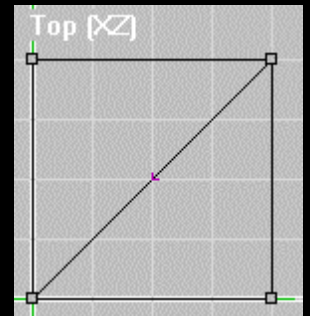
is the current room) Now press the button **F** on your keyboard, press it until you reach the one you want

You have reached your face number. In my example 190. (F:190/214 tells us face 190 of 214 is current) Now switch to the room view and mark the face (**Space**). Now you click on this button:



Pay attention, they are different buttons!!! You can also triangulate planar faces with the other one, you can try it out. Ok, the face has now split into two triangles (two faces). (Picture right)

Now run Verify Mine again and you will see that there are no more non-planar faces (if you only had one non-planar face). This is how you have to proceed with all non-planar faces.



Have fun building levels!!!

Back to Section D

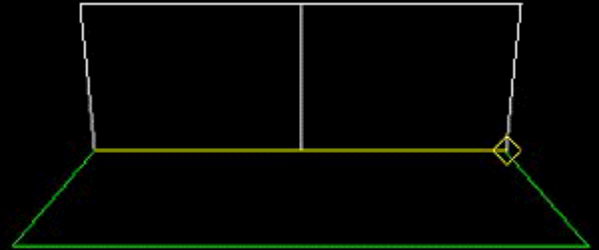
052 - T joint

Fischlein

In this topic I would like to explain how to find and correct T-joints.

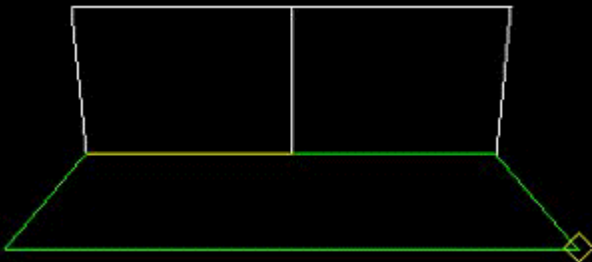
What are T-joints?

These are verts that are not connected to the neighboring face or edge. In the picture on the right you can see such a T-joint.



The yellow line is the edge, but it should look like the picture on the left.

Ok, now to find the T-Joints, go to the menu File/Verify Mine(World View view). Then click on **Close**.



Now open the Windows Editor and go to the Edit menu and then Paste. Now you can see all the details about your levels. Look for a line that looks something like this: "Room 33: Face 17

edge 0 has a T-joint", that tells us that Room 33, Face 17, Edge 0 has a T-joint. Now switch back to D3Edit and

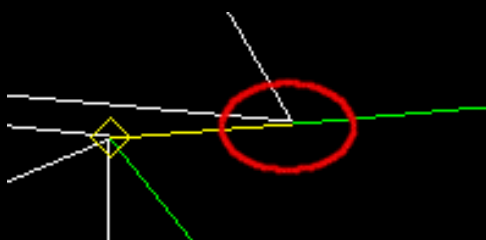
presses the **R** button, press it until the corresponding room number is displayed in the status line (see above). For face button **F**, for Edge button **E**. The following information should also be noted: "Room 33 Face 17: No connection for edge 0", this belongs to the one above and tells us that in room 33, Face 17 has no connection to Edge 0.

R: 33/32 F: 17/144 P: -1/2 E: 0/4 V (idx): 0/4 (18/188)
Ready

Now switch to the Room View view and zoom in on the face in the preview. Look closely at the edges of the face, especially where a neighboring face is adjacent. In my example it looks like the picture on the right.



Now click on the neighboring face and use the V key to select the vert that has no connection to the neighboring face, mark this vert and switch back to the face with the T-joint, select the edge again where the T-joint is and click on this button: - you can see the result in the image below.



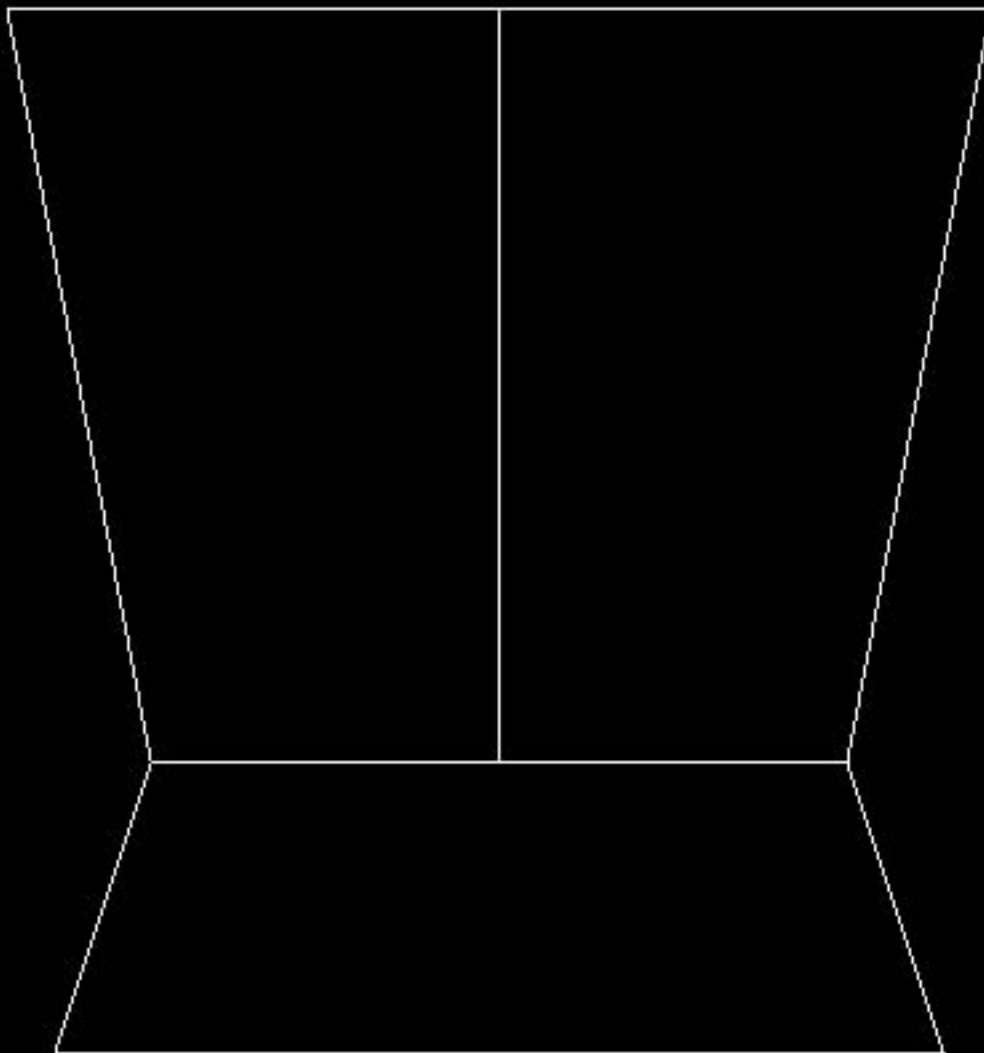
If you now again File/Verify Mine If you run it, you will see that there is one T-joint less. Repeat these steps until you no longer have any T-joints in your level.

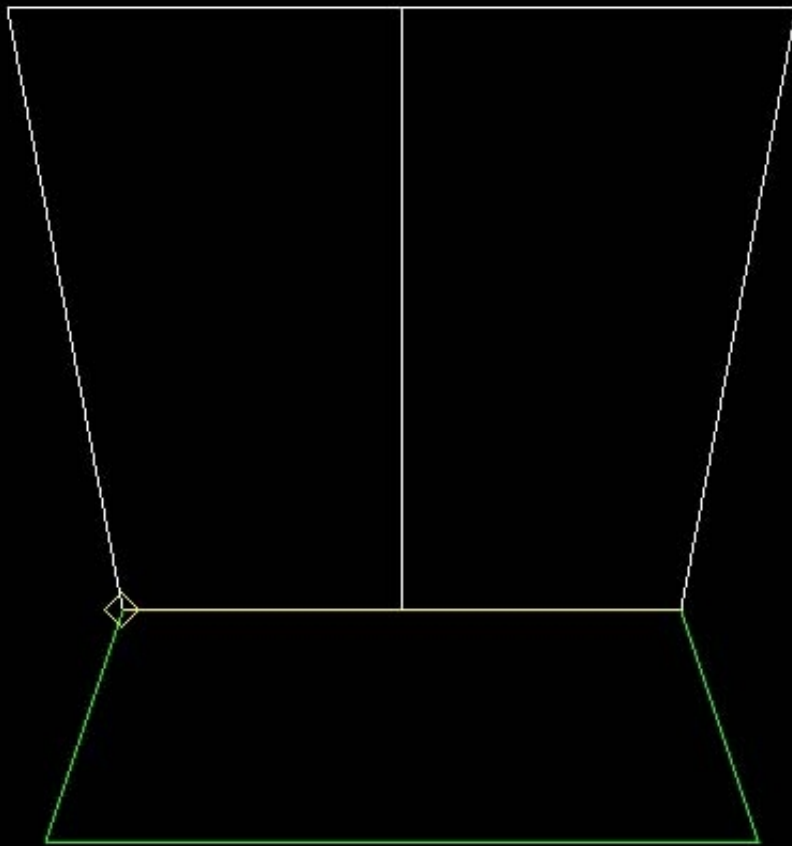
Back to Section D

053 - T-Joints: Second version

The Edge

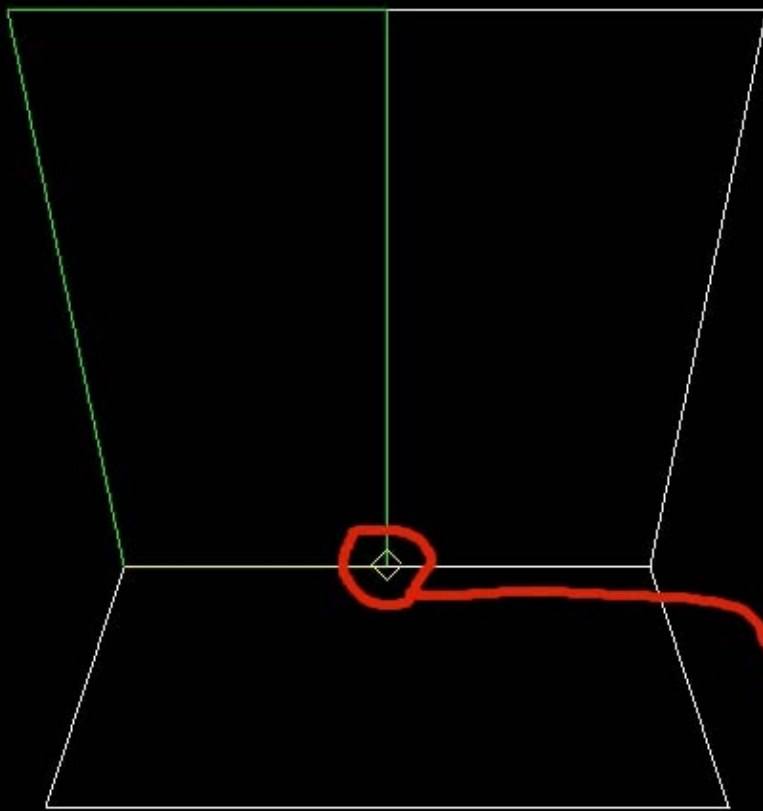
So here's another tutorial for repairing T-joints. We then have a situation in which 1 face on only one edge is in contact with 2 edges of the two neighboring faces:





There is no 100% agreement:

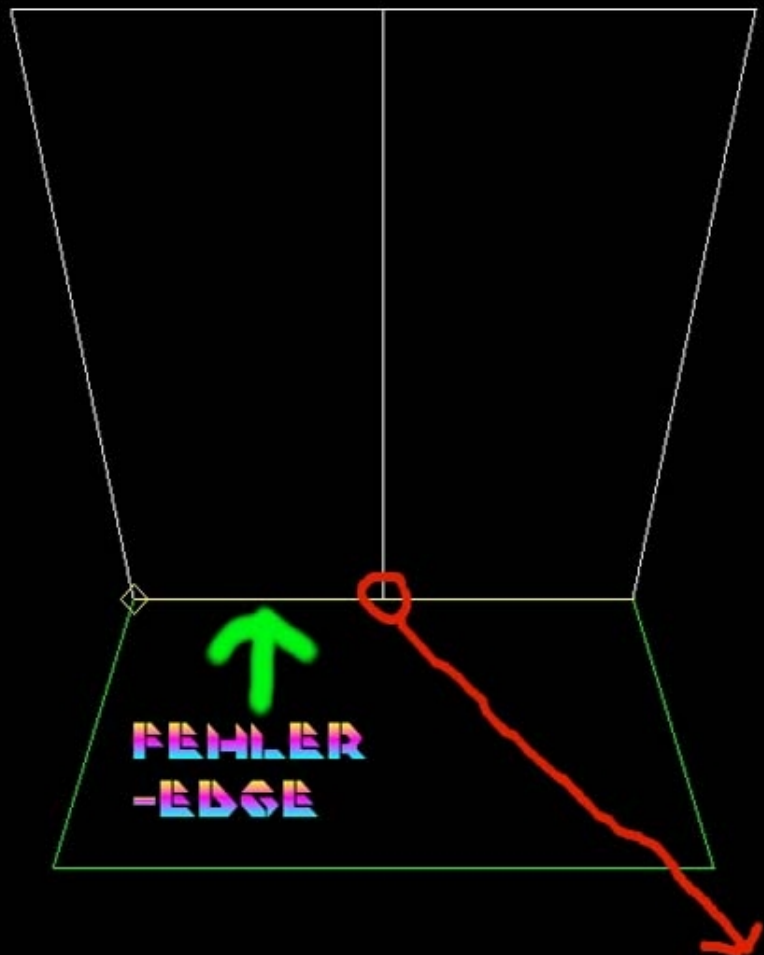




To fix this error, select the vert that partially causes the T-joint (red circle) and mark it **M**:

then click on the adjacent face (here, for example, below) and select the long edge **E** which also caused the error. then click on the button **Add Vert** (highlighted in yellow)

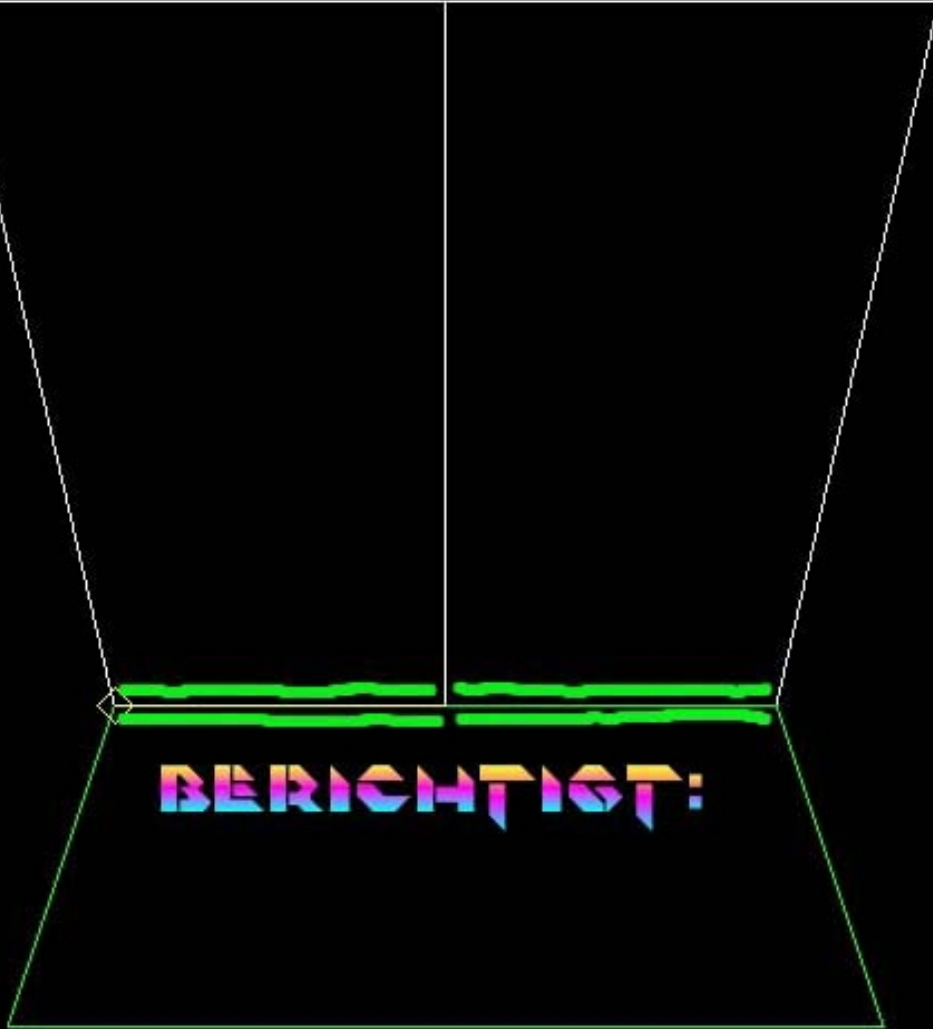
Grid: 0 90, 0, -100 -100, 0, 80 Vertex mode Marked: V: 1 F: 0



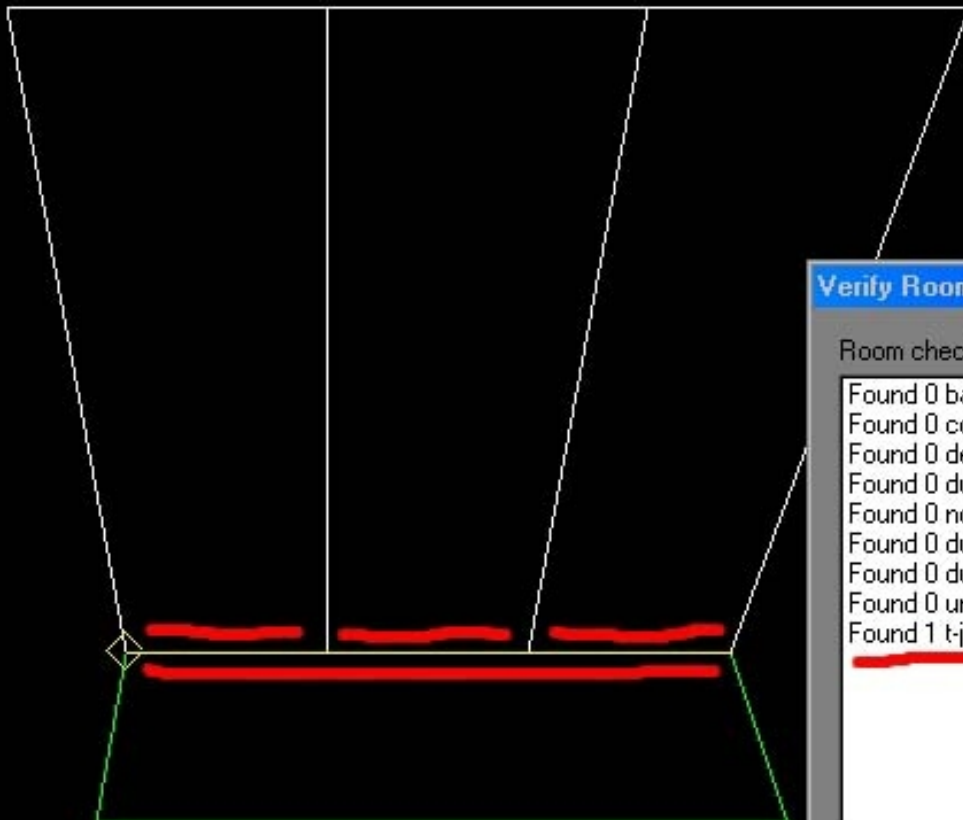
R: -1/0 F: 0/3 P: -1/0 E: 0/4 V (idx): 0/4 (0/8)

Grid: 0 90, 0, -100 -100, 0, 80 Vertex mode Marked: V: 1 F: 0

This way you divide the
error edge and two
Others that exactly
match the above:



BTW: even if an edge meets three, four, five (etc.) others, D3Edit only shows one T-joint:



Verify Room

Room check results:

Found 0 bad normals
Found 0 concave faces
Found 0 degenerate faces
Found 0 duplicate faces
Found 0 non-planar faces
Found 0 duplicate vertices
Found 0 duplicate face vertices
Found 0 unused vertices
Found 1 t-joints

Found 1 t-joints

NOTE: Check the system's clipboard
counts for this room.

Back to
Section D

054 - Verify your level

Kyouryuu

Introduction

Let's be honest. Be it chaos, Murphy's Law or whatever you want to call it, at some point something goes wrong. Sure, maybe that's not what you're aiming for, but it will inevitably happen, and you'll be forced to deal with the aftermath and repair. Descent 3 levels are no exception to this phenomenon. Sooner or later you will find out that the level you built has some errors in it. This tutorial will help you understand how to verify your level and fix the bugs.

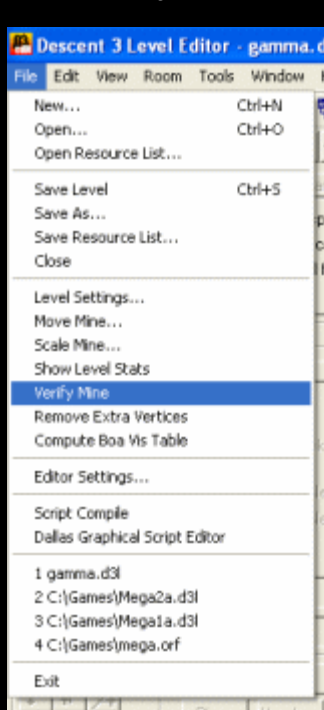
Why would I want to verify my level?

Making the level as bug-free as possible is always a good thing. Mistakes can allow the player to get beyond the walls of the level and enter the 'nothingness'. The walls prevent the player from seeing into nothingness and getting that hall-of-mirrors effect. Not to mention that you increase the overall speed and performance of the level. And it's good karma - you want things done right, not just from a gameplay standpoint, but from a technical one as well.

But Outrage didn't always do it!

True, but just because PTMC Proving Grounds (Retribution Level 12) has three concave faces and 62 bad shells doesn't mean you don't have to worry about it. However, it should be clear that there are a few levels (mainly single player) that are almost impossible to get right. Many of Outrage's door models, for example, come pre-loaded with attractive T-joints and other annoying stuff. You can fix the ones you want... but sometimes it causes more problems than it fixes. For the record, almost none of these bugs will actually kill your level or crash Descent 3. However, for performance reasons you should keep the number of errors as low as possible.

Verify Minecall



simply. If you're in World View, this includes the option `Verify e`, if you are in the room view it contains the option `Verify Room`.

basically do the same thing. However, it contains `Verify Mine`-Dialogue more information and that's why we will focus on that.

Understand the output of Verify Mine

If you select it, a small window will pop up. There will be some important ones

information about your

show vel. The following is an

ideal scenario where everything

is defective. You should on it

ielen that `Verify Mine`

This image provides. This is

the verify output from

actor Gamma.

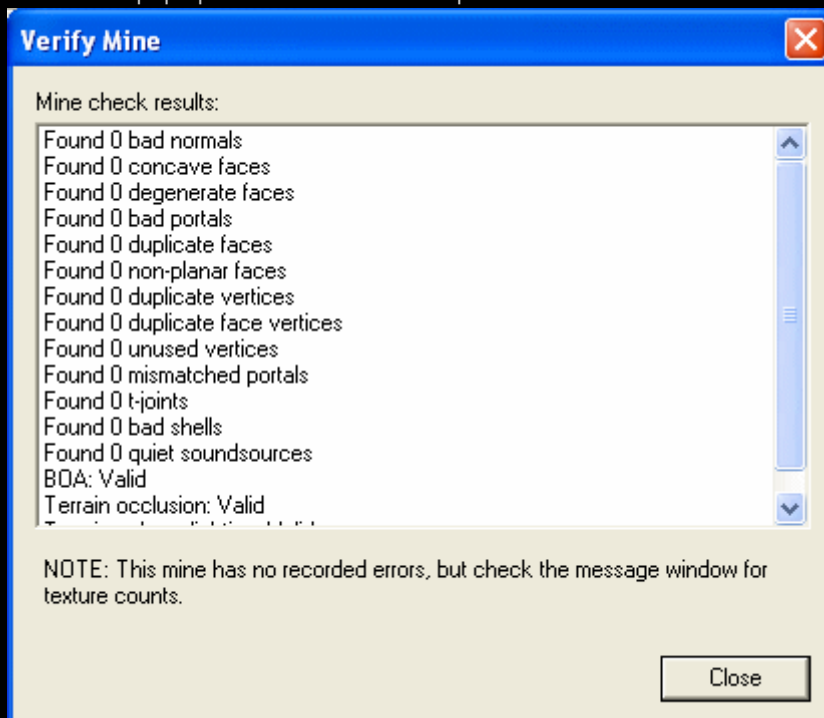
Specific information about

the errors is often provided

where they are copied to

the clipboard. You

you can paste this into Notepad or any other text program to see them. Or, you can use the new message bar in D3Edit (more later). Let us break down what each error means.

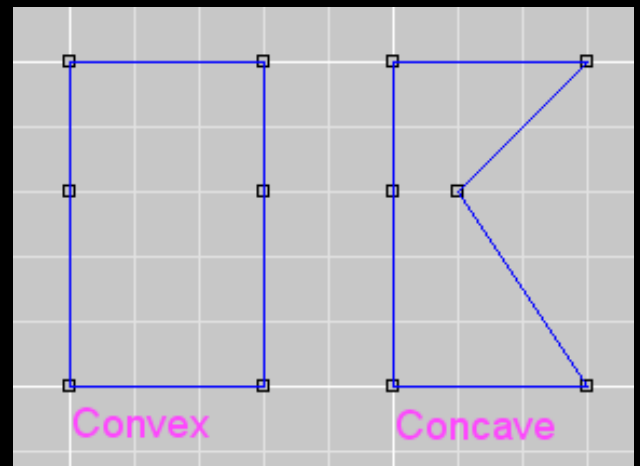


Bad Normals

A bad normal is a face normal whose direction cannot or can no longer be determined. This happens when faces are deformed into lines or even points; Lines or points have no normals, only surfaces (no bodies either, only their surfaces) 😊

Concave faces

In Descent 3 (and most if not all level building tools for that matter) faces must always be convex. Basically, think of a face like a pizza. It is round, whole and good. Then when I took a slice of the pizza it was concave. In other words, convex means that any point in a face can reach any other point without leaving the boundary of the face. For example:



The repair is usually a simple affair. You can either use Split Face to create the concave

Face to cut, or you can cut the face into triangles with Triangulate Current Face. This brings up another good point - a triangle can never be concave. Any shape after that, a rectangle, pentagon, hexagon, etc. can do this. By the way, this concept of 'concave' and 'convex' is limited to level design. They are oversimplifications of actual mathematical concepts.

Degenerate Faces

This is a face that has lost dimensions. In other words, it is no longer a 2d figure but either a one-dimensional figure like a straight line, or a zero-dimensional figure like a (mathematical) point. They most often occur when you move vertices around after an extrusion and you end up with 'collapsed' faces. You can delete them to solve the problem, but selecting them can be a real mess since they are - after all - just more lines or dots. You can have the faces in one room go through, or you do Right click in the world view and choose Select Face By Number. You can get the number in the message bar (more later).

Bad portals

Usually caused when the two faces that make up the portal are not exactly aligned. In other words, a face may have an extra vertex in it. Or maybe they are no longer aligned because you moved them in the spatial view. Or, worst case, the faces are far from matching. Careful planning will prevent this mistake from occurring. If it happens, you will want to delete the portal and recreate the faulty face or faces so that the portal can be made tidy.

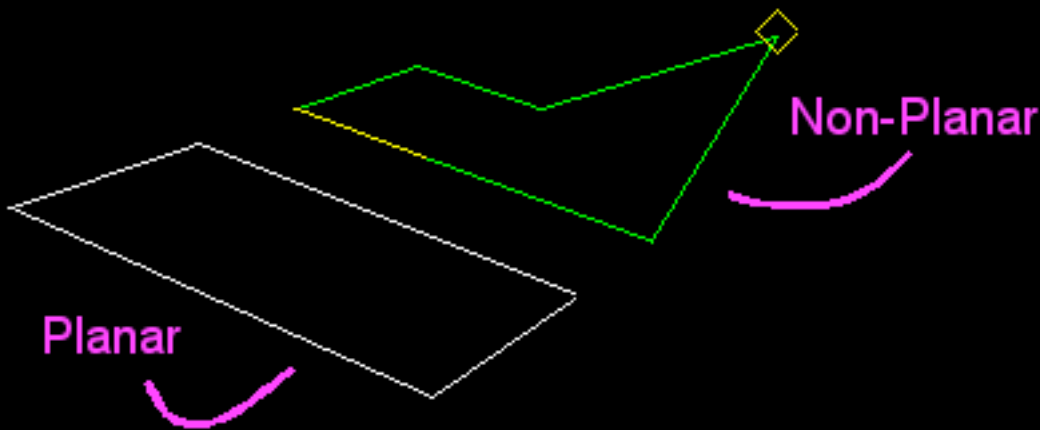
(Some doors may cause this error)

Duplicate Faces

This one is quite obvious. Two faces exist in the same plane, share the same sets of vertices and their normals face the same direction. Or in shorter words, you have one face on top of another. They are usually obvious in the textured views, as you can see the textures on the faulty faces 'fighting' each other as you move around the view. You can 'safely' delete duplicate faces.

Non-Planar Faces

The concept of nonplanar faces is similar to that of concave faces, only this concerns here the flatness, not form. An example:



Although it's a little hard to see since there's no grid here, seen from above are the vertices of these two

forms in the same position. Nevertheless, from the side, one of the points is moved up been. In this case became planarity

injured. A planar face is one where all of its vertices lie on the same plane. This means that a planar face should be completely flat when viewed from the side. Many non-planar faces are also counted as concave faces. The repair is again similar, although you will find it easier to split a face rather than triangulate it. An example:



Duplicate Vertices

This is rare but does happen. Basically, suppose you have a triangle but instead of it having three vertices, you have four - that is, one vertex sits on top of another. To repair this, you can select the affected face and press **v** pass through the verts. When you reach the point where pressing twice **v** doesn't seem to move on to the next vertex, then you know you've found the double face vertex. Use **Remove Vert** to eliminate them.

Unused vertices

They're not dangerous, but they're like sawdust on the cutting room floor - you may want to vacuum. Fortunately, this is among the easiest errors to fix. Just press **Remove Extra Vertices**. Complete.

Mismatched portals

They're like Bad Portals, except I think there's an emphasis on having a replacement vertex or vertices. This generally happens when you make a portal and later change the faces on one side of the portal (e.g. adding a vert on one side of the portal face but not the other). These are considerably easier to repair. To reshape the portal by deleting the current portal and then **Join Rooms** usually solves the problem.

T-Joints...

... were a nightmare to repair. Luckily, the latest developer versions of D3Edit (specifically 8j) have a tool in the Geometry panel that eliminates the T-joint when you do the face current that contains the T-joint.



A T-joint is typically defined as a place in the level where one face of the space shell is not flush with another, producing a hole (a 2d 'line' hole) into 'nothingness'. Or, in more common terms, if I flooded my level with water, T-joints would be the places where the water would seep out. Visually, the surefire sign of a T-joint in a level is a thicker-than-usual line in wireframe mode. (I couldn't reproduce with v039,

but you can see a difference to normal edges in wireframe mode) T-joints are a common enemy

in almost every level building software (especially annoying in UnrealEd2). Note that rooms that have the External Flag enabled are immune to T-joints.

Therefore, external buildings and structures are exempt from T-joints and you don't have to worry about it.


Bad Shells

Each room in D3Edit consists of two main types of faces – shell faces and detail faces. Shell faces typically have normals that point inward and define the outer 'shell' of the room. If you look around where you are sitting (probably in any room), the shell faces would be the walls of the room - the walls facing inward. Then detail faces are those faces whose normals typically point outwards and exist in space. These are things like lighting fixtures and the monitor on which you are currently reading this text. A bad shell occurs when the shell faces are not 'airtight'. The usual suspects are T-joints or double vertices. Look in the appropriate places to remedy this.


Quiet Soundsources

Sound sources are a rarity in custom levels, but they can add a lot of atmosphere if used correctly. Basically, they are invisible objects in a level that emit a sound you specify, which you fly into their area of influence. Ultimately, waterfalls should sound like waterfalls. However, a sound source is useless if there is no sound associated with it. A quiet sound source is one of these. It can be deleted, or you can assign a sound to it.

BOA

That is either **valid** or **Not valid**. The BOA Vis Table is a necessity for D3 levels. It tells the game how to load rooms into memory based on the player's location, thereby optimizing level performance by not having to load the entire level at once. But there are other advantages too. BOA Vis is automatically calculated when you do the lighting, when D3Edit notices a change in the level geometry or when a BOA has yet to be calculated. You can start the BOA calculation tool without the level lighting by clicking the button at the top of your screen. You do not need  calculate a new BOA unless you change the level geometry in some way. The game will then fuss over a BOA error when it loads the level, so you have no excuse for forgetting it.

Terrain occlusion

Either **valid** or **Not valid**. This Error is irrelevant unless you can see the outdoor area in your level. If your level is completely indoor and you never venture out onto the terrain, you don't need to worry about it. When you open the terrain panel (), press Calculate Terrain Occlusion. This process can (becomes) take a few minutes. I'm not entirely sure what terrain occlusion is - although it undoubtedly has something to do with optimizing performance when you're outside.

Terrain volume lighting

Read this as "Have you calculated the lighting for the terrain?" Again, it's either **valid** or **Not valid**. This is also irrelevant if the terrain is not used or cannot be seen from the level. Go to the lighting dialog and turn **Dynamic Light** and **Light it!** at. Note that you only need a few iterations to calculate the light of the terrain (I do 500 but only because I can). Convergence is not mandatory.

Use the message bar to identify errors

Thanks to the collaboration between Nirvana and OtherOne, the latest versions of D3Edit (especially 8J) have the Message Bar. Press **C** es a window to appear at the bottom of your D3Edit window.

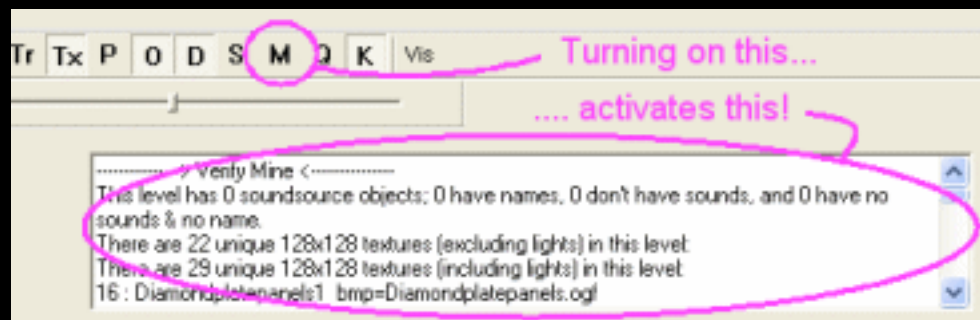
If you are your brave
mine expanded with
the message, you will
be invulnerable. d
Message Bar shows
Verify Mine so far in
the clipboard k

has. Here you have to have

more between the two

Applications jump back
and forth to fix the errors

read. the message bar tells you exactly where each error is, usually the space first, then the face, then the edge or the vertex. Learn this! It will make your life a lot easier.



Back to Section D

055 - Verify your mine

WillyP

You need the latest version of D3Edit (v39). You also need your existing, over-L33t, but slightly broken level. And a magic potion made from three

One part perseverance and three parts hard work 🧪

Note: Vert, Verts, Vertex, Vertices... all the same.

Building a Flawless Mine

It's not that hard...really!

Why do this to yourself?

Simply put, a poorly constructed mine will not play well. Shifting textures and faces, disappearing faces, players outside the level, these are some things you you might experience at a level like this. But to tell you the truth, a few mistakes won't kill you. The Outrage levels have a bunch of errors, T-joints, bad shells etc...

Usually these problems are easy to find and fix, but it takes patience and persistence! There is no reason not to have an error-free mine. If you encounter errors for which you can't seem to find a solution, ask for help (box at the top right).

www.descentbb.net

English-language Descent forum

www.descentforum.de/forum/

German-speaking Descent forum

both have a level development department.

Where do errors come from?

Try as you will, some mistakes are unavoidable. Most of the functions we use to create rooms and levels builds can produce errors, even the simplest operations like moving a few verts can produce a variety of errors.

When should these errors be repaired?

As soon as possible! It's never a good idea to add errors to errors. Most errors can be eliminated as soon as they occur. If you created a room separately from your mine, verify it before attaching it. As you become more familiar with the processes, you will get an idea of which operations are likely to produce errors

and which ones don't.

Verify Mine

Mine check results:

- Found 0 bad normals
- Found 2 duplicated rooms
- Found 0 concave faces
- Found 0 degenerate faces
- Found 0 bad portals
- Found 0 duplicate faces
- Found 0 non-planar faces
- Found 0 duplicate vertices
- Found 0 duplicate face vertices
- Found 0 unused vertices
- Found 0 mismatched portals
- Found 0 t-joints
- Found 0 bad shells
- Found 0 quiet soundsources
- Found 12 RoomCenter not inside Room(s)

NOTE: Check the message window for details for this mine.

Level Stats

Level statistics:

- 227 Rooms (10 external)
- 45805 Faces
- 53740 Vertices
- 288 Portals
- 36 Doors
- 95 Polygon Objects (94 inside, 1 outside)
- 15421 Object Faces (13539 with lightmaps)
- 59344 Total lightmap faces
- 1514346 Total volume bytes
- 1577890 Total bytes in lightmaps
- 0 Total specular faces
- 1895518 Bytes wasted in lightmaps
- 3 Energy Centers
- 0 Red Goals
- 0 Blue Goals

Statistics gathered.

227 rooms and 45805 faces - error-free!

Can every error be eliminated?

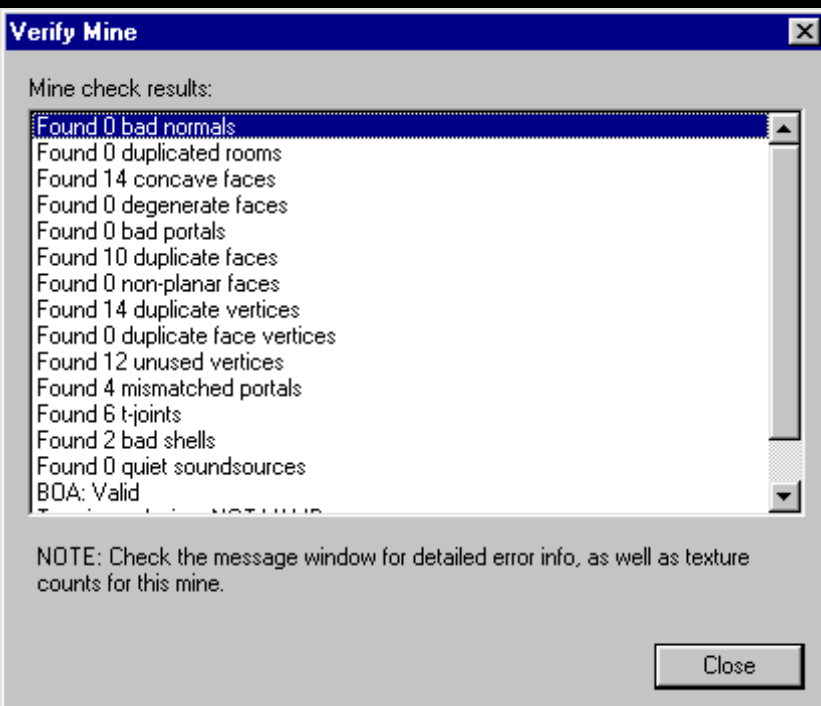
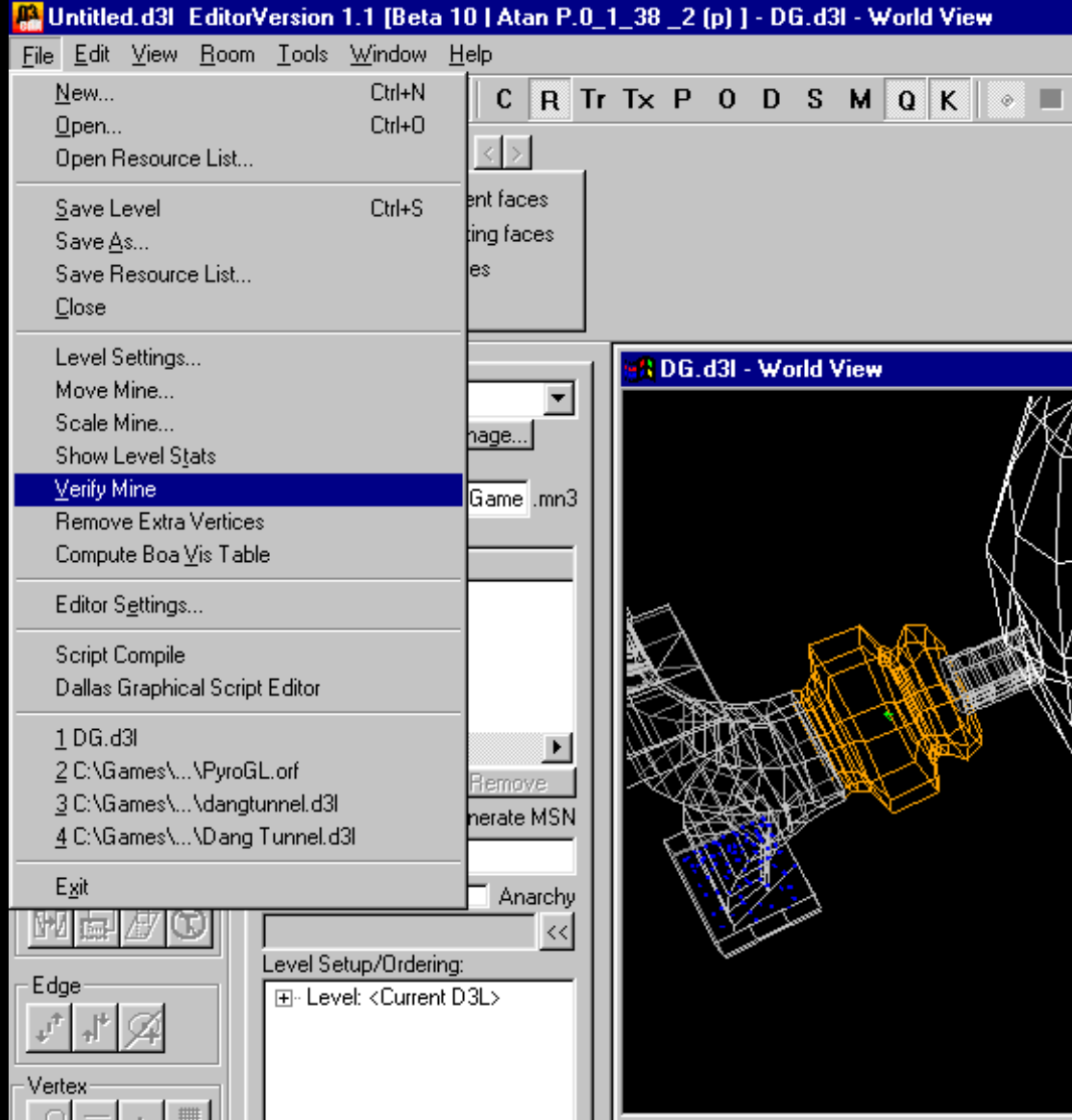
I think so. But it should be mentioned that some 'mistakes'...Verifyreported are not really errors. For example, that in external spaces it is reported that the center of the space is not in the room. And if you make a space on the terrain, then bind all of its faces to an external space,Verifywill report this as double space.


When a mine is built, it is inevitable to introduce errors. But we can and must remove these!

A small mine with typical errors

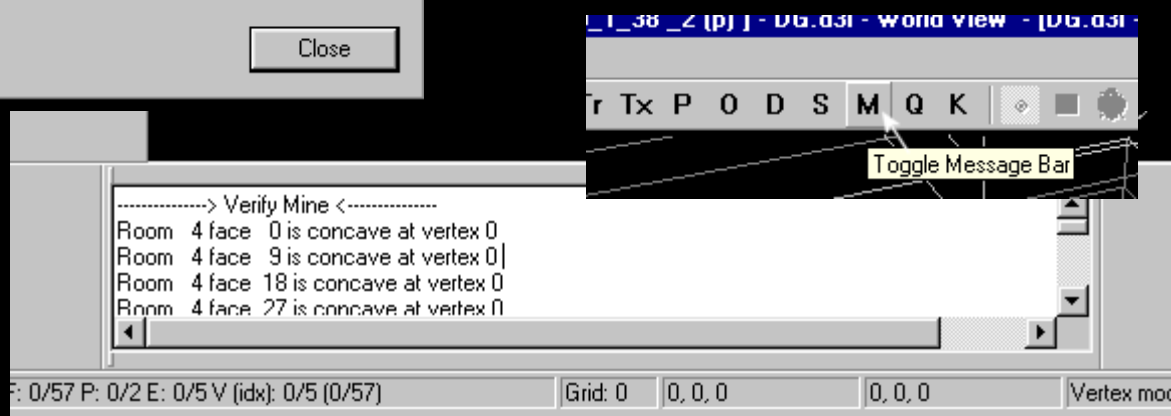
It is the first step to let it run. Open your mine and in the world view choose Verify Mine out of the File-Menu. Or choose Verify Room, when you work on a room.

Verify reports



Here is Verify's report. There are no bad normals or duplicate rooms here, but we'll cover them anyway. Press the button . This opens the Message Bar. In this window is additional information from Verify. It tells us which space Face and Vertex has the error. Right-click in World View, then select Select Room by Number, and enter the room number from the message bar. Click OK to go to this room. You can then select your faulty face or portal by number, and both cameras, in World View and Current Room View, will zoom in on the affected face. This sweet tool (thanks, Atan!) also works for objects.

Now let's discuss what are the errors and how to use them repaired.



But first another good tool...



This one is under the button **Hidden**. You need to be in a spatial view. Choose the one **Errors** card and select the bug you want to destroy. Then press **Mark**. Is that cool? It gets even better, you see the little buttons **<** and **>**? Press **>** and the first marked face becomes the current face! Smooth! Squeeze him again! Ah! Hit me!... Uh, I mean let's move on...

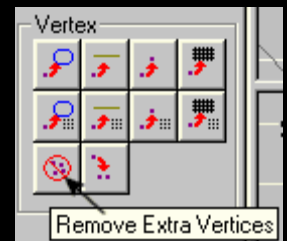
Common procedures

A few keyboard shortcuts

To move on to the next room press **R**, **Shift-R** for the previous one. Works in Room or World View. To cycle through faces in Room View, use **F** or **Shift-F**. Traverse the edges of the current face with **E** or **Shift-E**. Loop through the verts of the current face **V** or **Shift-V**. Use **P** and **Shift-P** for portals.

Remove Extra Verts

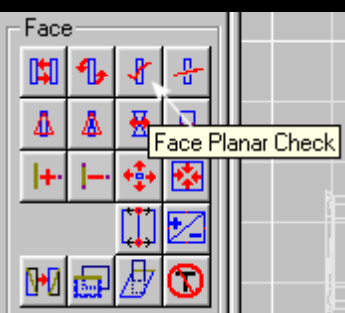
More than one vert can occupy the same space. But if these belong to shellfaces, that would be considered a hole in a shell. It's easy enough to eliminate extra verts, just click the button **Remove Extra Verts**. Many repairs can leave extra verts. And removing extra verts may also be all that's needed to fix some things. So if **Verify** says there are extra verts, remove them first.



Run Verify again

After making repairs, you should verify your room again, and when you're done with the room, verify the entire mine. To keep an eye on your progress and update the message bar.

Non-Planar Faces



Imagine a piece of paper on your desk. All four corners are on the same plane. Lift a corner and the paper will bend. It will 'non-planar'. Descent cannot handle non-planar faces, and will show unexpected results. Now imagine a diagonal fold in the paper so that three corners stay on the desk while one is lifted. The paper has been divided into two sections, each is planar. In our level we can split a face into smaller ones that are planar. A face that only has three verts will always be planar.

Once you have marked the non-planar faces, you can **Face planar check**

Use to split the faces. This is the quick and easy way, but you will need to re-texture the new faces. You can also split faces manually. Mark two verts in a face. The next button on the right is **Split Current Face**. Use this to control the way faces are split. For example, to maintain symmetry.

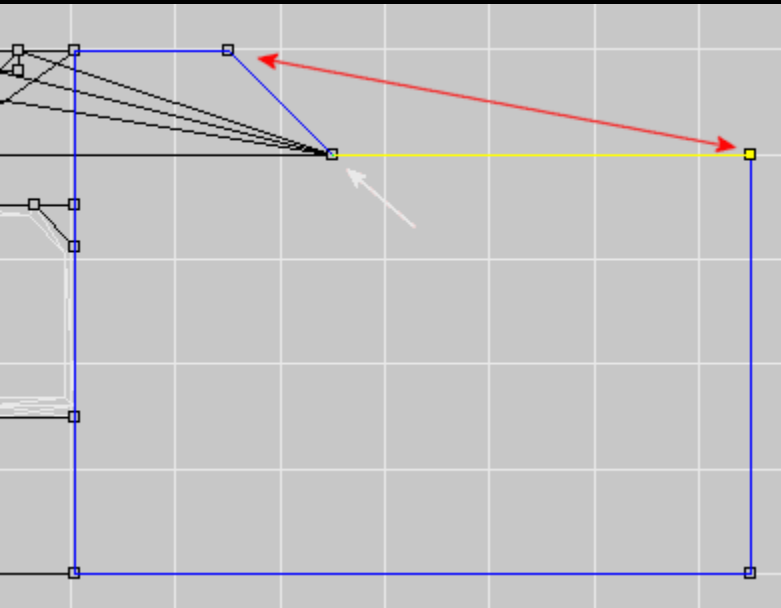
Another way to get a planar face again is to bring its verts to the grid, then move them to the same grid line. Use **Snap Marked To Grid** after you mark the verts, or you can also snap **Current Vert To Grid** use. These buttons can be found in the **Vertex**-Section

Yet another way to correct non-planar faces is to delete them and build new faces. This is sometimes the easiest way to get specific geometry.

Concave faces

Note, you can set the tolerance level for concavity in your editor's settings dialog. A lower number could detect more faces. At the lowest setting, 0.0010, you may have a little more work to do, but you will end up with a better shell.

If you can draw a line between any two vertices on a face and it is outside the face, you have a concave face. The best way to deal with this is to split the face. Use the Mark Tool to make the concave face Current. Check the message bar to see which vert is causing the concavity if you are unsure. Select this vert and another one (but not the next one), and press Split Current Face. A



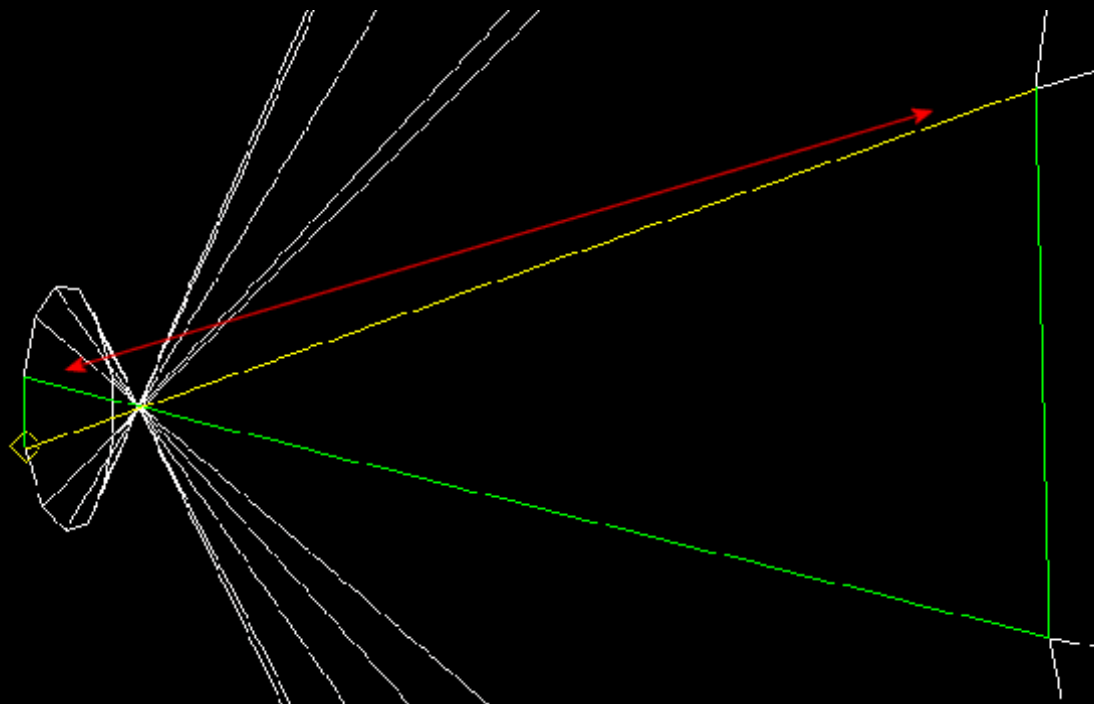
Face that only has three vertices will never be concave. Some concavities are difficult to see, but using this procedure you can easily remove all concavities from your level. As with nonplanar, you may sometimes prefer to delete the face and make new ones.

Here on the left is a view of a face that is concave. The red line I drew between two verts is clearly outside the face. The white arrow points to the vert, which is concave. I *could* just move the vert up, but that would change the architecture of the room. And it could cause more errors. It's easy enough to mark the concave vert and another to split the face.

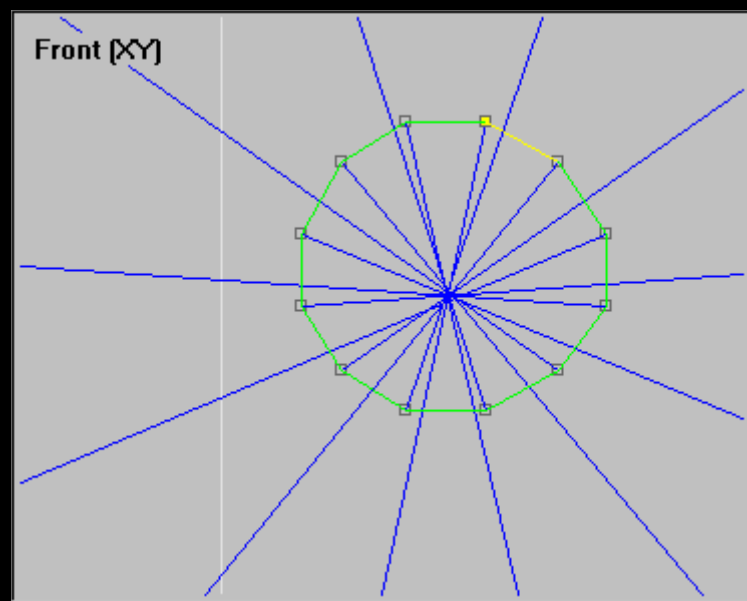
Here is another example of a concave face. This one has four vertices. The current edge crosses the one opposite lying. All four vertices are on the same plane, so the face is planar.

I have a red line drawn between two vertices. This is a unique situation in which the face is twisted in on itself. There are actually twelve faces here in the same dilemma shown. This mistake

is so serious that it allows the player to fly out of the level and makes the level very laggy. Since the shell is not properly defined, there is no way to know when to stop the player. I haven't tried it, but I think if I had tried to simply split all twelve faces as described above, I would end up with a mess of confused faces and possibly other errors. But I have another idea.



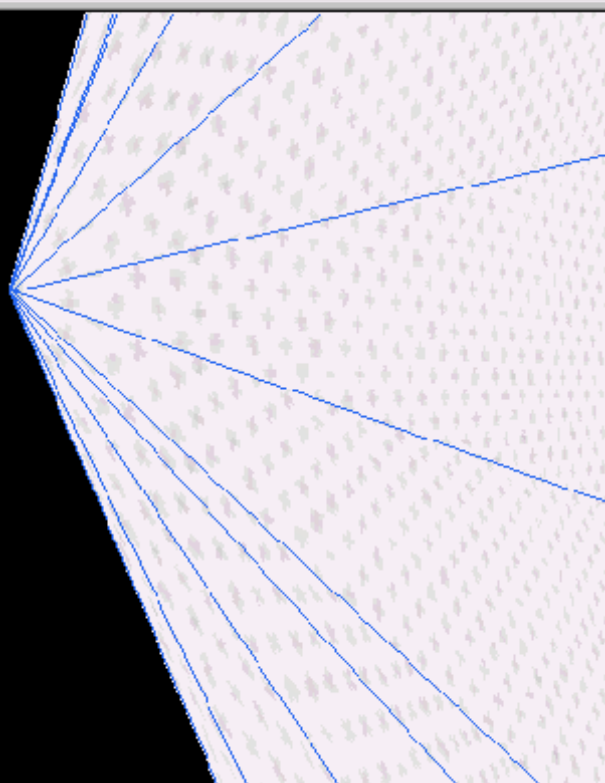
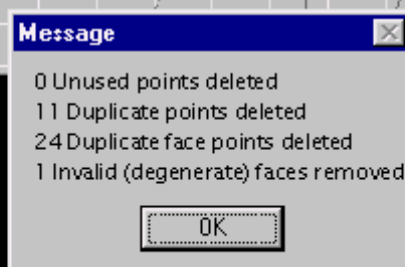
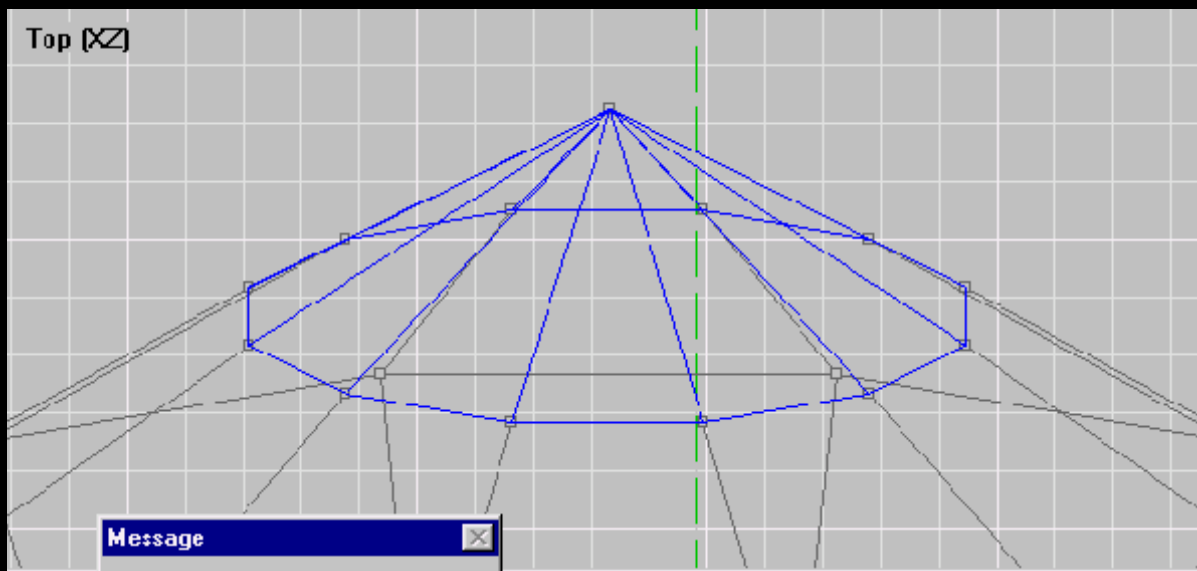
In this view we can see that there is a twelve-sided face (wouldn't that be a dozen-sided face?) attached to our problem faces. In Vertex Mode I will use the lasso to mark all of his twelve verts. Make sure one of them is current (click on the face to make it current if it isn't). I press **Snap Marked To Vert**. Twelve verts, all of them



in the place of one. And the area of the twelve-sided face was reduced to the size of a dot (a Degenerate Face).

The edges of the concave faces, which are in common with those of the twelve-sided face also became zero in length. They all disappear and become a vert when we do that

Remove Extra Vertspress.



Duplicate Faces, Bad Normals, Degenerate Faces

If two identical faces occupy the same space, they are duplicate faces. Mark them with the Mark tool and delete them. Mission accomplished. Bad (bad) normals are not good normals. Degenerate Faces were good faces, but they fell on hard times; there is little to no hope for either. You can send them packing using the old mark-delete-rebuild routine. Sometimes snapping a marked vert to another also works; then it works too. **Remove Extra Vertices**, although you may have caught these first. Although, not exactly a small hangover treat!

Mismatched portals, bad portals

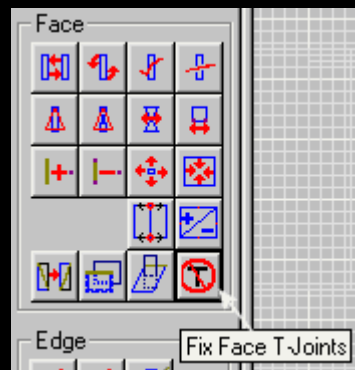
Most often caused by moving a face or vert that is part of a portal. Another reason is to attach a room to a face that is adjacent to a portal; When this face is split, a vert is added to the existing portal. This breaks the portal. You can try deleting the portal, reconnecting all the faces that were split through it, then join Rooms to use to make a new portal. Or, you can delete one of the rooms (save as a room file first!), reunite the faces, then place the room again. In the future, if you want to add a portal to a face that already has one, you can try splitting that face so that it doesn't affect the existing portal, then add the new portal. Or better yet, build the room with built-in insulation.

(compare Sirian's Tuts)

T joints

This one is relatively simple. A T-joint is a connection between two faces. One face has a vert in the split edge, the other does not. There is a button to remove T-joints from the current face. Now you should know how to find and mark the faulty face.

You just added a vert to the edge that was missing one. If the face you repaired is a portal, you have also destroyed the portal. Look at the paragraph about portals. Many of the doors included in D3 have errors, including T-joints. If you delete the door and reinstall it, you will have the same problem. You can try deleting the portal, repairing the T-joint, then reconnecting the rooms.



Important: Never delete a door with Delete Room. Delete a door ALWAYS via the door panel!

Bad Shell Faces

These can be the trickiest to repair. Often they come as a result of other errors and are repaired when the other errors are eliminated. Sometimes they result from repairing other errors. If you repair all other errors (and **Remove Extra Verts** done) but you still have a bad shell error... well, good luck, you're on your own... no, just kidding... But... Are you sure that everyone else has fixed any errors? Make sure you run Verify Mine again to be sure. Mark the faces using the **mark-Bar**, and carefully examine the area in question. By going through the verts of the edges identified in the message bar, you may find a tiny gap that isn't really visible... either snap the verts together or do a New Face.

BOA and lighting

Once you have satisfactorily removed all errors, you will need to recalculate the BOA and run the lighting again. If the BOA is not valid, it will be charged when the lighting is executed.

Back to Section D

Section E–Terrain

... and there is light at the end of the tunnel!

(Original sound Interplay/Outrage, from the promo video)

It is strongly recommended to practice integrating terrain several times, at least until you are fluent in it. If the link between the mine and the outside world is not made properly, it can be assumed that the level will not function properly.

056	Terrain implementation	Explanation of how to make terrain and edited	Kyouryuu	218
057	Create outside world	A step-by-step walkthrough, how to create a terrain and integrates	Fischlein	221
058	Outside World – Update	Mine and outside world with the new one Connect D3Edit	Ragil Ral	227
059	Building	Make buildings on the terrain	Ragil Ral	229
060	Player start in the outside world	Different ways to let the player start on the terrain	Ragil Ral	231

056 - Terrain implementation

Kyouryuu

Introduction

One of the key features that Descent 3 offered beyond its predecessors was the addition of outdoor areas. This part of the Fusion engine works differently than the indoor part that you are most likely used to using. This tutorial will guide you through creating a terrain elevation map and implementing it in D3Edit.

Quick reference

For those of you who already know how to work with terrain, here is a brief list of the controls used to work with terrain.

Ctrl-T–Toggles the terrain in the World View. **Left click** on terrain cell – Selects the terrain cell.

Ctrl-Alt-Shift-Left click on terrain - 'flood fill' which makes all terrain textures of the same texture the current texture.

Shift.(Point) – Rotates the texture of the current terrain cell 90° clockwise

shift,(Comma) – Rotates the texture of the current terrain cell 90° counterclockwise **Shift-**

double left click to Satellite – Changes the satellite texture. **Shift-left click** on Skydome –

Changes the Skydome texture to the current texture. **Shift-left click** to Terrain Cell – Sets the texture of the next terrain cell to the current texture.

Create a height map

Terrain in Descent 3 is defined using a height map, which can be created with any painting program. Basically, the way it works is that you create a grayscale image that is 256 pixels square. Pure white (255, 255, 255 in red, green, blue) represents the maximum height, solid black (0, 0, 0) represents the minimum height. In between you have your hills and cliffs. The player ship cannot fly over pure white as this is outside the invisible barrier that lies on the terrain.

To make good terrain, you will want to have a mix of gray and black areas where you want the player to be able to fly through, and white everywhere else. Be sure to keep the boundaries of the terrain white, otherwise the player could see the 'nothingness' outside the terrain. I like to use the regular brush to draw the main paths in black. Then I have some gray to line these paths. Finally, I use the smudge and push tools to blur the edges of the colors together, maybe even run a Gaussian blur over them. This appears to produce fairly natural terrain. When you're done, save your terrain as a 256-color grayscale.pcx-File. If you have the option, make sure it is a version 5 PCX file.

Import the height map into D3Edit

To make this clear, let me make one point. D3Edit will not allow you to manipulate the shape of the terrain in and of itself. The only thing D3Edit can do inherently is texture and light the terrain. Anything else that needs to be physically changed must be handled via your painting program and then imported back into the editor. If you have one.pcxIf you re-import, you won't lose your texture information.

That said, now you can open D3Edit and load a level and go into World View. At this point, turn on the terrain panel with Tr if you haven't already. Click on the Import PCX button and navigate to the folder where you want your.pcxFile inside and open it. After a short while D3Edit should show you the terrain in the view. If you can't see it, **Ctrl-T** switches the terrain on/off. In wireframe mode, the terrain is represented by a grid of points. In the textured modes, unsurprisingly, you see the terrain with its textures.

Edit the Skydome and the satellites

You'll notice when you have the textured view in World View that there is a gigantic semi-circular shape that extends over your terrain. This is, of course, the Skydome. This also contains the satellite elements. These in turn are the little decorations you see in the sky, like the sun, planets, moons and so on. For most purposes, you will want to have at least one satellite that is a light source and that casts light onto the terrain. The problem is, there aren't very many satellite textures that have light values per se. You can find a complete selection in the texture selector, below

terrain. I find that MercSun serves well as a base sun. To use it, first look for your satellite in the Skydome (easier said than done - you may have to move across the Skydome before you can see it). By default the satellite is a rather ugly blue and purple wall texture. by pressing the **Ctrl** button on the satellite **double click** will change its texture to whatever the Current Texture is. Similarly, you can also edit the Skydome by **Shift** press and on it **left click**.

Optionally, you can also play around with other settings in the terrain bar. Stars for example, places small stars in the sky that blur when you move. This is great if your skydome is just black space, rather than having clouds or other atmospheric features. After all, stars beneath the clouds just aren't a realistic sight. Another parameter worth mentioning is that Red. Deg.-Field. What it does is gradually rotate the sky texture around the skydome. A setting of .50 is slow and natural. Of course, if you want to rotate your sky like a hot tub, be my guest.

Texture your terrain

Just like your elevation map, your terrain consists of a grid of 256 by 256 cells. That gives a total of 65,536 cells in total. Each cell measures 16 units square. Thankfully, you don't have to sit there and color each one. The main way you texture terrain is by applying the base texture first, and then adding the other textures sporadically. For example, if your terrain has a snow theme, you'll start by laying down a basic snow texture, and then sprinkle other types of snow textures here and there to break up the monotony. You simply texture a terrain cell the same way you would texture a mine face. As mentioned at the beginning, **Ctrl-Alt-Shift-Left click** on the terrain is like a 'flood filling'. All attached terrain textures with the same texture will be changed to the current texture. Since the terrain is completely covered in an ugly purple texture by default, it's easy to 'flood fill' the entire terrain initially.

Some terrain texture schemes, like the canyons and especially the Piccu Station Sand/Grass/Earth set, contain textures that you have to rotate. On the terrain you can only rotate textures at 90° angles, and it may take a few tries before you learn which texture rotates so that it seamlessly connects to the next. Rotation is done by first selecting the terrain cell you want to rotate by right clicking on it and then **Shift**. or **shift**, (Period or comma) to rotate clockwise or counterclockwise.

Terrain occlusion and lighting

This is the easy part, although it can take a long time. For the final version of your level you will want to calculate the terrain occlusion and also illuminate the terrain. For the former, simply go to the Terrain panel and click Calculate Terrain Occlusion. This may take a while but unless you replace the PCX elevation map you don't have to do it again. You can then open the lighting dialog (Window->Lighting) Then when you light the terrain, iterations is the only relevant factor. Set this number greater than 100 (2000 is quite decent for the final version of a level). Then, in the Terrain section of the Lighting dialog, click Light it! If he asks you if you want to calculate Dynamic Lighting afterwards, let him do it. Then sit back and wait.

You might stare stupidly at the progress window that goes up to -480% until convergence is reached? Yes, well - the truth is that convergence is not necessary when you treat the terrain. It only matters if you light the mine. That's why we get away with so few iterations.

Connecting the inside with the outside

Let's assume for a moment that you have a mine and you want to attach it to the outdoor areas. First you have to create a structure for the mine that is visible on the terrain. Think of it like a metaphor - you have a standard office building that is your 'level'. The corridors and offices correspond to the interior of your mine. But you don't necessarily see them from outside the building because of the 'shell' that's on the outside. In this step you have to build this shell.

The exterior shell of the building is quite simple. It is a standard ORF room with all normals flipped outwards (an 'inverted' room). One point worth noting - keep the base of your structure relatively wide. You'll see why later. If you put him in your level, you have to go to the Room Properties for this room and External flag tick. By using Room View and World View together, you can move your structure across the terrain quite easily. Voila - buildings on the surface.

Finally, you have to punch holes in the terrain. If the terrain is visible from the player's location and the player is in the mine, they will see the terrain cutting through the middle of the mine where it meets the terrain. To clear this blockage, select each terrain cell by left-clicking on it and then use **Toggle Vis.** in the terrain panel. Now you see why I urged you to keep the building base wide. Because you're cutting one square out of the terrain at a time, it's a rather imprecise process. Cut away a little too little and you'll see the terrain in the mine. Cut away too much and you'll have holes all around your building when you look at it from the outside.

Objects and the terrain

This part is a little obscure. If you want to place an object on the terrain, select the object you want to add. In the object panel, select the terrain radio button. Then select the cell where you want to place the object and click on the Insert button while you are in World View. Most of the objects are used neatly in this way. Some of them, like the InvisiblePowerup, deploy depending on your camera position, which is a little clumsy. If you want to place the InvisiblePowerup, first insert another object and then convert it to an InvisiblePowerup.

If you want to move objects on the terrain, first select the object by right-clicking. Then use the following controls:

Alt-X - Locks movement on the X axis (left and right, default) **Alt-Y** - Locks movement on the Y axis (up and down, standard) **Alt-X** - Locks movement on the Z axis (back and forth, standard) **Alt-CursorUp** - Moves positive on the current axis **Alt-CursorDown** - Moves negative on the current axis **Alt-CursorLeft** - Rotates clockwise on the current axis **Alt-CursorRight** - Rotates counterclockwise on the current axis

If you right-click in the World View, you can also set the increments for the movement of the object and see which axis is currently active.

Back to the overview of section E

057 - Create Outside World <>

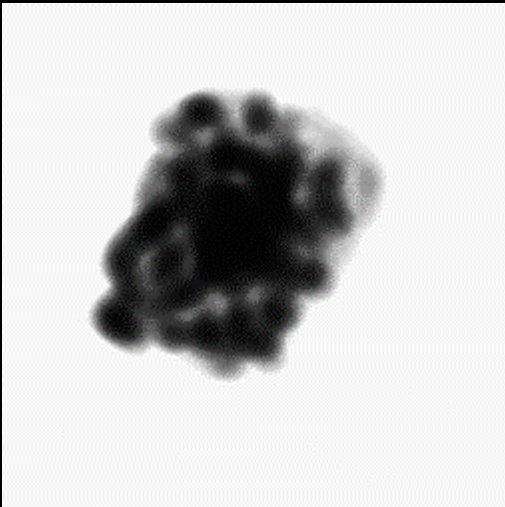
Fischlein

(revised)


To create an outside world you need D3Edit version 1.0 or higher!

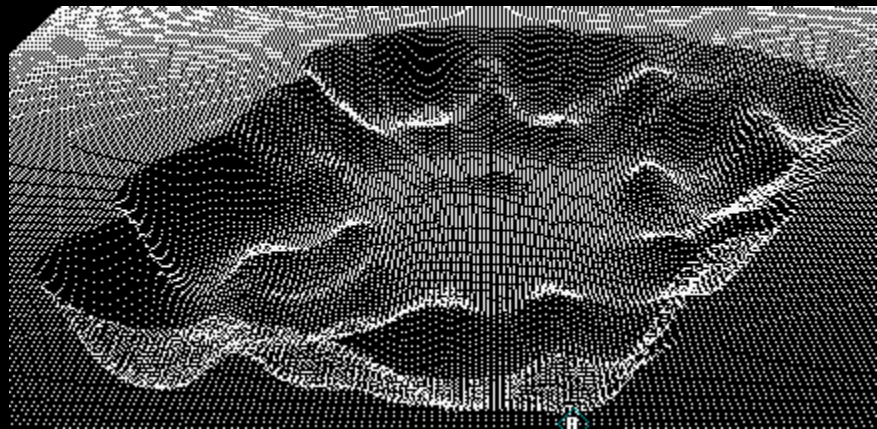
It's actually not difficult to incorporate an outside world into your level. You need a graphics program that supports the format **PCX** supports.

First of all, something to understand. You create a two-dimensional picture, now you paint a spot on the sheet (with black paint), now you have the colors white and black, with black making the valley and white making the mountain. D3Edit calculates the outside world according to the color gradations.



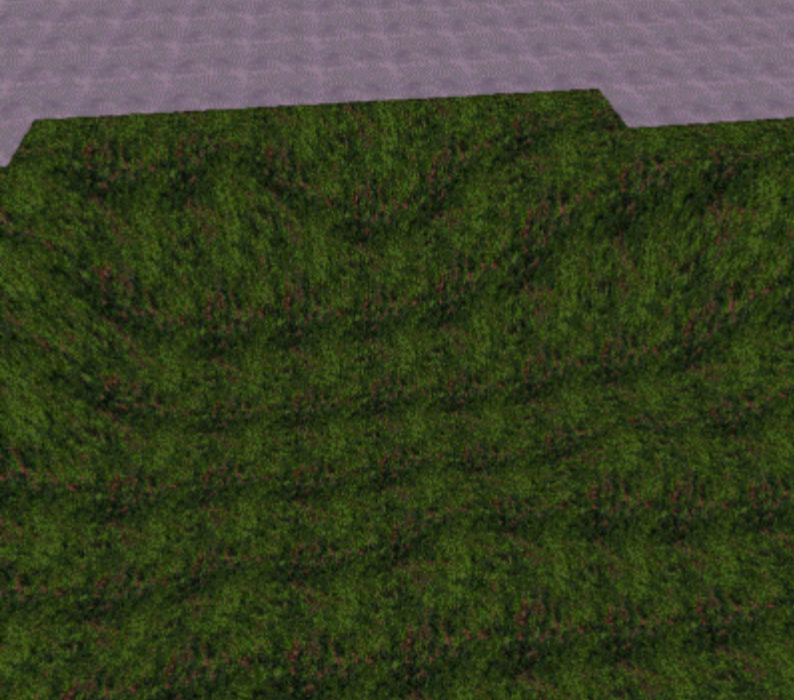
I once prepared an example. Creates a PCX file of size 256 x 256 pixels in grayscale (left).

Now you have to import the image into your level and this is done as follows. Open D3Edit, load your levels. Now click on the button  the **Terrain** menu opens. Click on the button **Import PCX** and import your file. The result looks something like the image below. (D3Edit – World View)



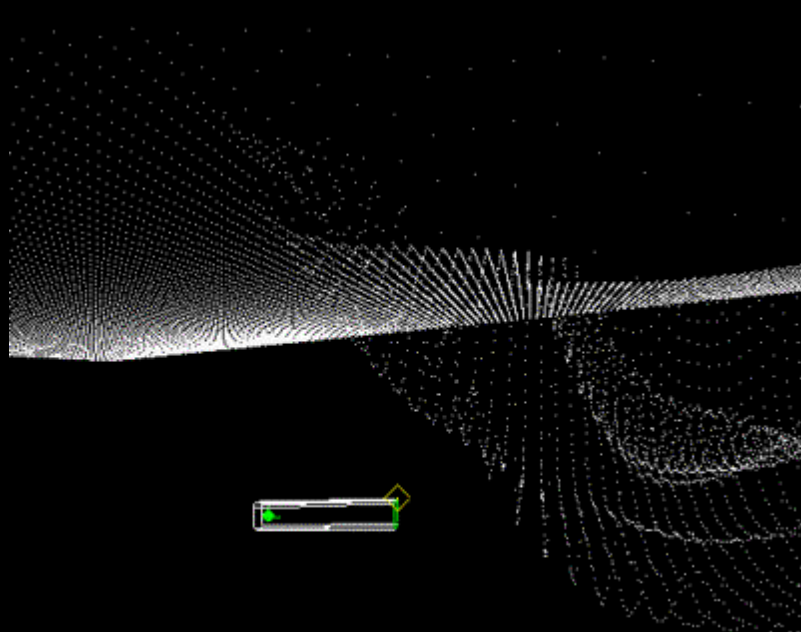
Now you still have to texture the outside world and it works like this. First click with the right mouse button in the World View view. In the menu you choose the following menu **Texture with outline**. Now you see the "landscape" with the standard textures. (Don't be taken by the depiction irritating) You can now join in to **Z** switch to the "Z-Buffer" mode, you can get a better display this way, but it requires a lot of computing power. But you have to try it out for yourself to see which is better.

Continue with the texturing. You only need to texture the "valley" and the "slopes" and possibly the upper edges. The rest that you don't need will be "cut off" anyway, but more on that later. First select a texture from the **Texture Bar (Terrain Textures)**. You should actually know how to do this, if not, look further ahead. In my example I choose the texture **Dan_Moss1**. Now hold this **Shift** Press the key and click in the "valley", now you have textured the first piece. Repeat until your outside world (valley and slopes) is completely textured. For me it looks like this at the bottom left.



Once you've done that, click on the button **Calculate Terrain Occlusion**. In the following dialog box you will be asked again whether you want to calculate it **Yes** click. Now it appears that the editor has crashed or hung. Don't panic, he's still working! If you have a tool that shows processor usage, you will see that D3Edit is still working. When D3Edit has finished calculating, you will get a dialog box confirming this. You will also be asked to save. Now let's take another look at our outside world and see that everything that wasn't needed has been cut off.

So, the outside world is now finished. But how can we fly from the inside of the levels to the outside? To explain this to you, I made the default room a little longer and placed it on a "slope", whereby the **Rainbow Texture** is close to the slope. (see picture on the right) I did it like this. I have in view **Current Room View**, in Vertex mode (**Ctrl+R**), all vertex with **M** marked and in the plane **Side (ZY)** (Grid 100), the room using the buttons **2,4,6** and **8th** placed in the correct position. Now we have to get the player starting point, to do this we switch to object mode (**Ctrl+G**) and open the **Object Bar** and check whether the player starting point is Current, you can see that in the object bar, where it says something like "PLAYER 0". Now back to the plain Slide (ZY), using the keys mentioned above Position the player starting point back in the room. Check this by looking in the World View.



Now we need another one **External room**. Switches to RoomMenu and selects Add Room at Current Face or press the button **Insert**, but note that you have clicked on the face that points towards the outside world.

It's now the turn of the new space, stretching it so far that it protrudes into the outside world. Texture it to your taste.

Then click on the portal that lies between the two rooms and select Room -Menu Delete Current Portal. Now click on them in the first room **Rainbow Texture** and chooses in Roommenu Link Room to New External Room. The editor then asks if he should create a new room, click here no. Now we have to use the second room **External** To do this, we click on the room in the World View and select it in the menu Window.../Room Properties. Put the checkmark like in the picture on the right.



Then joint the rooms.

Now we have to delete the face of the second room on the outside (the exit). To do this, we switch to the Current Room view **Ctrl+F** Go to face mode, select the face, mark it with the **Space** key and delete it with **Remove**. Now we seem to have the exit, but there is still no hole in the "slope". You could fly outside, but we wouldn't be able to get inside.

Switches with the button **Z** the Z buffer we need it now. Go to World View and position your field of vision close to the exit as in the picture on the right. You can already see that "Cell" is selected (clicked).

After we clicked on the cell, you can also see that something is displayed at the bottom of the status line, shown in the picture below.



In this image we see that Cell 20095 is clicked.



So now click on the button **Toggle Vis**. The cell you clicked will then be deleted. Note, if you want to insert cells again, you need to know the cell number, so always look at the status line and make a note of it

Numbers! Repeat the steps until the opening does not occur is blocked anymore and it looks like the picture on the right.



That would be done, now we have to build something around the exit so that you can't fly out of the level. I'm solving this Problem as follows:

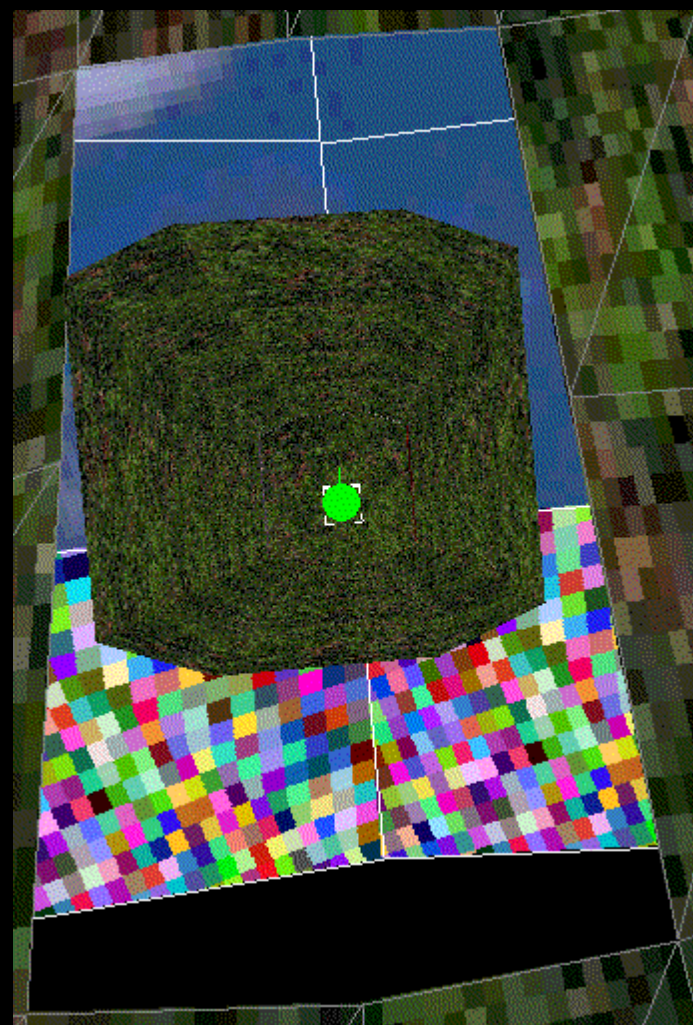
Go to the Current Room view, mark all faces (with the key **M**) and selects in the menu **Edit.../Copy**. Then you choose the menu **File.../New** and creates a new room, **New Room**. After that's done, go back to the menu **Edit** and chooses **paste**. Now we enlarge the room by calling up the Room menu with the button **R**. (you can close the terrain menu)

Now click on the button marked in small red.



Picture

Click on it about 10 times.



After this is done, save the room, e.g.: "temp1.orf", closes the room and opens it again. I don't know why, but that's just how it works. Now we have to work on the room a little more. The room doesn't need to be that long for our project. First we'll make it shorter by half. Mark the vertex on one side and then make it using the keys **4** or **6** shorter. **Before you continue, check whether all faces are there, including the outer ones!!!** (otherwise we'll build a "bad shell" into the level)



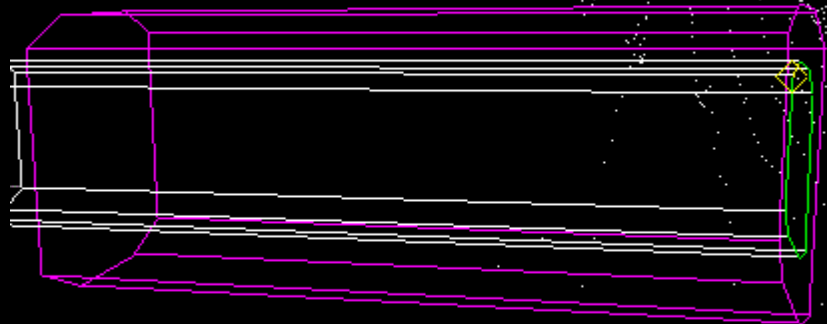
Now we switch to face mode (**Ctrl+F**). Selects all faces (key **M**). In the **Room** There is a function in the menu **Flip Faces**, (**N**) click on it and you will receive confirmation in a dialog box. Now we close the room and move to the second level room (the external one). In this room we previously deleted the outer face, we now need it again. Switch to vertex mode and mark the vertices where the face is missing. You can recognize this by the small red lines that are always in the middle of a face. Then return to face mode (**Ctrl+F**) and the button **Insert** press. Now you have to look in which direction the face is, it has to point outwards.

Saves the room and closes it.

Switch to World View and see if the outer face has a green border. If that's not the case, then it clicks. Now the roomtemp1.orf open, here we have to open the outer face mark (click and **Space** press).

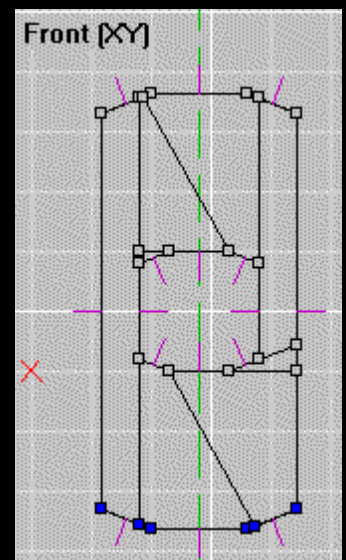
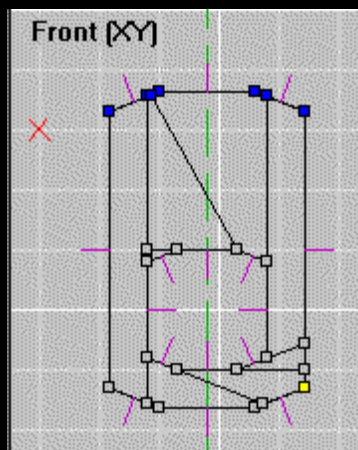
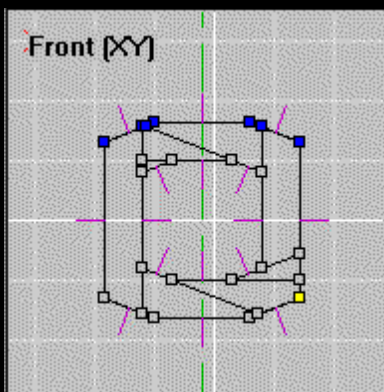
Now back to the world view

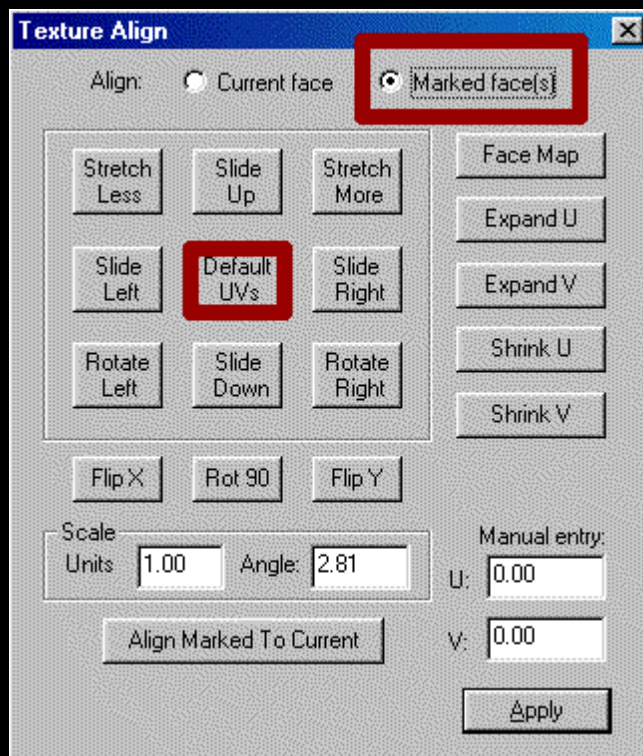
View, into the menu Room.../Place Room at Current Room. Your world view should now look like the picture on the right.



Let's continue, go back to the menu Room and clicks on Attach room, the room is now inserted. Before we do anything different, press the button **M** to mark the room, then we click on any face of the attached room and go back into that Room Menu and select there Combine Rooms. Ok, now let's close the roomtemp1.orf. Now switch with us **Ctrl+Tab** into the Room View view and marks the top vertex. (Picture below left)

We still have to cover up the places that we deleted earlier. In the middle of the picture below, I moved the marked vertex up by 20 units (Grid 10) then using the button **U** demarcated the vertices and did the same on the bottom side (bottom right image). Before we continue, let's delete the back face of the attached and the second room. Because otherwise you can't fly in anymore.



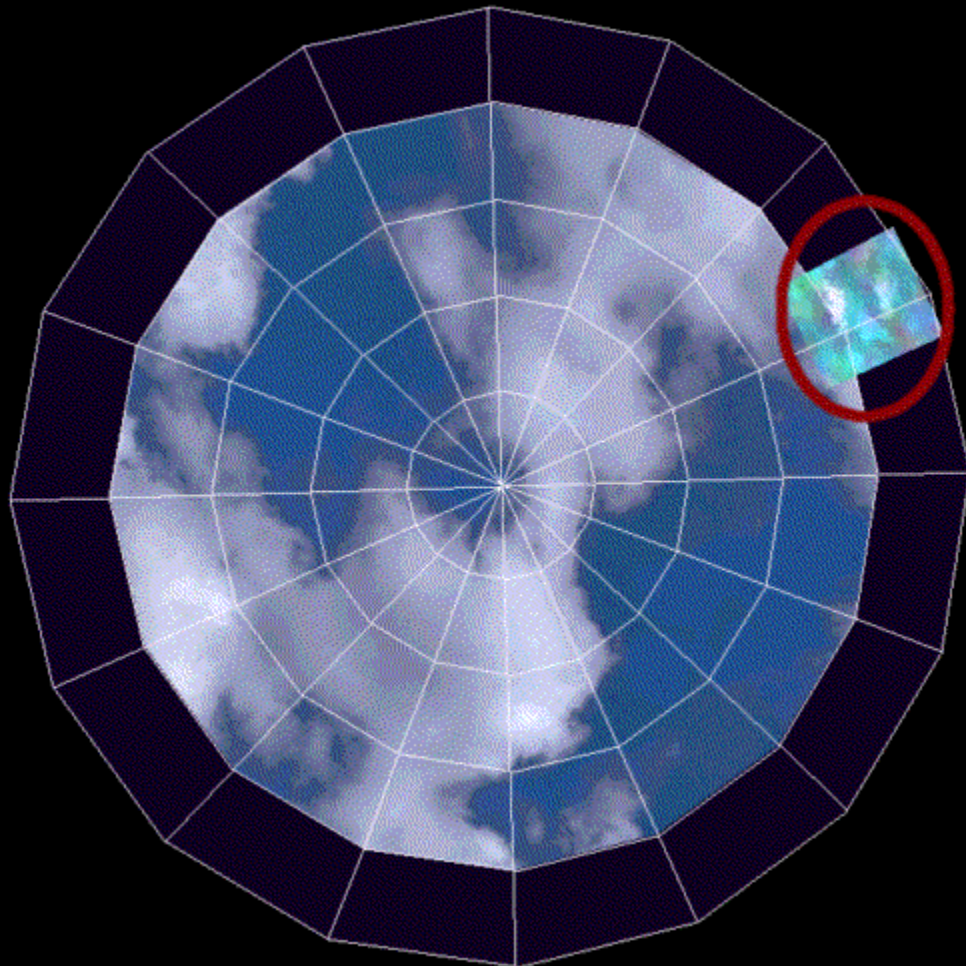


We unmark the vertices after this action and switch with them **Ctrl+F** in face mode, mark with **M** all faces and click in the texture bar **Align....** As you can see in the picture on the left, I have it at the top
Marked Face(s) clicked and then clicked the button **Default UV's** clicked. (because of the texture display)

So now we need a sky and a sun. It's supposed to look good too! Move your position (camera) as far as possible back, so you can see the sky and the position of the sun can recognize. The

This is the position of the sun red-bordered square. In order to place a sun there, we have to look for a sun in the texture bar. I chose the texture Sun1 (not shown in the picture). Now hold down the Shift key and click on the square. I'm going to leave the sky as is, but if you want a different texture in the sky, you have to choose one that **Shift** Hold down the button and press it several times

Click sky.



Last but not least we have to calculate the lights. Switches to the menuWindow and chooses Lighting. Fill everything out like I did in the picture on the right. Then click on the button labeled 1 in the picture, answer the next dialog box with Yes and the light in the mine will be calculated. Now click in the box terrain on **Count Lights** (Determine the number of lights) and click button 2 again and you're done.

Note: The terrain light calculation can take a long time if you are at iterations Leave 999999!!!

Lighting

Lightmap spacing: ☒ Combine lightmap faces
Iterations: ☒ Volume lights
Ignore limit: ☐ Super detail

Current Room
Ambience: R: G: B:
Current Face
Multiplier: ☐ Has Corona ☐ Touches Terrain

Multiplier: ☐ Fill Coronas

Entire Mine
Ambience: R: G: B:
Multiplier: ☐ Fill Coronas
 1

Terrain
 2

Once you've done that, save the level and put it in the MN3 Packer, fly through the level and take a look.

This was pretty extensive, but I think there are too few levels with an outside world and wanted to explain it to you from start to finish.

Have fun building levels. :-)

Back to the overview of section E

058 - Outside World – Update <>

Ragil Ral

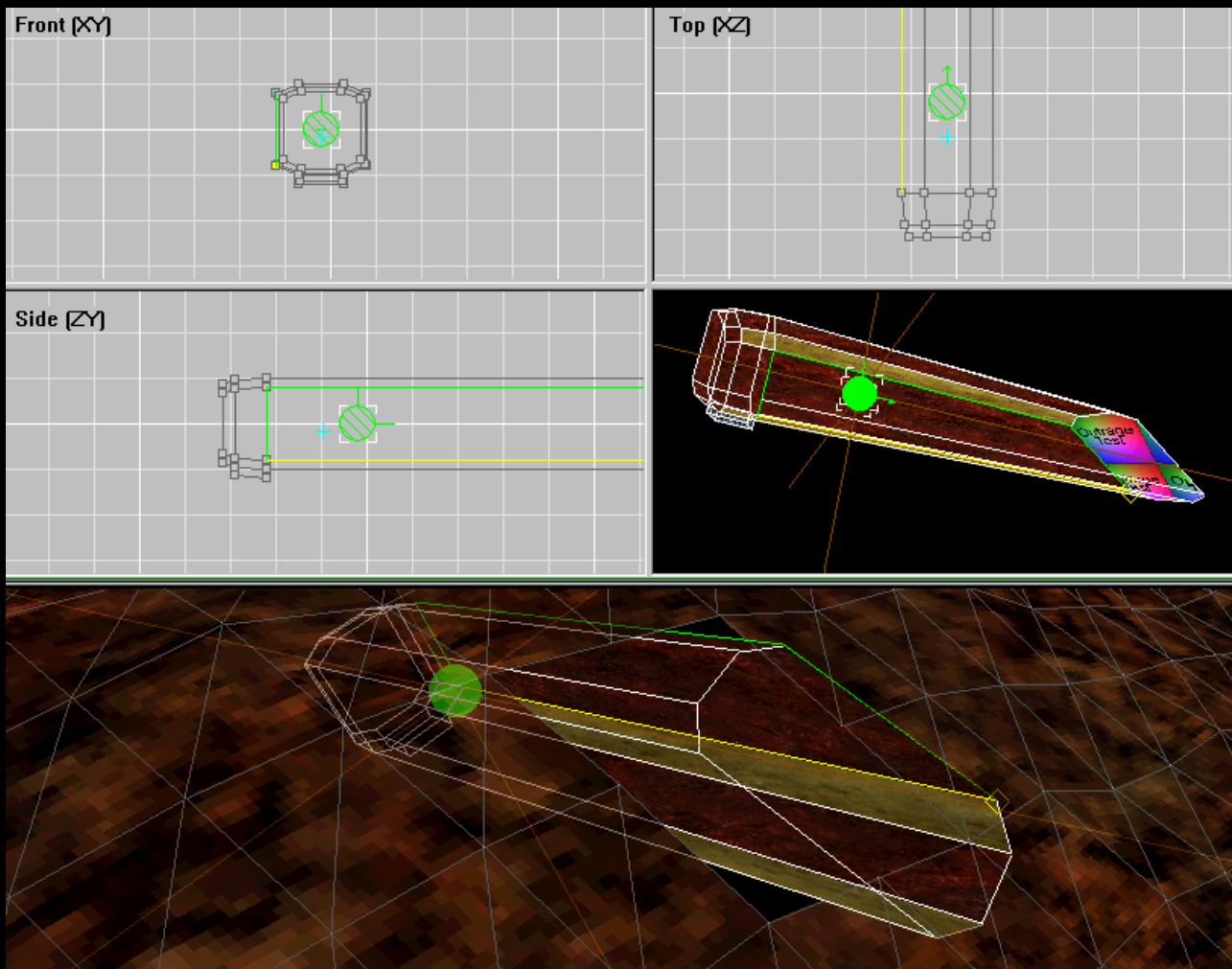
In the current version of D3Edit (1.1 [Beta 10 | Atan P.0_1_39(p), I don't know about earlier ones) there is another way to realize the connection between terrain and mine.

First a few words about the terrain.

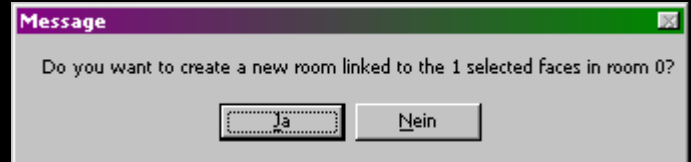
The player can still fly over terrain if the height map is 97% white. Whiter areas are too high. I have a test file for this.d3lgenerated that you in the.zipfinds. On the other hand, if the gray value difference between two neighboring pixels is more than a few percent, this leads to unsightly spikes in the terrain; Of course, you can also use this as a design element. The terrain cells are created in such a way that each terrain cell is created from four pixels, so a single pixel whose gray value deviates greatly from the surroundings becomes visible as a spike on the terrain. You should also go to the editor settingsUse Terrain LODcheck off so that you can see all terrain cells, otherwise cells that are on the same level will be grouped together, but will be separated again when you switch visible/invisible, which makes marking very difficult. Also, stay away from the terrain boundaries. If you get too close to those with rooms or objects, strange, strange things happen...

Terrain connection new

In the previous tutorial you attached a room, marked it as external, deleted faces, etc. etc.; The new editor now offers a more convenient way to do this. So build a terrain and create a space so that it looks through the terrain. Don't forget, the side where you're supposed to fly out MUST have the texture 'Rainbow Texture' (the first one under 'Terrain'). The screenshot shows a default room, which I moved to an appropriate location and lengthened have drawn. I then cropped it using the 'Cut Room' function (&) and turned on the corresponding terrain cells.

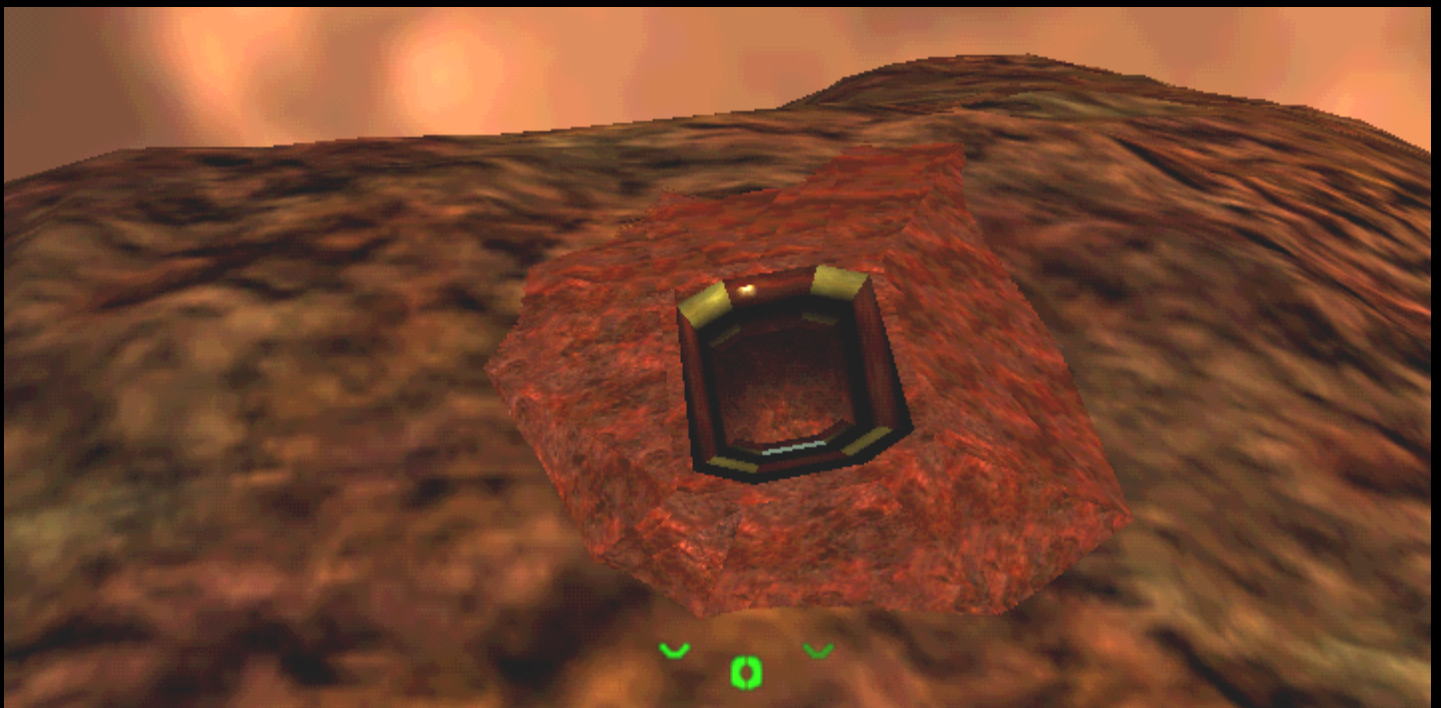


Now do the face current that you should fly out of. Then, in the world view, rises Room->Link Room to new External Room, then you will get a dialog box that you can see here **Yes** confirmed. In the previous tutorial you had to confirm with No because you had already created an external room.



D3Edit now attaches a space that consists of only one face and is marked as 'external'. To cover the gaps in the terrain, proceed as follows: Create a suitable outside space, for example using Fischlein's method from before. The face through which the cover is to be attached to the new external room must have the normal to the outside.

Then go with me **R**to the external space, which currently only consists of a single face, the normal of which faces outwards. Since it's a portal, you can't flip it, so add a second face whose normal must then point inwards. The cover is attached to this. Now joint the rooms are completely normal as usual Room->Place Room at Current Room and Room->Attach Room. Tweak the cover so that the gaps are completely covered, voila, done. Texturing, aligning the textures, calculating lights... etc. etc.



Back to the overview of section E

059 - Building

Ragil Ral

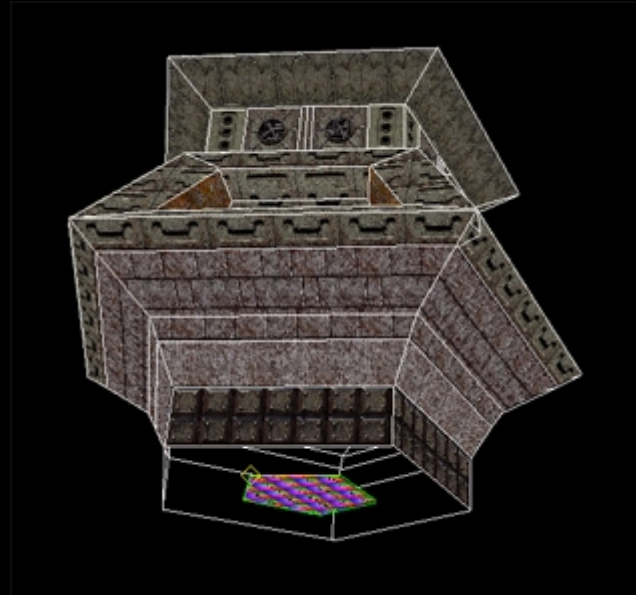
Sometimes you need a focal point on the terrain, an antenna, a ventilation tower or something else to break up the outside world. Now that you know how to connect the inside with the outside, this will be a walk in the park.

First you need your building that you want to place on the surface. I have a ventilation tower here.

Note the face at the bottom, it has the rainbow texture and is directed outwards. The fact that the 'floor' of the structure has a hole isn't a big deal; it's not a problem for rooms on the ground. (rainbow texture mandatory for external?)

Then go and mark a terrain cell in the area where you want the structure (**Alt-Lclick**).

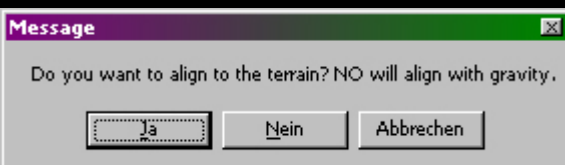
If applicable switches in **Tr** the **LoD** out of.



Make sure that you have opened the room to be placed and that you have selected the face that should go to the 'building site' and goes into the **wireframe** Mode. Then says

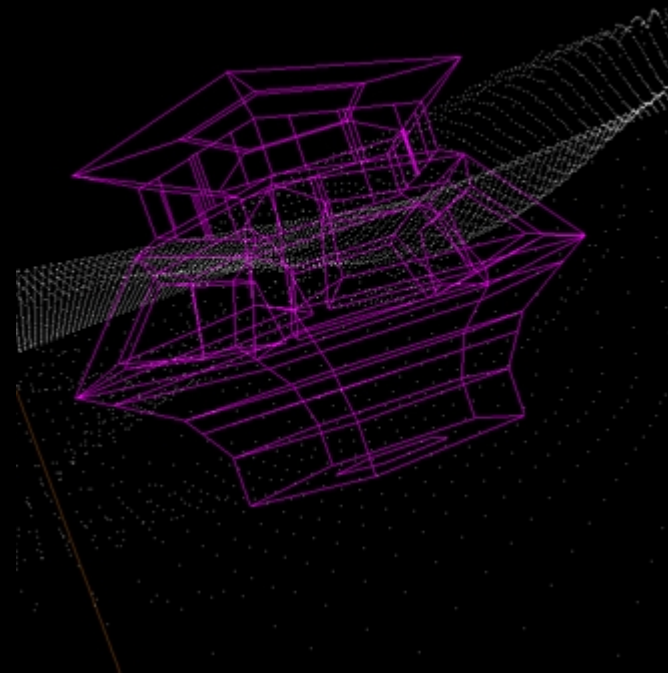
in the editorRoom->Place Room at Current Terrain Cell.
You can take the room with you **2/8th, 4/6** If necessary, position it and use it **1/3** twist.

After that says Attach room, and D3Edit wants to know from you how the booth should stand:



At **No** the building is aligned according to gravity ('down'). **Yes** it will be aligned to the plane of the selected terrain cell.

Then the thing is on the ground:





As you can see
he can throw it
No tower
Shadows, though
the terrain
Radiosity calculated
became. it takes time
sometimes several
Calculations up to
the shadow
becomes visible.

Here you can see pipelines under which another external room has been placed. It can be clearly seen that the shadows on the terrain are more diffuse than on the external space. If you clearly visible have shadows if you want, you have to the terrain with a room cover.



[Back to the overview of section E](#)

060 - Player start in the Outside World

Ragil Ral

There are several ways to move the player start to the outside world, but unfortunately there is no silver bullet.

The starting point is this: you can't put the Player0 directly in the outside world, you can't move it there either. It is also not possible to move the player to an outside room, i.e. a room with the 'External' flag - the graphics engine then completely freaks out.

The following methods are known and tested:

On the one hand, link an internal room to the terrain

'Invisible space': Interior space that is completely linked to the terrain so that it is no longer visible.

'Garage': Interior, mostly linked to the terrain.

These two methods have the disadvantage that light and sound are calculated by the 'inside' engine; So you have to adjust here so that little or no difference can be seen. For example, if the starting room has no light, the pyro there will be completely black; Not a problem in pure single-player missions, but it doesn't look so good in COOP missions or if you have an intro.

'Method Thomas': Extract a level that has Player0 outside; This mostly only occurs in the original missions and therefore has a legal component. 'Method Boogie': simply beam the Player0 outside.

The practice of the methods is now described here.

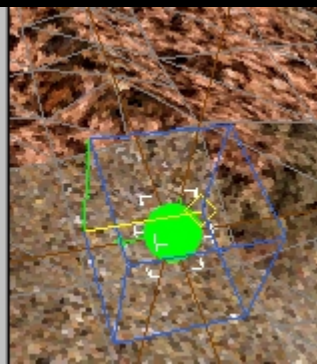
Internal space left to the terrain

It's the easiest workaround.

A) Invisible space

Either just start with oneNew->New Level-Default Room and import your terrain.pcx, or if you already have a level, set up a room the height of the terrain. Take a simple shape, cube or tetrahedron; You put the Player0 there. Give the space on each face the rainbow texture, and select all faces (right). Dark gave us the tip to name the room to avoid sound problems. Now the trick: in the world view Room->Link Room to New External Room, The editor then asks you whether you want to link the marked faces to the outside world, which you confirm with yes.

The room now becomes invisible as all of its faces are transformed into a portal to the outside world. Then go to the Room View, open the lighting dialog and



checks the box at 'touches terrain'.

Ingame it looks like this (room with **outlinemdisplayed**):

As you can see here, the lighting conditions are not right.



Player in the starting area



Player outside the starting area

Now it's time to experiment to adjust the light. Another thing is, if you have weather on the terrain, this doesn't happen in the invisible starting room; That also seems a little strange; However, you could make the room much smaller so that the lack of weather is not noticeable.

If you look in the editor, you will see that there are now two rooms; one is the cube we started with and the link room.

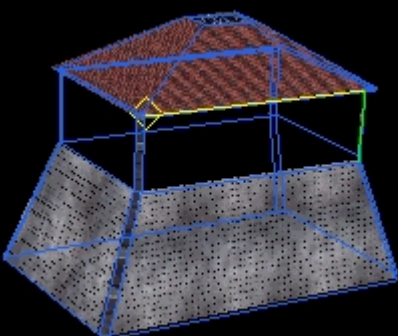
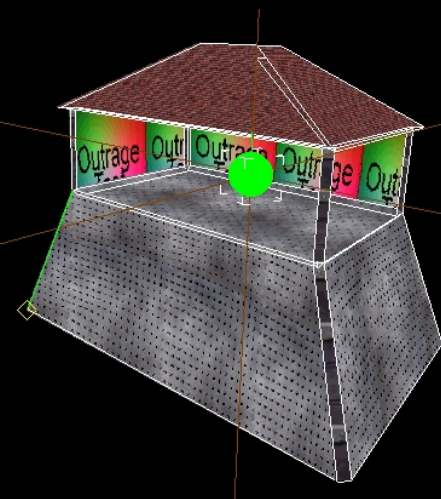
B) garage

If you put in more work, this is a good workaround. You simply build a shed or something like that, link it to the outside world and cover it with an external room.

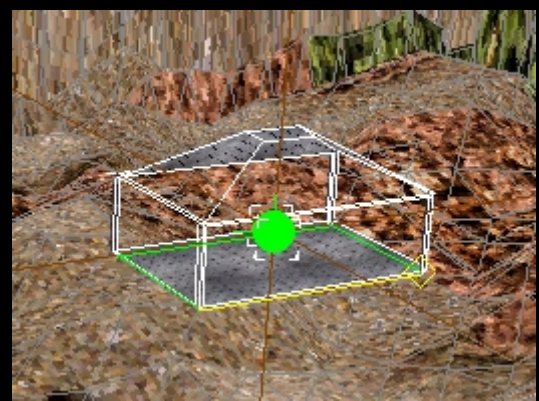
Build the entire building first, both inside and outside (left). Give the faces that should be linked to the 'outside' the rainbow texture again and link them to the terrain as before. Don't forget to put players in the room.

Now do the following: mark all the faces of the room that should be outside. Cut this out first **Ctrl-X** out of. then go with me **R** into the link space and add the cut faces **Ctrl-Shift-V** back in place.

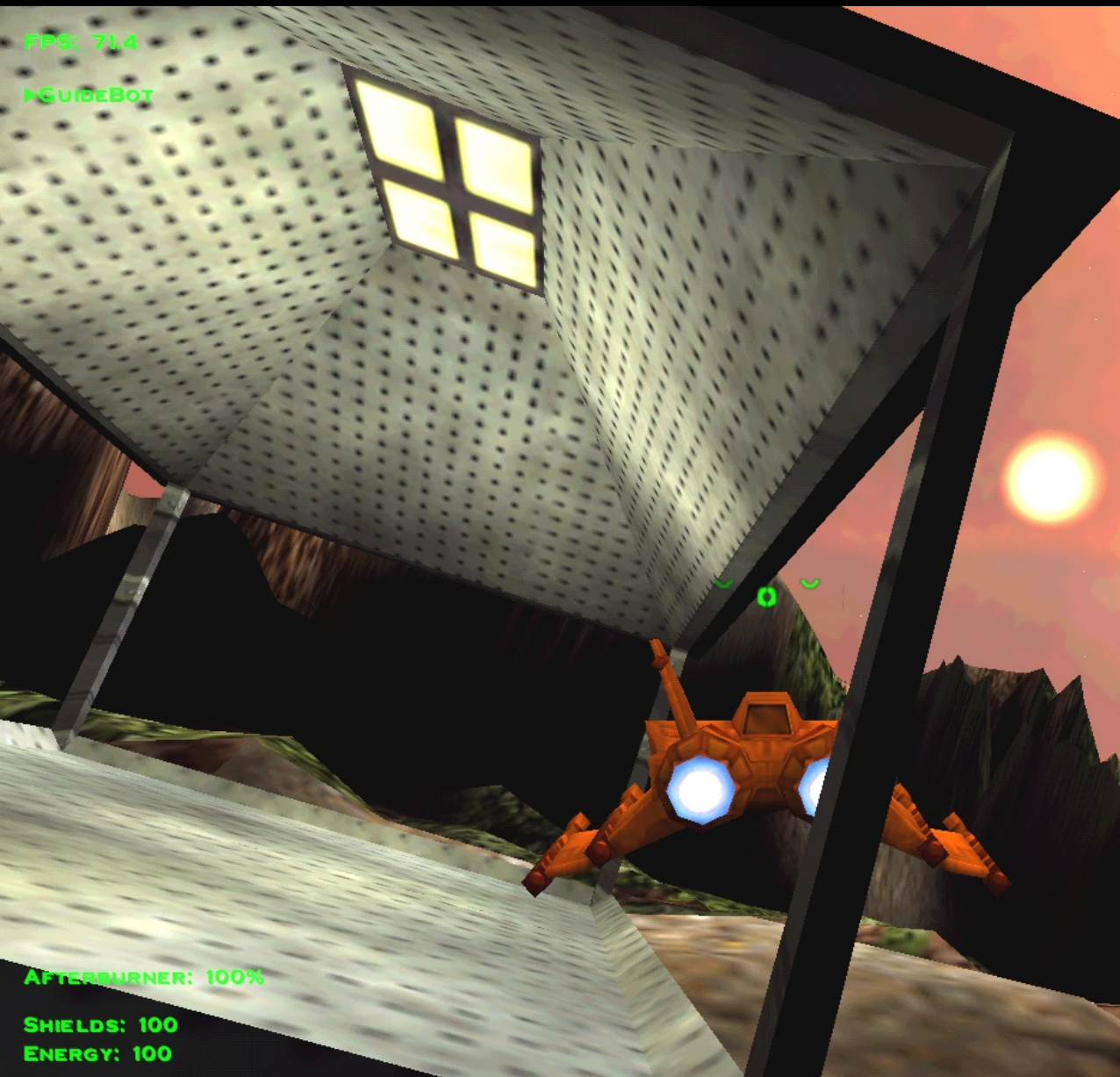
From there it continues as before.



Now you should have something like this: the outside space on the left, the remaining interior space on the right.



What remains are the different lighting conditions, but the whole thing looks a bit more 'real':



And that it
under a
No roof
snowing or
It's raining too
more like a-
bright.

Furthermore you have
here freedom
the thing
'airier' or
more
closed to
build.

Thomas method

Personally, I like this best because the Player0 is really on the ground here, but as mentioned at the beginning there may be legal problems here.

The way it works is that you take an original level, delete everything (rooms, objects, etc.) so that only terrain, an interior room and player 0 are included.



In this example I use level15.d3l from the Retribution missions, but basically it doesn't matter. You can extract the original missions from the d3-hogs with Quickedit.

Clearing the level

First remove all rooms. The easiest way to do this is to select them in the World View and cut them out; This saves you the query that comes up every time you simply delete the rooms. So click on any room and go Edit->Select all Rooms attached to Current Room. Then go to room 0 and deselect it (**Space**); an internal room must remain in the level. Then delete all portals in room 0. Then mark a face in a room adjacent to room #0 **M**. Then you just say Edit->Cut selected rooms or **Ctrl-X**. The advantage of this method is that the objects contained in the rooms are also included

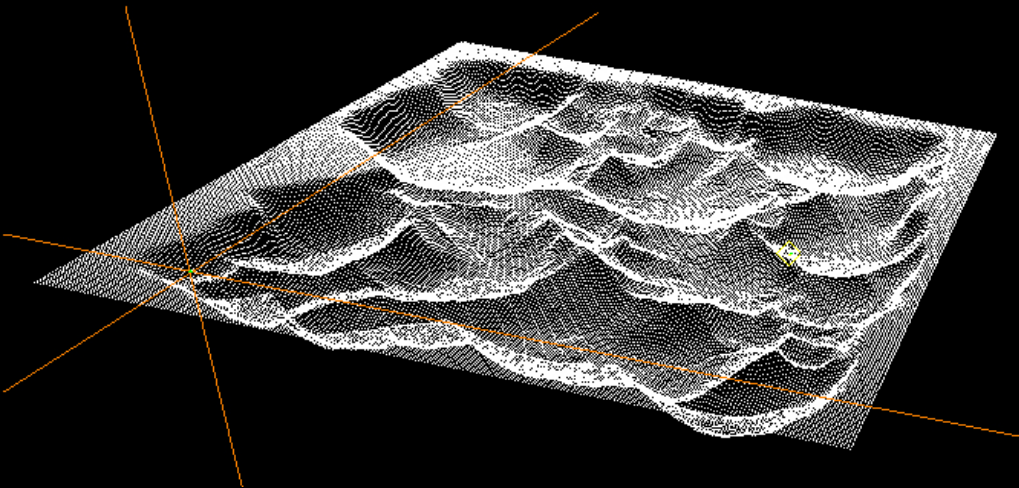
can be removed and there are no problems with door spaces. You then throw any remaining rooms with you **Del** out.

This is just one way, just try it out a bit.

Objects now remain on the terrain; goes into the and ck on the objects there, but you have to delete them using the Delete button in the objects panel, all individually... Leave only the Player0; in the c you see below Room NoEnter the cell above which Player0 is located. You should now save the level under a new name.

Now go ahead Window->Goals and throws them all out too.

Then import your own terrain.pcx.



Now you have a completely cleared level which you can build your own things into.

You can then use this as a basis for future levels; so you only have to do this procedure once.

Boogie method,

he solved it with teleportation; First you start as normal in an indoor room. It contains an object that beams the player via DALLAS to a corresponding object on the terrain. In the starting area, wind causes the player to be pushed onto the object, whereupon he immediately reaches the terrain. You will learn how to teleport the player - this requires scripting - under Error: Reference not found - Error: Reference not found. Wind, in turn, is a property of space and can be used without scripting Window->Room Properties be set.

A certain disadvantage of the method is that scripts are not always compatible with COOP multiplayer.

Back to the overview of section E

Section F-Textures

Everything you see or can see in the game is 'covered' with textures. However, sometimes you don't find exactly what you need or want in Descent 3's standard textures. Using custom textures isn't difficult, but read section #117 and 'Having custom objects visible in D3Edit' in the appendix.

061	Own textures	Integrate your own textures with the Gamtool and the OGFTool	[DCG]Roadrunner	236
062	Custom textures Apply	The whole thing is a little deeper.	Schplurg	238
063	Custom textures: Tips	Tips for Texture sourcing & creation	Schplurg	242
064	Textures tileable make	How do I get it to the tile?	Ragil Ral	245
065	Partial transparency Textures	Their creation and integration	Digijo	248
066	Animated textures - Screenshots	in-game(!) screenshots of the Animated textures	(LL)Dark	254
067	Animated textures - Homemade	Explains how to use Animate moving textures created	Ragil Ral	261
068	Textures @ 256x256	Large textures and their Application	Ragil Ral	262

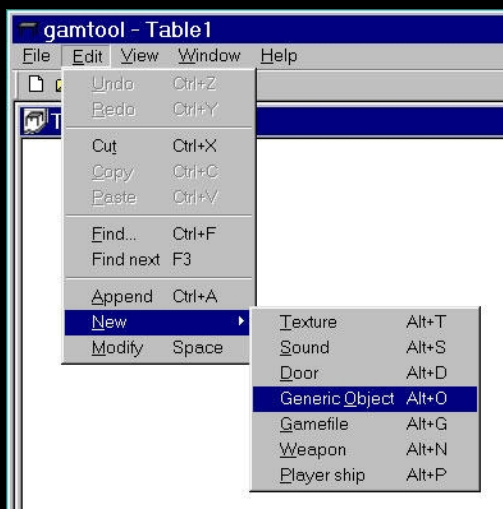
[Toc](#)

061 - Custom textures

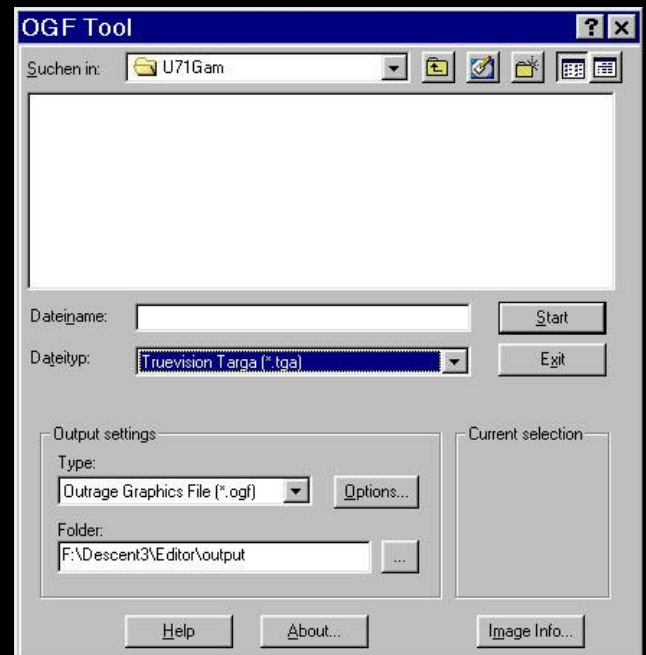
[DCG]Roadrunner

Proceed as follows: Create a texture with your graphics program (Photoshop, etc.). It should be 128*128 pixels or 64*64 pixels with a resolution of 72 pixels! Save the whole thing as a TGA texture! Now open the OGFTool and convert the TGA image into an OGF texture.

Now open the GamTool and choose new texture!



Open the texture to edit (picture on the left).



Here you can now make some settings for the texture (picture on the right):

First of all, include the name **Path specification** (maximum 34 characters are allowed!!!) a . You only need to specify the path if your own textures are not stored in the D3 path!!! You can now add a glow to the texture. Play with the colors to get the right glow for your texture. For example: A value around 10 in all three columns results in a bright white light. If all values are 0, no light can be seen. OK now select the flag. This means that you have to look for the texture under this group, for example. Save the whole thing and you're done. The Gamtable file must have the same name (also

Upper/lower case) have level MN3 like yours.

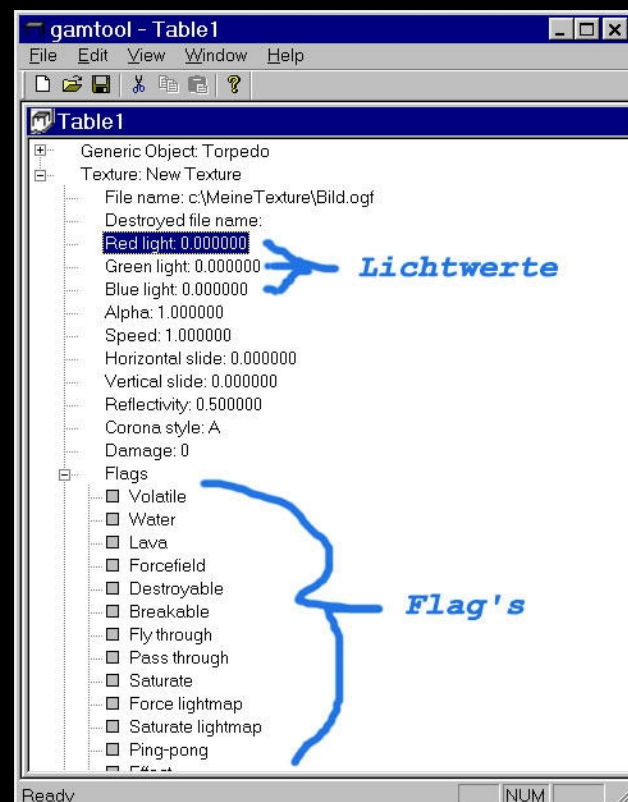
Now start the level editor (do not open a file!) and click on in the menuData. Choose that oneTable file manager. First is the base table file (do not change). In the

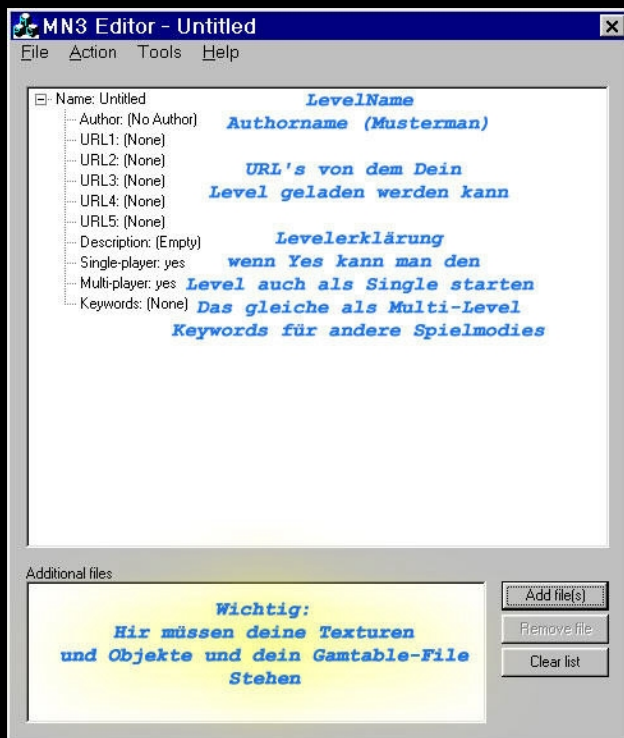
second line you can

Enter gamtable file. **File Browse** etc... Now your textures should be visible in the editor and you can assign them

Tip: Create a folder in which you can put the textures and the Put in the GamTable file.

Now start the MN3 editor.





Proceed as usual and fill in all the necessary items! Now comes the most important thing: Select the 'Add Files' button below. All new textures and the Gamtable file must be located here
 The name of the Gamtable file must be the same as the level name of the MN3!!!
 Now save and that's it.

[Rip]DwnUndr showed me another way to bring your own textures to your own level. Only you cannot add lighting effects to the texture. It basically overwrites existing textures from the Hog file. You all know his levels, "Worldsapart, Canyon Siege, Canyon Aftermath". He opens the HOG file and chooses a texture that he doesn't use in his level. Then it looks for the size of the original texture (pixel E.g. 128*128) and creates its own, in the same size (must have the same name). Now insert your texture into the MN3 editor in the same way as the Gamtable (without a Gamtable file). You can unpack the HOG file with the HOG Viewer to determine the size.

Have fun !
 [DCG]Roadrunner

Back to the overview of section F

062 - Apply custom textures <>

Schplurg

Prerequisites:

- Besides the game and D3Edit (ver 1 or higher)
- OGF Tool and MN3 Edit come with the editor
- GamTool or something similar to create .gam files
- A good painting program to create your textures (Photoshop, Paint Shop Pro...)

Let's do it

The texture we will use is 128x128 pixels with a color depth of 24 bits. You can make your own, or use mine (.zip). If you're making it yourself, make sure it's a 128x128.tgafile is. A.tga we need for the OGFTool to turn it into a usable texture (.oif). On the right is a picture of my texture. I think since D3 really needs a better wood texture, this one would come in handy for you.



Ok, we have a 128x128 24 bit.tga-Picture. Now start the OGF tool either in D3Edit Tools->OGF Tool(You may have to configure it first) or via the MN3 packager Tools->OGF Converter.



In the picture on the left you can see that I have the texture woodply128.tga have selected. I like to give my images detailed names... and yes, I have some Three Stooges textures 😊

Make sure the Output Settings is set to 'Outrage Graphics File' (.oif) are set. By changing the file type selection you can.tga after .oif or convert vice versa..oga- Textures (animated) are also supported. If necessary, set the output folder.

When everything is set the way you want, press the button **begin**. Even though you don't receive any corresponding message You just one.oif-Texture generated.

Navigate to your output folder and find woodply128.oif. Click on it once and it will appear in the small preview window. If you see it, you've done it right. Now we just have to get the part into the level! 😊

In order to use the texture in a level and see it in D3Edit, we need to create a few more files.

The GAM file

The .gam file tells Descent3 to use its own texture. You can also add your own sounds and objects to the level. They must all be listed in the .gam or D3 will not use them.

Open GAMTool. If a new window isn't already open, check it out File->New. under the menu Edit choose New Texture (note the other choices while you're there!) Then double-click the new entry so you can type and replace 'New Texture' with woodply128 or something meaningful. Press **Enter** to complete the change.

Then click the small '+' sign and expand the listing (picture on the right). There where File names says write woodply128.ogf for something descriptive. Make sure you write it exactly like the file name and append the extension... this is where the texture is called by the game. The change again with **Enter** complete.

Now look for the +Flags and expand the listing. Make sure the Mine box is checked and all the others are unchecked. This tells D3Edit that the texture will appear in the texture library under the Mine section. This is the default selection.

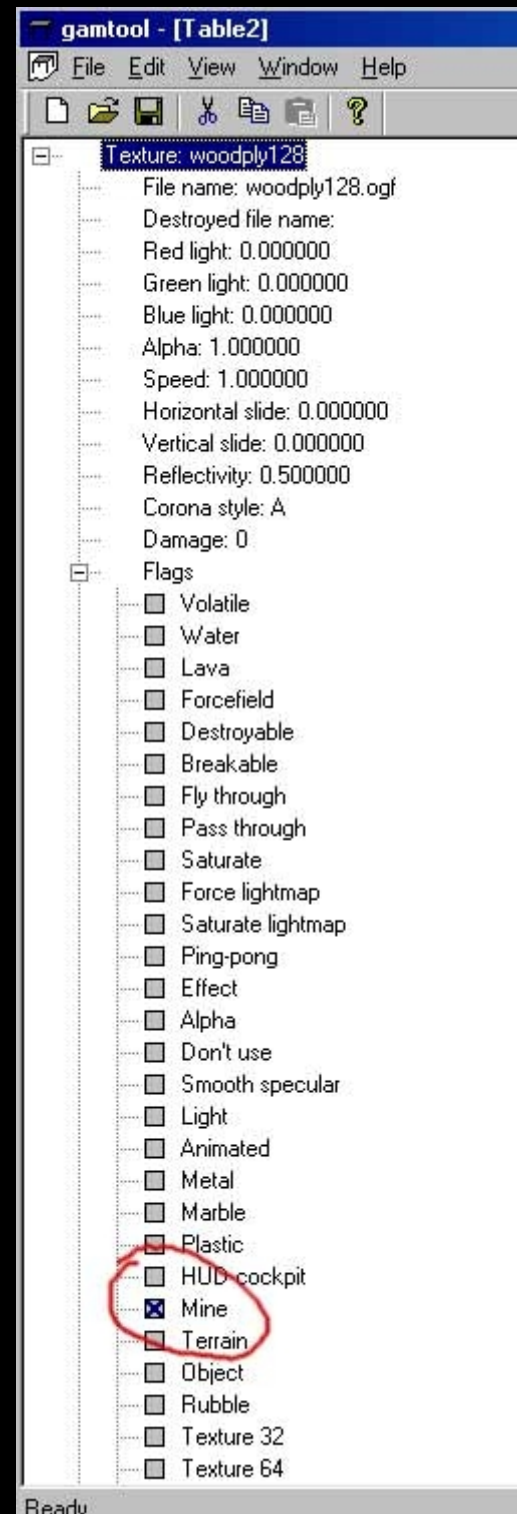
You can see in this picture that you have many options like fly through and pass through. This lets players fly and shoot through the texture. Other settings are not so easy and require different settings to work.

Save your .gam-File under texturetest.gam to your hard drive. The .gam MUST have exactly the same file name as yours .mn3! We don't have one yet, but will soon. Next Step...

The MN3 file

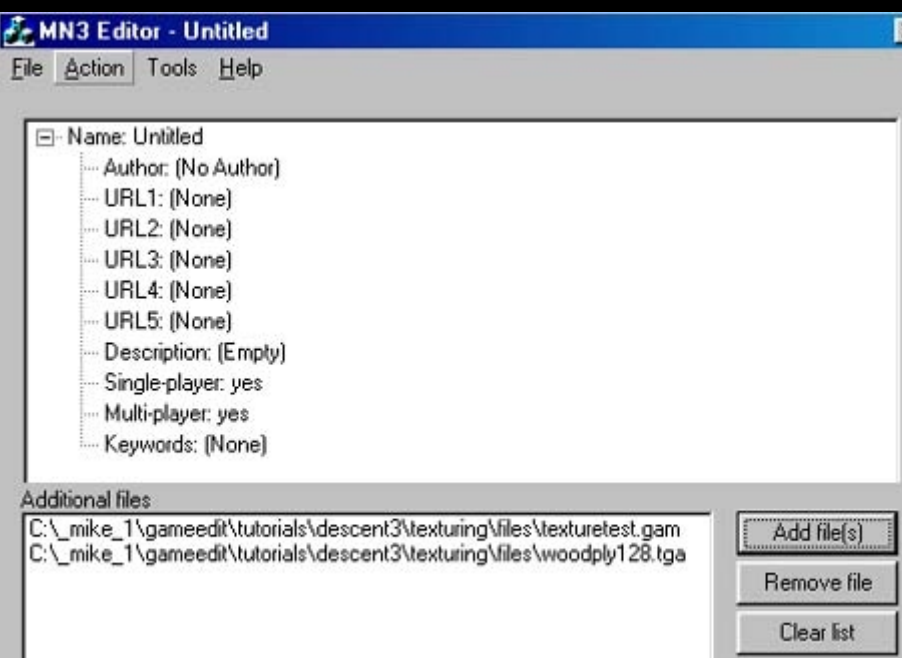
It's usually the file you use as a finished, playable level. But the MN3 format is also used by D3Edit, so that your own textures are visible in the editor.

Open the MN3 Editor.



Use the Add File(s) button and find your texturetest.gam and the woodply128.ogf.

Otherwise you don't need to do anything else here. Save the .mn3 as texturetest.mn3 (same name as the .gam file.) Now it's time to use our texture!



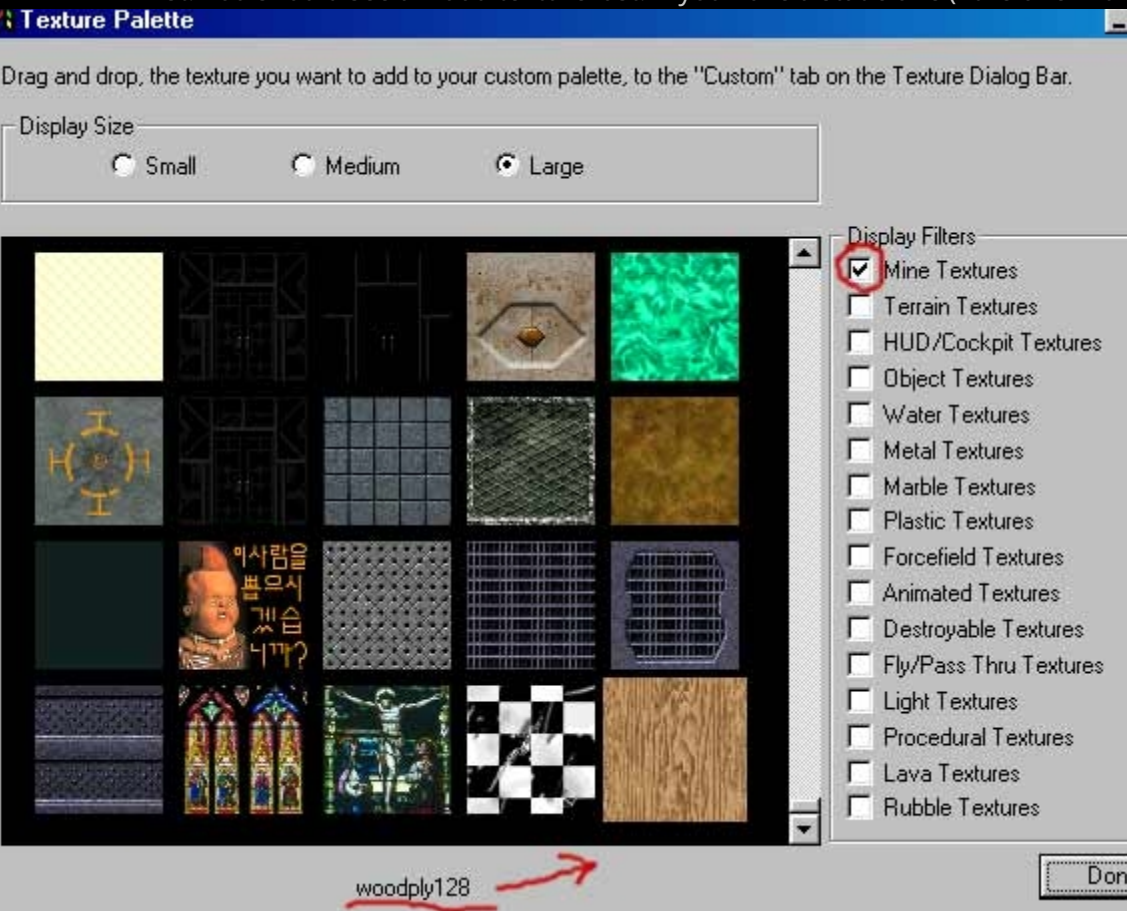
Use the texture

Open D3Edit. Don't do anything else yet! To see our textures in the editor, we need to tell D3Edit where they are. That's why we have them.gamand the.mn3generated.

Go to the menuDataand chooseTable file manager(our.gamis a table file). There are three rows of buttons. Click the File Browse button in the "Mission table file (optional)". Find yours.gamand confirm with**OK**.

Then go to Data again and select the second selection, Configure a Mission Hog. Click the Change button and locate your .mn3 file. Confirm with OK. A dialog pops up, just click OK to continue.

Now open D3Edit's Texture panel, select Mine Textures and scroll all the way to the bottom of the list. You should see a wood texture last. If you have a static one (=the one with the



color noise)

See texture, have it

You were wrong

made. If you

neither a static one
nor the wood texture

Well, sometimes it is
the new texture
in the next row
further down, though
You until
End of list scrolled
have. Choose one
other display sizes
and she will
turn up. That is
a 'scrolling' bug
n some
Versions of
D3Edit.

You can now use your new
texture like any other
use others!

One more step!

If you have your.d3If you're building and want to try out the textures in the game, you have to.gam-File
and the.oif-Textures in the final.mn3load. This is done with the button**Add Files**made in the MN3 editor.
Then add yours.d3Add it normally and save your MN3 with the same name as the game file!

Do that.mn3in yourD3\missions-Folder, play your level
and check out your new texture Here 🤖 a screenie
of the texture in my level 'Mad Science'.



Important:

- You have to `.gam` and `.mn3`-Load files into D3Edit EVERY TIME you edit your level. If not, and you save the level, all textures become static and must be applied again.
- The `.gam`-File MUST have the same name as yours `.mn3` have!
- Textures can be 32x32, 64x64, 128x128 and 256x256 in size and have up to 32 bit colors.
- Using lots of large 256x256 textures can slow down graphics cards. When deciding the texture size, it is important to consider the size of the area where the texture will go.

Have fun texturing and use your imagination. You may never use standard D3 textures again!

[Back to the overview of section F](#)

063 - Custom Textures: Tips

Schplurg

I shortened a little here. Original version at: http://gameedit.net/custom/GameEdit_OG/tutorials/descent3/texturing/texturing_tips.htm

introduction

I learned a lot about texturing while building 'Mad Science'. Although I'm not a texture master, I think I have enough valuable information to fill a website or two. For me, making textures is perhaps the most enjoyable part of level building. It's an important aspect because everything you see is textured. The inside of a barn can be made to look like a church if the texturing is done well.

To make textures you need a good painting program (Photoshop, Paint Shop Pro...). You should have a decent understanding of your painting program, especially how to use layers, cloning tools, filters and effects, and your imagination.

Where are these textures?

Where do we get them from? Assuming you have an idea of what you're looking for, there are many methods to get what you need. Sometimes you can design the entire texture in Photoshop. However, some textures are a little too difficult for the average person to simply draw with a mouse. I am one of these. For those with less than spectacular artistic talent, alternative methods exist.

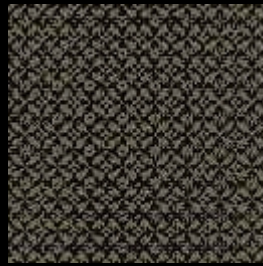
We need some basic material to get started. As a texture artist, you will discover that the Internet is your best friend! You can find any type of image you need with a little patience and a search engine. In my level 'Mad Science', which takes place in a house, I wanted some funky wallpaper for the living room. I had a good idea of what I wanted, but didn't think I could make it look real by drawing. I thought about taking photos of wallpaper with a digital camera (another valid option), but I was too impatient to borrow a camera, let alone find the right design in anyone's house!

So I went to Google's search engine and typed in 'wallpaper images catalog' or something similar. And then came the wallpaper dealer websites. It took a little time to find a site that had a good online catalog of their designs, but I finally found a good one. In fact, on one site, I found thousands of cute .jpg's of wallpapers from all over the world, in every color and pattern I can imagine.

Here are a few that I downloaded... I downloaded dozens so I have a good selection and for future use. If you find a good source, download what you can get!



Finally I used the texture on the left for my level. I optimized it a little. I mirrored them, then made them tileable, which wasn't too hard with this particular pattern. Later I decided that I wanted a really unusual rug for this room. Off to the Internet!



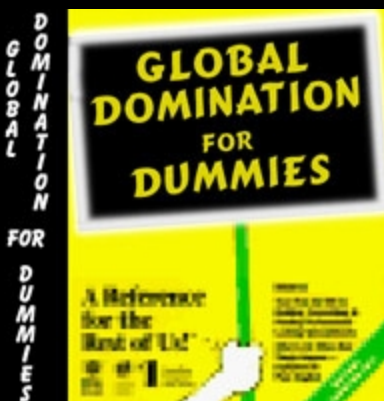
I think I used number two or something similar. I don't think it needed any extra work to be tileable. Here and now, believe me later, there is a website for EVERYTHING! Look closely and you'll find a site that has exactly what you need, including game editing tutorials :\

These aren't textures you'll find included in Descent3, at least that's what you can say! There are a few texture websites that offer free textures ready to use, but I prefer to make my own stuff, plus a lot of these textures suck or are spat out by programs that generate kaleidoscopic random textures. STILL, I'm not above using an existing texture from a library as the basis for a new one, or just using it unedited. I found some nice wood textures somewhere that needed to be made tileable (using Photoshop's clone tool) but other than that I didn't make any changes. By the way, if you're using someone else's work, you should use textures that are offered for free or give the original author credit if they allow you to use them. Of course you can still use them, it's up to you and your conscience. If you plan to sell your work, don't rip anyone off, it can get expensive.

Many of the images I use as base images are simply photos from the internet that I later edit as needed. The previous examples are samples from an online catalog. I doubt they'll mind if I use them (or so I say). Anyway, to hell with legal stuff! We're on a mission (and I don't make a dime anyway, so sue me)!

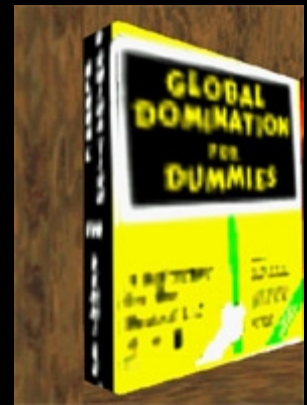
I designed some textures from photos that I took with a digital camera. I'll get to that in a future tutorial, but it was fun driving around on rural roads taking photos of old barns and cabins and seeing the looks on people's faces when they saw me taking close-up photos of their stuff (mostly the security ones). in my truck which I'm sure looked even more suspicious!) What would happen if they were playing D3 and saw their old barn in my level? Photos are also easier to get from the web.

I took a small image from a Programming for Dummies book and modified it slightly for the shelf in 'Mad Scientist':



Our mad scientist has big goals indeed! The actual texture I used in the level is only 64x64, so detail wasn't my main concern. I wanted the title to be legible and the book to be recognizable and close to the original. The book photo that I downloaded had mostly blurry text as you can see in the image to the left. I made a new 'slate' for the cover and added my own text. The front was close enough to the original and still legible.

One thing to note is that I used this single texture for both lid and spine. This creates less work for your graphics card because it doesn't have to load two separate textures to render the book. The picture on the right was taken in-game.



In another room is a Donkey Kong arcade machine. I found photos of the original arcade game's respective stickers on the web, and chopped and glued them together into just two separate textures. This includes the side of the case, the top title panel, the panel where the controls are, the coin slot area and the plain blue areas that cover the rest of the machine. Instead of using a bunch of small textures, I used a few larger ones, creating a more defined appearance.

Here on the left is one of these textures. I drew the speaker and the "25" thing, the rest is vintage Kong!

This includes the front of the case, the control panel and the speaker as well as the coin slot.

On the right is the finished product (except the joystick). This machine is many times larger than the ship you fly in the game. The texture had to be stretched even though I used the largest possible size. So I used as few as possible. Not bad, right?



[Back to the overview of section F](#)

064 - Make textures tileable

Ragil Ral

It doesn't really belong here, but anyone who messes around a lot with textures can't help but wonder how to quickly get a graphic or part of it into a texture suitable for Descent. Often you don't need it *THE* texture, but just something that looks like rock, sand or metal or something like that, which you can span over a larger number of faces or terrain cells in order to texture a larger area without it looking too monotonous.

Formats and integration have just been covered, Schplurg covered the extraction and creation of textures; But one problem is the tiling: how do you get that baked quickly?

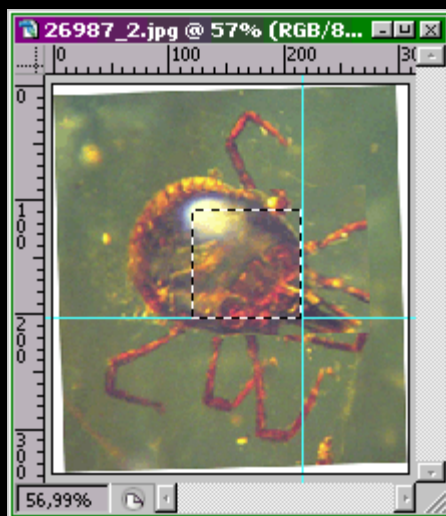
Here's a quick, not to say cheap, method for doing this, but unfortunately it's not suitable for everything.

1. Select source graphic

You've probably seen pictures or graphics somewhere online where you thought 'that would be cool as a texture'. Here I found a picture of a prehistoric tick in amber, whose metallic shimmer and Reflection point made me think that immediately. Wouldn't it be appropriate for a bot in D3?



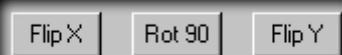
2. Format



Textures are square in nature in D3. This means that you either have a texture that is tileable on all sides, or a quarter of it, which then allows larger areas to be textured, but is only tileable when aligned (kind of like the bullseye textures under Mine). So find the part you want as a texture (unless you care about the entire image anyway) and put it in a square format. Now you have to decide: full tile or quarter? (You can of course also make larger texture groups) Then make sure that the resolution and size (in pixels) of the graphic (or the part you are concerned with) is not less than the texture size, otherwise you will have to enlarge it - whatever can create very unsightly artifacts in the graphics according to the editing program. At.jpg's as a raw material, pay attention to the quality; jpeg compression itself causes such artifacts if it is used incorrectly. So open the graphic with a graphic editing program of your choice. Maybe still

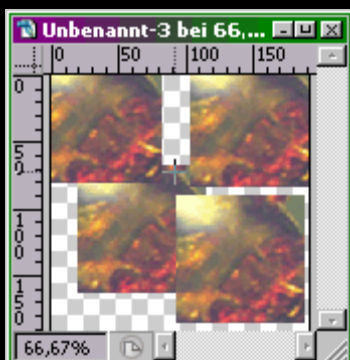
Pre-edit so that it corresponds to your ideas. Then copy the desired part from the image; It's good if you can set the selection to be square, like in Photoshop. But it usually works if you do it while marking **Shift** presses.

Quarter tile: in this case, simply paste it into a new file whose pixel dimensions correspond to the copied part. When texturing in D3Edit you will then be in the buttons **Flip X**, **Red 90** and **Flip Y** Find new friends quickly with Align



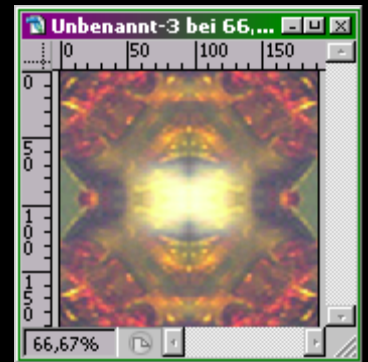
tile:

Here too, you paste the copied part into a new file, but make sure that the work area is exactly twice as large as the pasted part. To make your work more comfortable: not all graphics programs allow you to manipulate graphics outside of the work area - in this case, just use about four times the size so you have space to 'maneuver'.



You can now do the part that you do in D3Edit with the buttons under Align here: paste the copied part a total of four times. Mirror or rotate the respective pieces so that they fit together and put them together seamlessly. It doesn't work with rotation alone. You're best off with mirrors!

Now I have my texture. I will use these as Light and Forcefield.



In case you decide later that you want something different, In any case, save the work files, because deleting them afterwards is easier than complaining about deleting it too early and then having to do it again 😊

If your graphics program works with layers, now reduce all layers to one. Then scale the image down to a size suitable for D3 (I really strongly advise against scaling anything up, it usually looks really terrible), with values suitable for D3.tga Save with the OGF tool.oifconvert into the.gamininsert, possibly set flags, feddich.

Always look at the texture in-game! Something that looks good outside doesn't necessarily have to look good in-game. You may have to play around with the color values, alpha channel or something else until you get it right, not to mention the light values.

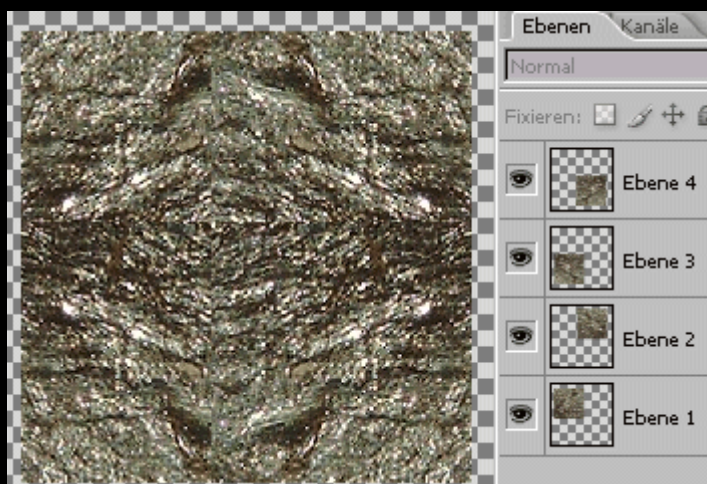
Theory: By inserting the quarters and aligning/positioning them correctly, it is very easy for the edges to fit together because of the mirror symmetry - which is a property of this (current) universe - so the thing becomes tileable. This works with all graphics, since the texture is a copy of itself and the edges will therefore always fit. In the worst case, if you have chosen an unsuitable motif, you can just see the 'tile joints', but the symmetry will always be preserved. The mathematical laws of nature are alive, hehe.



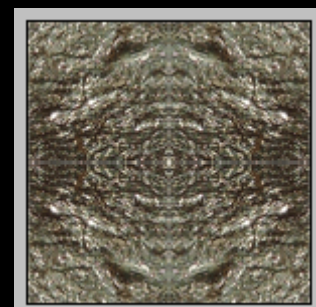
Tileable without symmetries

One problem is that this method creates striking symmetries (right). To get rid of this, get ready for work...

Remedy:



First push the individual parts into each other, but make sure that the basic circumference remains square (left):



Now things get a little more difficult:

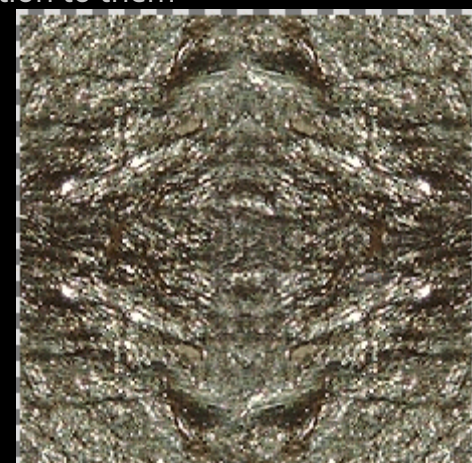
Grab a soft-edged eraser tool and erase parts on the edge of each quarter. Be careful not to get too close to the edges, otherwise the tiling will suffer, and pay attention to them

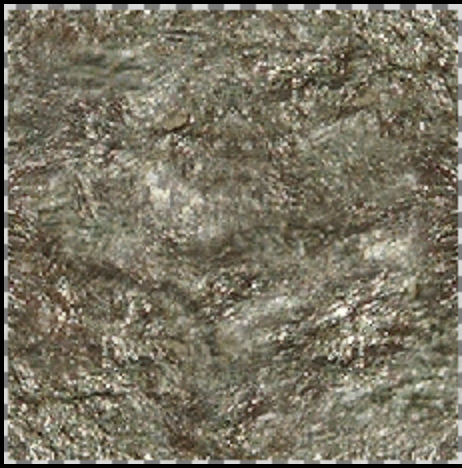
Level you are currently on.

This is how you break it Symmetry a little and

the hard edges disappear (right).

Next, reduce to one layer.





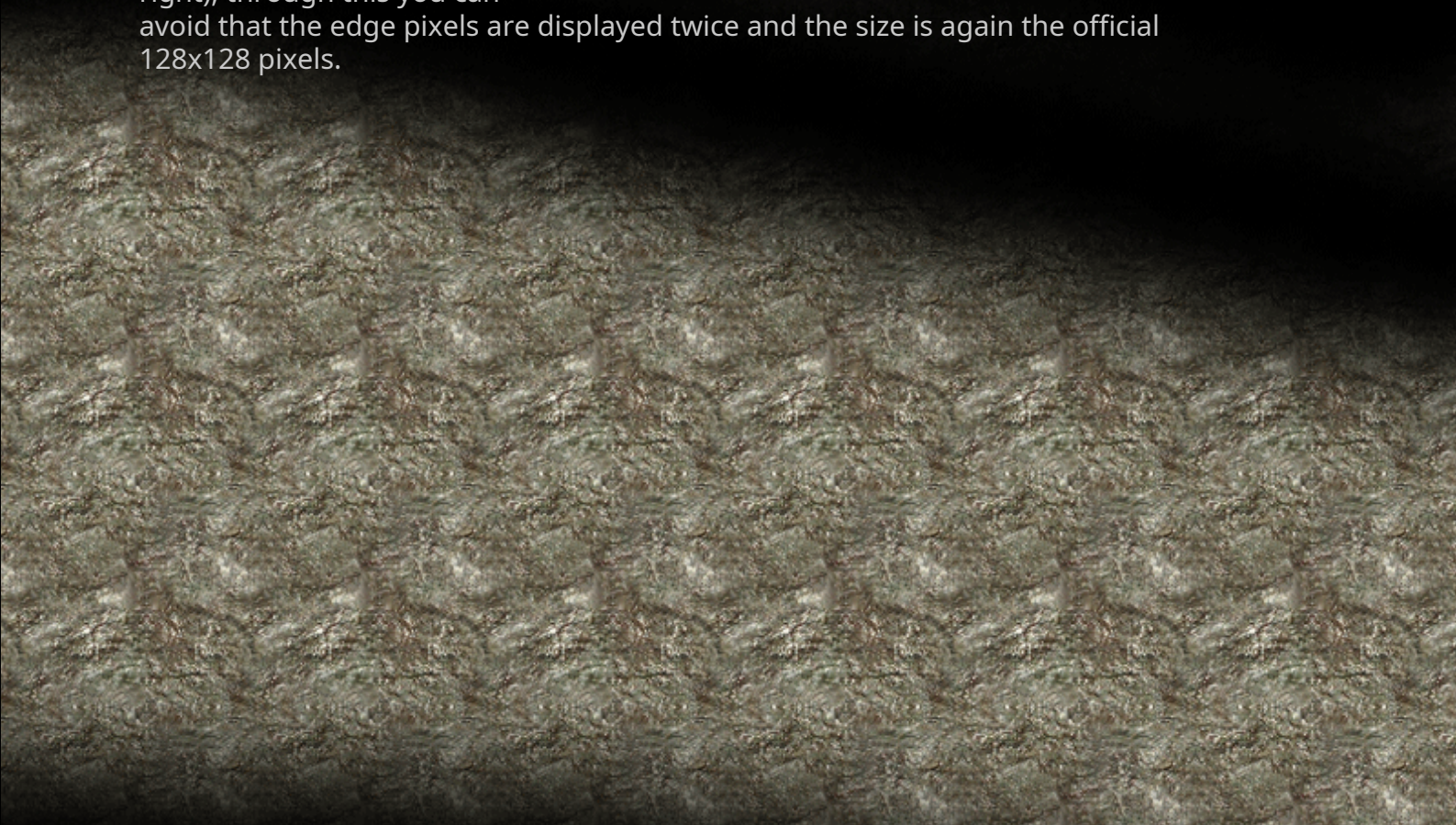
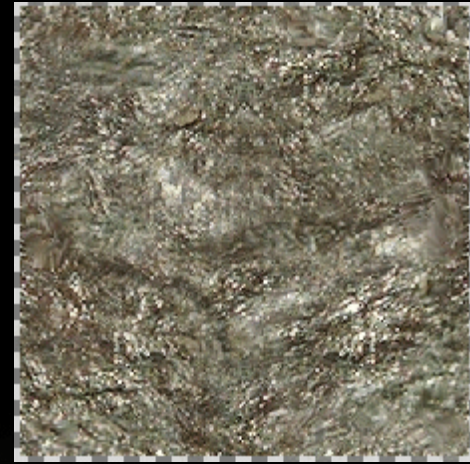
Then get a clone stamp and change the parts that appear twice. If your painting program allows this, get the source from the image from which you got the texture part. You will notice that the closer you get to the edges, the more difficult it becomes (left). You may 🤖 have to paint over almost the entire picture.

When tiling, symmetry would now only appear at the edges. To get rid of them, do the following: Take your wiping finger (or a

similar tool) and now edit an upper and a side edge so that they are also different (right):

If something gets out of focus, it's not so bad, the image is scaled down to the texture size anyway.

Now trick 17: Scale the image to 129(i.e. one more pixel). It then crops one pixel at each edge (top or bottom and left or right); through this you can avoid that the edge pixels are displayed twice and the size is again the official 128x128 pixels.



The texture is repeated, so (unfortunately) it doesn't quite work without symmetries. On the other hand, it helps to design the texture rather structurelessly. Also be creative when editing, so try different settings of the tools or other tools at all every now and then.

It's hard to believe how much text you can create with something like that :o)

In the meantime, a T-joint-free construction for you!

[Back to the overview of section F](#)

065 - Partially transparent textures

Digijo

revised

This tutorial specifically covers semi-transparent textures in Descent 3 levels. There is already plenty of information here at Fischlein for basic information about custom textures in Descent 3, so I want to focus exclusively on partial transparency here:

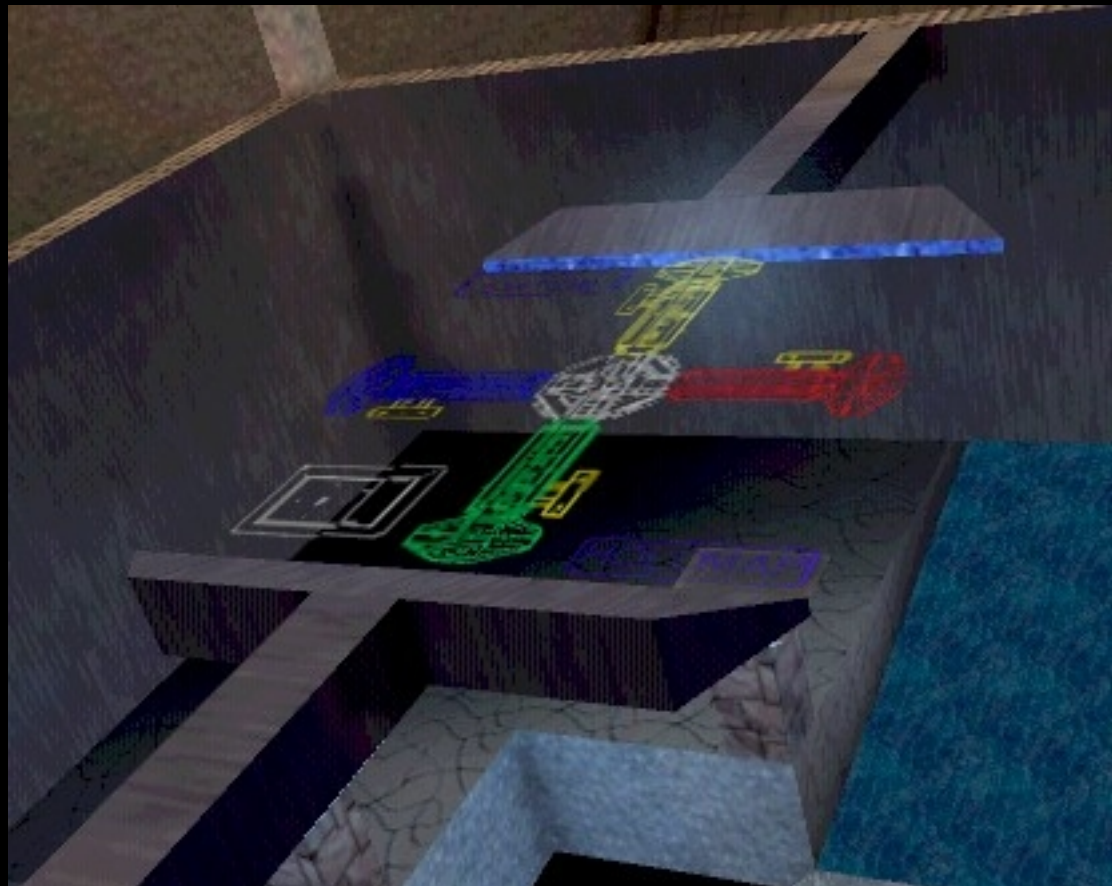
First of all, what are partially transparent textures? Partially transparent textures can create some nice effects e.g. plants (bushes, tree leaves), fences, windows etc. and many more, this means that parts of these textures are transparent and allow you to see parts of the level behind them. In principle it is the same as with transparent GIF files, which, for example, allow a view of a background graphic when integrated into a website. The advantage of this method is clearly the saving of faces/polygons, since, for example, creating a garden fence or tree with polygons/faces would require a few 100 polygons. With partially transparent textures, a similar effect is possible with perhaps only 10 or 15 faces, even if it doesn't look so good.

A disadvantage of this method should also be mentioned, it seems that due to a bug in Descent 3 this method only works on newer graphics cards when they are operated in OpenGL mode. Users of older Voodoo maps with Glide or users of D3D mode see a black area instead of the transparent texture part. This error only seems to occur when the transparent textures are placed on structure faces, on objects or (e.g. as a moon like the IO moon) in the terrain sky, they work everywhere.

In my opinion, these textures can still be used well because almost all Descent 3 players use newer graphics cards in OGL mode. (D3D in Descent3 is really garbage) 😊

Examples

Here we have an example for Partial transparency Textures, these Holomap out Monolith Station exists, however because of the higher Resolution from 8 Individual textures (4 on the front and 4 mirrored on the back) and has additionally one more overall transparency, more on that later.





Another nice example, this time a tree from Tower of Isengard. Here you can clearly see the "holes" in the forest of leaves, these trees only have 16 faces each and protect the frame rate...

Basics

Now for the technical part, Descent3 basically knows 2 types of textures .oifformat and that.oafFormat. OGF is used for standard textures and means "Outrage Graphic Format", OAF is used for animated textures and means "Outrage animated Graphic Format" or similar. OAF's are nothing more than OGF's connected in series. Both formats support partial transparency. The graphic data in these graphic formats

are in turn compressed with the so-called ARGB compression, Descent3 knows 2 versions, the 1555 ARGB and the 4444 ARGB. The abbreviations ARGB stand for:

- A=alpha channel
 - R=Red channel
 - G=green channel
 - B=Blue channel
- The numbers in front of it represent the respective bit depth of the individual channels:
- 1555=1 bit alpha, 5 bit red, 5 bit green, 5 bit blue
 - 4444=4 (3) bit alpha, 4 bit red, 4 bit green, 4 bit blue

The alpha channel is interesting here because it determines which pixel appears transparent and which pixel is not transparent. With 1555 ARGB compression there is only 1 bit for the alpha channel, which means that each pixel can either be made fully transparent or fully visible. I haven't experimented with the 4444 ARGB compression yet, but it creates a transparency for each color channel. This means that, for example, only the red portion or only the blue portion of a face or object behind it shines through.

Total transparency

I just want to briefly talk about the overall transparency of textures in Descent3. In addition to the partial transparency via the alpha channel in the texture itself, Descent3 also has an adjustable overall transparency for each texture via the gametable entry "ALPHA" of the corresponding texture, regardless of whether this texture has an alpha channel or not. The alpha value in the game table can be adjusted from 0.0 to 1.0, where 1.0 means fully visible and with a value of 0.0 the texture is invisible. With a value of 0.5 it is semi-transparent, such as in stained glass. This value then basically applies to all pixels of the texture.

The alpha value for the overall transparency of the texture is set at the point marked red in the game table. This works for every texture and basically applies to all pixels of the texture.

```

... Texture: Sol-BuildWindows1
-----
File name: Sol-BuildWindows1.o
Destroyed file name:
Red light: 0.000000
Green light: 0.000000
Blue light: 0.000000
Alpha: 1.000000
Speed: 1.000000
Horizontal slide: 0.000000
Vertical slide: 0.000000
Reflectivity: 0.720000
Corona style: A
Damage: 0
+ Flags
-----
Sound name:
Sound volume: 0.000000
  
```

Creating semi-transparent textures

To create partially transparent textures you basically need 2 things: a graphics program that supports alpha channels and the "Tagged Image File Format" (TGA), and a program to convert the texture to OGF (or OAF). Descent3 provides the OGF tool (ogftool.exe), but the D3 Imagetool Ver.1.03 from Supersheep is much better. Of course this is also available at Fischlein. I usually use the graphics program Paintshop Pro 7 (PSP7) to edit the graphics; Proggy is simply brilliant.

First you need a corresponding motif from the Internet (there are millions of templates to be found everywhere) or draw your own texture (assuming you have the appropriate 😊 talent). The graphic is cropped and/or resized to texture size, it should then be square (height=width).

Usable formats (in pixels):

- 32x32
- 64x64
- 128x128
- 256x256

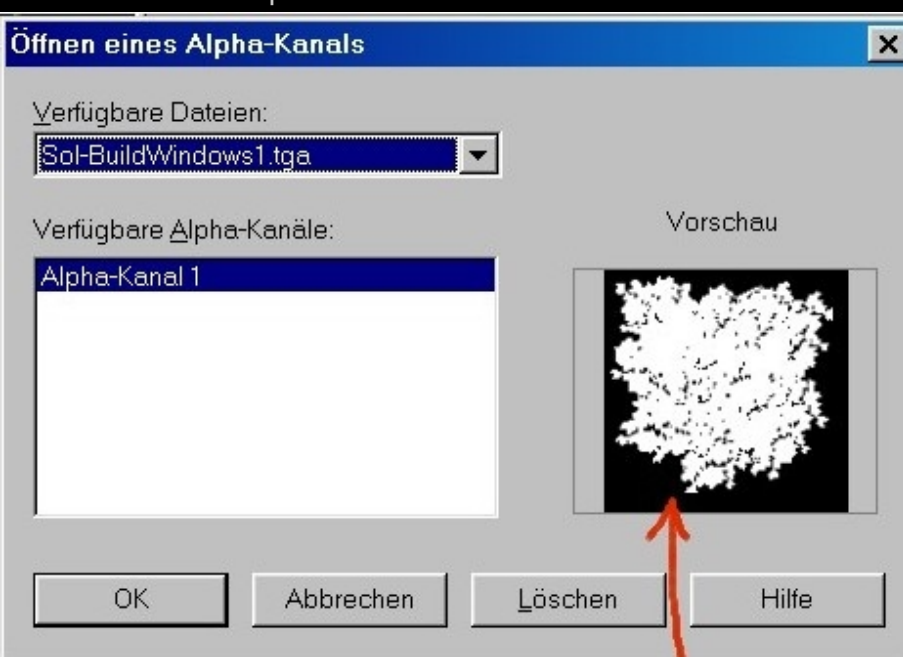
As far as I know, other formats are also possible but not recommended. Briefly about the TGA format, this format is ideal for creating textures because it is supported by all converter programs and offers exactly what you need for a texture, up to 32 bit color depth and an 8 bit alpha channel.

This one on the right is the template I used for the foliage in the "Tower of Isengard" trees. This texture is already cropped to 128x128 pixels and has a black background.



Now everything that is now black should be transparent later in the Descent3 level, for that we need an alpha channel. For this purpose I created a mask in PSP7 containing every black pixel and saved this mask as an alpha channel.

To do this, I did a CMYK separation with PSP7 and reduced the black channel to 2 bit color depth. The black channel is then inverted and saved as a mask.

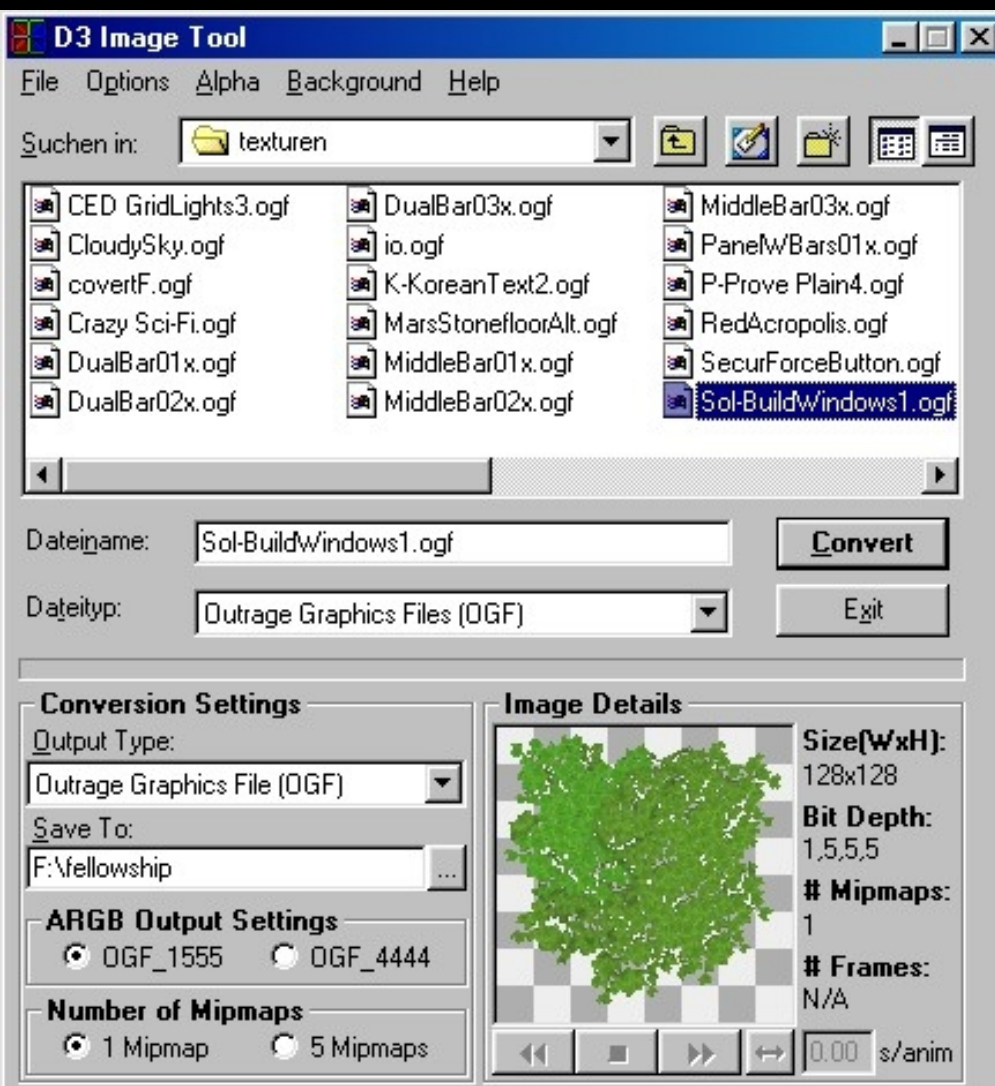


This alpha channel only has 1 bit depth (not transparent / fully transparent) because we want to convert the file later as OGF (1555 ARGB).

In this PSP7 menu on the left the alpha channel saved as a mask is added, on the right you can see the preview of the channel. Later in the TGA and in the finished OGF texture everything will be black in it Alpha channel be transparent.

Here on the right you can now see what the finished TGA looks like with the alpha channel displayed. Like a GIF file, the alpha channel is integrated into the TGA file and does not need to be saved separately.

Now that you have saved the texture as a TGA file (I chose Sol-BuildWindows1.tga as the file name here because I want to replace a corresponding file in Descent3 later), all that's left is to convert it. To convert I use the D3Imagetool program mentioned above, but the included OGF tool also works.



This screenshot from D3Imagetool gives a good overview of the settings. The input file is selected at the top (of course it has to be the TGA, I accidentally chose the finished OGF for the screenshot), at the bottom right you can see the preview and the image parameters, at the bottom left is the interesting area with the compression parameters.

Settings. For these textures we choose 1555ARGB, i.e. 1 bit alpha channel. It would also be possible 4444ARGB

Choosing compression, since the alpha channel of the TGA only has a depth of 1 bit, would not be possible. Difference to 1555 can be seen. Mipmap level doesn't actually matter because with only one MipMap the remaining maps are created in the graphics card. 1 or 5 is just a question of loading time and quality of the graphics card.

Now all you have to do is click convert and the leaf texture with partial transparency is ready. To integrate the texture into Descent3 levels, it must be mentioned that there is a flag in the corresponding gametable entry for the texture, the TMAP2 flag. This flag indicates to Descent3 that the texture has an alpha channel and should be used, so don't forget to activate the TMAP2 flag. If you simply want to replace a texture to save yourself the gametable entry (see Linux and MAC compatibility), you should look for a texture to replace that has this flag set. (In this case, the properties of the original texture and the custom texture must be exactly the same, see ARGB, mipmap and resolution). For example, the Sol-BuildWindows series has TMAP2 flags.

- ☒ Object
- ☐ Rubble
- ☐ Texture 32
- ☐ Texture 64
- ☐ Texture 256
- ☒ TMap 2
- ☐ Water procedural
- ☐ Procedural

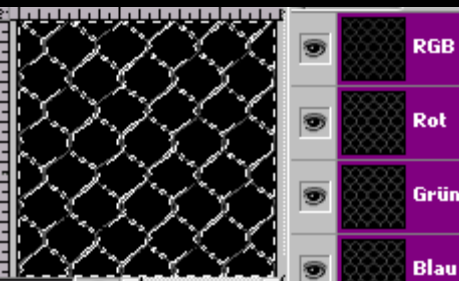
This is the TMAP2 flag in the gametable that tells Descent3 whether there is an alpha channel in the texture and whether it should be displayed. Strangely enough, I've also had it that the alpha channel is displayed even without this flag, but better safe than sorry. (D3 seems to have problems with transparent/animated textures in general, see bug list 1.4)

I hope that this little article is useful for the level builders' guild.

With this in mind, I wish you a T_Joint free level building,
[NuB2]Digijo(digijo@gmx.de)

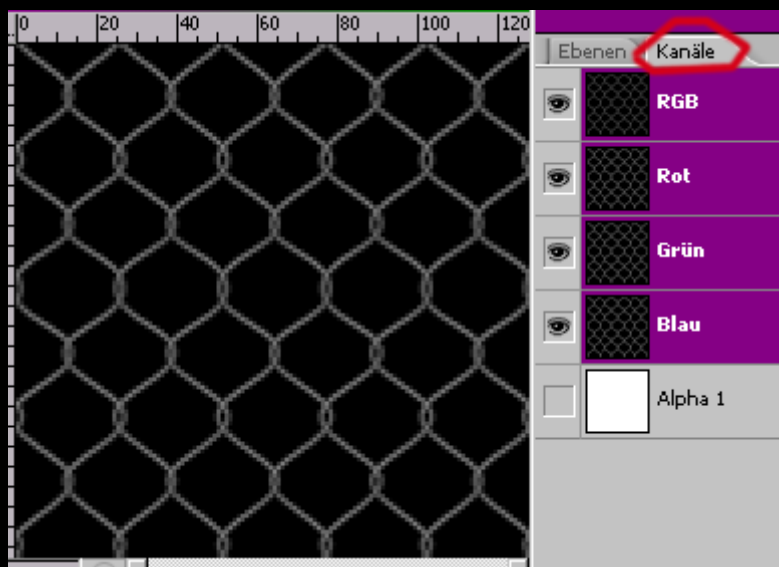
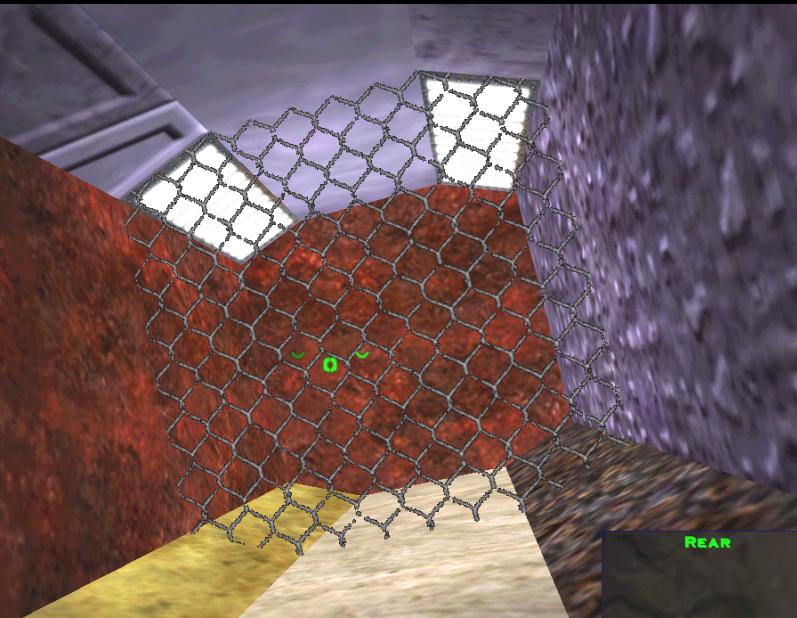
Addendum for Photoshop:

In the PS the whole thing works so that the channels are in the layer palette. The image must be 32-bit.tga-present (right); If necessary, you have to create the alpha channel first: click on Channels and create a new one there. An alpha channel is automatically created. Black means transparent, white means visible (if you don't change anything when creating the channel).



Ctrl-Shift-I flip.

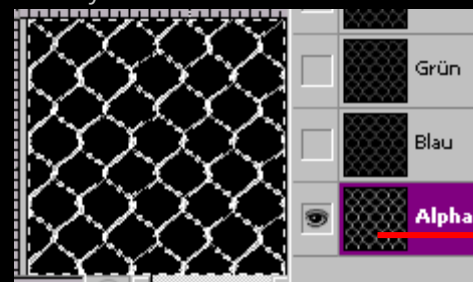
Then you click on the alpha layer and press the button there **Delete**;
With this you have created your transparency mask. Now save it
.tga-File.




In the color channels
So now choose with that
Wand those

Areas that you do NOT want to be transparent. Be sure to set the magic wand to 'Smooth' so that the boundaries between opaque and transparent are not too hard (left).

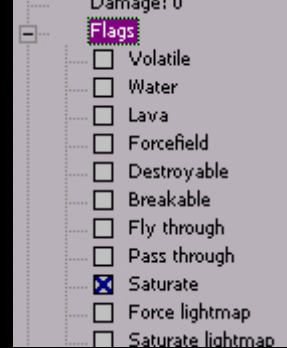
Alternatively - if that's easier - you can also select the areas that you want to become transparent and then mark them with



The rest like
had... change to.oif, integrate into one
.gam, these as.mn3with the.oif
save together, load in the editor etc. etc.

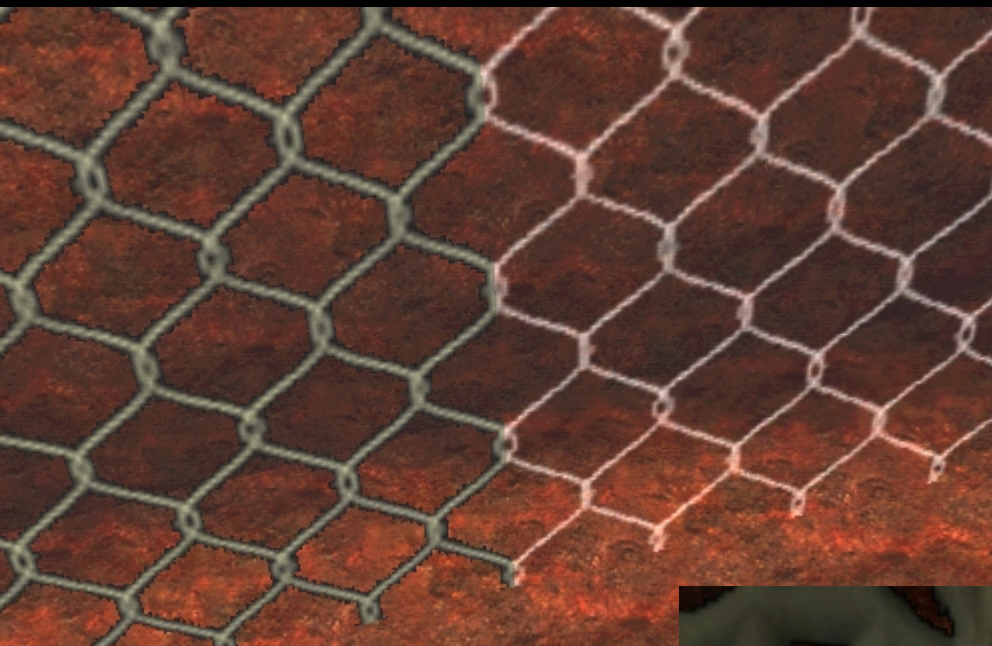
Result: chain link fence in D3 

But if you look closely, you'll see that it doesn't look that great. On the one hand, the transparency mask didn't turn out properly, and on the other hand, the whole thing doesn't look particularly good at the edges.



To get rid of this unsightly transition, you can do this in the .gamthe flag for Saturateset (right):

If you look below, you can see that the transparencies are much smoother (the texture in the right part of the picture below has thisSaturateflag).



Unfortunately, this flag also has a disadvantage, namely that the entire texture now becomes transparent.

Unfortunately, various attempts to set flags in all combinations have not yet brought a solution, so you have to think carefully about where you want to use a partially transparent texture and set the flags accordingly.

As you can see on the right, the texture is created by thisSaturate-Flag relatively transparent; which is not desired in this case (right).

If such a texture also has the alpha flag and you make it a mirror, you even get broken mirrors (below).



Back to
Overview
from
Section F

066 - Animated Textures - Screenshots

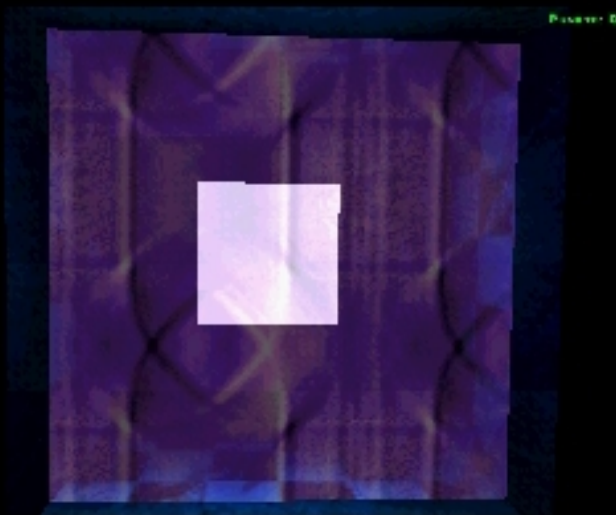
(LL)Dark

The following images show the animated textures that are only displayed as "rainbow textures" or displayed incorrectly in D3Edit. You can see the transparent textures in the white square in the background.



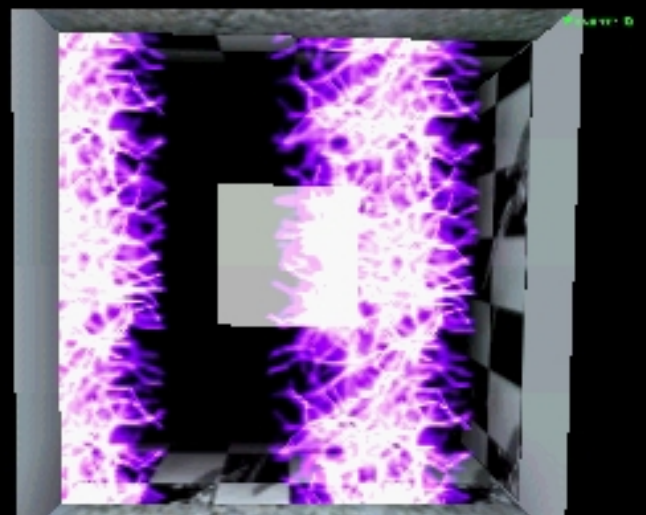
Tankwater

Light: 0.1 / 0.7 / 0.6 - normal, bewegte sich krauselnde Oberfläche mit seitlich darueber - laufenden Wellenfronten, dampft bei Beschuss



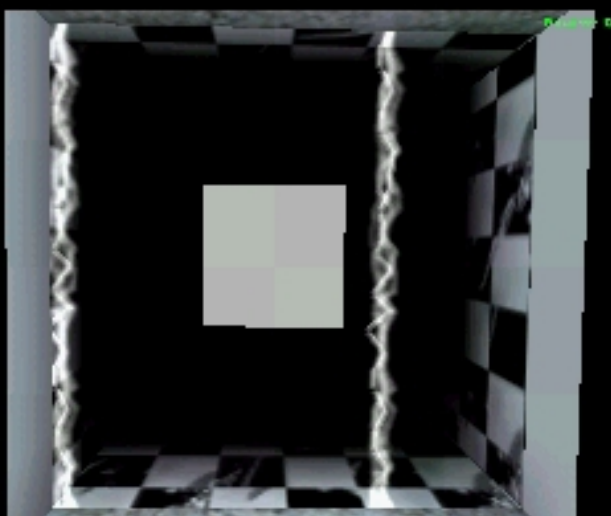
SS - Forcefield1

Light: 0.0 / 4.0 / 8.4 - Schild, sich ueberschneidende Kreise und Wellenfronten



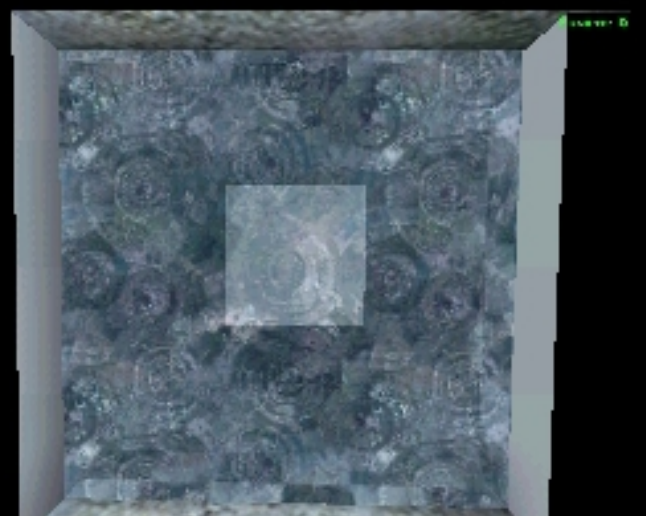
ThinMatzenLightningPurple

Light: no Light - normal, von unten nach oben durchlaufende Straenge von Blitzen



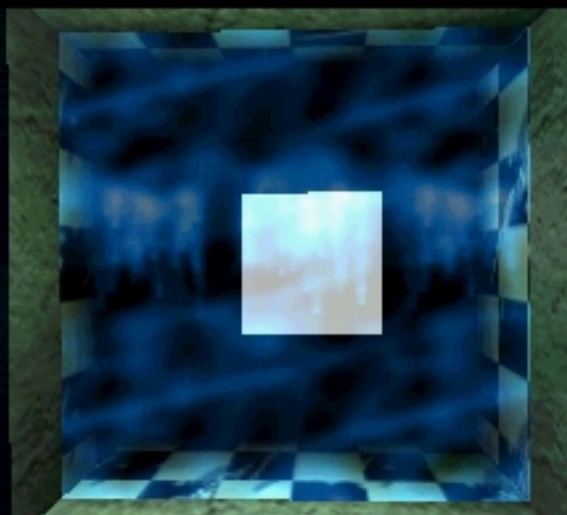
ThickLineLightning

Light: no Light - normal, schmale bänder die sich bewegen wie elektrische Entladungen



Waterfall1

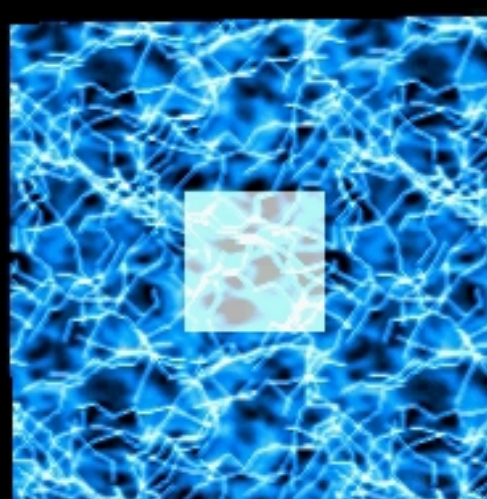
Light: no Light - normal, von oben nach unten fließend mit Tropfkreisen, dampft bei Waffenbeschuss



Picture 0

EXP_Shield_alter

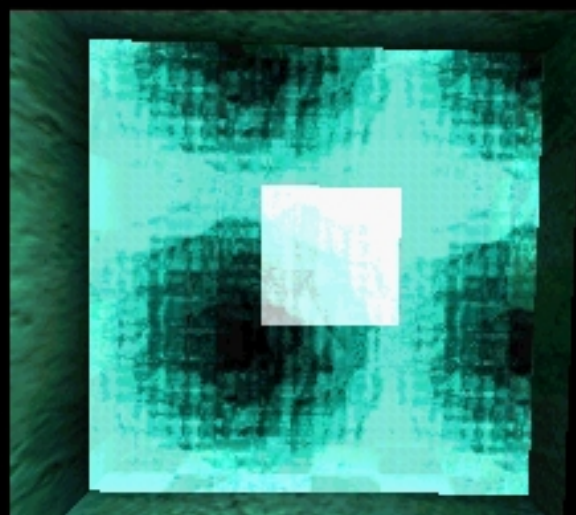
Light: 15.0 / 20.0 / 15.0 - Schild, aufblitzende sich kreuzende Linien und Fontänen , langsam von unten nach oben durchlaufend



Picture 0

ForcefieldBlue

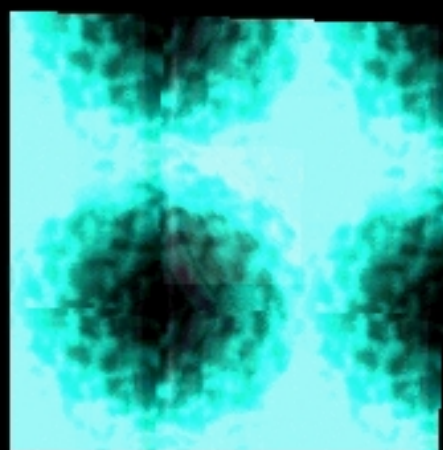
Light: 0.0 / 2.0 / 4.0 - Schild, durcheinanderzuckende Blitze , von links nach rechts durchlaufend



Picture 0

Alien Force Field_1

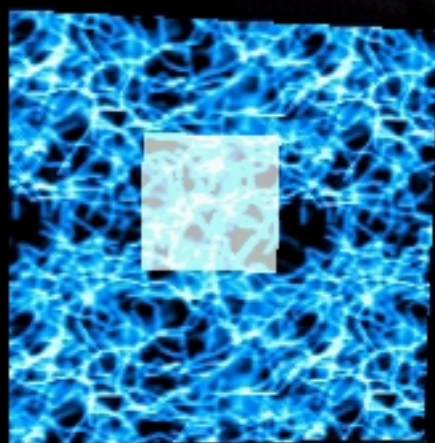
Light: 4.0 / 10.0 / 8.0 - Schild, schnell bewegte sich konzentrisch überschneidende Kreise



Picture 0

ALIENCPULINK

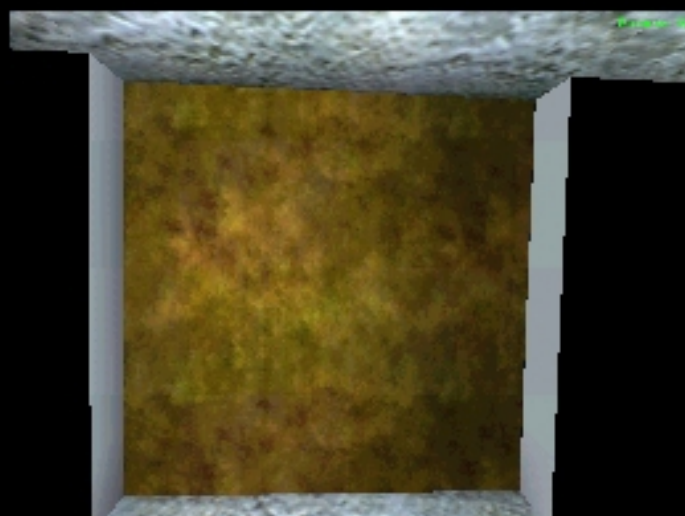
Light: 0.0 / 3.0 / 2.0 - normal , mittelschnell auseinanderlaufende Wellenfronten



Picture 0

BlueMagneticField - V

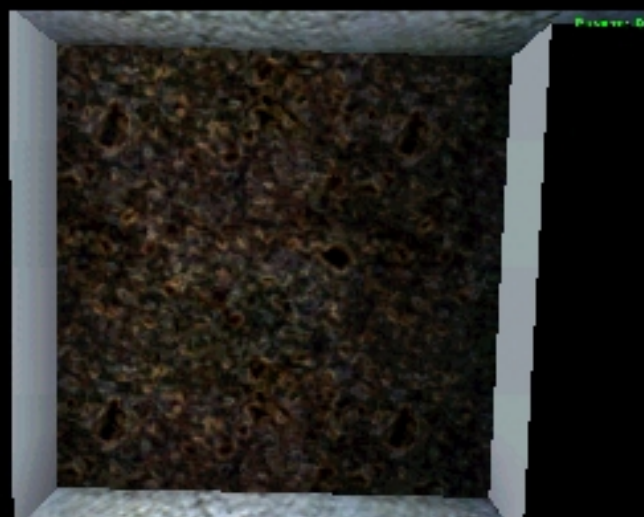
Light: 1.0 / 1.5 / 2.0 - Schild, mittelschnell durcheinanderzuckende Blitze



Picture 0

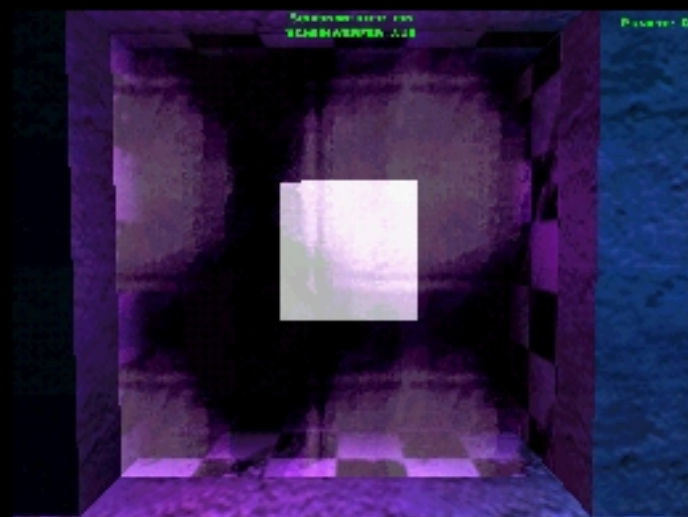
BrownSewage

Light: no Light - Saeureschaden, Inagsam von oben nach unten fließend mit leichten seitlichen Wellenfronten, dampft bei Waffenbeschuss



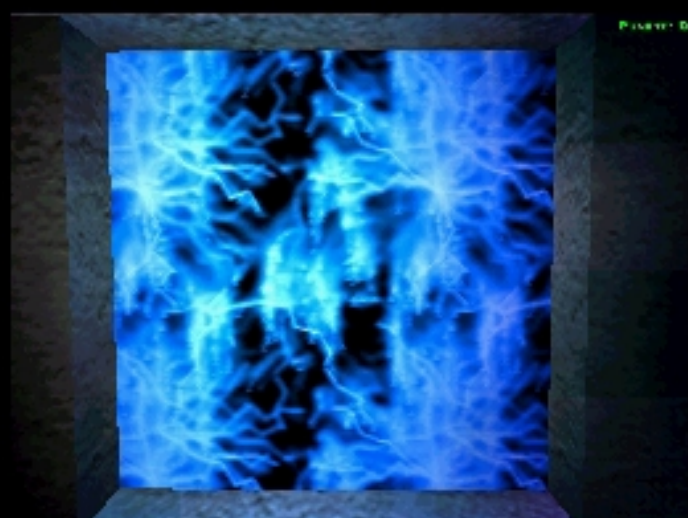
BrownVolcanoRockShimmer

Light: no Light - normal , enthaelt kleinere (hitze) wabernde Stellen



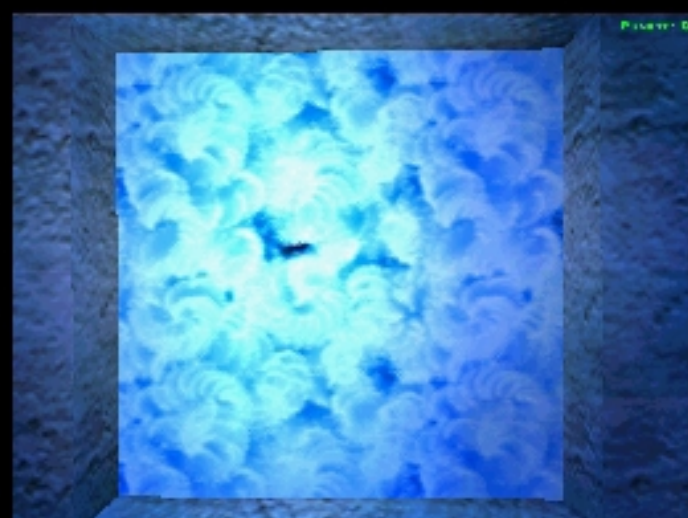
CED - Forcefield

Light: 15.0 / 5.0 / 20.0 - Schild , bewegte sich ueberschneidende Kreise und Wellenfronten



CED_CoreSkin01

Light: 2.0 / 5.0 / 9.0 - normal, von Punkten ausgehende Blitze mit herabrieselnden Fontainen dazwischen



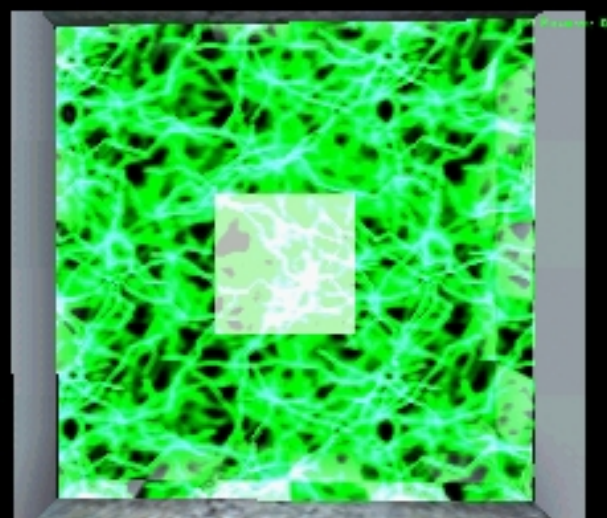
EnergyConvpro

Light: 20.0 / 50.0 / 100.0 - normal, sich ringelnde Energieentladungen (oder Plasma) von links nach rechts durchlaufend



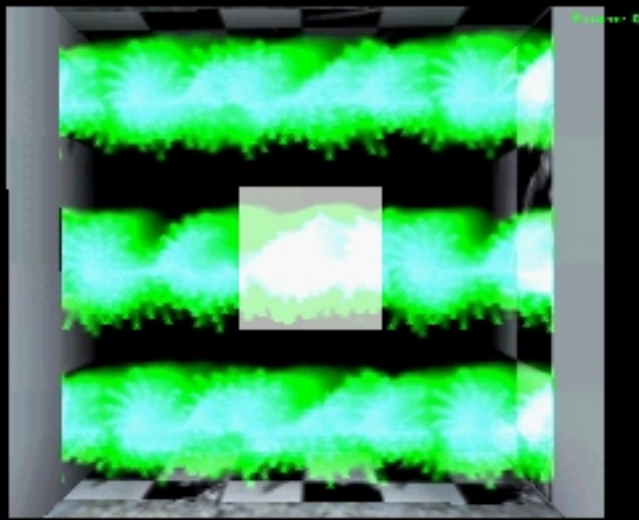
EuropaWaterGlowing

Light: 1.5 / 2.0 / 3.0 - normal, als ob kleine Tropfen ins Wasser fallen, dampft bei Waffenbeschuss



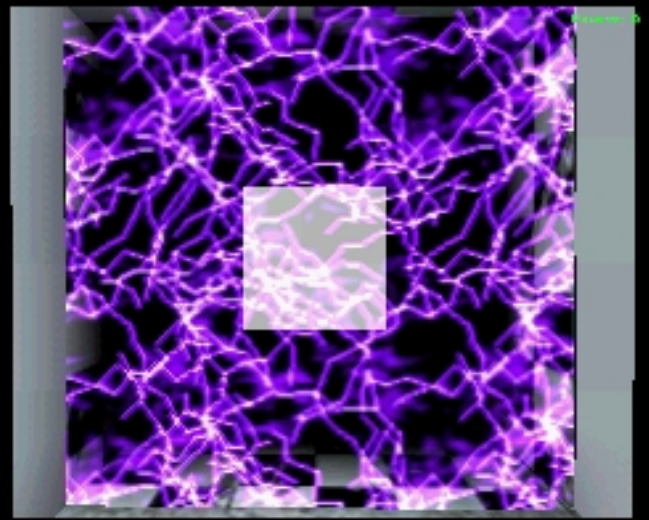
ForceFieldGreen

Light: no Light - normal, durcheinanderzuckende Blitze



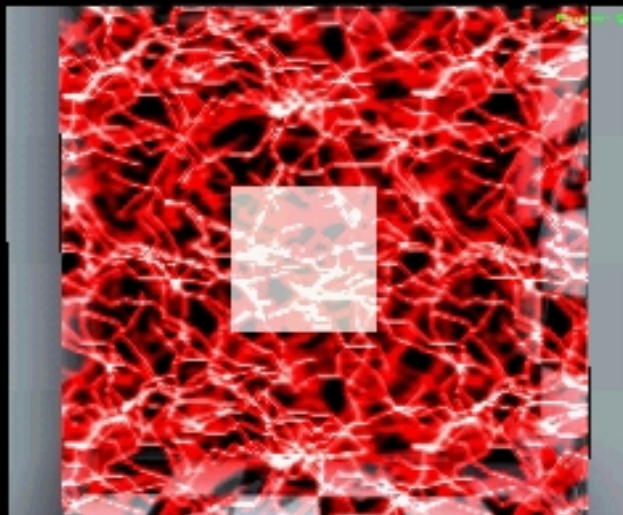
ForcefieldGreen2

Light: no Light - normal, sich drehende Spiralen aus kleinen Blitzen



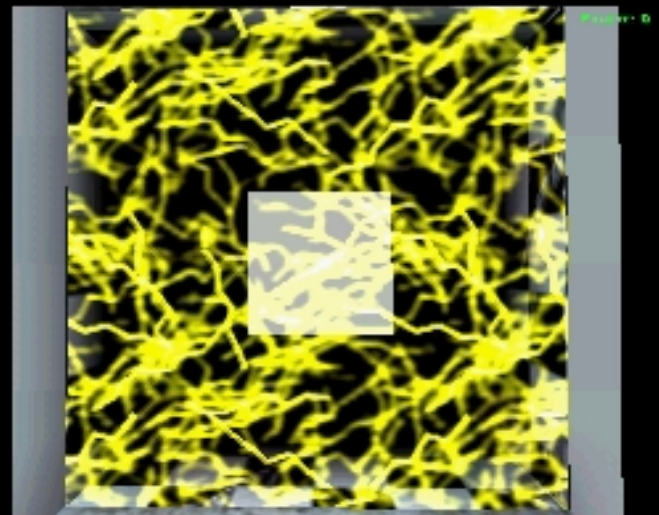
ForcefieldPurplelightning

Light: no Light - normal, durcheinanderzuckende Blitze, langsam von unten nach oben durchlaufend



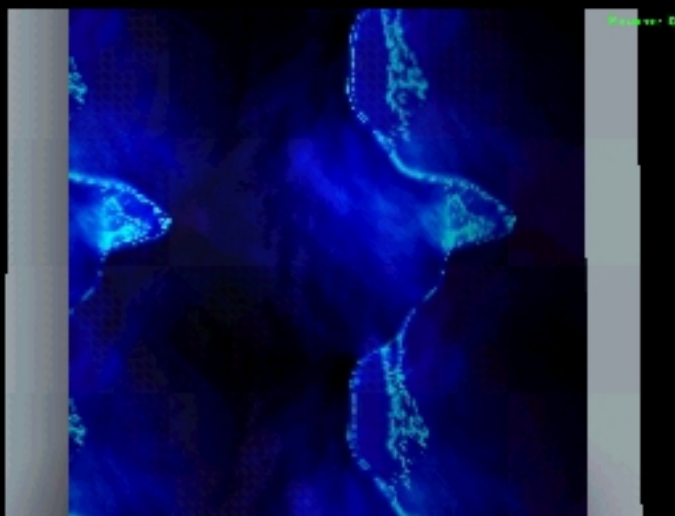
ForcefieldRed

Light: no Light - normal, sehr schnell durcheinanderzuckende Blitze



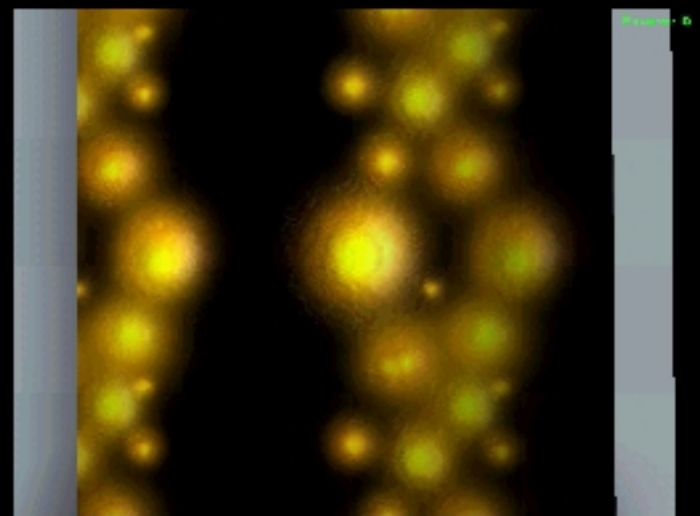
ForcefieldYellow

Light: no Light - normal, durcheinanderzuckende Blitze



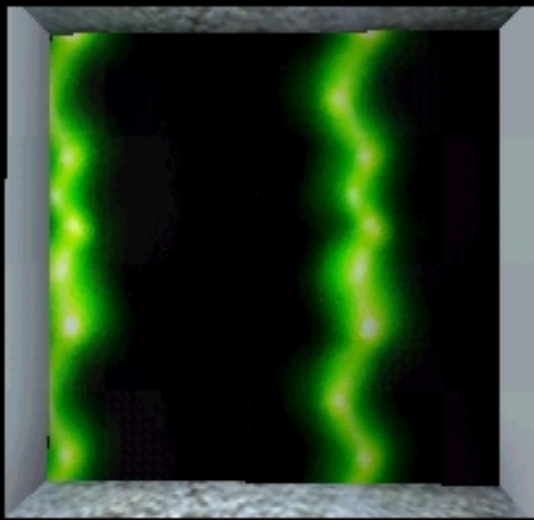
FunkyEffekt1

Light: no Light - normal, an den Strängen die zu sehen sind entlangkriechende el.-Entladungen



FunkyEffekt3

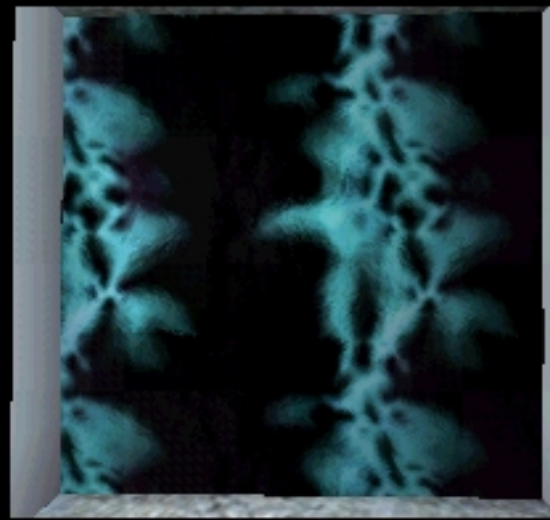
Light: no Light - normal, stillstehend, mit sanft nach aussen dringender Strahlung (wie soll man das sonst beschreiben ...)



Plasma-0

FunkyEffekt4

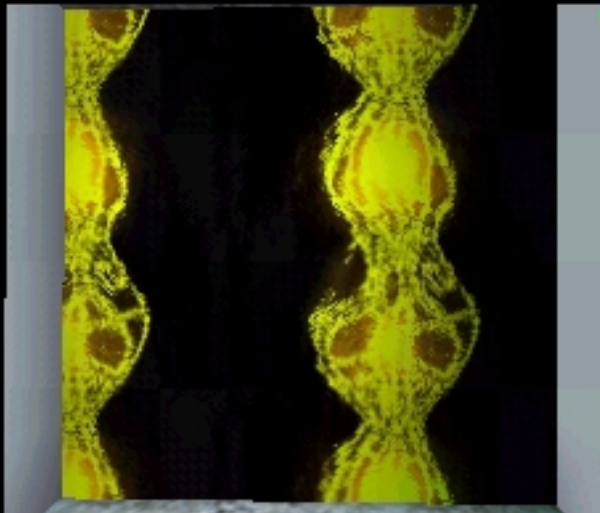
Light: no Light - normal, stillstehend, mit sanft nach aussen dringender Strahlung (wie soll man das sonst beschreiben ...)



Plasma-0

FunkyEffekt5

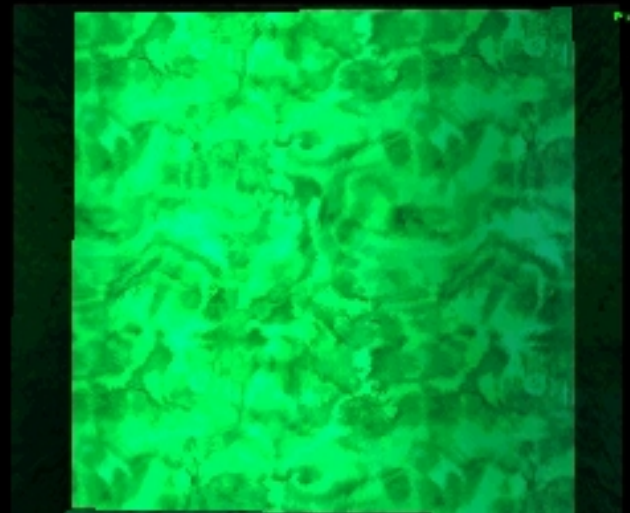
Light: no Light - normal, stillstehend, mit mäßig stark nach aussen dringender Strahlung (wie soll man das sonst beschreiben ...)



Plasma-0

FunkyEffekt9

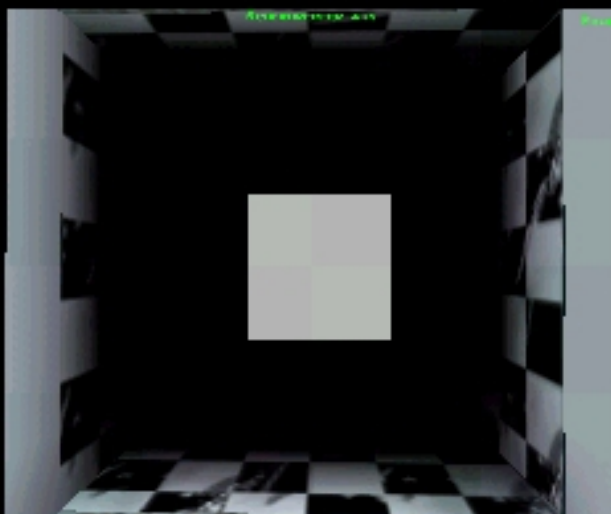
Light: no Light - normal, die Strahlen bewegen sich ganz langsam wellenförmig von unten nach oben



Plasma-0

GreenMysterySurface

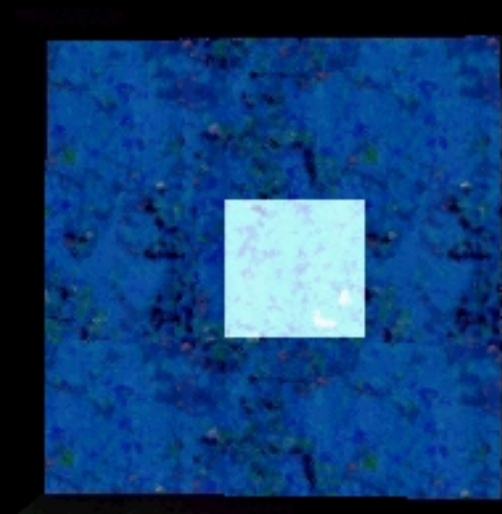
Light: 0.0 / 10.0 / 5.0 - Schild, prismenartig wabernde Oberfläche



Plasma-0

Korea_Matzen_Shield

Light: no Light - normal, absolut durchsichtig, ohne sonstige Besonderheiten



Plasma-0

Large Water Body

Light: 0.0 / 2.2 / 2.4 - normal, leichte wellen, gekrauselte Oberfläche, dampft bei Waffenbeschuss



Picture 0

MarsForcefield

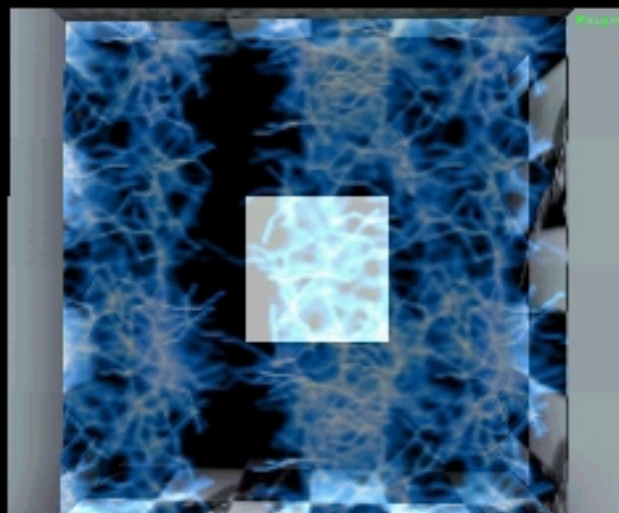
Light: 1.0 / 0.5 / 0.5 - Schild, starke durcheinanderlaufende Wellen und Kreise



Picture 0

MarsVolcanoRockShimmer

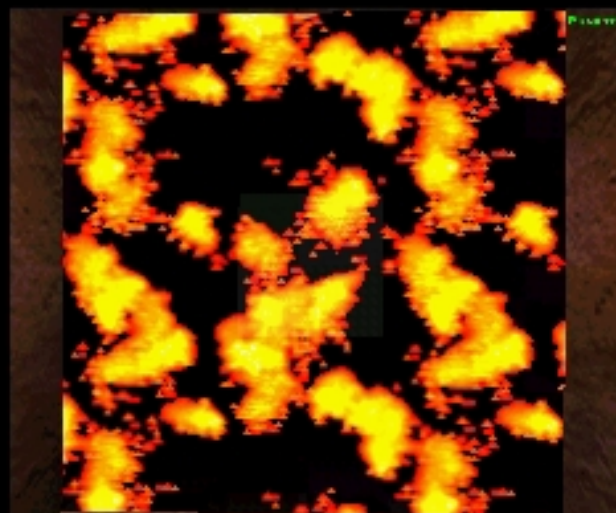
Light: no Light - normal , mit ausgedehnten darueberhinweglaufenden (hitze) wabernden Flaechen



Picture 0

Matzen Lightning

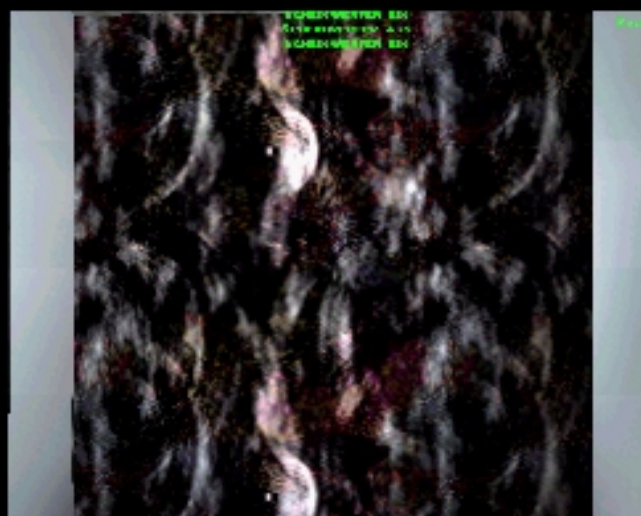
Light: no Light - normal, Blitzstraenge von unten nach oben durchlaufend



Picture 0

Nano Plasmic Cesspool

Light: 18.0 / 10.0 / 8.0 - verursacht Hitze schaden, sieht aus wie aufflackernde gluehende Kohlen, dampft bei Waffenbeschuss



Picture 0

OilSlick1

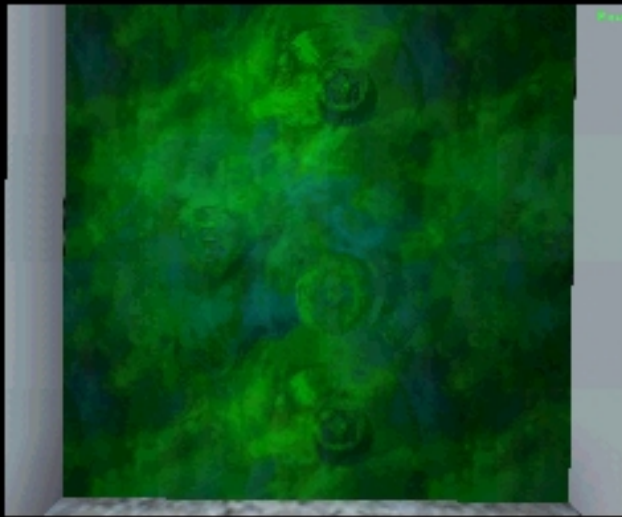
Light: no Light - normal , als ob viele grosse Tropfen hineinfallen



Picture 0

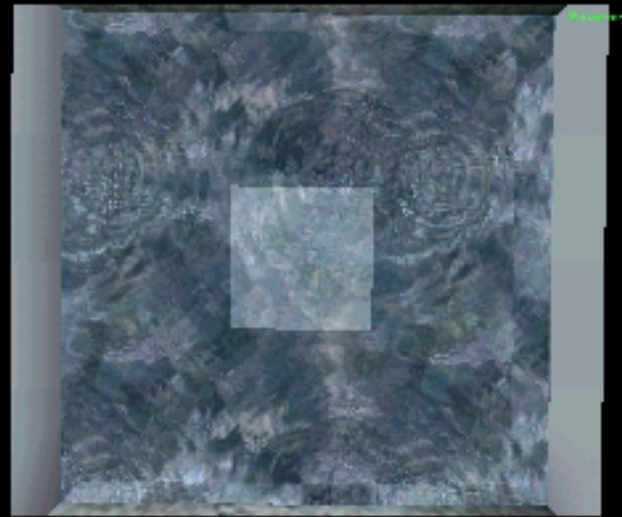
OrangePlasma

Light: 2.0 / 1.5 / 1.0 - normal, leicht bewegte Oberflaeche, dampft bei Waffenbeschuss



PhobosAcid

Light: no Light - verursacht Saeureschaden, konzentrisch auseinanderlaufende Kreise, dampft bei Waffenbeschuss



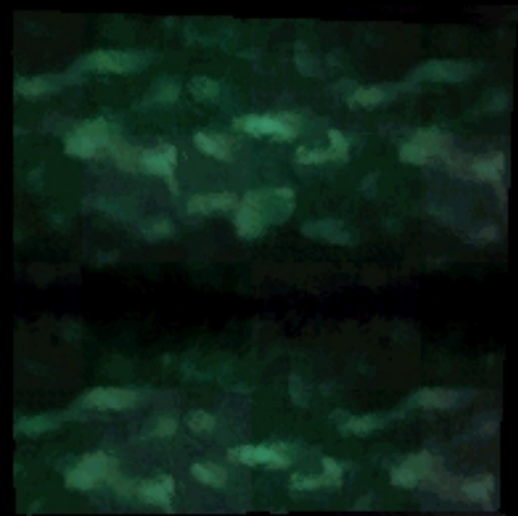
PondRipple1

Light: no Light - normal, auseinanderlaufende konzentrische Kreise (Wasser eben), dampft bei Waffenbeschuss



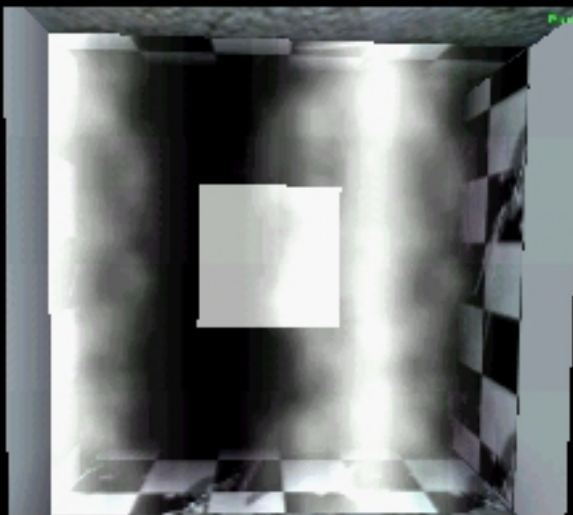
Rippley1

Light: 0.5 / 0.4 / 0.2 - normal, schnell von unten nach oben durchlaufend



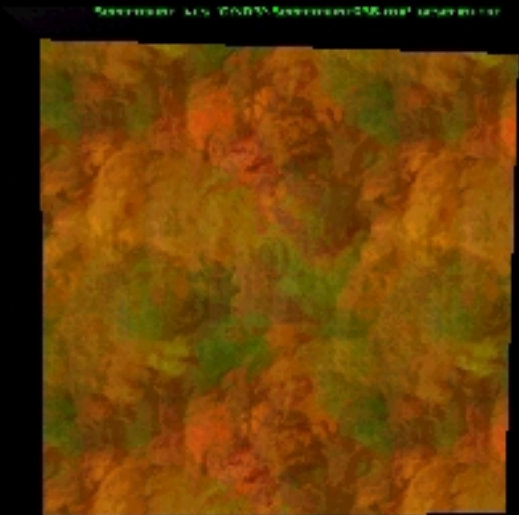
Rippley2

Light: 0.2 / 0.5 / 0.3 - normal, sehr schnell von unten nach oben durchlaufend



rsa

Light: no Light - normal, schmaler und dann wieder breiter werdende dampfartige Straenge



SewerWater

Light: 2.2 / 2.1 / 1.0 - normal, leicht bewegte sich krauselnde Oberflaeche,dampft bei Waffenbeschuss

Thanks again to (LL)Dark!

(From me too)

067 - Animated Textures - Homemade <>

Ragil Ral

The process differs only slightly from creating/incorporating homemade static textures. You just create not one, but several .tga's, which then accordingly .ogf's being transformed. From there things go differently.

Like an animated one.girlie in one.oaf(Outrage Animated File) the partial images of the animation are presented in a row and are in one file summarized.

First step

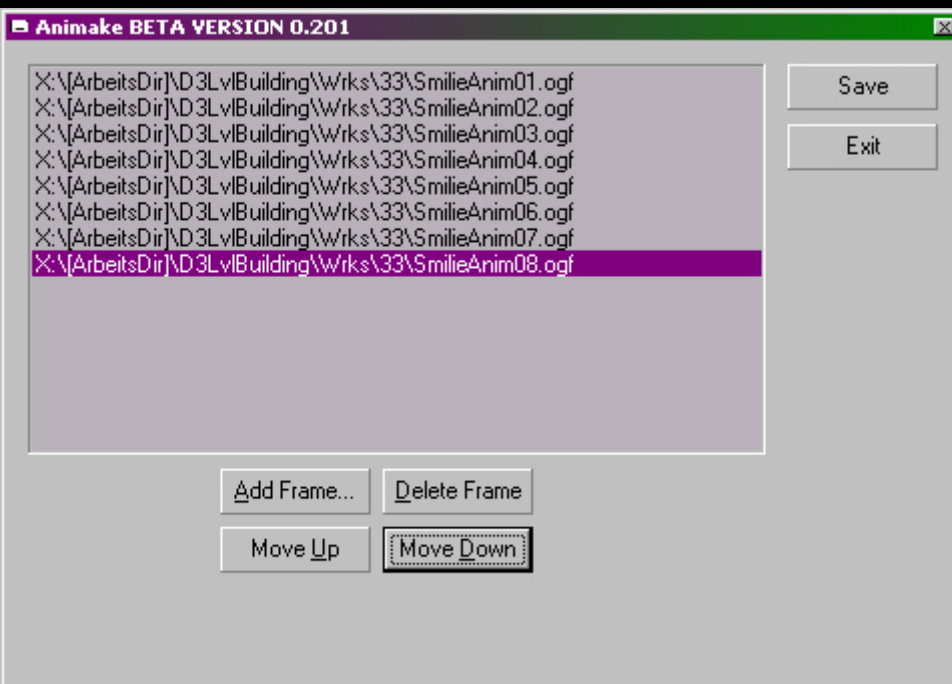
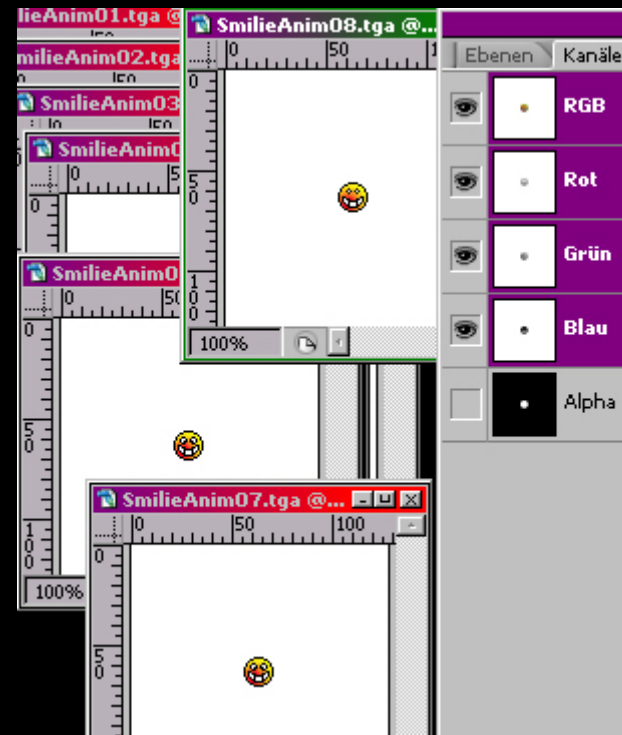
Create your individual images. They can also have transparent, semi-transparent as well as other flags. I took an animated smilie from the Descent forum, extracted the individual images and created an alpha mask for each one. There are eight pictures.

If certain frames are the same, you can of course use them multiple times, we are lazy by nature 🤪

Second step

is short, with OGFTool or D3 Image Tool the .tga-Files after .oif walk.

Third step



Now you need the tool 'Animake', currently in version BETA 0.201. Start it, click **Add Frame...** and select your individual images (left). You can come with me **Ctrl** and click to select several at once. Animake will then show you a list of files. Arrange them by marking an entry and using **Move Up** or **Move Down** move in the list.

Then just open it **Save**, You will then be prompted where you want to save, and the file extension will be with .oif immediately given.

fourth step

It's just a matter of integrating it into one.gam or ..mn3, and, of course, using it in a level 🤪

Since a screenshot doesn't help here, here's it.oaf in the packed file.

Back to the overview of section F

068 - Textures @ 256x256

Ragil Ral

There is the option to use textures with an edge length of 256 pixels, but D3 has to load four times as much data per texture than with the standard 128x128 textures. Therefore, you should think carefully about where you use these textures.

The primary areas of application are textures for the terrain, partially transparent ones and if you want an image in the level that should be higher resolution; For example, a map or a screen with content that is necessary or helpful for the player to solve a mission.

Terrain textures

All textures of the outside world are extremely stretched, so it is recommended to use 256x256 textures for satellites, the sky and the terrain itself. Especially with the sky, as it consists of a single texture that is stretched over the skydome and not tiled.

Partially transparent textures

Here's the chain link fence again, this time @256.

If you
the picture
considered,
should
actually
understood
be...



Again
the same
But with textures
demSaturate-Flag:

Compare to this
the chapter about
Partial transparency.

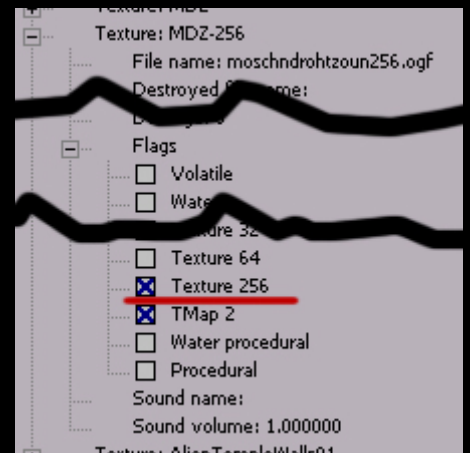
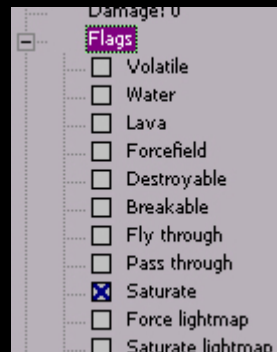
Create

You do the same thing as described in the previous sections, just with 256 pixels square.

The GAM entry

The flag for 256 and possibly for Saturateset:

That was it!



Back to the overview of section F

Section G-sounds

Noises, sounds and music enliven the game, especially in single player. Here you will learn how to use them. Sounds are also important for custom doors and other level objects.

069	Play sounds	What you have to do so that D3Edit can play the sounds.	Atan	265
070	Insert sound sources	Inserting noise sources	Ragil Ral	268
071	Terrain sounds	Noises in the terrain	Ragil Ral	269
072	Sound sources on the terrain	Sources of noise on the terrain	Ragil Ral	270
073	Music in your own Descent3 level	Insert music files into D3 levels	Toxxeon	272
074	Sound: problems avoid	An explanation of how to solve problems can handle the sound	Atan	276

[Toc](#)

069 - Play sounds

Atan

(V1.0)

Atlas Graphical Script Editor v1.0 - licht2*

Scripts:

- Script 000: Sound
 - Owner: Level
 - Event: Level Start
 - If the following condition is met:
 - ALWAYS
 - Then perform the following actions:
 - Play 2D Sound NOT SPECIFIED for player IT, volume = 100.0%
 - Sound: NOT SPECIFIED

Sound Selection Prompt

Available Sounds:

- AfterBurner
- AtrBmrTail
- AmbAcidBurning
- AmbAnimalGrunt
- AmbBigUnlock
- AmbBirdA
- AmbBirdB
- AmbBirdC
- AmbBirdD
- AmbCaveDebrisA
- AmbCaveDebrisB
- AmbCaveDebrisC
- AmbCaveDebrisD
- AmbDistantBangMany
- AmbDistantBelch
- AmbDoorCellShut
- AmbDoorCellSlide
- AmbDripsA
- AmbDripsB
- AmbDripsC

Play Sound Stop All Sounds

Room Properties

Name: <none>

Flags

- ☐ Refueling
- ☐ Secret
- ☐ Red Goal
- ☐ Blue Goal
- ☐ Green Goal
- ☐ Yellow Goal
- ☐ External
- ☐ Door
- ☐ NoLight
- ☐ Flicker
- ☐ Strobe

Audio

Ambient sound: Electric

Damage sound: Melee Attack

Reverb Preset: <none>

Play sounds needs to extract all ways from the d3 hog into a custom wav folder. See settings dialog.

Sounds/Waypoints

SoundSources

Current

Name: AmbAnimalGrunt "Start"

Sound: AmbBirdC

Volume: 1

Play

Insert Delete

Waypoints

Insert Delete

Play sounds needs to extract all ways from the d3 hog into a custom wav folder. See settings dialog.

In the dial shown above, audio files can be installed in levels and thus also played (red arrows).

But before we can play the sounds we first have to do some preparatory work.

First we create a new folder .wav's should record.

For this you need approx. 70 MB!!! free space on the hard drive.

We now enter this folder in the Editor Settings dialog (right):

Editor Settings

Rendering

In textured world view:

- ☒ Display all rooms
- ☐ Display rooms to specified portal depth
- ☐ Display one room

Use Software Z Buffer

Render Shell Faces

Render Non-Shell Faces

Use Terrain LOD

Render Portal Faces

Render Floating Triggers

Render Face Triggers

Miscellaneous

Allow objects to be pushed through walls

Autoselect Mode On

AutoArrange Windows Mode On

Popup Message Alert Level

Concavity Toleranz

Scripting Configuration

Warning Level

Debug Info

Virtual Compiler Path:

Script Directory:

Descent 3(tm) Directory:

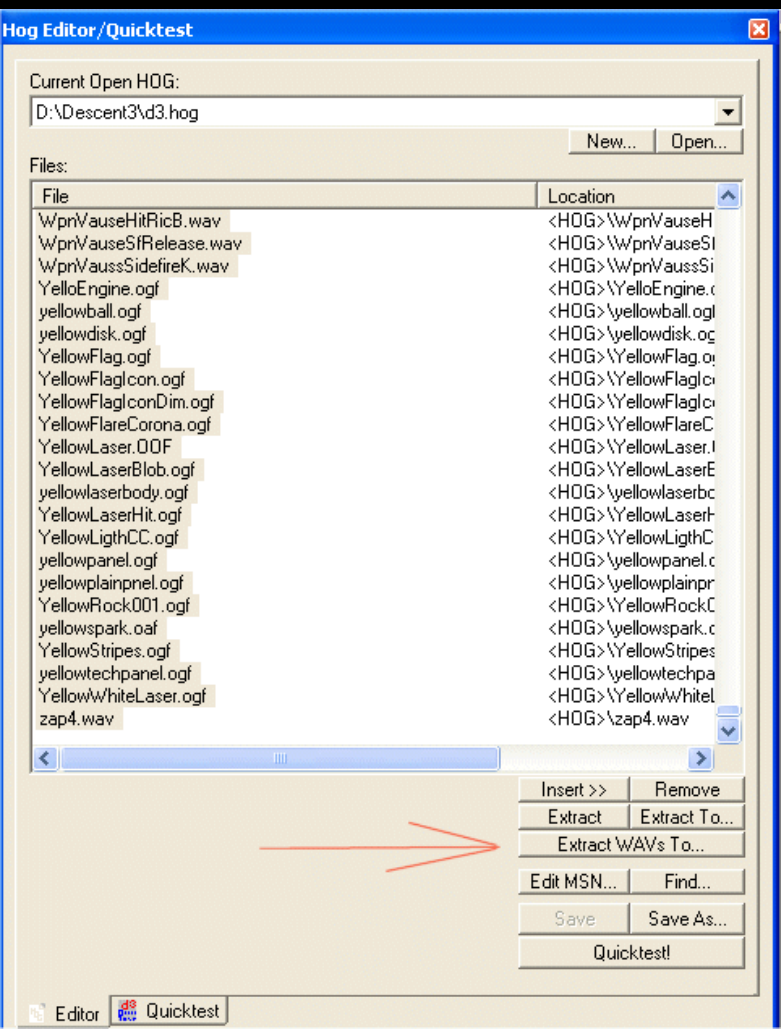
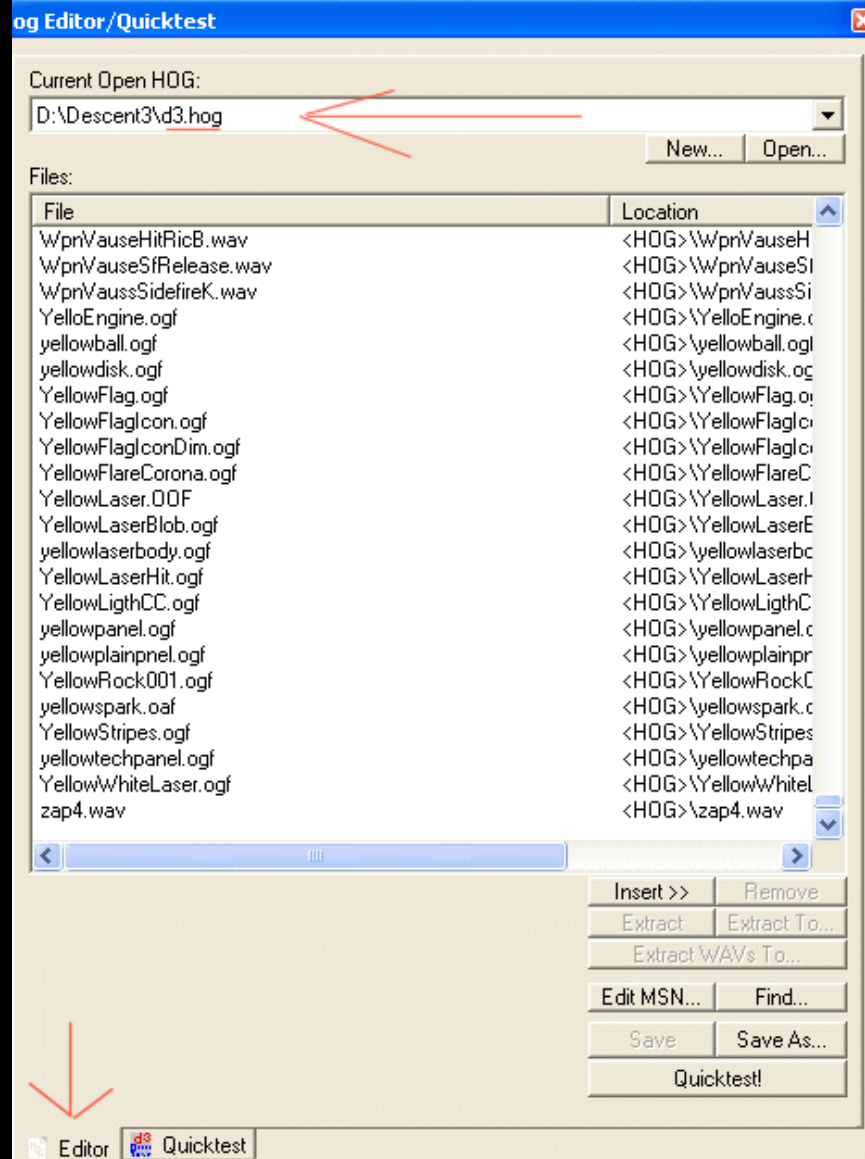
Tools Directory:

Sound Directory:

Play sounds needs to extract all ways from the d3 hog into a custom wav folder. Use Q dialog for extracting.

OK Cancel

We use the Quicktest to extract the wavs. So we start the editor and open the tab card in the quick test editor, and there the D3 hog file:



Then we mark (left mouse button) the lowest entry, pull the right scroll bar all the way up and mark while holding it down **Shift**-Key the top entry. Now it takes a lot of patience

Depending on the computer performance, it takes a very long time until all entries are marked automatically.

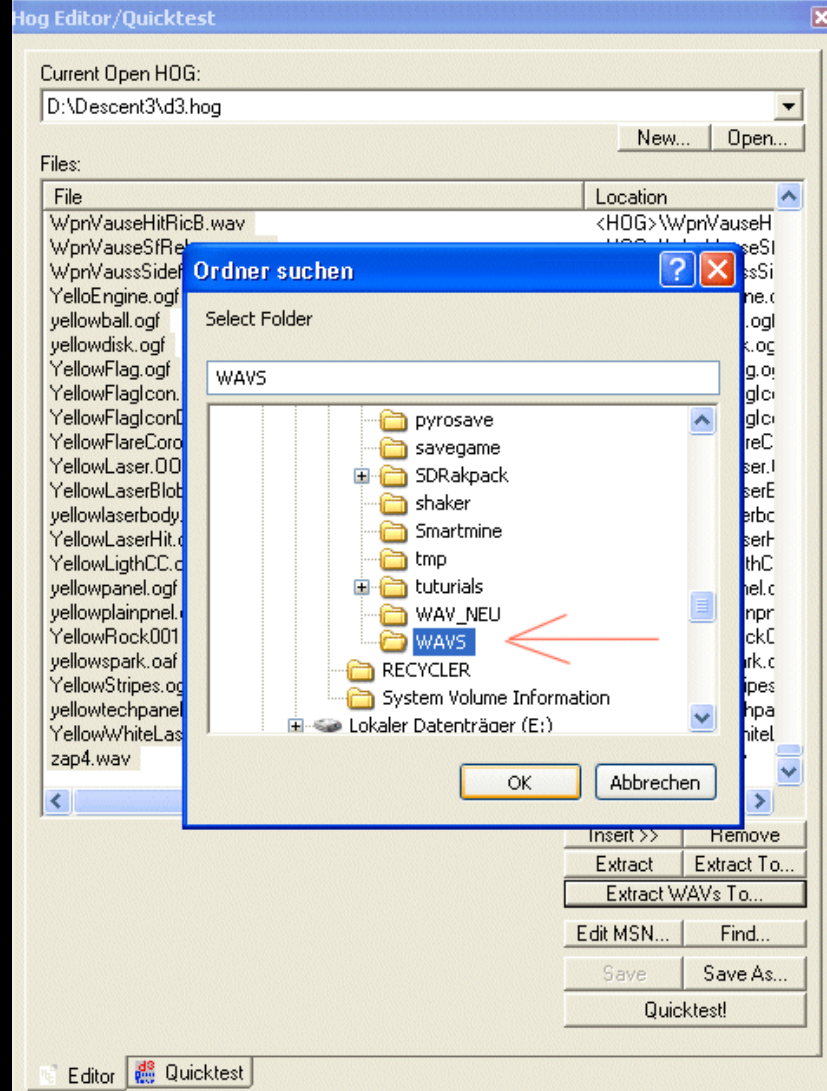
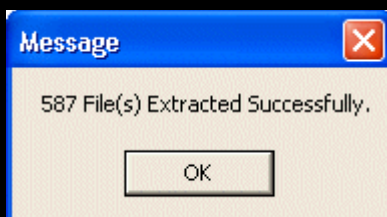
Alternatively, you can also go to Filetype sort, the .wav's are then last in the list and only mark those.

After all entries have been marked we click **Extract wav's To**.

A selection window opens and we select our desired WAV folder:

D3 Edit now extracts all wav's from the D3 HOG and copies them into our WAV folder.

After some time the job is done and should bring up a message dialog like this:



We only need to carry out this action once, then the sounds are available to us in DALLAS and thatRoom Properties-Dialogue always available.

Back to the overview of section G

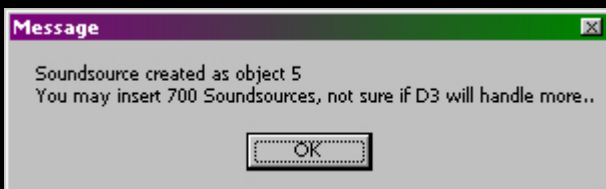
070 - Insert sound sources

Ragil Ral

A natural environment is riddled with noise. D3 is no different, there are fluttering fans, whirring force fields and gurgling lava. In the case of the ones just mentioned, it is the textures that have a sound as a property. There is another way to add sounds to the level, namely using the sound panel, shown on the right.

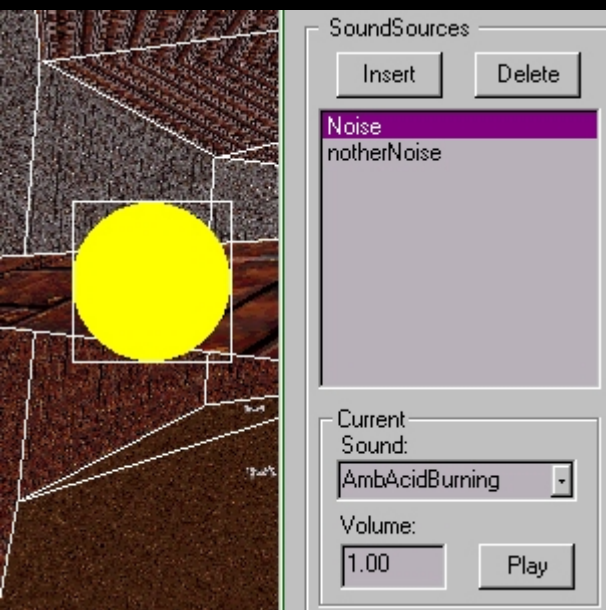
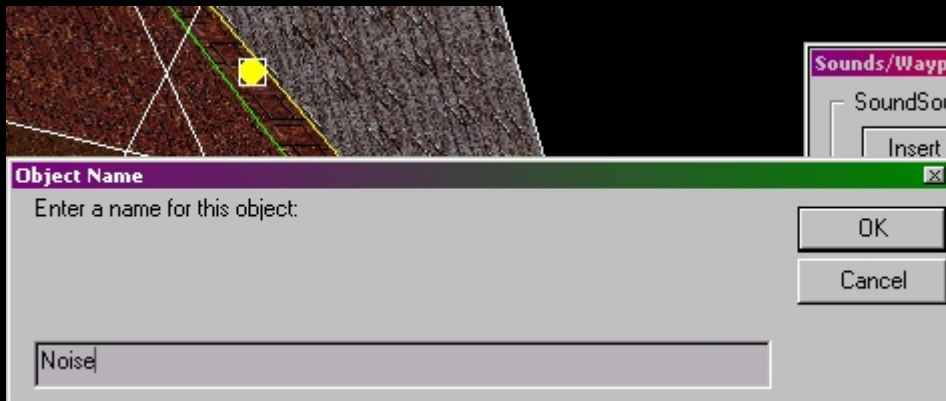
S

The whole story is very simple.



You click on a face in the room view and on in the sound panel **Insert**, whereupon you will receive a confirmation dialog (left)

You will then have the opportunity to name the source of the noise. The yellow dot is the inserted noise source, whose name now appears in the list.



Now double-click on the list entry and you can in the drop-down list box **Current** sound assign a sound as well as under **Volume** adjust its volume (left).

Of course, you can also make these settings before inserting a SoundSource.

To move the noise source switches to Object Mode (**Ctrl-G**) and moves it in one of the three 2d views **Shift cursor** to the desired location.

One more thing, the sounds are played starting at the start of the level. So if you want a permanent sound, the flag must be in the GAM entry of the sound loop be set, otherwise the sound will only be played once.

Thomas' tip is that if a sound cannot be heard through a portal, you should make sure that the 'Palmleaf1' texture is applied to all portal faces.

That's it, there's nothing more 🤖

[Back to the overview of section G](#)

071 - Terrain sounds

Ragil Ral

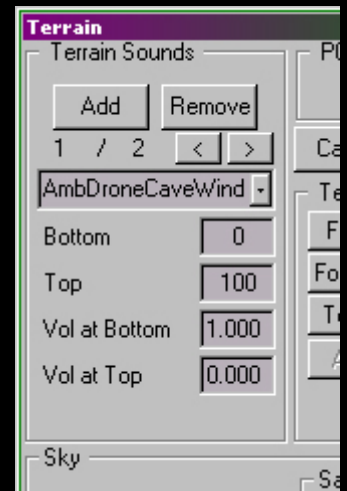
The **Terrain** Panel contains an area that allows you to assign one or more sounds to the terrain. They can then be heard across the entire terrain.

Just on **Add** Click and first select a sound, then set the range.

Bottom: At what height can the sound be heard, from 0-255.

Top: At what height can the sound be heard, from 0-255.

Vol at Bottom/Vol at Top: the volume at the respective trebles, values from 0-1.



If you want a smooth transition, you have to insert the same sound twice, swapping the volumes for top and bottom once:



This ensures that the sound initially becomes louder and then quieter again as you move higher.

The elevation 255 indicates the top of the terrain and is 350 units away from the terrain elevation 0.

072 - Sound sources on the terrain

Ragil Ral

This chapter was created at the suggestion of a thread on descentforum.de:
<http://www.descentforum.de/forum/viewtopic.php?t=4442>

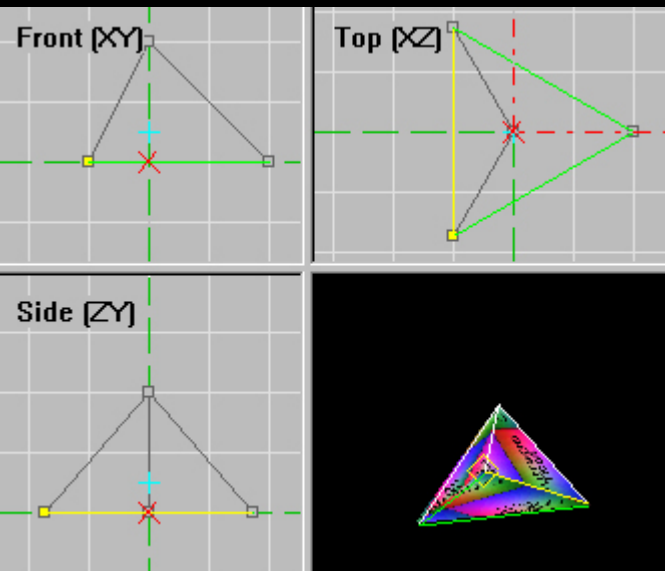
Problem statement

Sound sources on the terrain are a little problematic. They can only be used via Roomview; You can therefore place sound sources in external rooms, but you won't hear them. The same goes for textures that carry a sound; They also remain silent on the terrain - in an external space.

A solution

The simplest workaround is to take an internal room, put the sound source there and connect all faces of the internal room to the outside world.

Practice



First create the simplest possible construct, a cube or, better yet, a tetrahedron.

Build this construct as an internal space and move it to the location in the terrain where the sound should go.

My approach is as follows:

A) insert a room anywhere in the level (**Into the** in the world view),

b) I copy the figure into this new space, **C)** then I delete the portal of the newly inserted room,

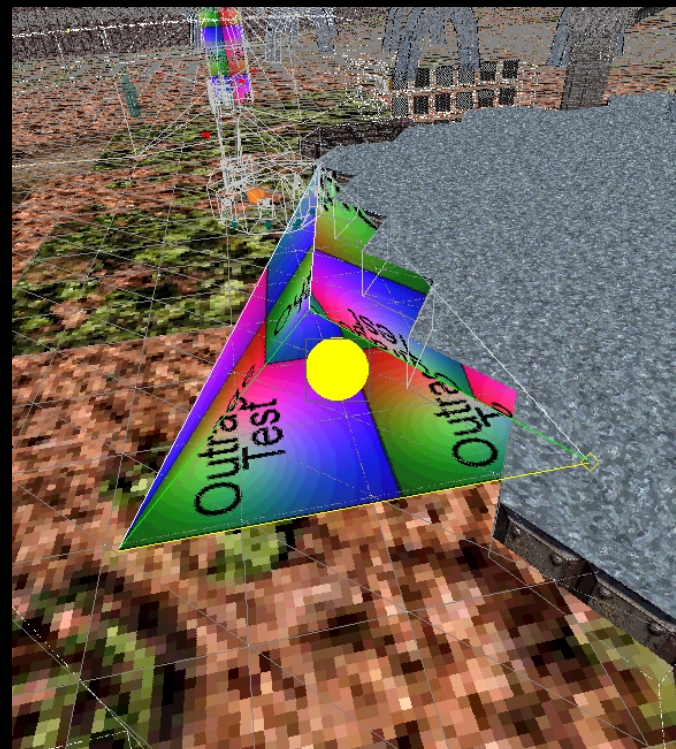
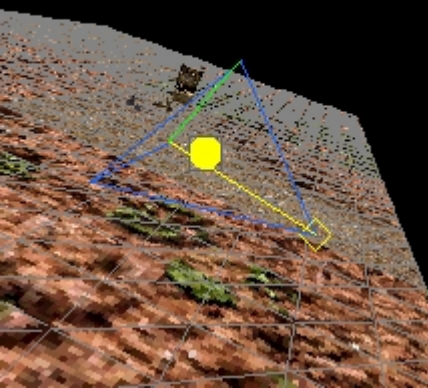
D) then you can delete the other faces. **E)** Then position the remaining construct there

where the sound source should be,

F) finally place a sound source in the room (right),

G) then connect all faces of the room to the outside world, making sure beforehand that the rainbow texture is on it.

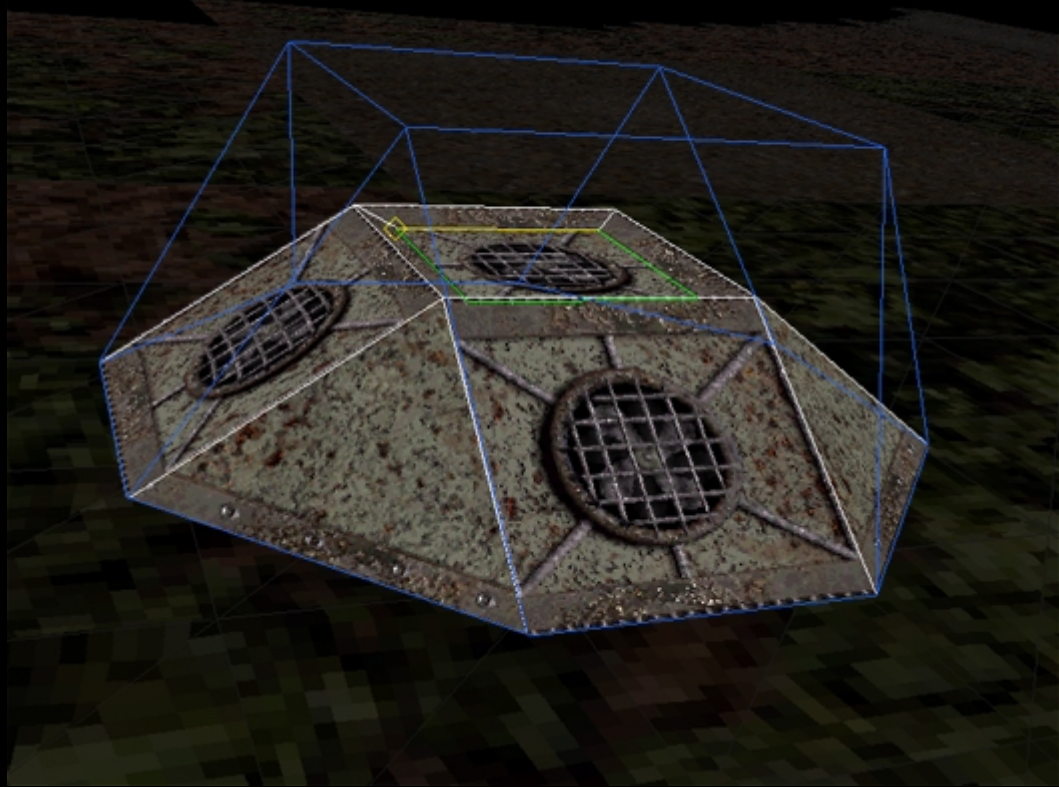
What remains is an invisible but internal space (left) and the sound is audible.



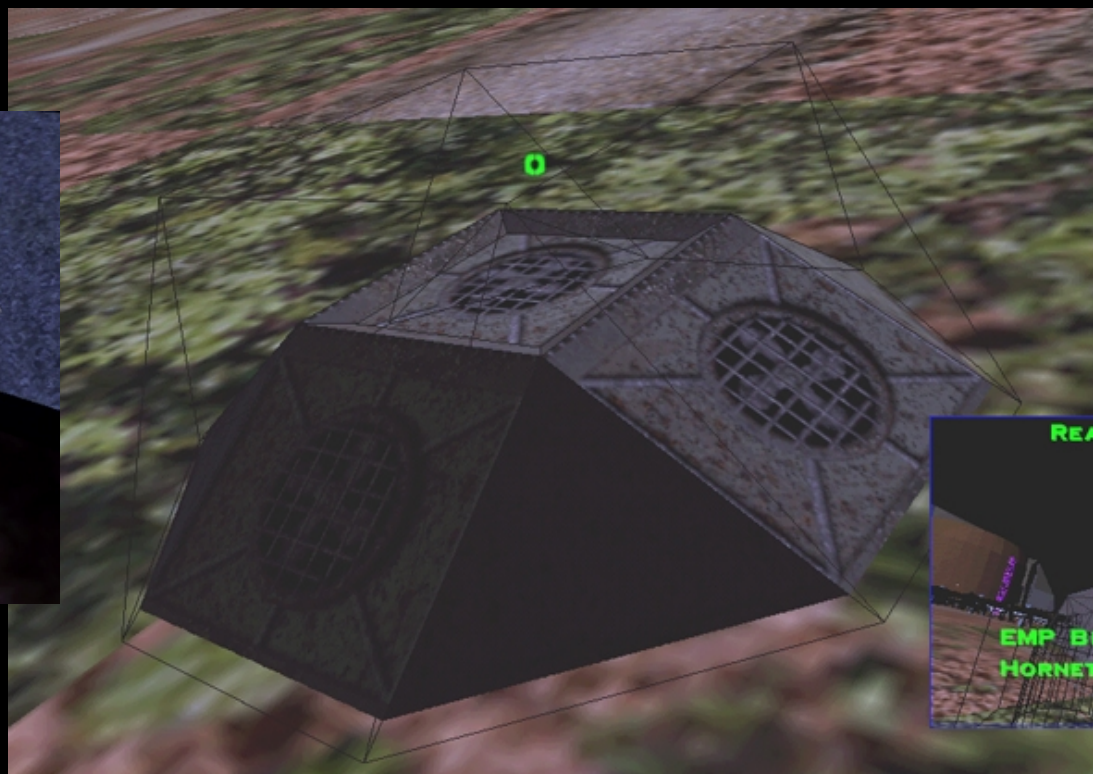
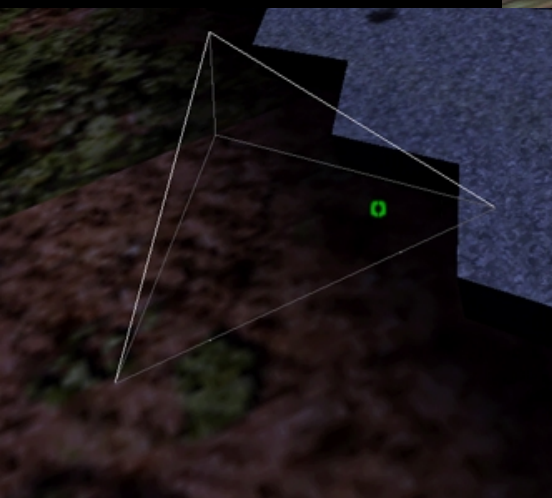
You can do a similar thing with external rooms that have textures with sound.

In this example a ventilation outlet that has sound-assigned textures:

This was initially an internal space
The marked faces are those that were then used to link to the outside world.



Here are some more screenshots
Descent 3, withoutlinem:



Here too, Thomas'

Tip that the portal faces the

The 'Palmleaf1' texture must be worn if the sound is to be heard beyond the room, but this does not apply to the portals that connect to the outside world, which must still retain the rainbow texture.

This method always works.

However, strange things can still arise when working with sounds. If in doubt, you have to try a little.

[Back to the overview of section G](#)

073 - Music in your custom Descent 3 level

Toxxeon

This guide explains how to insert sound files into your custom D3 levels.

In order to integrate our custom sound into the level, we need the program *MusicTester*. If you have the original D3 editor (D3edit v1.1 and later) installed, then this program should be in the *editor*-Folder located or via the *Tool* menu can be accessed.

Furthermore, the *Audio Taunt Manager* Highly recommended by Alex "LEX" Leuthold. This makes converting any .wav files child's play... You can get this ingenious program either on Fischlein's tool page (which also includes the "Original Editor" package *MusicTester*...) or download directly from LEX.

Short overview:

In order to insert your custom sound into a D3 level, you have to use the original .wav file into one .osf-Convert file (Outrage Sound Format). In addition, there must be one more .omf file (Outrage Music File) can be created. This file is used to control the sound in the level.

First of all, you obviously need an appropriate music sequence. It's best to have an endless loop so that the players don't have to hear an annoying "break" or something like that. Because things like that get really annoying over time! Also remember that an average of 22.5 Kbytes of sound is generated per second. So for a piece of music that's 1 minute long, that's about 1.3 MB of data. (in the unzipped level)

Important!
Be sure to note any
Copyrights of the
music you use

or similar!

Preferably, you should ask the creators whether you can use their music in your level. Or you can create your own with appropriate music programs or an instrument...

With this all out of the way, we can now start actually creating the level sound. The sound file you have selected must first be "intermediately converted". Namely in the 8-bit mono .wav format.

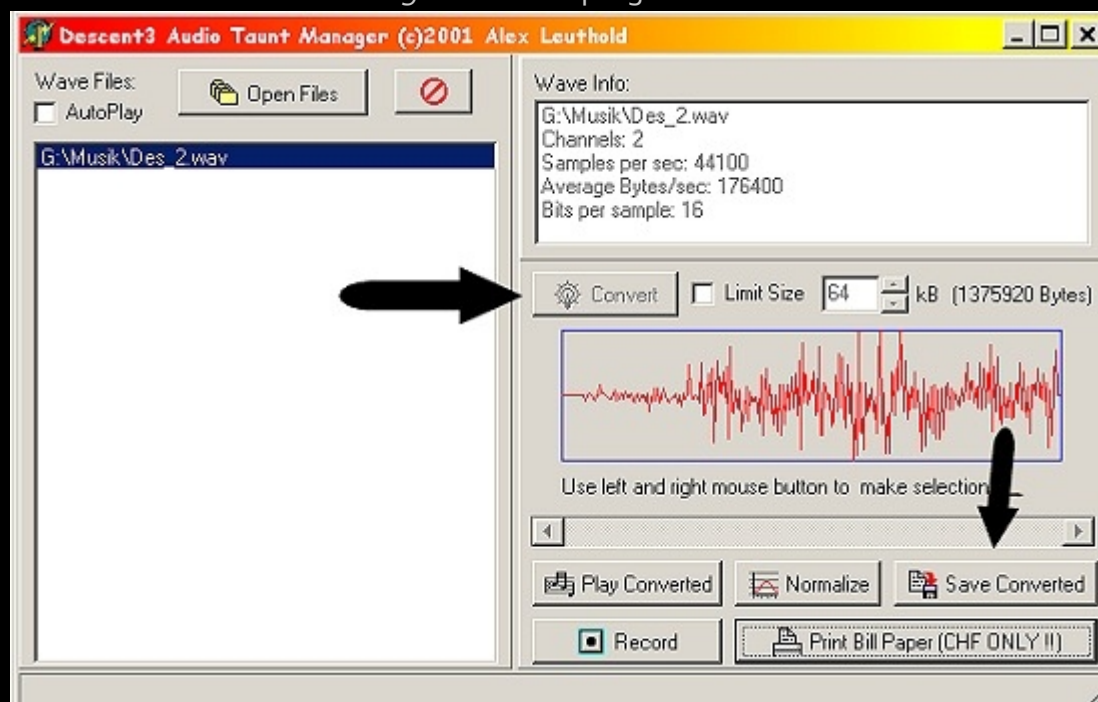
The easiest way to do this is with this *Audio Taunt Manager*. Start the program and load

by means of **Open Files**
your sound file into it.

Now click on **Convert** to save the sound file convert. You can't and don't need anything further set, the correct format will be selected straight away.

By the way, that one

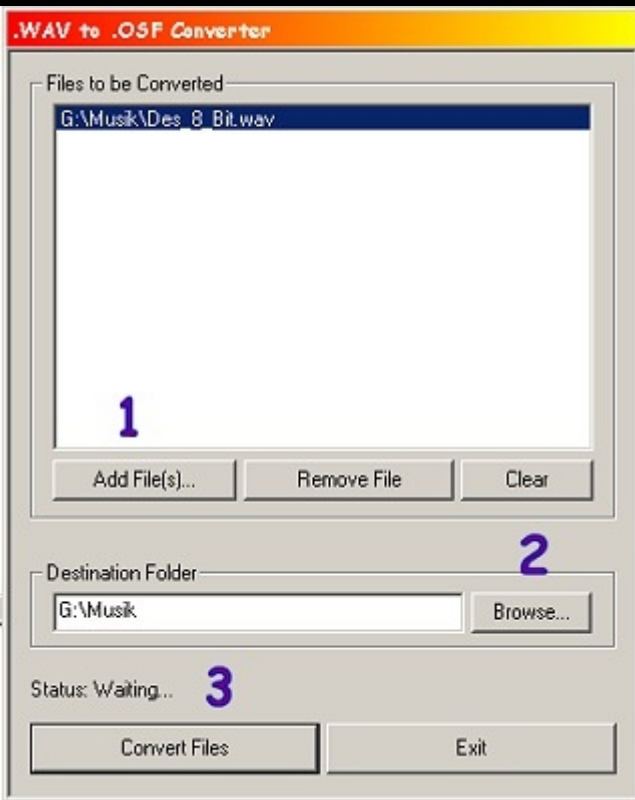
AttitudeLimit size is important if you want to create your special audio mockery 😊
These sound files namely not allowed be larger than 64 Kbytes!



Play Converted You should definitely use it because sometimes the converted sound sounds completely different than the "original" because it has to be the quite economical 8-bit mono format. The option **Record** is there to record something using a microphone, or you can play a CD or an MP3 file... and "record" the sound at the same time *Taunt manager* in the correct format. **But be careful: As already mentioned, be sure to respect the copyrights of others!** It is advisable to use the converted sound **Normalize'n...** To ultimately save the whole thing... **Save Converted** to use.

Now we have to further process the sound...

This happens in **Music Tester**, a program that is "included" with the original D3 editor.



Now start it **Music Tester** and convert your 8-bit sound to that .osf-Format. Proceed as follows: Click on **Actions** and then up **Convert Files...** It now opens **WAV to .OSF Converter** (Left).

- 1.) Select the corresponding file(s).
- 2.) Find a storage location for the OSF file
- 3.) Convert sound.

If there is no error message, everything is OK. By the way, converting can take a while depending on how large the file is... Closes with **Exit** the converter.

I recommend that you make an important setting now: Click on **Test Theme File...**, in the new window that appears **Options** and then up **Edit File Search Path...** Now on **Add Path**, and find the location of your .osf files out. If those

Once the path mapping is done, you can later listen to your sound here in the Music Tester, and the OMF file will be correctly compiled by the program.

Now let's create it **OMF**-File.

First of all, to understand: This OMF file is a kind of script that controls which OSF file should be played, how often and for which event. It may sound complicated, but in principle it isn't 😊

Just do the following...

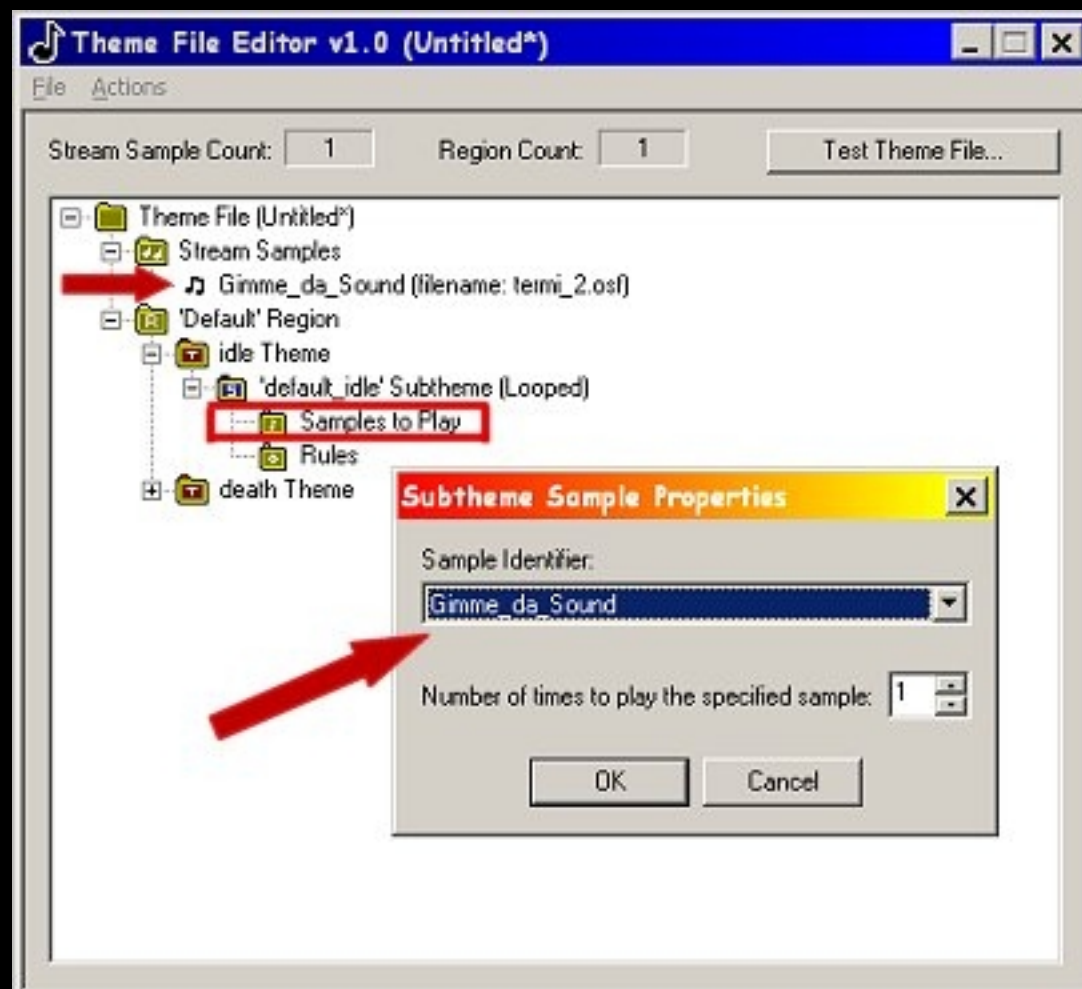
Click with the *right* Mouse button up **Stream samples** and then back to normal **Add New Stream Sample...** A new dialog window opens. In this you give a unique name in the upper part and then search **Browse** your corresponding OSF file.

Next, click on the plus sign **Default region**, then click on that again **idle theme** and last but not least again **'default_idle' Subtheme (Looped)**. And now again with that *right* Mouse button up **Samples to Play** click.





In the dialog window that appears, the name you just assigned should appear in the pull-down menu. See image below:



Of course you can too multiple OSF files simultaneously integrated into the OMF file become. How nice described above **Stream samples**. For example, you can have one *Intro*, one *Background music* and a *death theme* agree at the same time. There is only "room" for one in the level one OMF file...



A little trick:

If you want that when entering the level *Welcome* sound is played, and then the background music "normally", then proceed as follows:

First of all, two OSF files **Stream samples** Integrate, then right-click **idle theme**, and now up **Add new Subtopic....** Enter a distinctive name in the dialog box (e.g. background), and click on **OK** click

At **'default_idle' subtheme** specify the intro music file **'Background' subtheme** the corresponding background music. Now right click on **'default_idle' Subtheme (Looped)**, then up **Edit....** The tick at **Enable continuous playback of this subtheme** remove.

Overall it should look like this:

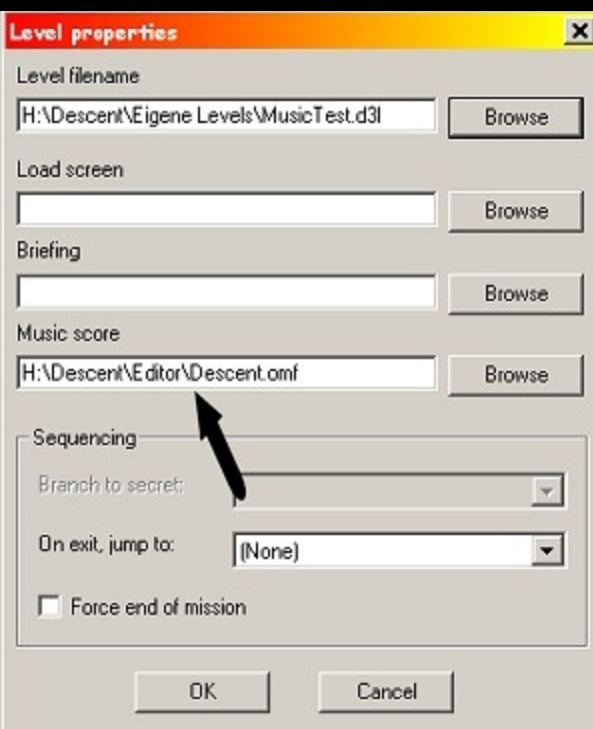
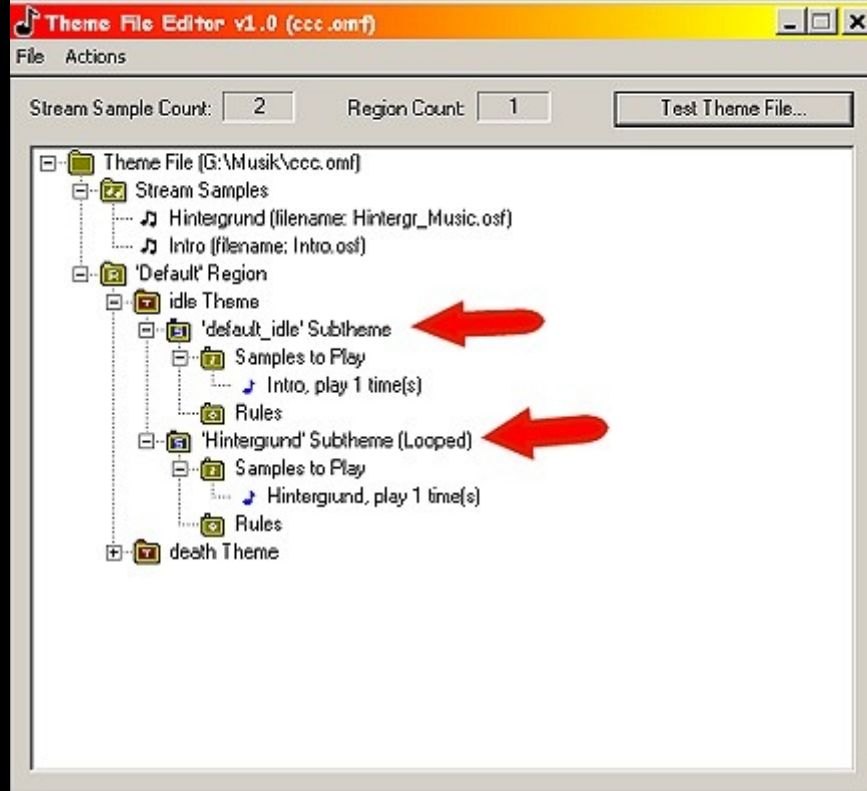
Strangely, the very first time the player enters the intro sound is played twice, the next time the player enters (after being shot) it only plays once... But give it a try.

Save now! Here is another note from me:

If you are under File --> Save as When saving the OMF file, I have found that you also have to write down the file extension. So not just as usual, for example Level music, but stop Level music.omf. Then it should save the program correctly.

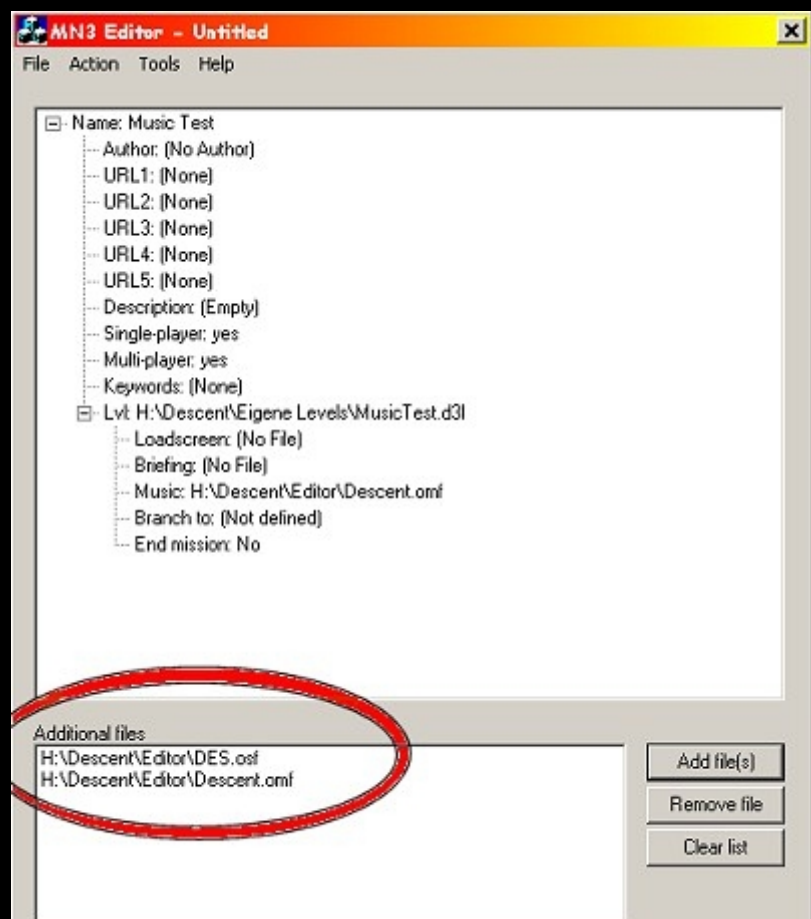
Now you can listen to your music. To do this, click on the button **Test Theme File...**

An interesting thing is **death theme**. Here you can set an alternative sound that will be played when a player is shot down in the level. The sub-point **Rules** appears to me in *Music Tester*. Unfortunately it's a bit half-finished, there you may have the opportunity to refine the playback of the sound according to certain criteria. But you'll have to test that out yourself...



Finally add the sound to the level. This happens in *MN3 packer* (mn3edit.exe), i.e. when putting together the finished level.

Both *Level properties* (left) below *Music score* the one created .omf-File indicate.



Now all the relevant ones .osf-and .omf- files under *Additional files* indicate. Then everything is done. 😊

Have fun setting your levels to music!

Back to the overview of section G

074 - Sound: Avoid problems

Atan

Problem:

A shot at an opposite wall produces no sound when it hits. The cause lies in the level design and can be remedied by paying attention to a few points or avoided when creating the level.

A few basics:

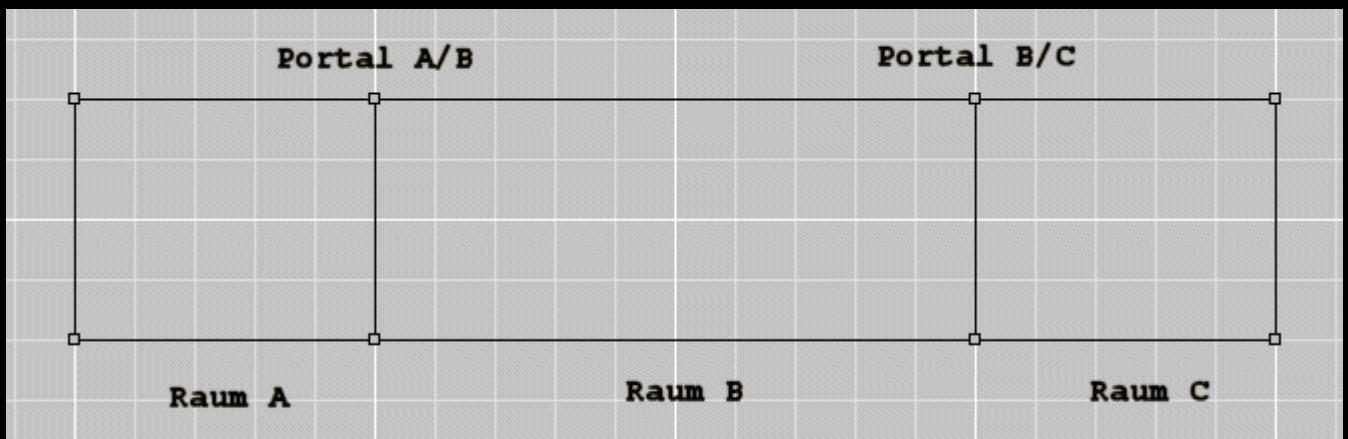
While creating the BOA Vis, D3Edit also calculates the sound distances within the level and enters them into a table. This table is saved with the level and used by Descent3 for sound generation. If the maximum distance of 400 units is exceeded, this distance will be declared invalid in the sound table. Descent3 does not produce a hit sound if the distance is invalid.

How is this sound distance table created?

Each room is checked to determine which room can be seen. The distances between the determined start and end rooms are added together and (if valid) entered into the sound table.

How does 'seeing' work from room to room?

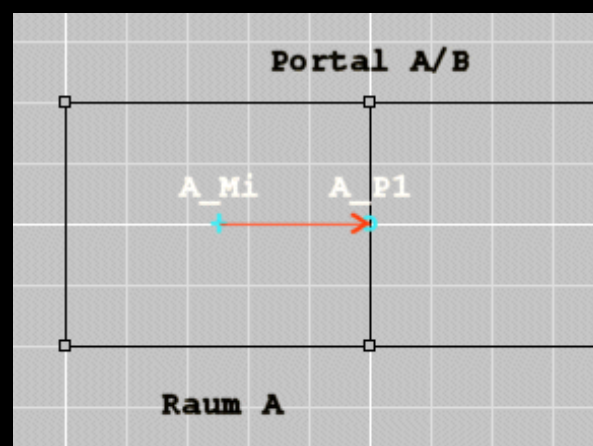
Our example level should consist of 3 rooms in a row (A, B, C) through portals



How is the room-to-room distance determined?

Let's start D3Edit with room A as the starting room. Based on the table, the distance from the center of the room to the center of room C is now determined. To do this, the distance (Dist) between the room A center and the first portal of this room.

The distance AC is set to 0. Then a straight line is drawn from the room center to the center of the portal face A/B



If this line does not reach the portal face, this distance is invalid:

Dist AC = invalid.

If the line reaches the portal face, this distance is temporarily saved:

Dist AC = Dist AC + (Dist A_Mi -> A_P1)

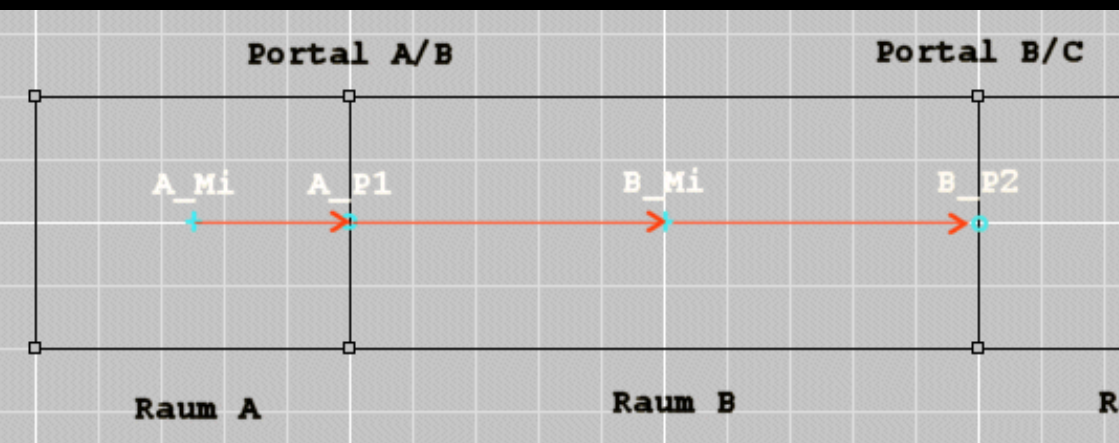
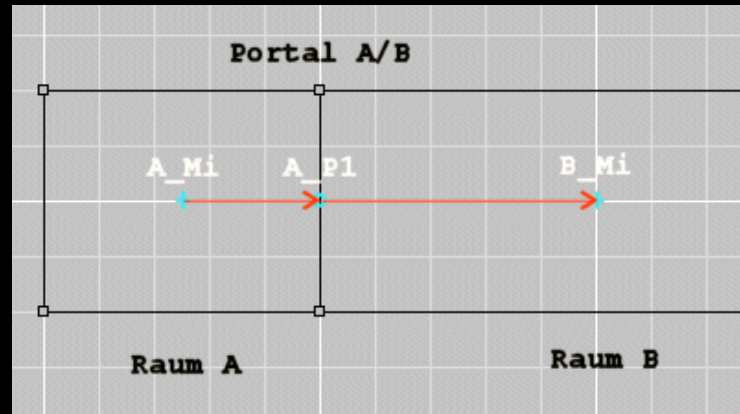
Now there is a straight line from the portal center to

If this line does not reach the center of space B, the distance is invalid.

Dist AC = invalid.

If successful, the distance is added to the one already saved:

Dist AC = Dist AC + (Dist A_P1 -> B_Mi)



Now the portal

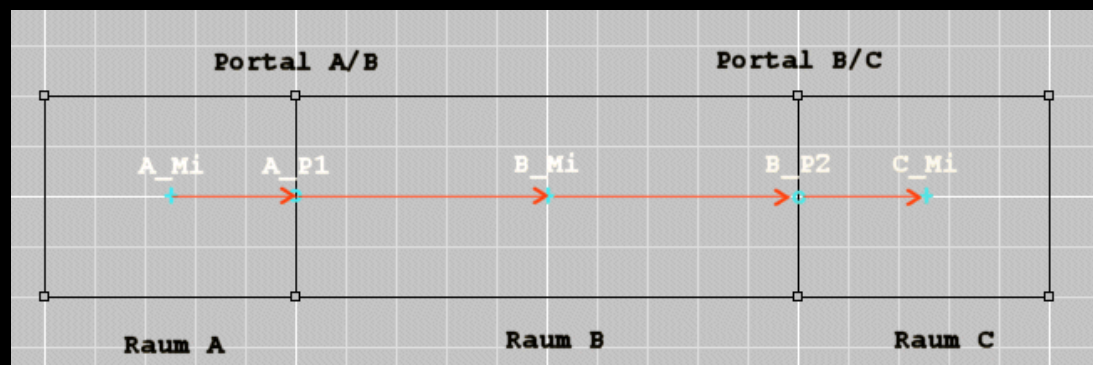
This line reaches the room C center not, the removal is invalid.

Dist AC = invalid

If successful, the distance is added to the one already saved:

Dist AC = Dist AC +

Now D3Edit checked



If this is the case, distance AC is declared invalid. Otherwise, the determined distance is now entered in the Max Sound Distance table under A/C. Room A is fully calculated because it only has one portal.

D3Edit switches to room B and, as described under room A, determines the distances to room A and room C. After room C has also been edited, the list of max sound distances is ready. It is part of the .d3l-File and saved or loaded together with the level.

The problem...

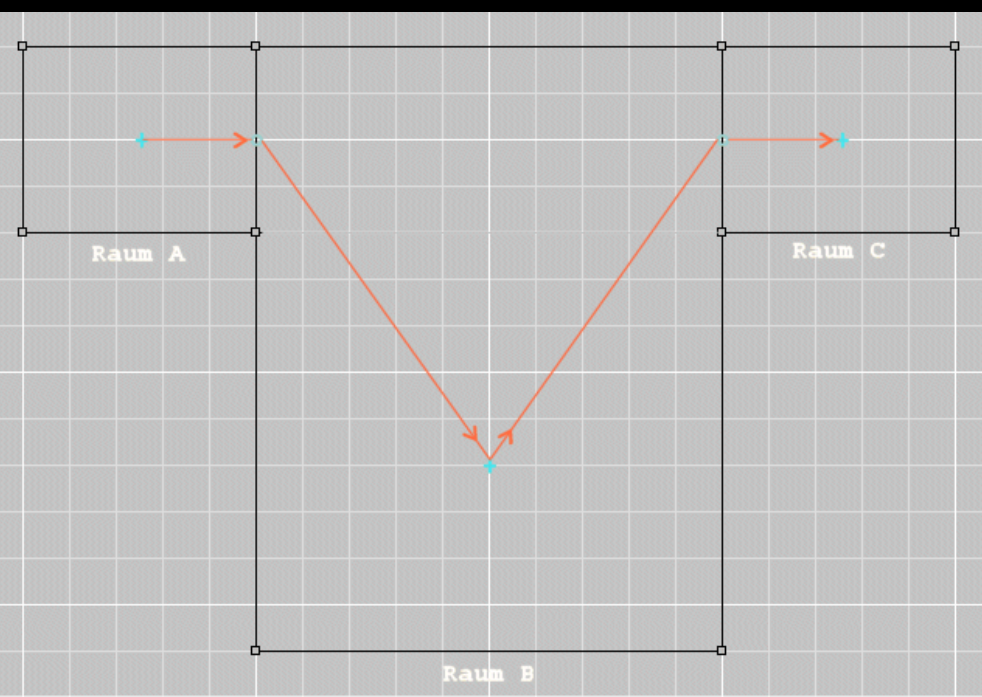
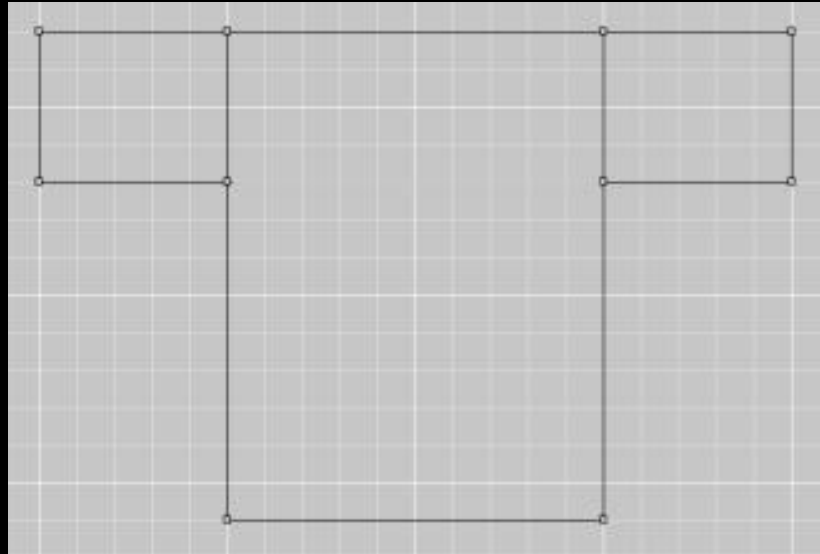
In the previous example, if a laser shot from room A to room C, you would be able to hear the impact in room C, provided the distance

Let's take a look at this level:

Here too we shoot from room A to room C, but we don't hear any impact in room C. However, another test shot can be heard directly in front of portal B/C. If we fly into room B and shoot again into room C, the impact can be heard. Now we fire from room C into room A, but even now we don't hear any impact, a shot directly in front of portal B/A is ok again. If we fly from room C to room B and shoot into room A, it works too... The rooms A and C are close to each other, what's going on?

We remember that with the MD Zoom we can make a distance measurement. So we take a look and lo and behold, (for example) 250 units are displayed, i.e. a distance that is well below the magical 400 limit. But why don't we hear any sound?

The solution is relatively simple if you are somewhat familiar with the above procedure. Let's take a look in the picture at how the distances are determined in this level: I deliberately drew the room center of room B a little further down here to highlight the problem more clearly.



board to the portals

calculates. That's what lies here too
hon the problem, because
the direct air line between
room A and room C

If the example only has 250
units, D3Edit determines more
than twice the distance as can
easily be seen from the image.
But that means we are well
above that
casual sound distance and
3 plays, when shooting
from room A into room C,
and vice versa).
impact sound.

Remedy for this problem...?

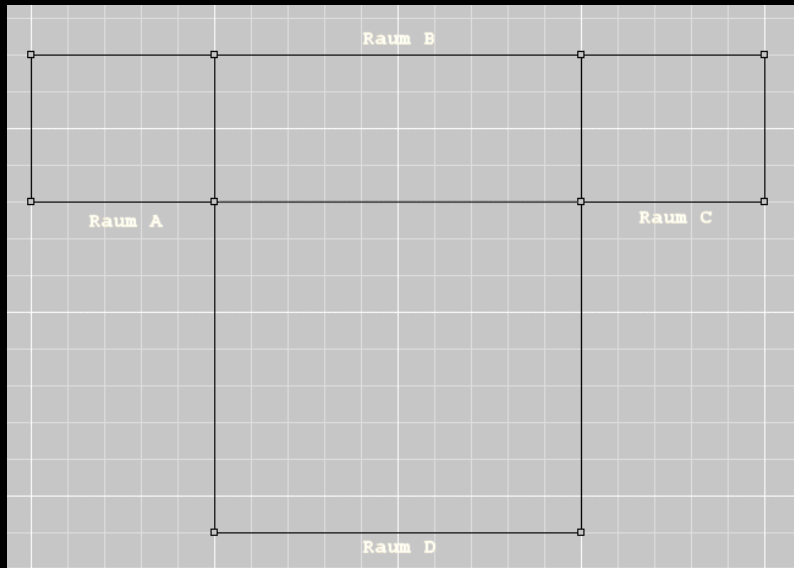
As long as D3Edit performs the calculation in this way, we can only change it

level design. Here it is easiest and create another room.

The following image shows the changed level:

The view table created by D3Edit now looks like this:

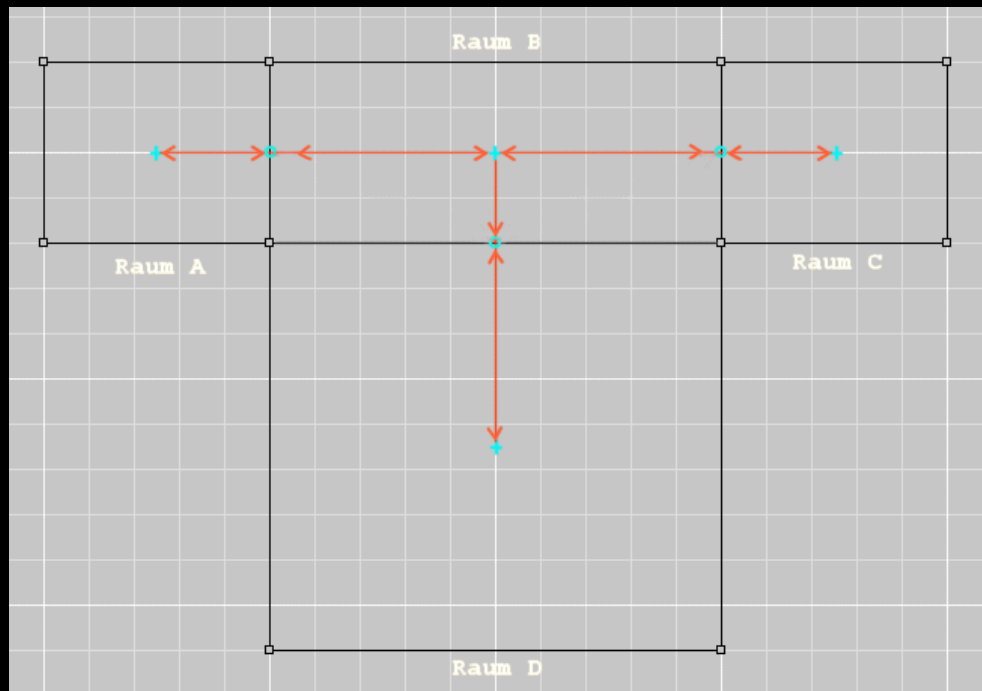
Start room	End space
A	C
A	D
b	A
b	C
b	D
C	A
C	D



In this level, if you shot from room A to room C (and vice versa), you would now hear the impact. In the game itself you wouldn't be able to notice any difference in the level design, the level looks exactly the same as the previous version.

The next picture explains why

You can see that the distance from room A to room C now corresponds to the straight line and is therefore safely within the permitted range.



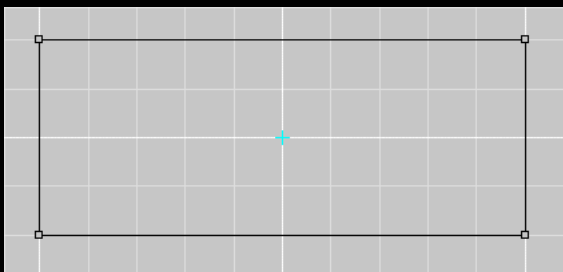
Another Problem...

One feature in D3Edit that is almost never used even though it could prevent sound problems is the View Room Center feature. Is she When switched on, a small cross appears in the middle of the room in the Room View. You can see where it is in all 3 ortho views

Center of gravity is located.

Why am I calling it all of a sudden?

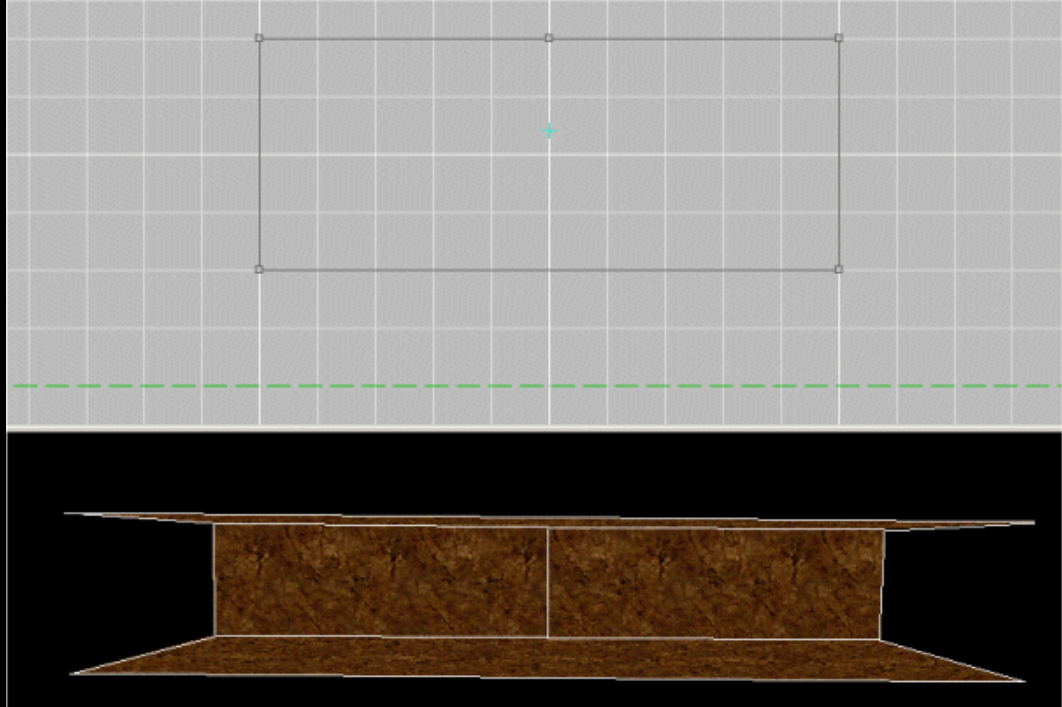
Center of gravity and not room center or room center...? Let's look at another example (top view):



As expected, the Room Center is in the middle of the room.

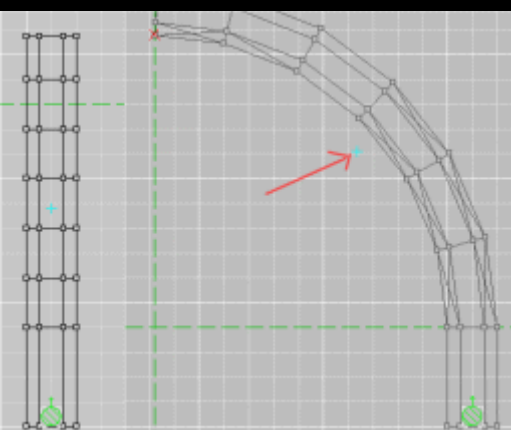
Now let's split a face of the room:

Although the room still has the same shape, the room center shifts towards the new verts. The Calculation of the room Centers is more of a weighting of the verts. A face with lots of verts magically attracts the room center. This property can create sound errors but can also help to fix sound errors.



Example:

After bending, the room center shifted. The weighting calculation determines a center of gravity **outside** of the tunnel (see red arrow). This has serious consequences for the sound calculation. Let's assume that the curved space is room C in our

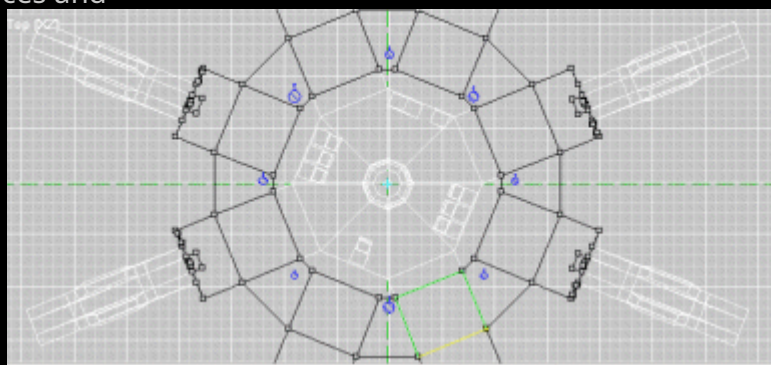


be the example above. Then the line would run from room A through room B to portal B/C, but the (line of sight) between portal B/C and room center C (C_Mi) is interrupted. D3Edit would declare the sound connection between room A and room C invalid, the result would be that there will be no hit sound as described above. A solution would be to make two or more rooms out of the arch, or, if the distance is not too large, to increase the weighting of the verts on the opposite side of the wall. It is also conceivable by sticking a monitor on which you can then make part of the room using the Merge Object into Room function. The new faces and Verts pull the

The focus may be so far back into the room that the Room Center can be seen from the portals. This means the sound works again.

Here's another nice example of what not to do:

The center of the room is perfectly in the middle, but unfortunately in the wrong room 😊

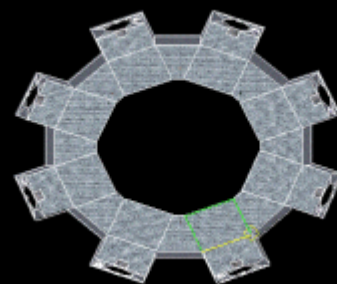


Portals:

The newer D3Edit versions now take into account that the portal textures have to be transparent if you don't want to have sound problems in the level. The Palmleaf1 texture is automatically set to portals.

This is the only D3Texture that is available for the above

Line of sight is permeable and thus enables correct sound calculation by D3Edit. If you want to cut the sound, you just have to put a different texture on a portal; you will no longer hear the hit sound behind this portal. Using the Pamleaf1 texture also brings advantages for the BNodes.



Back to the overview of section G

Anticipation:GETTING DALLAS WORKING

Although scripting is actually a later topic, Dallas is needed from here on out. Therefore, here are the instructions on how to install it, set it up and get it running.

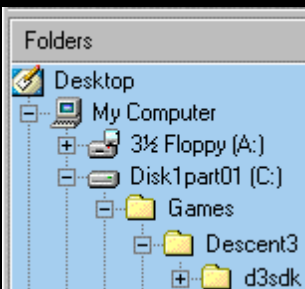
It is Fischlein's translation of a manual from Papacat.



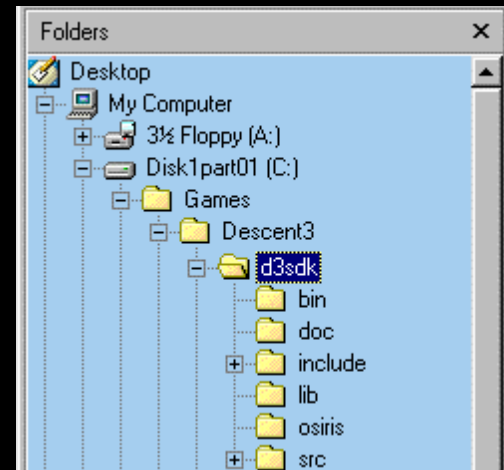
I'm not repeating word for word what Papacat from Gameedit is in English wrote !

All images I use here are from Gameedit

First you need the file `d3-sdk14.zip` (You should look for the current version: http://descent3.com/4_downloads.html), which is installed as follows.
First you create a directory (left).



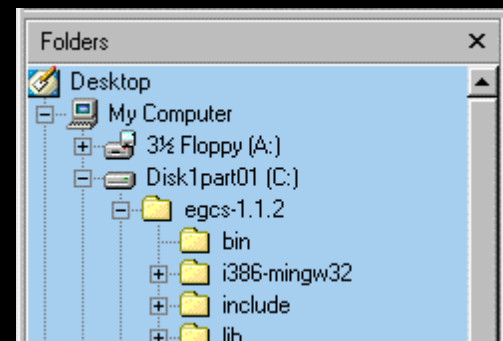
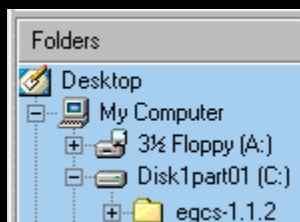
Now extract the file `d3-sdk14.zip` with directory structure (right).



Next you need **EGCS-1.1.2**. If necessary, google 'egcs-1.1.2-mingw32'. This is actually for people who are in C/C++ programming, but since Dallas generates C-like code, you need this to turn your code into a DLL file (=compile).

Creates a directory "egcs-1.1.2" (left).

You also unzip this `egcs-1.1.2-mingw32.zip` with the directory structure (right).



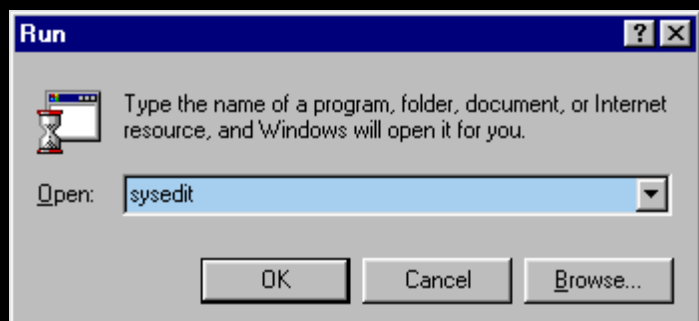
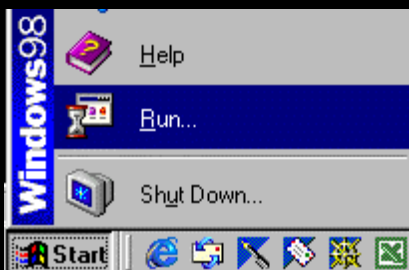
Now you have to add the following instructions to your `Autoexec.bat` insert:

```
"SET PATH=C:\EGCS-1.1.2\BIN;%PATH%"
```

then save the file and restart.

For those who haven't tinkered with the startup files yet:

In the German version Carry out...when you open it you write "sysedit" pure and **OK** click



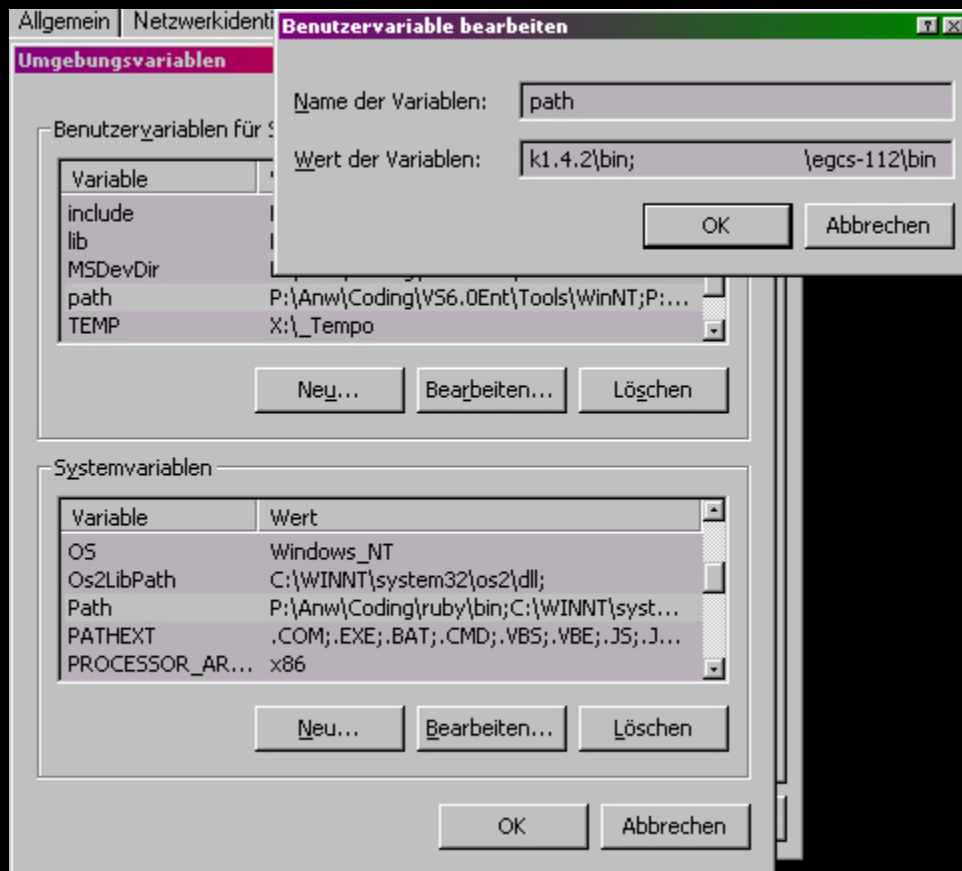


insert the above statement, then File -> Save. Now restart, done.

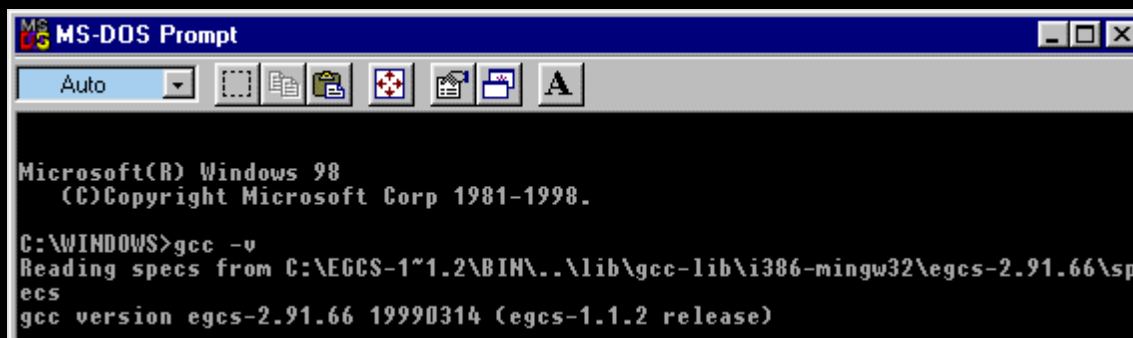
Alternatively, you can also do the following:

My Computer, right click -> Properties. Select the Advanced tab there and the button there **Environment variables:**

Mark the entries with a lighter background on the right and click on **Edit...** then add the path to the compiler there, for both. Your path goes where there is nothing in the screenshot. Don't forget the semicolon! 😊



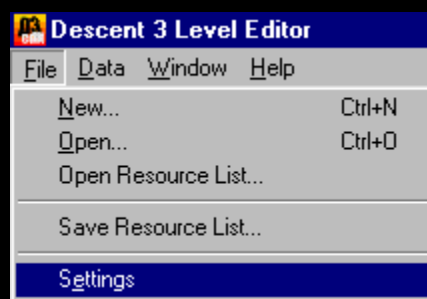
After you restart, a Dosbox or "MS-DOS command prompt" opens. Now you enter the following: "gcc -v". Between **GCC** and **-v** must be a space! After entering this you should see the following:



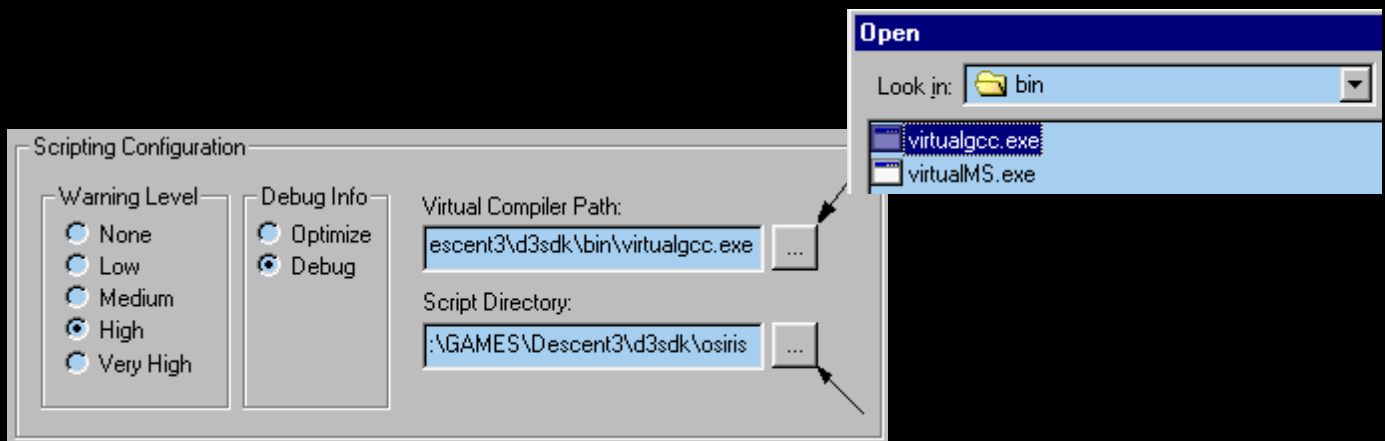
so if the GCC Version number is displayed it installed correctly.

Now open D3Edit, here we also have to make a few settings.

Call up the configuration:



At the bottom part, "Scripting configuration", you have to specify the path to the compiler and the directory in which you want to save the scripts and DLLs. With "Virtual Compiler Path" You have to register `virtualgcc.exe`. At "Script Directory" the directory `osiris`, as in the picture.



So there you have it, now you can incorporate teleporters, stargates and other effects into your levels.

Section H–Doors & Objects

While doors are something you can easily use in level building, they are also objects you can create yourself; The thematic transition is so fluid that both are dealt with here.

In addition, from here on it makes sense to have worked through the section about sounds in the game.

Furthermore, from here on: with 'GAM' is the one in the.gam-File registered storage page meant. See also the specification file at the end (page 518) of this file.

Doors				
075	Doors tutorial	How to install doors in levels.	Kyouryuu	285
076	Own doors	How to build, create and integrate doors yourself.	Toxxeon	288
077	Sound for your own door	The corresponding ones for a self-made door Miss sounds.	Ragil Ral	294
Objects				
078	Own objects 1	Create & integrate your own objects	Fischlein	295
079	Own objects 2	Deepening into GamFile & Objects with Dallas "Breathe life into.	Fischlein	297
080	Own objects 3	Add gun points to the object & the script <small>in addition.</small>	Fischlein	301
081	Custom Robots Part 1	So that the object can defend itself	Papacat	305
= Papacat's Custom Weapon Workshop =				
Very comprehensive walkthrough for creating and incorporating a home-made primary weapon, from which Theory to action!				
082	Introduction & Overview	Overview of the workflow	Papacat	308
083	The Powerup GAM	Explanation of the GAM page for powerups		310
084	The Weapon GAM	Settings in the weapon GAM		311
085	A simple primary weapon	Integrating the weapon		315
086	The ship's GAM	Ship GAM settings		319
= Papacat's OOFEdit digression =				
A detailed explanation of what you can do with OOFEdit.				
	• OOFEdit digression	An overview of what, why & how	Papacat	321
087	OOFEdit: Terminology	The most important terms		323
088	OOFEdit: Features	What can OOFEdit do?		324
089	OOFEdit: Getting started	Operation of the program		325
090	OOFEdit: A powerup	A powerup.oofmake		328
091	OOFEdit: Turrets	Create a battle tower		330
092	OOFEdit: Animation	Basics of animation		333
093	Ship tutorial	Adding a custom ship	Darkside Heartless	337

075 - Doors Tutorial <>

Kyouryuu

In this tut you will learn how to add doors to your Descent3 level. Here you have to understand spaces and know how to add them.

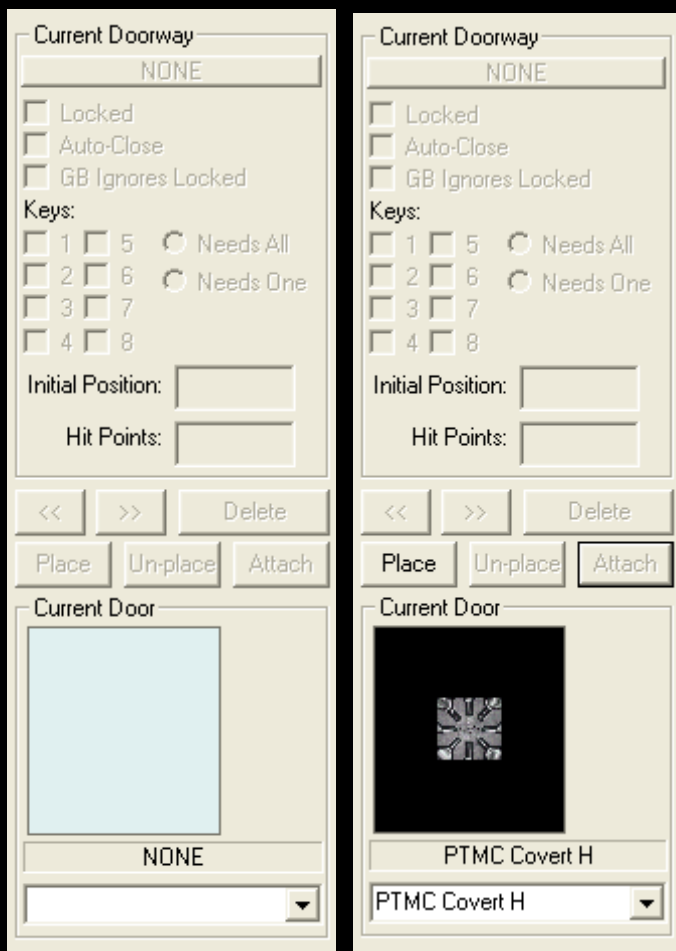
Introduction


Doors in Descent 3 are essential for single-player missions. They are great for separating areas and preventing the player from getting to certain areas prematurely. Doors are usually not used in multiplayer levels. Many players complain about the lag caused by doors, since doors are actually considered an object in the level, just like powerups and the ships themselves.

To understand this, we have to remember that in Descent 1 and 2, doors were completely flat polygons with an animated texture on them that looked like a door. When a player shot at the door, the game would play a predefined series of animation clips that resulted in the door 'opening' and allowing the player to pass.

Descent 3 is different. Here the doors are completely 3D, just like all the other level architecture is. The pieces of the door even retract into cracks when it opens (think of an elevator in an office building). Specifically, a door in D3 contains a special room (the frame around the door) and a door object. When you use a door, you are actually using both.

The door panel

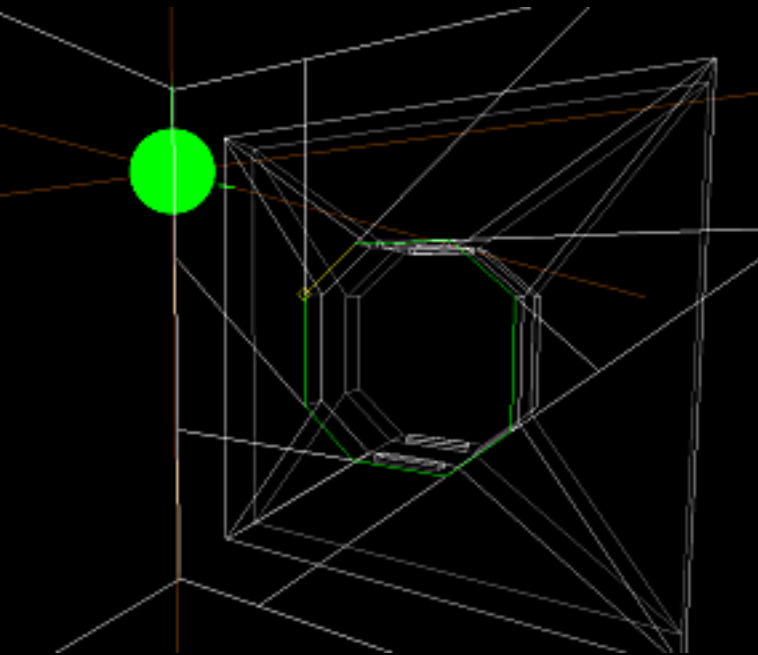


Doors have their own toolbar in D3Edit. It is that  at the top of the editor. The button activates the panel shown here.

When you first open the Door panel, you will probably see the left of the two images, everything is grayed out. Although if you open the menu at the bottom you can choose from a whole menagerie of doors. For example, select PTMC Covert H. Now the menu comes to life as a whole option becomes active **-Place**. Now, if you know how to add spaces, this should be pretty obvious to you.

Note: A door could not be added to face 0 of the first room until v40!

In the image on the right I have activated wireframe in my world view. Choose a current face in the room. If you **Place** If you click, you will see a purple line representation of the door on this current face. You may wonder why the door is so big; this is because the frame must have gaps for the door to retract into when opened. Like an elevator in an office building, the door components slide back into the hiding places in the wall.



Now will **Attach** available in the door panel, so click it. You will see in D3Edit how it inserts the door.

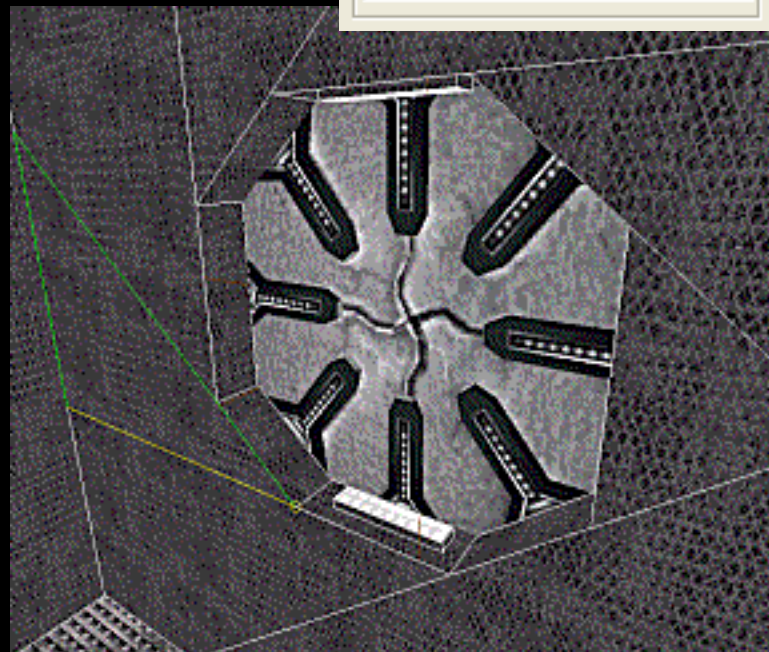
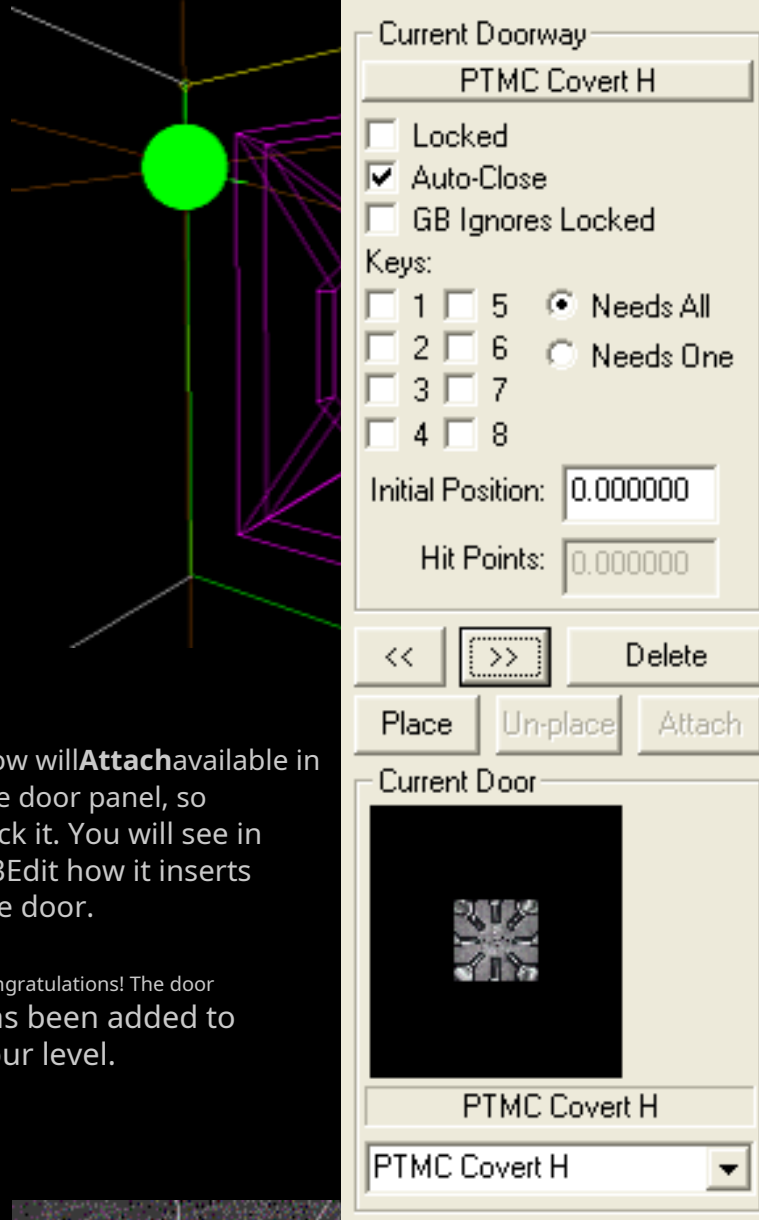
Congratulations! The door has been added to your level.

Back in Textured wh Outline Mode, you should notice a few things. First, the door frame takes on the current texture that you set in the Texture panel. If you don't want that, you can click on the door room and edit it in the room view. Note that if you are in one of the textured views and click on a doorway the editor will center the view on it. Secondly, you will notice that some doors have fixtures on the doorway sides. These are the places where you can apply a light texture if you want. I did this in the picture on the right.

One more thing. You can use the buttons << and >> Use in the Doors panel to cycle through the doors in your level.

You can do some interesting things here.

If you click on the button labeled "PTMC Covert H" you can give the door a name. This is useful if you want to reference the door in a DALLAS script.



'Locked' tells the game whether the door is locked or not and displays the standard message on the HUD when the player touches or shoots the door. You can use the DALLAS command UNLOCK Door <name> to unlock the door (or vice versa, lock it with LOCK).

'Auto Close' determines whether the door remains open or closes again after a predetermined time.


TheKeys:-Stuff is a bit strange and I haven't had a reason to use it until now. I'm going to digress here, so skip to the next point unless you're curious and understand DALLAS.

It is used by DALLAS in the player command "Give player [PlayerObject] key [KeyObject], key number [KeyNum], name = [KeyName], show HUD message = [Yes/No]." used. So they have in

Novak Corporate Prison (Retribution Level 2) gave the keys names (like "X-1 Security Pass"), and the [KeyNum] most likely references a number between 1 and 8 from the Door panel. This way you should then have real keys for doors that actually function as keys. This means that in a COOP game everyone must have their own key. In normal SP levels, it is sometimes preferable to simply unlock the door when you pick up the key from the player, rather than adding it to the inventory and fiddling with names and numbers. I didn't do the latter, but you can check out the Level 2 DALLAS Source Script if you want to try it. (075-Level2-

DALLASourceScript.rar)

Initial position lets you determine which position a door is in when you start the level. Use this to half-open a door.

Hit Points Control how many times you have to shoot the door before it breaks. This is applicable to lattice and rust-like doors, such as the manhole covers in Seoul Level (Level 4, that's the one with the nice subway) 

Doors can also be deleted using the button **Delete**. It is strongly recommended that you use this delete function instead of deleting the room like you would any other room. You want to make sure that you delete the door object with the room. Also make sure that the door you want to delete is also displayed in the Door panel.

[Back to the overview of section H](#)

076 - Own Doors <>

Toxxeon

This guide explains how to create your own doors and add them to levels.

In addition to a current D3 editor, we also need this **OOF editor**, to extract the Table.gamthe **HOG2 workshop** as well as to create your own.gam-Files that **gamTool**. If you don't have these programs yet, download them.

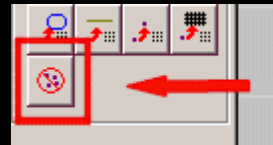
To start with, just an overview:

A door ...Door... is a room with one or more animated objects. The movement of these objects is controlled by script. In order to integrate a self-made door into the level, we have to replace an existing door with ours. This has the advantage that we don't need to create an extra script for our door. We simply use the one from the original door. But always in order...

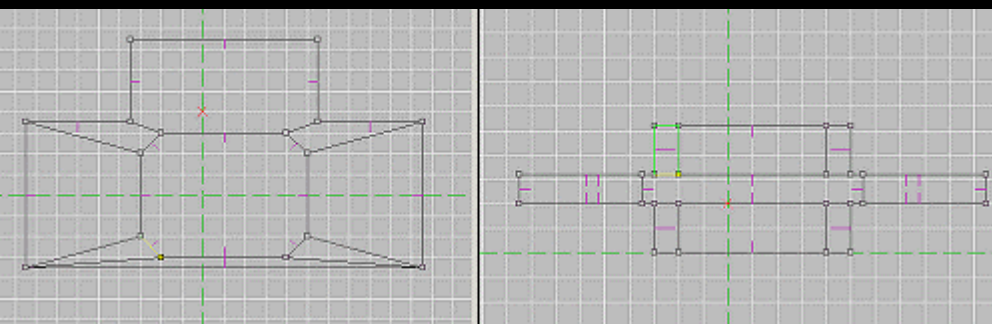
In these instructions we will build a three-part door in which the individual door objects open in different directions.

Opens a new room in the D3 editor. First create the "door frame" as you specifically need it for your level. You need a passage that you want to fly through later, and pockets on the side (or above or below). The door objects then slide in there. Remember that this room is closed all around. Because otherwise you can "look outside" the level.

VERY IMPORTANT: Must herefor nowthe facesBe sure to form a closed polygon network!So there shouldn't be any extra vertices left when you copy faces and move them together. If you have duplicate verts, click here. (Picture right)



You can texture your "door frame" here according to your taste to get a better overview in the 3D view. Sometimes, however, those defined here are



Textures not included accepted. Then you'll have to do that later, when the door is in ours*.d3ifile is inserted, do it again. But this is no problem. OK Our door frame is almost finished...and looks like this (picture on the left)

As already mentioned above, everything up to this point must be a closed polygon network. However, now we have to **one**Area that later than *Front faces* serves, detach from this network! This area must be a later portal!

The easiest way to do this is to click on an area in the 3D view that should become this front face and mark it with the **Space bar**. (next picture) Then copy it to the clipboard. Now mark the face again and delete it. Then click in a 2D view and insert the face from the clipboard back into the editor. Now align the face on the grid and move it back exactly to the location of the face you just deleted.

Of course you can also take the face with you **Shift-Ctrl-V** Insert it so it ends up in exactly the right place 😊

ATTENTION: From now on you are NOT allowed to DELETE THE EXTRA vertices!!! Otherwise the polygon mesh will be connected again!

What we have modeled so far will be our door shell and front face. But more on that later.

Now to the door wings:

The cheapest way to create them is at one somewhere else in the same room and only moves it into the correct position at the very end. More precisely, the positions that the door should look like when closed. These door panels could look like this on a three-part door. (Picture on the left) The picture shows the three parts just pushed apart a bit to make it easier to see...



And do not forget: **Don't delete the extra vertices!** Each individual door must be an independent polygon network. You also have to decide here what texture the door leaves should have. The textures that you assign to the wings are applied and can no longer be changed!
Saves the room as "own_door.orf".

Now let's create our door with the door objects...

First, we look in the D3 editor to see which door we want to replace. In this example we take this *Novak B* (Picture right)

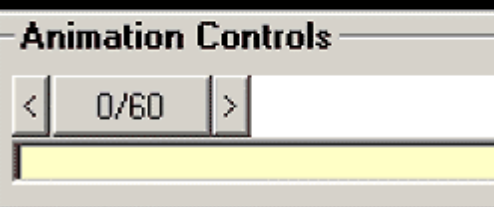
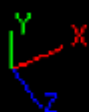
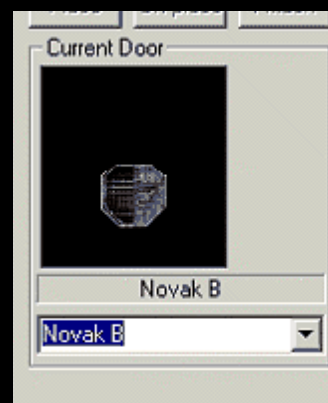
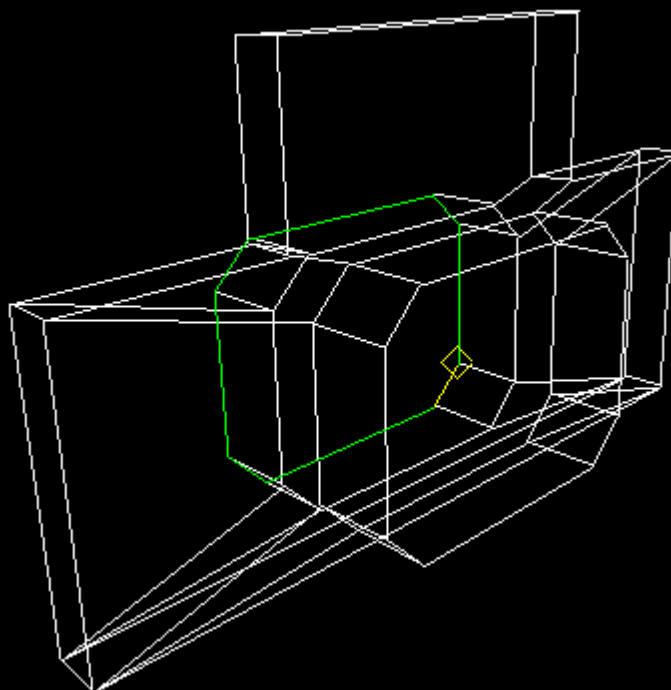
Then you open with it *HOG2 workshop* the file *d3.hog* and extracts the files inside *Table.gam* and *novak_b.oof*.

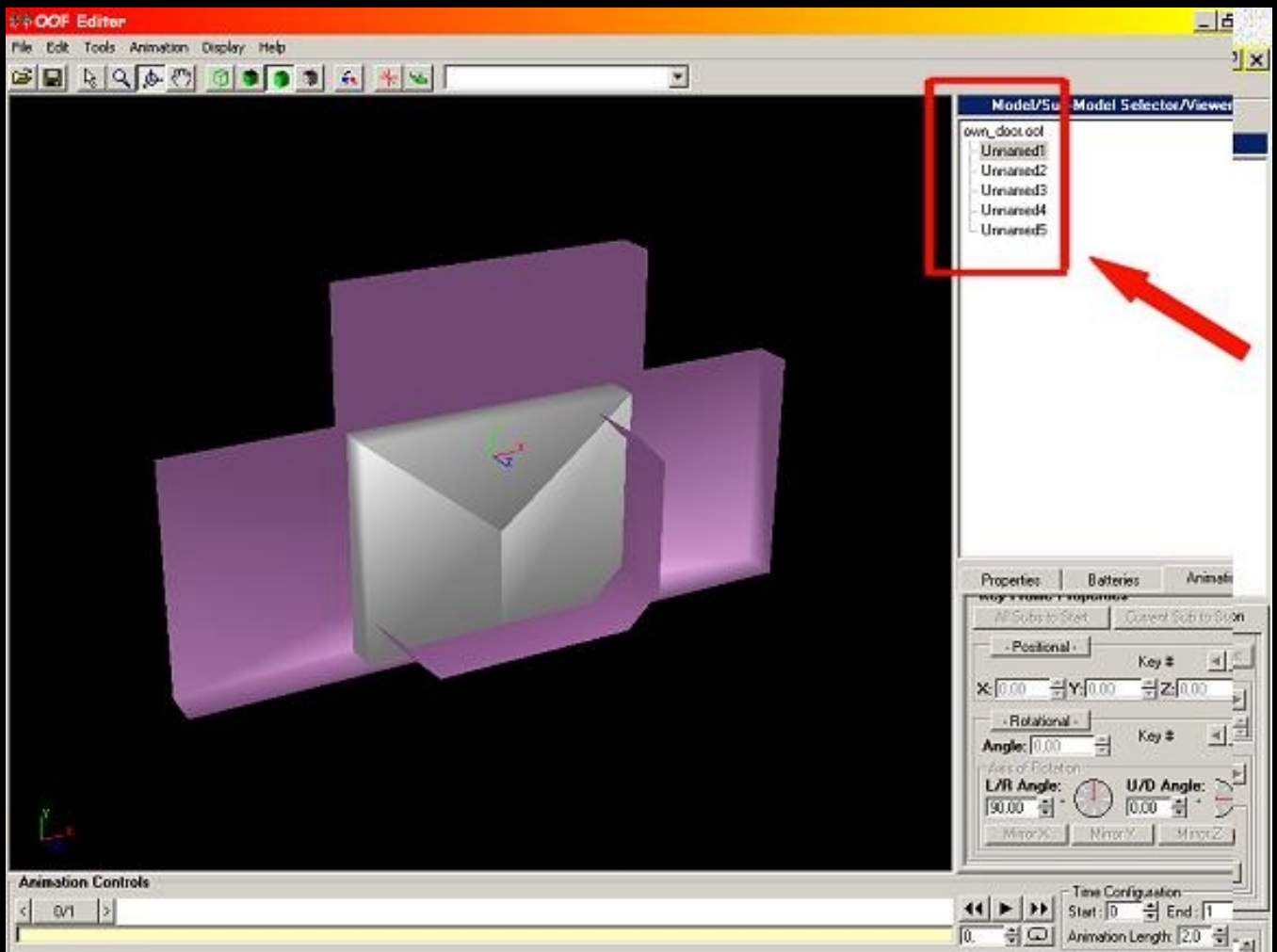
Now you start the program **OOFEditor.exe** and loads the object *novak_b.oof*. Look in the bottom left corner to see how many keys the animation of this object has. Here the last key has the number "60" (picture on the left). This is important because **our new door must have the identical key**

Have a number to make it work!

So, now you simply open our room in the OOF editor *own_door.orf*. You can see the result in the following image...

More about the OOF editor later, in Papacat's OOFedit excursion.





On the right side (top) you can see the name of our object with all its sub-objects = *Sub models*. For us there are 5 pieces, they are all included *Unnamed* designated. If you click on them one after the other, they will turn lilac in the 3D view. Here you can also see that the door frame can be seen as a body... and that the individual portal area is available as its own sub-model. Now you can view the sub-models under *Properties* First give any name you want. I call you:

- door frame
- portal
- Wing_up
- wing_left
- wing_right

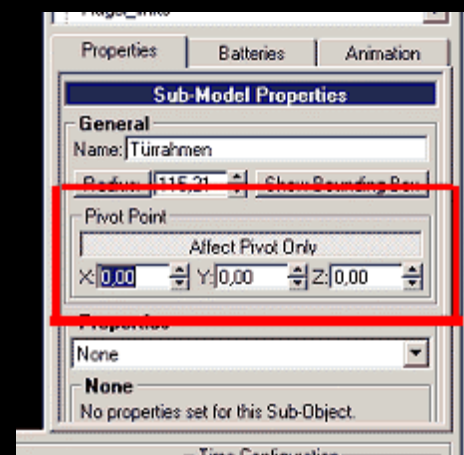
Now everything is under *own_door.oof* save!

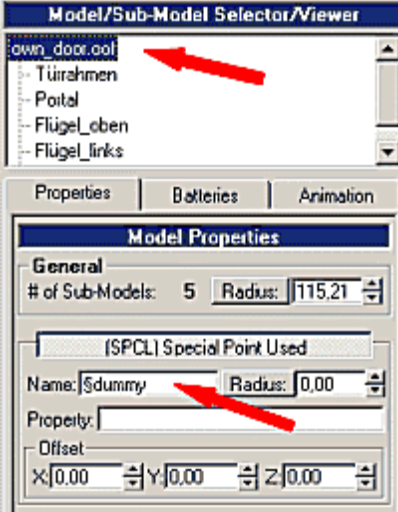
Annotation:

If you create very complex doors, the door object can have many sub-models. To get a little more overview, you can also view the sub-models one below the other *merge*. These can also be spatially separated from each other. So take a close look at which door parts should have the same movements, for example.

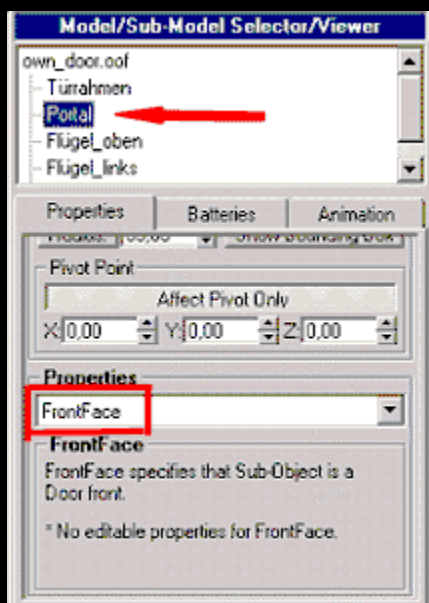
Then move in **Model/Sub-Model Selector/Viewer** While holding down the shift key, drag and drop one sub-model onto the other. Complete. This means that these object parts have the same properties and movements...

However, in our example here we are not allowed to do that! Next, activate the first sub-model and then click in the register **Properties** the button **Affect Pivot Only** at. Then you sit there all Coordinates (X,Y,Z). **0.00** (Picture on the right) Do this for every sub-model, otherwise the door leaves will not be in exactly the desired position later. And leave with every submodel **the button activated!**



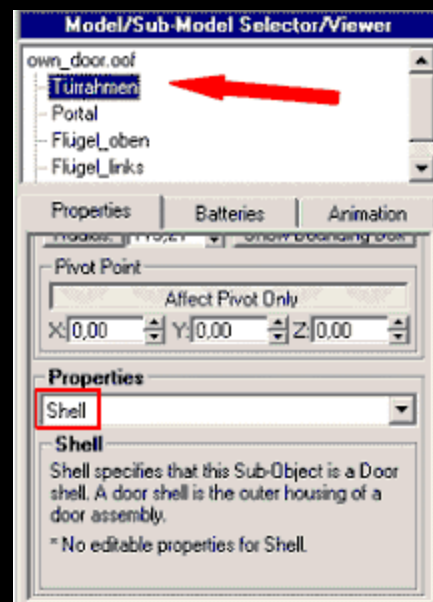


Now mark the (main) model in the Model/Sub-Model Selector/Viewer and then click on the button (SPCL) Special Point Used, and then simply enter there under Name \$dummy (Enter the dollar sign and dummy). As you can see in the picture on the left. Then leave the button activated!



Important: Now we need to assign the Shell property to the Door Frame submodel in the Properties submenu. Click on the pull-down menu and select the appropriate one. (Picture right)

Do the same with the submodel portal (this is the individual area from before). Select the Frontface property there. (Picture on the left) For the other sub-models, i.e. the door leaves, you have to select None! However, this is the default setting, so you don't need to change anything.

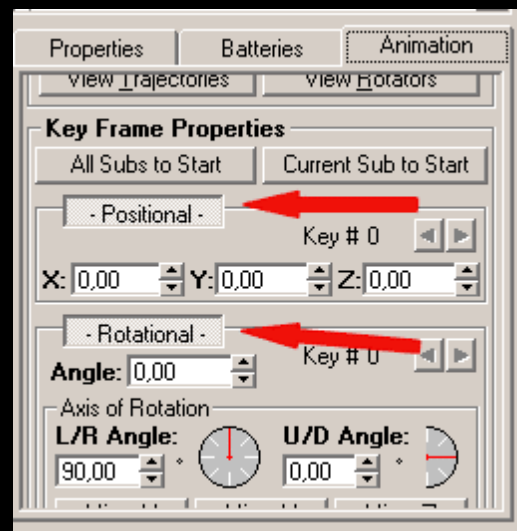


Cache!

So now let's move on to the movement of our door: Switch to the register **Animation**.

At the bottom right below *Time configuration* give her **End 60** one, these are the animation keys already described. At *Animation Length* You don't need to change anything, the time is controlled using a Gam file! Here you can only test whether you like the animation of the door, for example if you want the opening process to take 3 seconds. To start the animation *Play button* press.

Then you activate the individual sub-models one after the other and simply click on them **Positional** and up **Rotational**, without changing the values there! Make sure each one **Key #0** (the first key) is displayed. (Picture right)



Important: Also the door frame and that Portal need one Key #0 have activated. But no other keys!

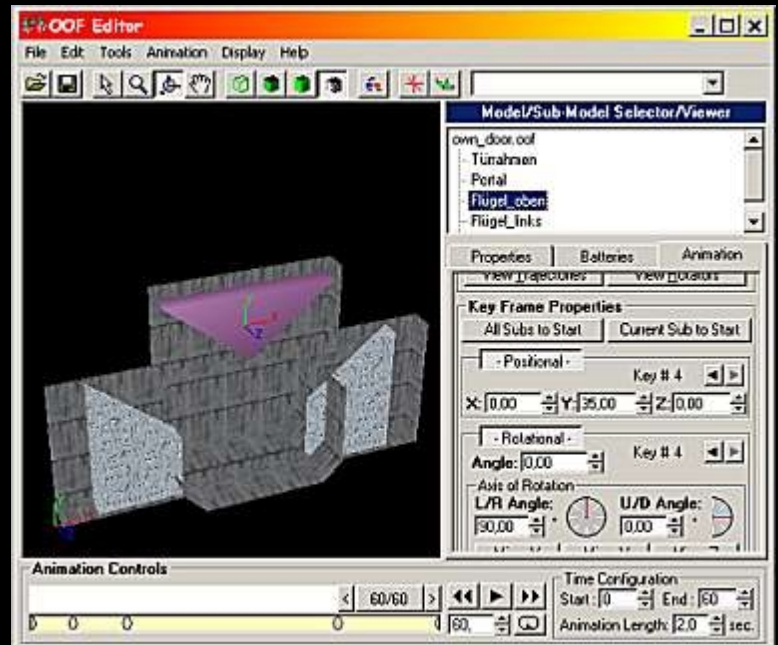
You simply set a key by clicking *-Positional-* and or *-Rotational-* clicks. You can recognize set keys in the bottom bar by the oval markings.

Also important: For key frames, always sets both positional and rotational. If you don't set both, Descent 3 won't animate the door, but will just show it either closed or open.

Now you can experiment until you find a movement sequence for the door wings that you like. The sub-models *wing_up*; *Wing_left* and *wing_right* are with the coordinates X, Y, Z or with the **Angle information** movable. You don't need to activate all 60 other keys, maybe just 2 or 3 or more, depending on how sophisticated you want your door's closing process to be. It doesn't matter where these keys are within the animation. Each individual door leaf sub-model can also have different keys. However, at These sub-models at least the one **Key #60** be activated at the end. (Only at this one Sub-Models!) These keys can be used very well to synchronize individual movements.

It should also be mentioned that only the movement from the closed to the open state needs to be defined here. So with Key # 60 the opened state of the door must be visible. The level then simply plays the animation backwards when the door closes again. We don't need to take that into account here with our animation!

Our door looks like this when open in the OOFEditor:



Saves the object again and closes the OOFEditor.

Now let's open with that **gamTool** the previously extracted file **Table.gam**. We look for the section in there **Door: NovakBout**, copy this along **Ctrl + C** into the clipboard, click on **File -> New** and add with **Ctrl + V** the section into the new game file.

(Of course you can also get a new one **gam**-open file, **Alt + D** Press and then write in all the necessary data yourself.)

Under **Model File Name** the path to our object **own_door.oof** indicate. (Just like it's stored specifically for you...)

The options *Opening time*, *closing time* and *Open time* you can adjust accordingly. **Leave everything else under flags as is!** That's what we want **generic.dll** use for our new door.

Don't change the door name (NovakB) either!

The **gam**-Then save the file and **note that this file has the same name as the later MN3 file.**

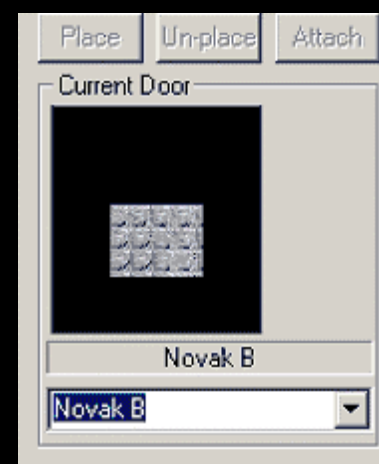
Specifically in our example: **own_door.gam ==> own_door.mn3**

Done...actually that's it!

We now open the D3 editor again and load under **Data -> Table File Manager** the file **own_door.gam** into it.

If we now in **Door menu** under **Novak B** If you look, you will see that the original door has been replaced by the new door (picture on the right)

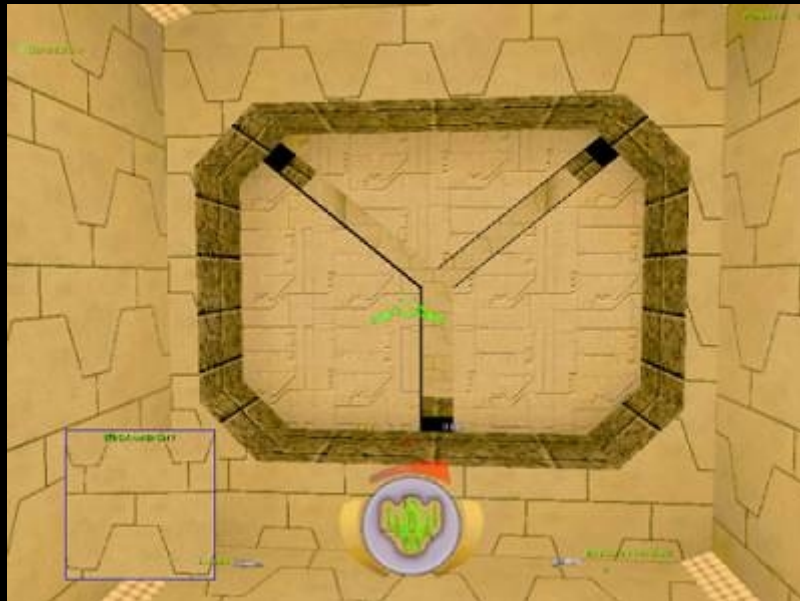
Don't be surprised by the display, only the door objects (i.e. the door leaves) are shown in the door preview in the editor.



Now insert the self-created door into your level, and texture the door frame again according to your taste, if necessary... When you finally pack the MN3 file, then submit **Additional file** the files `Sown_door.gam` and `down_door.oof` with on.

Have fun experimenting! There are actually hardly any limits to what kind of violent doors you can incorporate into the levels! For example, various locks could be released before a door opens. Or you build yourself *topic-specific* Doors...

If you have any questions or improvements to this guide, write todescent@toxxeon.de



Update:

It is now no longer necessary to replace existing doors with your own. Newer D3Edit versions now allow you to save your own creations as normal `.gam/.oof/.mn3`-Package integrates and in the `.gam` can be given your own name. These doors then appear - similar to the textures - at the very end of the list.

It is also no longer possible to simply delete the Door object, see right



I also refer you to the chapter here 'Set up D3Edit & Quicktest' from WillyP, that takes a lot of work away from you.

Danger:

D3Edit can handle a maximum of 60 doors, which means you can use a maximum of 11 additional doors of your own. Entries that are further down in the Gamfile will then no longer be displayed in the Door palette! You could then start replacing original doors again.

Back to the overview of section H

077 - Sound for your own door

Ragil Ral

So now you have a door. The stupid thing now is that it either doesn't have any door sounds at all or has one of the standard door sounds. In order to give your door the appropriate background noise, you have to put in more work.

Create SoundFX

Get sounds from the internet, there are some sites that offer them as free DL; In most cases you will be required to credit the authors, which I think is only fair. However, you will hardly find any that you can use straight away, so you will need a program that allows you to cut and mix sounds together. If you have no experience with such programs, you just have to learn now.

Now analyze the animation sequence of your door according to time, ie. if something happens at frame 69, you need to know how much time has passed until then; Because that's exactly what you have to adjust the background noise to. It's best to make a short list of what happens at which frame/time. Creating the sounds is actually the most work and quite time-consuming.

The finished sounds should then be in the format 16 bit, 22 kHz mono.wavhave.

Integrate SoundFX

This is done quickly: you have to go to the.gamnow set two additional entries. Either you copy two 'sound' entries from the D3.gam, or doesEdit->New->Sound(**Alt-S**). Enter the values accordingly (screenie). The door is now closedOpening soundand Closing soundthe names of your sound entries as a value.

The entries for sound are as follows:

File name-name of the.wav-File.

Loop start-At which sample does the loop start?

Loop end-At which sample does the loop end (round up) (length of the sound file in samples = length in seconds times sampling rate, whereby 22 kHz is to be expected with 22050)

Outer cone volume- ...

Inner cone angle- ...

Outer cone angle- ...

Maximum Distance-Distance from which you can no longer hear anything.


Minimum Distance-Distance below which nothing can be heard.

Import adjust-By what factor the volume of your sound is adjusted in-game.

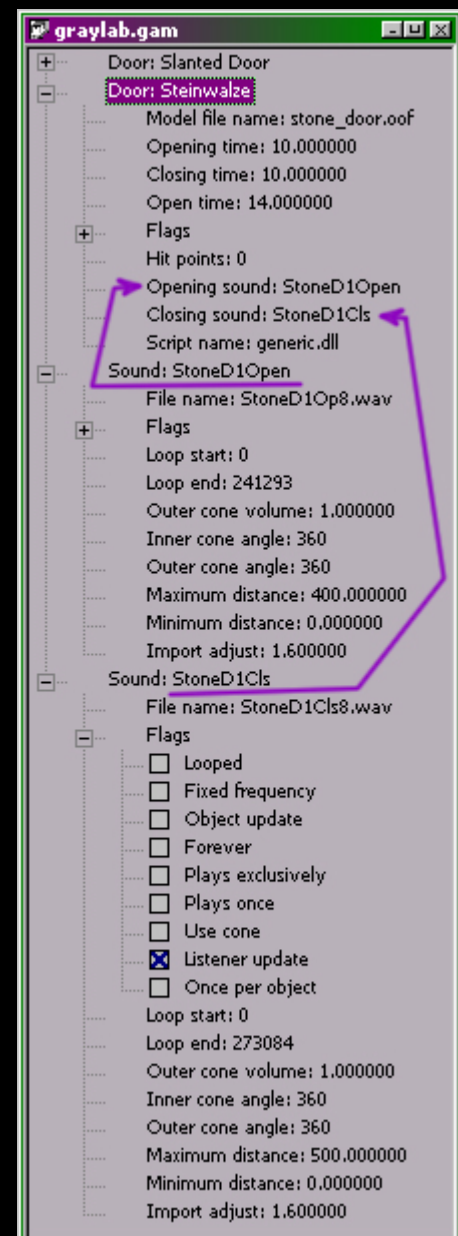
The flags are still thereListener update, so that the volume for the moving player also changes with distance.

A detailed explanation of the flags can be found in the.gam-Specification at the end of the document.

Then pack with that.oofand the.gamadditionally also the.wav-Files into the mission hog.

That was it 

Back to the overview of section H





078 - Custom objects 1 <>

Fischlein - Inspired by Papacat

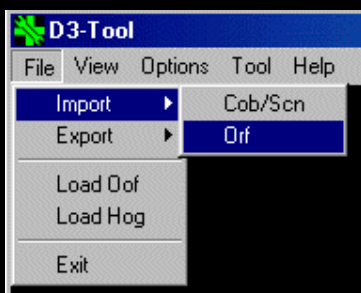
The files on the topic are in the zip file.

Creating your custom objects is not as difficult as you think. You need D3Edit, Gamtool and D3Tool.

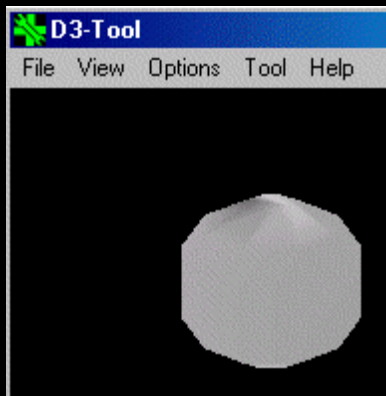
I have the space under "ball01.orf" saved, you can call it whatever you want.

Write the file name in lower case! I think that otherwise there could be problems with a Linux server (Supposition!).

Now open the D3Tool. Choose this oneimportfunction

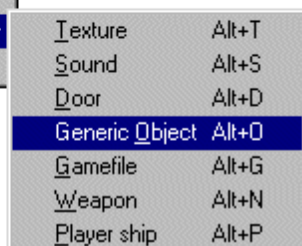
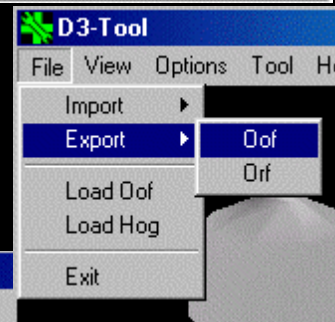


and "ball01.orf" out of. Afterwards it should look like the picture below.

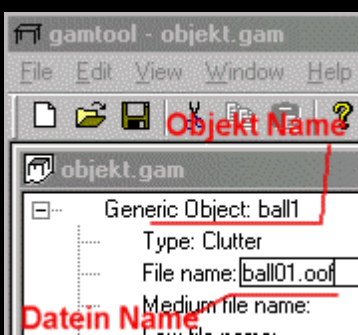


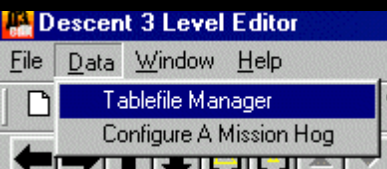
You then export the room as *.oorf (Outrage object file, right small image).

Open the GamTool and select Edit/New/Generic object. (right, larger image) First enter the name with path information (**a maximum of 34 characters are permitted!!!**) a. You only need to specify the path if your own objects are not stored in the D3 path.



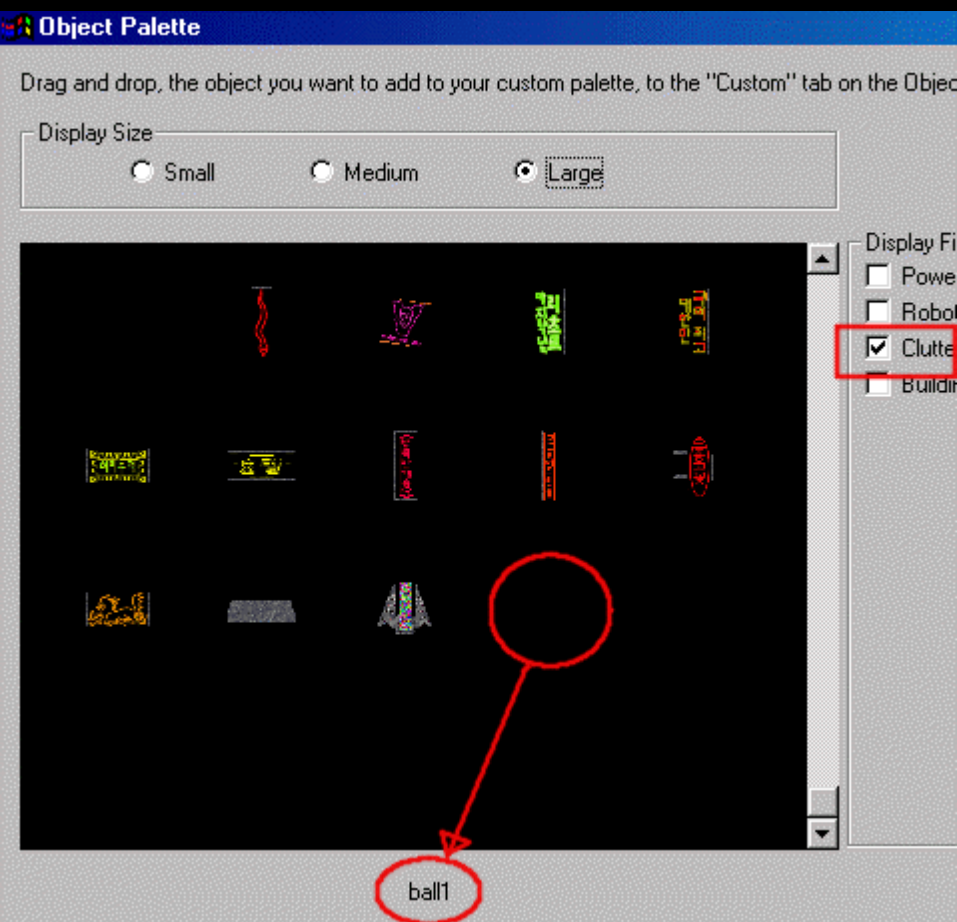
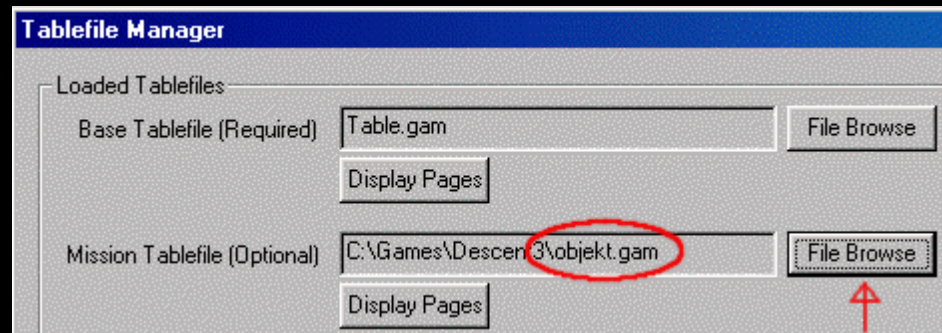
We now have an object named ball1 created. I have the table asobject.gam saved (**written in lower case**).





Now open D3Edit and select **Data** in the menu, picture on the left. Now you click on **File Browse** and look for your ***.gam** file. Now you can click on **Display Pages** and see what's in there ***.gam** file is defined (picture below).

Ok, now create a level or take an existing one and call up the object menu. Click on the button **Available Objects...**, then filter type: **Clutter** and scroll all the way down.



I don't know exactly why D3Edit doesn't display your custom objects graphically, but you can always find the objects or textures that you designed yourself at the end. Now insert the object into your level and save the level.

Update:

Newer versions are now able to display their custom objects. Please also read the sections **Making Custom Objects Visible in D3Edit** in the appendix and **Error: Reference not available found - Error: Reference not found by**.

Make sure that the level has the same name as your *.gam file!!!!

Starts the MN3 Packer and adds the object ***.oof** and that ***.gam** file.

Pay attention to level names!!!

Back to the overview of section H



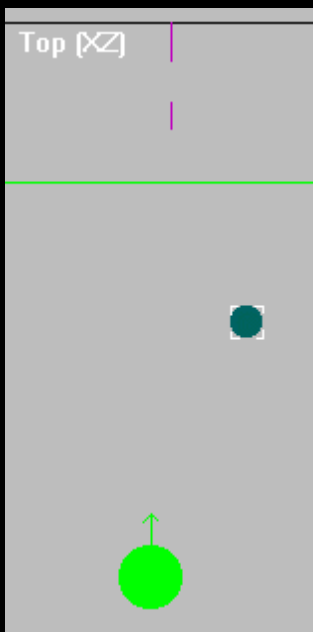
079 - Custom Objects 2 <>

Fischlein inspired by Papacat

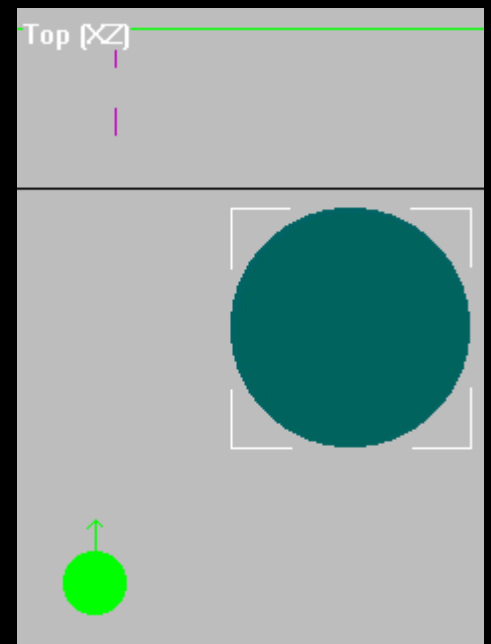
Some of the images on this page are from Papacat!

As always, the files are in the zip file.

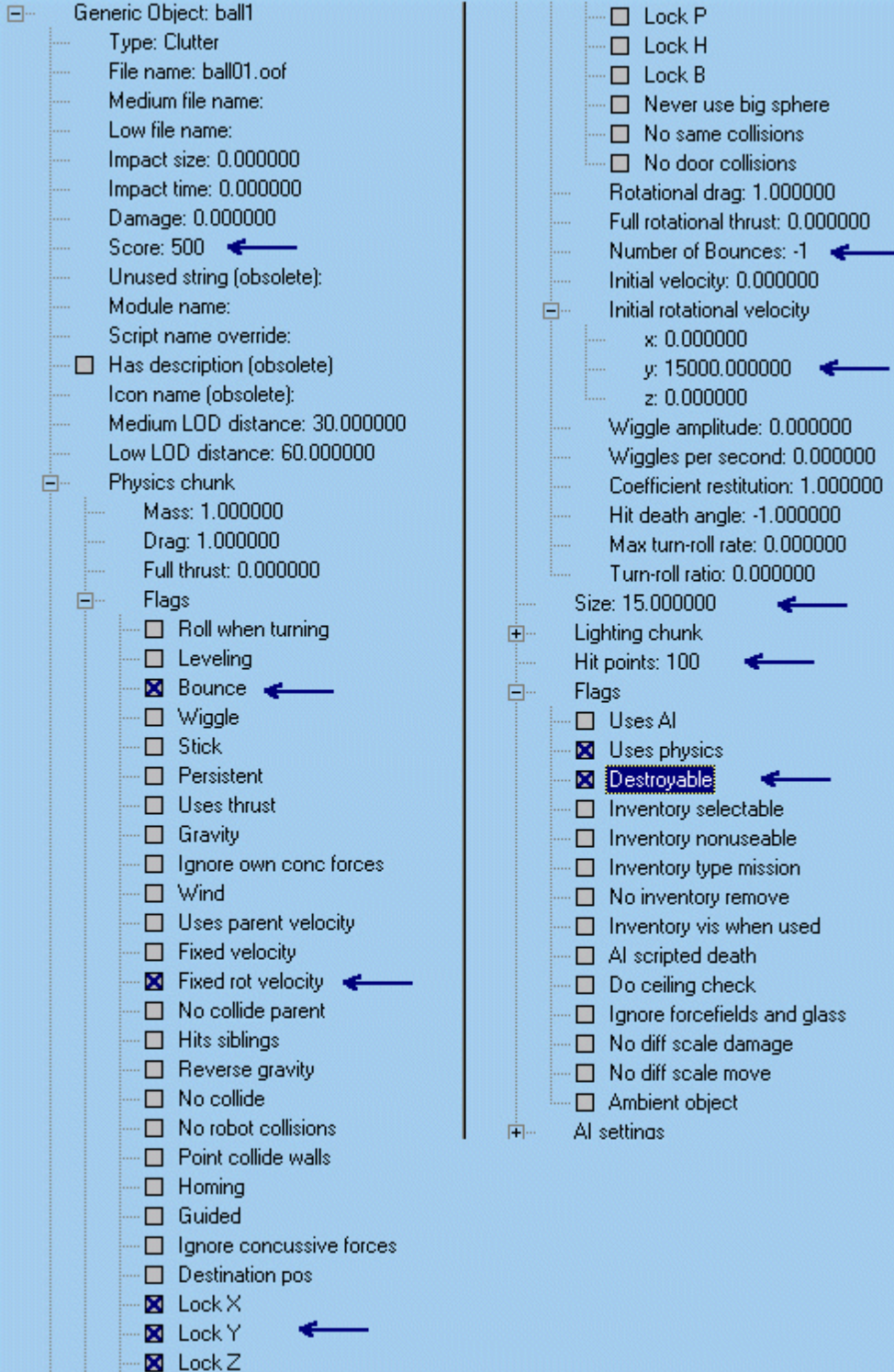
In the previous topic we created an object and the Gam File. In this topic we will set some properties for the object, which is again done in the Gam File. But first of all, a not so important feature, but one that is definitely useful, **SIZE**.



In the image on the left you can see what it looks like if you leave this value at default. In the right picture after I entered SIZE. The ball I built has a diameter of 30 units, which results in a radius of 15 units. So I have to enter 15 under SIZE. (not the value of the diameter)



In the following image you can see which settings we make:



Score: This sets how many points the player will receive if you destroy it. **Bounce:** Specifies that when the object is hit, it will bounce off the walls and other objects.

Fixed red velocity: This check mark means that when a player shoots the object, it begins to rotate at the speed specified under Initial rotational velocity.

Lock X, Lock Y, Lock Z: A check mark is placed to avoid the object being hit when hit shifts or bounces. **Number of bounces:** Set this value to -1, if you choose a higher value, the object could fly out of the room as it will then bounce off the wall very quickly. It would literally be thrown out of the level!!!

Initial rotational velocity: Selects the axis around which the object should rotate and the speed. You have to use high numbers!!!


Size: See above.

Hit Points: Specifies how many times the object must be hit until it is destroyed.

Uses Physics: That the options of the physics chunk are used.

Destroyable: Whether the object can be destroyed or not.

Make all the settings and save the GAM file.

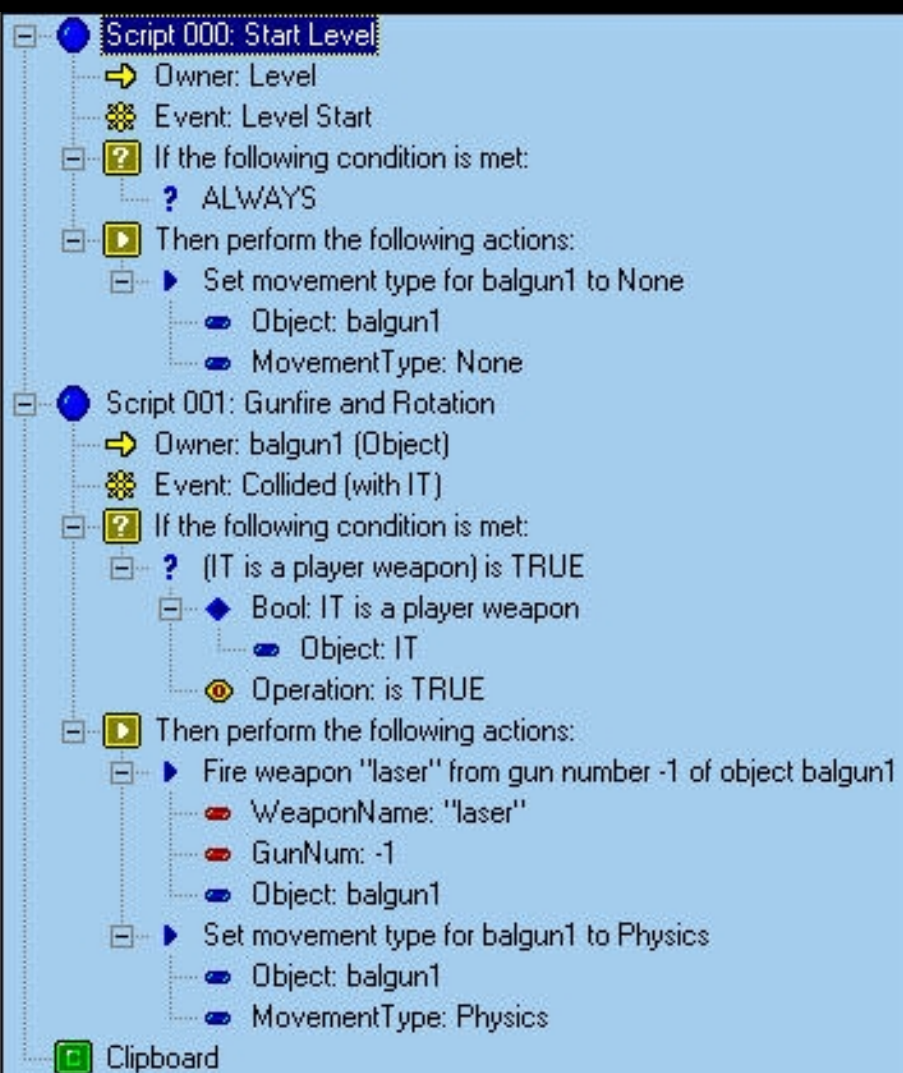
Now open D3Edit and load the Gamfile (Data/Table File Manager). Now open your level, call up the object bar  in the menu bar) and switches to object mode (**CTRL+G**). Now make your object current (for me it is Ball1) and click on the button that is outlined in red in the right image. We need to give our object a name so that we can access it later with the Dallas Script.

I named it "balgun1". Now switch to the World View view and then to the menuFile/Dallas Graphical Script Editor(You will now get two hint boxes because you haven't created a script for your level yet, just open itOK click.).

If you don't have Dallas Script Editor installed yet, read the topic

Anticipation: Getting DALLAS up and runningthrough!





For all of you who are already familiar with the Dallas Script, you can use all the settings from the picture on the left.

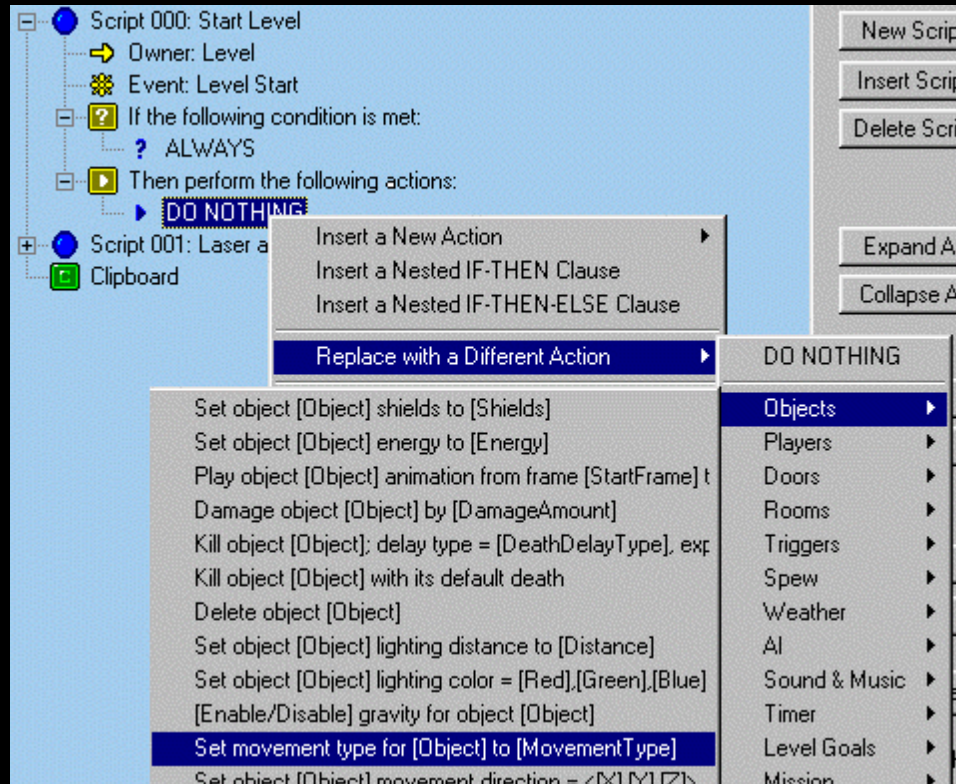
Explanation of the image:

Script 000: Start Level

If you have opened the Dallas Graphical Script Editor, you will see an empty window. Click on that button **New** at Scripts and give as a name **Start level**. Next has to be at **Owner** To enter a value, click with the right mouse button **Owner** and then select from the submenu **level**. Now it has to **event** to be specified, right-click again **event** click and from the submenu "Level start" choose.

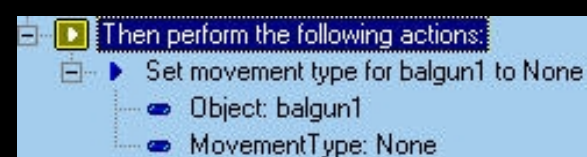
To the next, Then perform the following actions and take a look

The picture on the right shows how this is done.



After you do that in the previous picture

have done, you still have to specify the object. Right-click on **object** and then **Clutter** and in it the name of your object.



Script 001: Laser and Rotation

This works the same way as with the first script. Owneryou set it to your object name (for me balgun1). "event"put it on"Collided(with IT)". You can see all of that from the picture on the previous page, top left. Under "If the following condition is met"you click"Always" and then the right mouse button, in this submenu you then select "Replace with a New Condition", underneath Binary Statement with Query -> Objectschoose and then[Object] is a player weapon.

Further to "Then perform the following actions". First the laser has to go in, it works like this. Click next "DO NOTHING" and then right-click on it. Select "Replace with a Different Action -> Objects -> Fire weapon [WeaponName] from gun number [GunNum] of object [Object]"

Now we have to get our ball to spin as soon as it is shot at. Click along right mouse button "Then perform the following actions"and chooses"Add a New Action/Objects/Set movement type for [Object] to [MovementType]", is the same as can be seen in the second picture on the previous page, but under "MovementType"puts on"Physics". Now click on the button**Save**.

Save your levels and add all files to the MN3 packer. (for me these are:object.gam, ball01.orf,object.dllandobject.msg,I have the last two files in C:\Games\Descent3\Dallasscript\osirisdepends on whether it is the same path for you)

[Back to the overview of section H](#)

080 - Custom Objects 3 <>

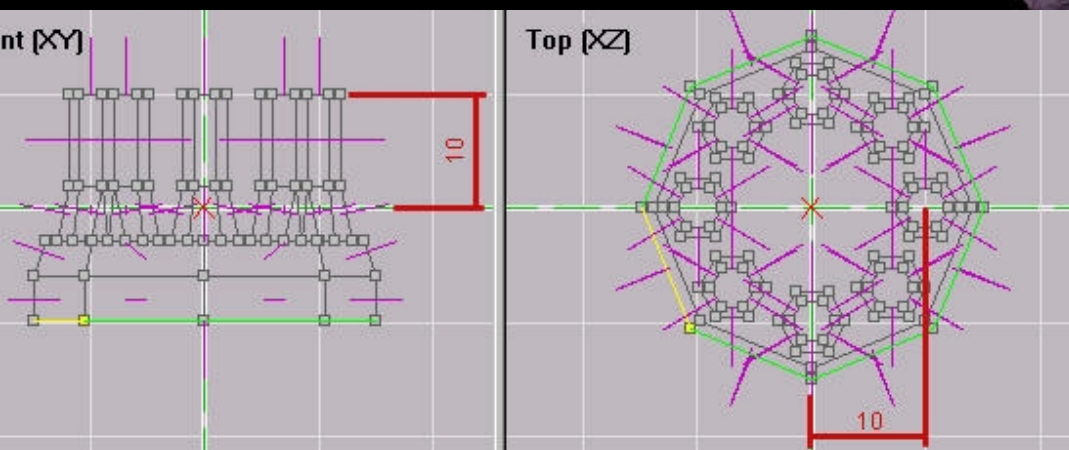
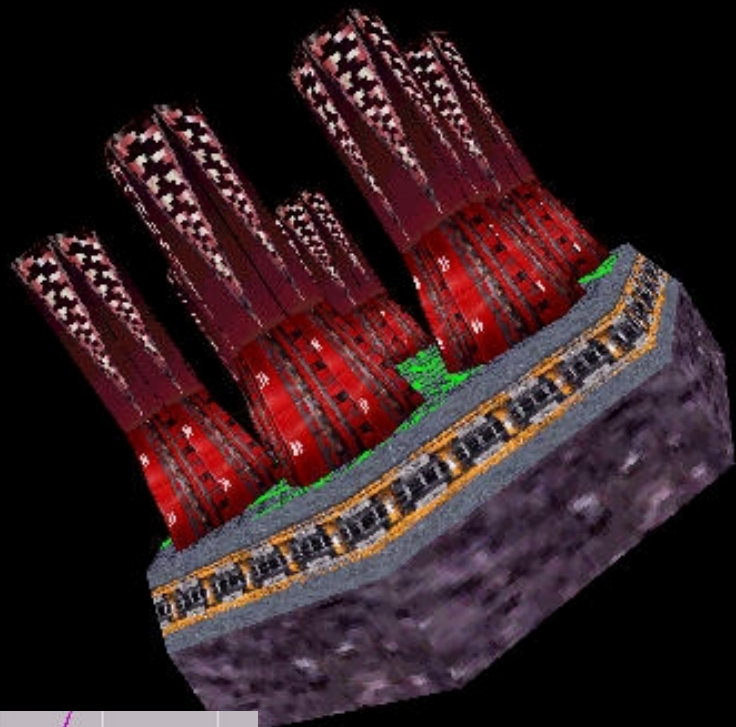
Fischlein inspired by Papacat

Now that you know how to create objects and integrate them into the level, all that's missing are a few nice effects. Well, I'll try to show you how to add a "gun point" to an object, position it and also bring it to "life". You need this for this topic **D3Tool**(V1.4).

I first created an object that can be seen in the picture on the right.

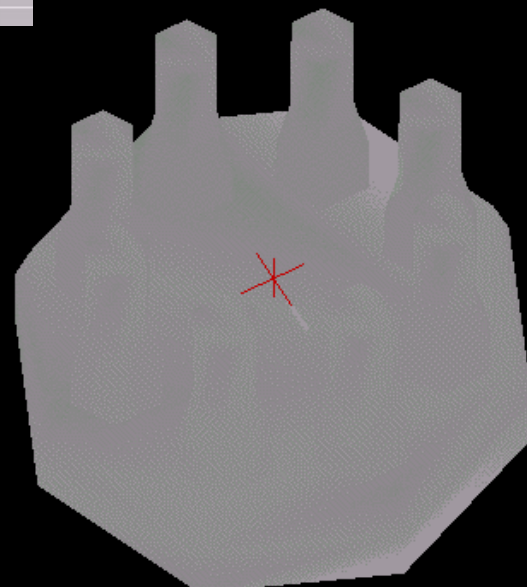
How you can create this is in the topics **Custom Objects 1** and **Custom Objects 2** explained.

To know where we need to position our Gun Points in D3Tool, we open D3Edit and the space we created for the object. We need two values, which you can see in the picture below.



Now open the D3Tool and choose File -> Import... ORF. Now select your ORF.

Now go to the menu Tool -> Edit Window, which is the selected tab GPNT. Click on the button here **Add point**, check the box below Show All Points. Now you see a red cross (picture on the right) in the center of your object. There is a lighter line on one side of this cross, which is the direction in which the shot is being fired.



Edit Window

GPNT | GRND | ATCH | NATH | SPCL | WBAT | SOBJ

1 + Of 1

Add Point

Delete Current Point

	X	Y	Z
Location	10.00000	10.00000	0.000000
Direction	0.000000	1.000000	0.000000
Submodel	1		

☐ Show All Points ☒ Show Local

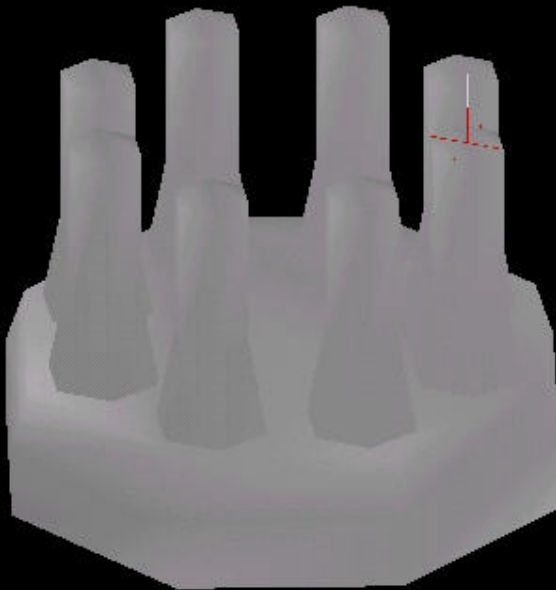
In the picture on the left you can see what needs to be set to position the gun point (X= +10, Y=+10, Z=0).

You don't need to write down the +, just the - (minus).

AsDirectionI chose the Y axis. (just put a 1 in, 0 is off, 1 is on)

Here is an overview of the positions of my gun points, I need 8:

Z=0
Z=0
Z= +10
Z= -10
Z= +7
Z= -7
Z= -7
Z= +7



annotation: Pay attention to which point you set the position for. (1 of 8 or 5 of 8) I also noticed that you shouldn't move the 3D view, because for me the gun points changed position, but only when I set the first two. I would recommend that you set all the points, save the object, then open it again and check whether everything is correct.

To save, go to the menuFile ->Export... Oof. Now create the Gam file with the following settings:

Physics chunk/Initial Rotation Velocity: Y=90000
Physics chunk/Flags/Fixed red velocity-Mark with a cross
Physics chunk/Flags/Lock:X, Y and Z

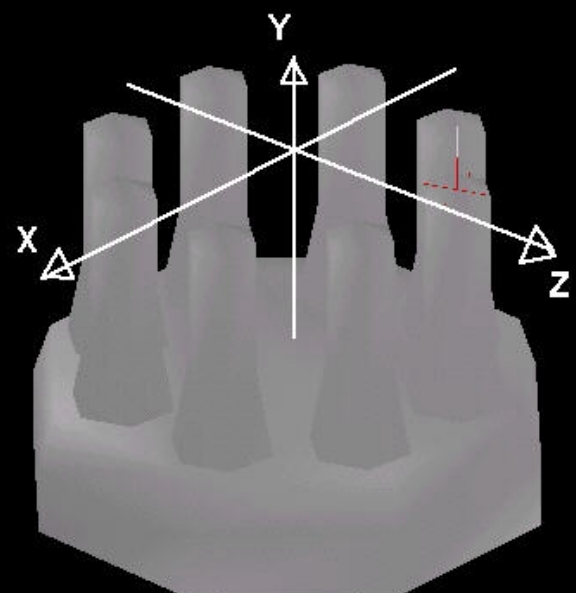
Save the game file with the same name as your level!

Now open D3Edit, go to thatMenu Data -> Tablefile Managerand select your Gamfile.

Annotation:In order to be able to see your custom objects in the 3D preview, create a finished MN3 file with everything that belongs in it, including *.GAM and *.OOF. If you now open D3Edit, go to the menu Data -> Configure A Mission Hog. In the following dialog box click on **Change**and select your MN3 and then click on**OK**and again the next time**OK**. Now againData -> Table File Manager and clicks on**Hog Browse**atmission Table file.

Update:

For newer D3Edits it is sufficient.oof to have lying there where in the .gam is referred. Please note the 34 character limit for path and file name!



Now open your level *.D3L and you can see it in the preview (insert objects). Add it to your levels and check it out. This is helpful when you need to align objects. Now you have to name your object. Press the buttons **CTRL+G** To switch to object mode, now click on your object in the room view so that you can see your object at the top of the object bar. Below this preview there is a button on which the name of the object can be seen. Click on this and enter a name of your choice in the following dialog box. I named the object "weapon". You have to do this in order to be able to access the object via script.

script on this topic

How do you "Dallas Graphical Script Editor" is set up under [The Dallas Graphical Script Editor](#) explained.

Switch to World View and then to the menu File -> Dallas Graphical Script Editor. Now click on **New Script** and names it in the following dialog box **Start level**. You can find out the rest from the script in the picture on the right.

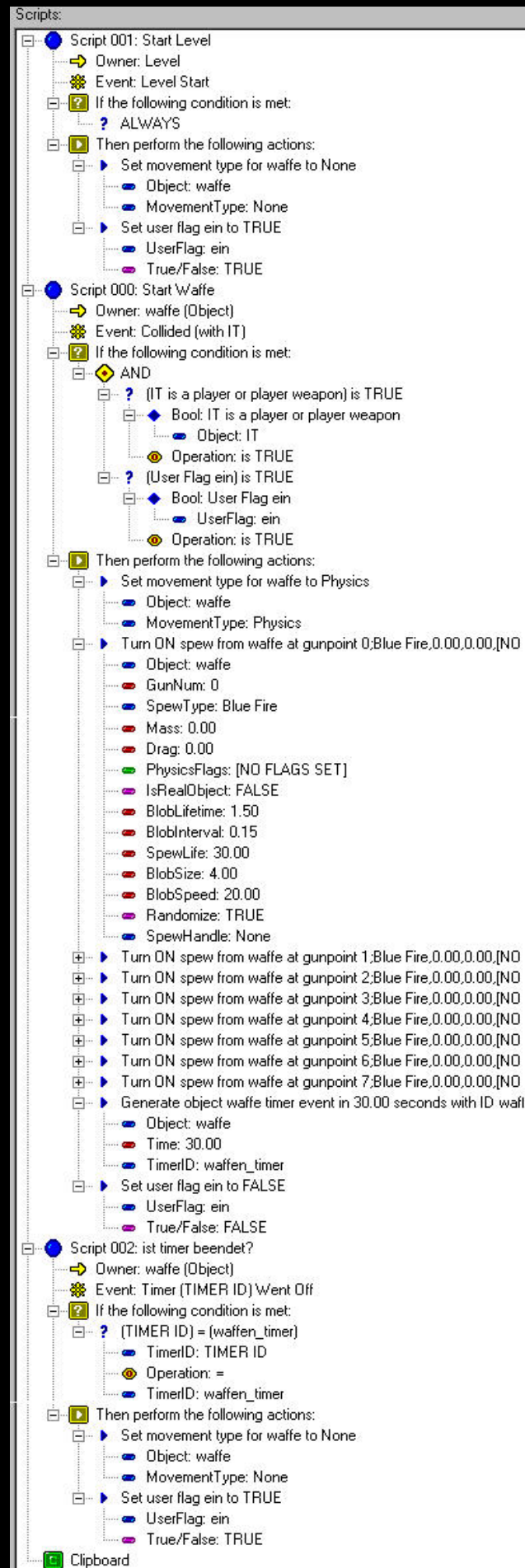
You get the user flag like this: Click **User types Workshop**, chooses at **Select the User Type to edit: UserFlag**, then clicks **Add New Value** and assigns a name (here "a") in the following dialog.

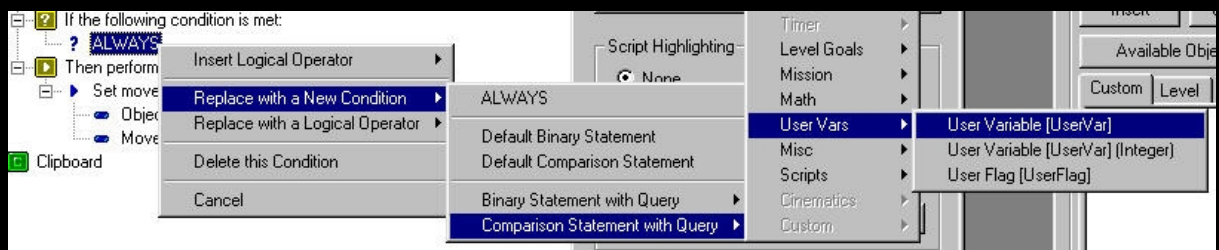
Once you've done that, click again **New Script** and names it **Start weapon**. Apply the settings from the right image again.

Explanation: Under **Then perform the following actions** stands everything that needs to be done. "Set movement type for weapon to Physics" says that the parameters that are in the Gamfile under Physics are valid. This "Turn ON spew...", I will explain in a separate topic because it is too extensive.

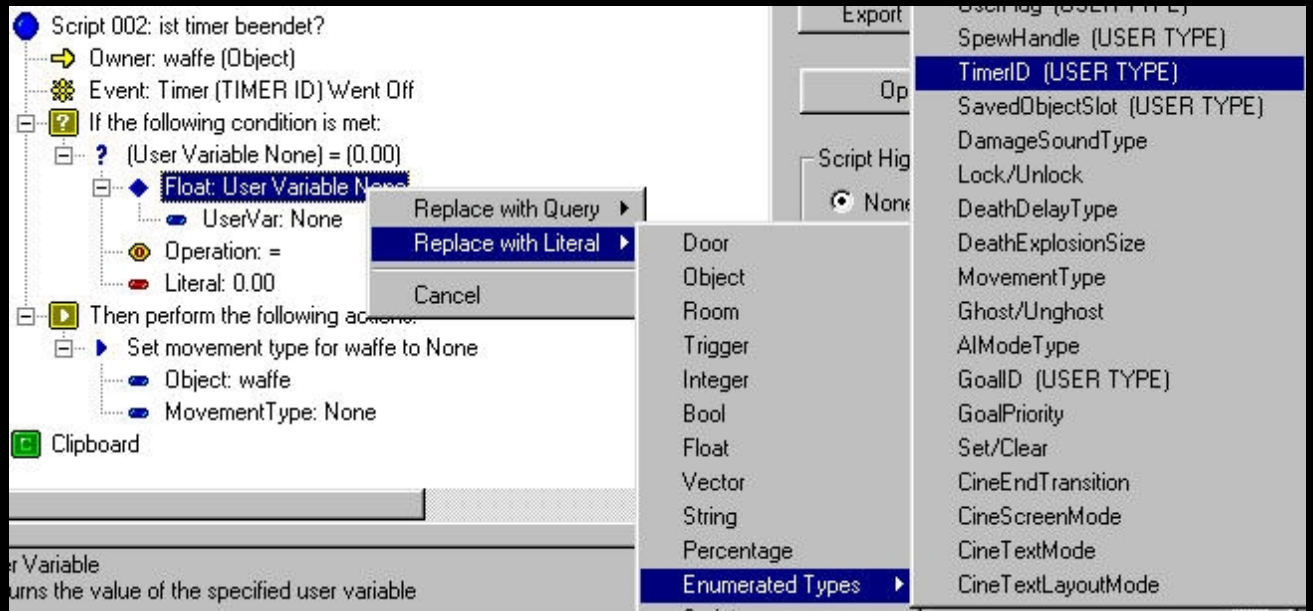
"Generate object...", starts a timer for 30 seconds and gives it the ID "**weapon_timer**". You create this ID as follows: In the script editor there is a button on the right that says, "**Open the User Types Workshop**" click on this and select "Select the User Type to Edit", "TimerID", now click on "**Add New Value**" and select any name related to the timer. I have it "**weapon_timer**" called. Now to the script "has timer ended?".

"If the following condition is met:" To do that in the To find extensive functions, I prepared the following two images:





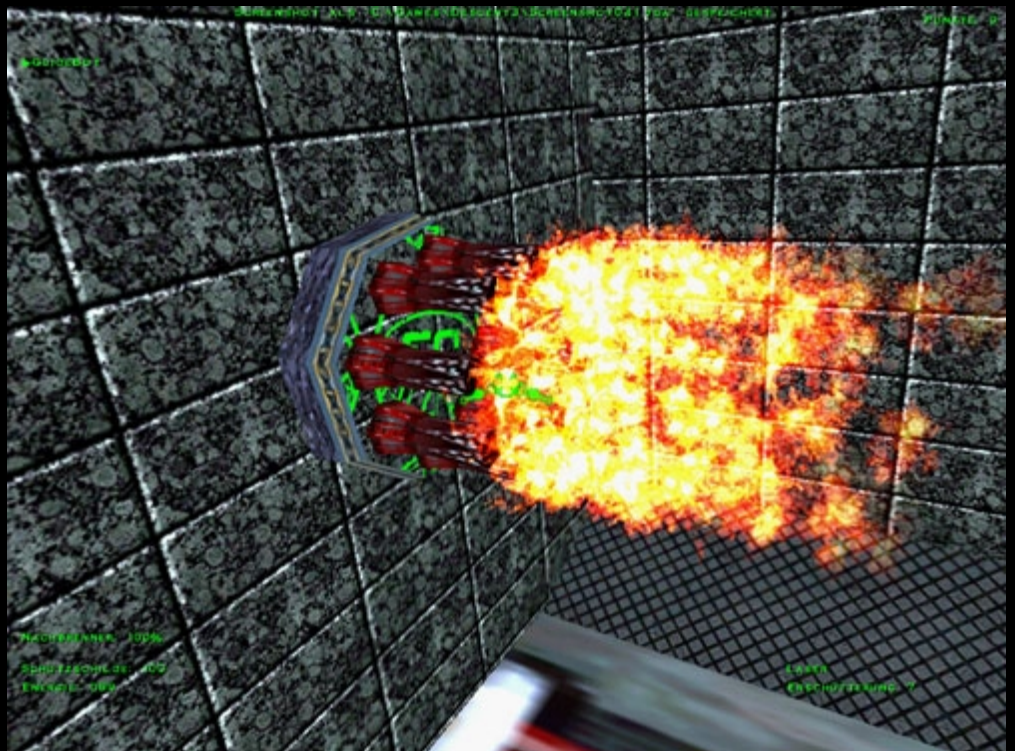
Now right click on "TimerID:None" and chooses "**TimerID**". Now click on "TimerID:None" (below) and your "User type" select. (in my case "weapons_timer").



Now to "Then perform the following actions:". Here the "movement type" for the object "**weapon**" on "None" is set. This says that the object should stop rotating.

OK, that was it. Script is not easy to explain, that's why I took a lot of pictures. Now save it by clicking the "**Save**" clicks. Creates the MN3 file into which you paste everything.
(Your.GAM,Your.ORF,Your.DLL,Your.MSG)

Back to the overview of section H



081 - Custom Robots Part 1 <>

Papacat

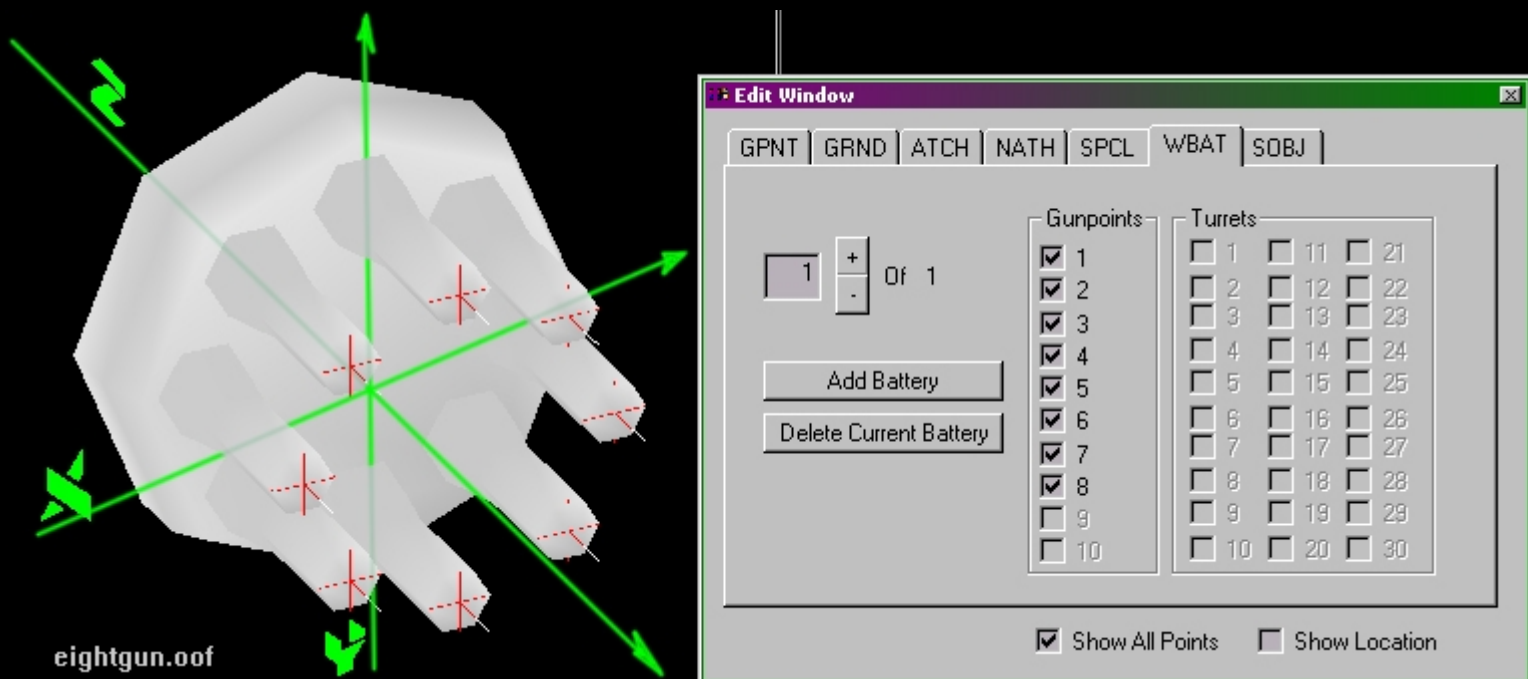
This continues where Custom Objects 3 ended.

When building a robot (or ship) the most important thing is to remember what orientation the object will have. Make sure it is facing the Z axis: when you look in the side view in D3Edit, your object should be facing to the right.

We use the thing from the previous tutorial. First I had to change the orientation. To do this, I went into D3Edit and got it.orfopened and rotated. Then I brought them back into the D3Tool and put the gunpoints back on - see previous tut.

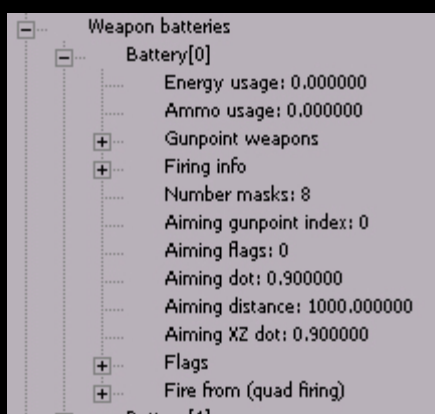
Once you have your gunpoints in you have to create batteries (groups of gunpoints).

Check in D3ToolTools -> Edit(same dialog box as for the gunpoints) and select theWBAT- Tab (Weapon Batteries). Under Gunpoints you should now have an active checkbox for each of them.

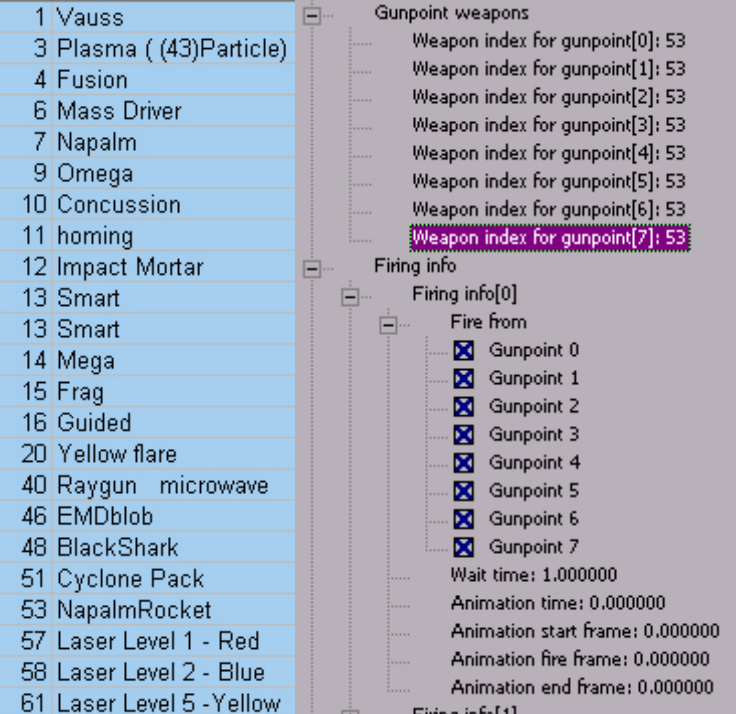


I've checked all eight here. Save (export) the.oof.

Next we'll make the .gam page for a bot, whose behavior is a good starting point for us. I copied the .gam page from a 'RAS1 Light Security Flyer' and saved it as a new file. First I have for the botFlags/Uses Physicsset so he will use this alongside his normal AI to rotate, so the bot will use both AI and physics. He uses AI to aim, fire and maneuver.



Next we need to configure the Weapon Batteries. There we tell the bot what to shoot, from where and how much or often. Open the branchWeapon batteries. Let theAiming-Settings at rest. We need to set the first (and in our case only) battery, so open the branch Battery[0].

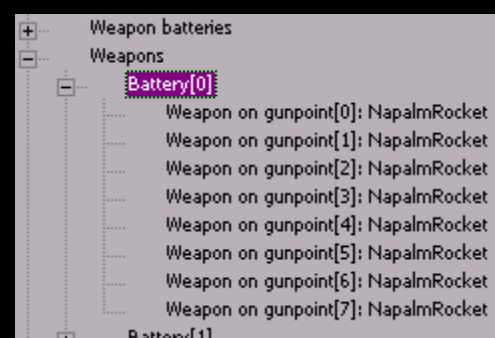


First, open the branchGunpoint Weapons. Here we assign each gunpoint a weapon using the weapon index numbers. (table on the far left)

Where did he get those?

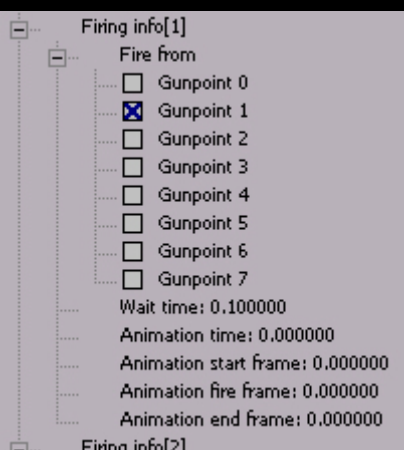
Firing info works in relationship with the Number Masks. These are like a fire sequence loop. 8 masks means there are 8 volleys in each loop.

Think of everyoneFiring info-Entry as a volley. To demonstrate this, we will fire the first of eight volleys from all eight gunpoints using theFiring info[0] as in the screenshot on the left.



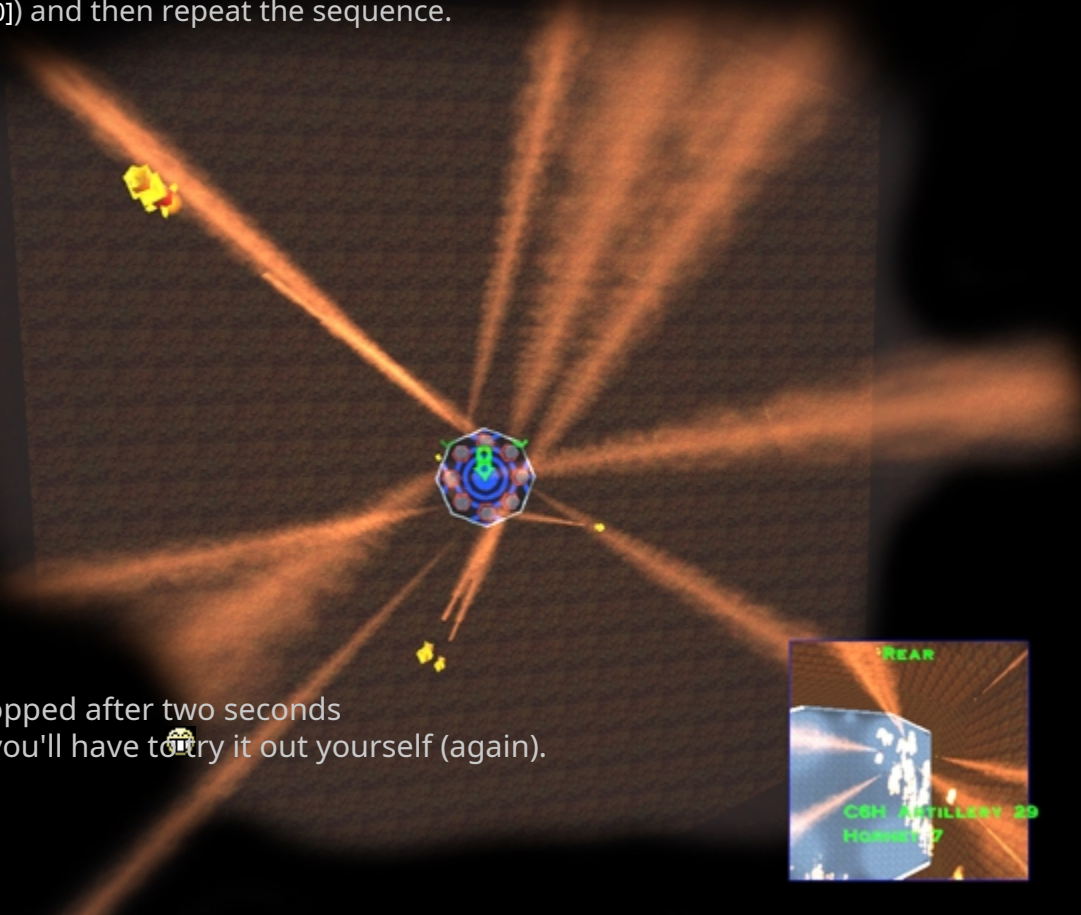
Update:

You also have to go under Weapons/Battery[0] Enter the weapons for each gunpoint as text, also according to the table on the far left.



Then sitFiring info[1]untilFiring info[8]like in Screenshot on the left, with oneWait timeof 0.1 (1/10 Second). Let[1]from Gunpoint 1,[2]from Gunpoint 2... to[7]Shoot from Gunpoint 7.

This setting will fire an eight napalm missile salvo, followed by 7 individual napalm missiles, one second pause, (Wait timein theFiring info [0]) and then repeat the sequence.



It worked, but D3 stopped after two seconds a memory error, so you'll have to try it out yourself (again).

Back to the overview of section H

Custom Weapons Workshop

By Papacat

The quick boiler version:

- 1) Create one.oof for your powerup
- 2) Create one.oof, .ogf or .oaf for use as a shot
- 3) Create a 64x64 and a 32x32.oif for your HUD image and icon
- 4) Copy the GAM page for the weapon and its powerup you intend to replace
- 5) Copy the ship GAMs if you want to make changes to gunpoints, fire details or payloads.
- 6) Change the GAM of the powerup to be yours.oof used. You may also want to change the Ammo Count if applicable.
- 7) Change the .gam of the weapon to use your fire image, HUD image and icon. Here you can also change their behavior, damage, sound and visual effects.
- 8th) Pack along **Add Files** the files in your.mn3

Complete.

What you will learn here:

- 1) The structure of events involving a weapon
- 2) The various files required to produce a weapon
- 3) How different settings affect the behavior of the weapon
- 4) the GAM settings which control almost all aspects of your weapon's behavior.

What you will NOT learn here:

Producing: OOF, OGF, OAF
modding work

I could no longer locate Papacat's original object files. So I created my own and deviated slightly from the originals.

082 - Introduction & Overview

Papacat

Personally, I think a lot of people confuse 'customizing' or using custom elements with 'modding'. Mods include C++ scripting. But there is a wealth of what we can do by creating .oof, .ogf, .oaf files and the GAM file

This workshop is in five parts. The first part (this one) will cover the overview of what makes a weapon. The second will deal with the powerup.gamoccupy. The third will be an overview of the weapon.gam. In part four we'll bring this all together to create a simple primary weapon. Finally we will gameexamine where we will set the firing details of the weapon. I won't go into countermeasures because they aren't fired from a ship, which limits them a lot. Most countermeasures (mines, seekers and gunboys) are actually robots that spawn from invisible weapons (this is done after the weapon.gam make more sense)

A quick overview of the basic definitions:

. oof	Outrage Object File	Physical object model such as a powerup, rocket, barrels, crates, etc... (see previous tuts)
. oif	Outrage Graphics File	A static texture (image). Note that a.oif'can glide', but that doesn't make it a 'real'.oaf.
. oaf	Outrage Animation File	An animation clip, similar to an animated one.gif. The difference between one.oafand a sliding one.oifis that in the latter ONE image is moved (shifted), in the former different images (frames) create the animation.
. gam	Game Table File	The GAM file contains the pages for each element, describing their properties and behavior.

You have toTable.gam(the main GAM file of D3) and the.oofs,.oifs and.oafs that are all involved in weapons. To extract the individual files from the d3.hoguse Hog2WorkShop.

You will need the following tools (in addition to D3Edit, of course):

Hog2WorkShop	Extract the sub-files
gamTool0.61	Editing the .gam file(s)
ModelView32	To edit.ooffiles
OOFEditor	

Once you have installed the tools you can get started.

First, remember that without modding with C++ we have to REPLACE an existing weapon. The biggest downside to this is that we have to keep the original names of the original weapon and its powerup so that a player can use them. If you've ever played Schplurg's Earthshaker 2001 or my level, Vortex 2, you'll have noticed that when you pick up an Earthshaker it still says 'Blackshark'. Even when you see a Hornet on your HUD, it still says 'Cyclone' above it. That being said, we can still have a lot of fun.

Here is a quick overview of the underlying files that make up a custom weapon:

- 1) a.oof for the powerup
- 2) a.oof,.oif or .oaf for the fire image (the shot)
- 3) a 64x64.oif for the image in the HUD (cockpit closed)
- 4) a 32x32.oif for the icon in the open cockpit
- 5) one.gam, which contains the following memory pages:
 - a) a PAGETYPE_GENERIC for the powerup
 - b) a PAGETYPE_WEAPON for the weapon itself
 - c) the PAGETYPE_SHIPs to set the fire details and payload, as applicable (optional)

Of course, you don't need a powerup or the HUD images if the weapon is for a robot.

To understand what these files do, let's walk through the lifecycle of a weapon. This applies more to primary and secondary weapons, countermeasures are slightly different creatures.

- 1) The player picks up the powerup.

This makes the weapon usable for the player ship. Depending on the nature of the weapon, it can also set the amount of ammunition. This comes from the .gam of the powerup.
- 2) The player chooses the weapon.

When the cockpit is on, it becomes 64x64.oif HUD image displayed when the cockpit is off is 32x32.oif Icon. Both images are specified in the weapon's GAM.
- 3) The player fires the weapon.

A 'shot image' (the shot) is generated by the weapon. The image itself is either a .oof, a.oif or a.oaf. Remember, it could just be a picture. The object does not cause the damage itself, the weapon does. Ammunition or energy is deducted from the player.
- 4) The weapon hits a player. Three things happen:
 - a) an explosion – The effect of the weapon hitting.
 - b) Damage to player or object - Shield is deducted from the object hit.
 - c) Impact, the hit - the physical reaction from the hit (example: MD throws you through the room)
- 5) The player runs out of ammunition or energy,

which renders the weapon inoperable until the player obtains more ammo or energy if it is a primary. If it is a secondary, it will no longer be available and will respawn.
- 6) The player is killed

The powerup is spit out by the ship. Always with primaries, only if there are still some left with secondaries.

We have now officially scratched the surface. Pretty boring so far, but now you have an idea of what you have to consider. Now we will look into the GAM's, first for the powerup.

Back to the overview of section H

083 - The Powerup GAM

Papacat

Let's see what we need to set in the page for the powerup. The GAM is of type `PAGETYPE_GENERIC`, it is not a weapon. When a player picks it up (collides with it), the weapon it represents becomes available, and if applicable, ammo is added to the player's payload. To get an overview, let's look at the GAM entry for the Vauss.

Generic Object: Vauss	
Type:	Powerup
File name:	VaussPowerup.OOF
Medium file name:	VaussPowerupMed.OOF
Low file name:	VaussPowerupLo.OOF
Impact size:	0.000000
Impact time:	0.000000
Damage:	0.000000
Score:	0
Ammo count:	5000
Unused string (obsolete):	
Module name:	generic.dll
Script name override:	
<input type="checkbox"/> Has description (obsolete):	
Icon name (obsolete):	
Medium LOD distance:	30.000000
Low LOD distance:	60.000000

Generic Object: The object name
Type: Powerup, Robot or Clutter
File name: The main.oof, fully detailed; as you see it up close (required)
Medium file name: that shown at a medium distance.oof
Low file name: Displayed at long distances.oof
Ammo count: Amount of ammunition added to the payload
Module name: The D3 script that controls the behavior of the weapon (don't touch it!)
Medium LOD distance: The distance from the powerup, where D3 has the medium detailed .oof instead of the main.oofused.
Low LOD distance: From when on D3 the least detailed.oof displays.

If you open the Vauss GAM page, you will see that there is a lot more there. The part shown above is all you need to worry about. Some people believe that in order to rotate the powerup you have to set up the physics block. Not true! Powerups rotate by changing the property `$rotateto.oof` adds yourself.

Above you can see that the GAM has three different ones.oofs indicates. To maximize game performance, Outrage figured that you wouldn't be able to see a certain level of detail from a certain distance, so why let the game draw that. If you are close you will see the fully detailed .oof. Based on the Medium LOD distance shown above, you will see the powerup between 30 and 60 units of distance as less detailed .oof. If you are 60 units and further away, D3 will show you the low detailed .oof.

You can do a single one.oofuse and the name fields for medium and Low Leave empty.

Continue to Weapon GAM.

Back to the overview of section H

084 - The Weapon GAM

Papacat

This section is a reference to understand the GAM page for a weapon. It is a culmination of information from the Outrage Tablefile Specification (from page 518) and various readme files from Aldel, Dark Knight and Peter Runge. Additional input and clarifications have come from SuperSheep and BlackPanther. Thanks to all.

I have tried to make the whole thing more understandable with examples where they could help. If an entry is empty I have NO information. If there is a question mark it is information, but I don't understand it and would appreciate an example to clarify the usage.

One thing that is confusing is when the name of something like the shot image or Particle name is required. Sometimes, as in the case of Fire image, it requires the name of a.oof,.oif or.oaf. In other cases, such as Particle name, it's about the name of a GAM entry in the.gam. I'll try to show which ones.

Note that energy consumption and rate of fire are not covered here. This is specified in the Weapons branch in the ship GAM.

Hud Image:Weapon image shown (64x64) when cockpit is on

Fire Image:The shot that leaves the ship (oof.oif, Or.oaf)

Particle name:Particles coming from behind the shot image (usually a weapon) Example: the 'plasma sparks' behind a Smart's homing plasma blobs.

Particle Life: lifetime of the particle.

Particle size:?

Flags

Hud animated: ?

Image bitmap: The shot image is a bitmap (.if). Example: The Omega

Smoke: There is smoke behind the shot. Example: Concussion Missile

Matter Weapon: The weapon is a physical object. Example Vauss, Concussion (requires ammunition)

Electrical: Electric Storm weapon (Omega)

Image clip: The shot image is a video clip (.oaf) Example: Flamethrower (the napalm)

spray: The weapon is a spray () Example Flamethrower Streamer: 'Band' effect (stripes) behind the weapon. Example: Streaks behind the particles of a Frag Missile

Invisible: Invisible weapon. For a Gunboy, an invisible weapon is dropped which spawns the Gunboy.

ring: Weapon is drawn ring style?

Saturate: Saturate this bitmap weapon?

blast ring: Creates a Blast Ring upon explosion?

Planar Blast: Exploded image always remains parallel to the wall (switch off = the exploded image always looks towards you)

Planar: The gun doesn't always look at you.

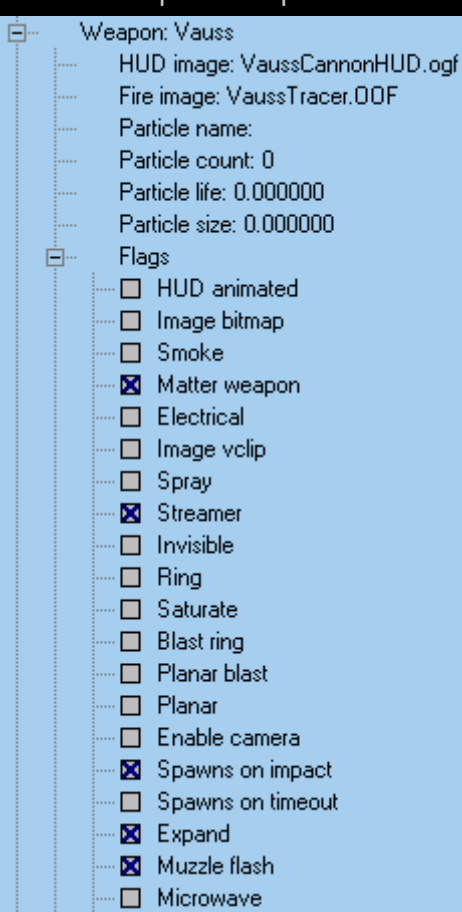
Enable camera: Camera view from the weapon. Example: Guided Missile

Spawns on impact: Spawns a weapon when it hits an object.

Example: Smart Missile

t a weapon when the lifespan expires. Example: Cyclone Id of the weapon increases in size

uer is displayed at the gunpoint from which the 'microwave effect' is shot on objects



- ☐ Microwave
- ☐ Napalm
- ☐ Reverse smoke
- ☐ Gravity field
- ☐ Countermeasure
- ☐ Spawns robot
- ☐ Freeze
- ☐ Timeout wall
- ☐ Planar smoke
- ☐ Silent homing
- ☐ Homing split
- ☐ No rotate
- ☒ Custom size

napalm: Napalm effect on objects

Reverse Smoke: Smoke trail becomes smaller. Example: shrinking rings of microwave

Gravity Field: Creates a gravitational field (BlackShark)

Countermeasure: Weapon is a countermeasure

Spawn's Robot: Spawns a bot instead of a weapon. Example: Gunboy

Freeze: Slows the player hit.

Timeout wall: A wall hit ends like a timeoutout.

Planar Smoke: Planar smoke trail instead of blobs. Example: Homers and Smarties

No rotation: This weapon does not rotate as a bitmap?

Custom size: Uses Custom Size?

Spawn name: Name of the weapon spawning on impact

Spawn count: Amount of spawns (in this case, five sparks spawn on impact)

Alt. Spawn Name: An alternate weapon to spawn.

Alt. Spawn chance: Chance (0 - 100%) that the alternative weapon will spawn.

Gravity Time: How long should the gravitational field last (-> BlackShark)

Gravity size: Size of the gravitational field radius.

Homing FOV: Cosine of the field of view for homing weapons.

Custom size: Custom size of weapon (override polymodel size)

Size: 'Blob' size of the weapon.

Thrust time: How long the weapon's thrust lasts.

Physics Chunk: physics settings (I will explain this separately)

Terrain Damage Size: Radius of damage the weapon deals to terrain.

Terrain Damage Depth: How deep does the damage penetrate into the terrain.

Weapon Alpha: Transparency of the weapon (0 to 1.0, 0=fully transparent)

Explosion image: D3 Texture Name (GAM) of the texture used when the weapon explodes.

Explode time: How long the exploded image exists.

Explode size: size of the explosion

Player Damage: Damage the weapon deals to players.

Object Damage: Damage the weapon deals to general objects.

Impact size: Radius of the effective sphere when the weapon hits and explodes.

Impact Time: How long the impact effect lasts.

Impact Player Damage: How much damage the impact causes to players.

Impact Generic Damage: How much damage the impact causes to general objects.

Impact Force: Amount of force released by the impact explosion.

Lifetime: How long the weapon lives. When its lifespan ends, it does a "TIMES OUT"

Recoil Force: Recoil released when fired (Mass Driver or Mega)

Sound name: D3 Sound name (GAM) of the sound associated with this sound slot.

Smoke name: D3 Texture name (GAM) of the texture for the smoke trail image.

Scorch image: D3 Texture Name (GAM) of the texture used for the weapon's shot tracks.

Scorch size: Size of the shot trail to be left behind

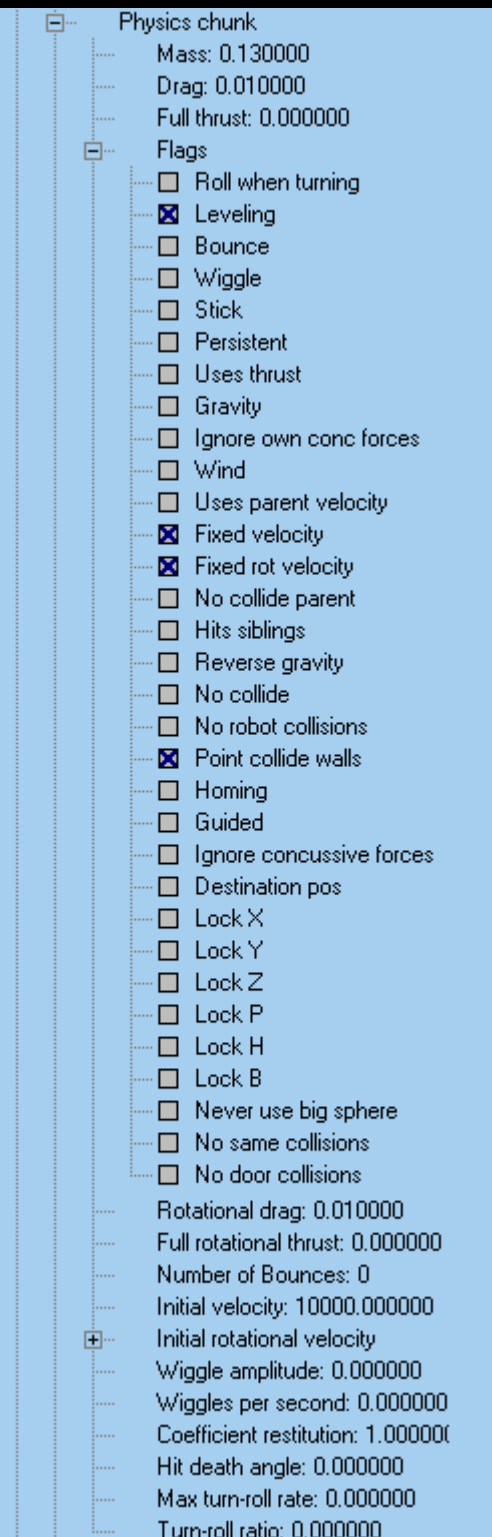
Icon name: Weapon image that is shown when the cockpit is off (32x32)

```

..... Spawn name: Vauss Spark
..... Spawn count: 5
..... Robot spawn:
..... Alt. spawn name:
..... Alt. spawn chance: 0
..... Gravity time: 0.000000
..... Gravity size: 0.000000
..... Homing FOV: 0.400000
..... Custom size: 1.300000
..... Size: 3.200000
..... Thrust time: 0.000000
..... +--- Physics chunk
..... Terrain damage size: 0.000000
..... Terrain damage depth: 0
..... Weapon alpha: 1.000000
..... Explosion image: vaussimpact
..... Explode time: 0.200000
..... Explode size: 3.500000
..... Player damage: 6.000000
..... Object damage: 5.000000
..... Impact size: 2.000000
..... Impact time: 0.000000
..... Impact player damage: 3.000000
..... Impact generic damage: 3.000000
..... Impact force: 500.000000
..... Lifetime: 0.250000
..... +--- Lighting chunk
..... Recoil force: 0.000000
..... +--- Sounds
..... Smoke name:
..... Scorch image: BulletHole011
..... Scorch size: 1.000000
..... Icon name: VaussCannonICON

```


Physics chunk opened



Dimension: Affects the ability to gain speed and turn.

Drag: Affects how quickly the movement of an object slows down (resistance cry -100, hehe)

Full Thrust: Amount of thrust force

Roll when turning: Rolls when it turns 😊

Leveling: Levels the object with the next side?

Bounce: Bounces off walls

Wiggle: Moves up and down (wiggle) while flying Stick: Sticks to walls or objects (does not bounce or be timed out)

Example: Napalm and Flares

Persistent: Object continues to move after collision with another.

Example: merger

Uses Thrust: Use his thrust

Gravity: Is affected by gravity. Example: Napalm, Impact Mortar, Bouncing Betty

Ignore own conc forces: Ignores his own Concussion powers?

Wind: is influenced by wind

Uses Parent Velocity:

Fixed velocity: Constant speed

Fixed red velocity: Constant rotation speed

No Collide Parent: Cannot collide with its parent object

Hits siblings: May collide with his siblings. Example: proximity bomb

Reverse Gravity: Moves against gravity.

No collision: "No collisions AND NO RELINKS -- DANGEROUS TO USE if not used correctly"

No robot collision: Does not collide with robots

Point collision walls: For collisions with the wall, calculate the radius as 0. This means that the only physical point of contact is the center of the object, not its sides.

Homing: This object (weapon) is homing

Guided: This object is guided; Guided Missiles

Never use big sphere:

No same collision:

No door collision: Collision won't open a door?

Rotational drag:

Full rotation thrust: Affects the rotation speed

Number of bounces: Number of ricochet before an impact timeout (explodes or dies). Example: Mortar (2 rebounds)

Initial velocity: Speed before release or spawn before thrust is applied

Initial rotational velocity: Initial rotation speed. Put the

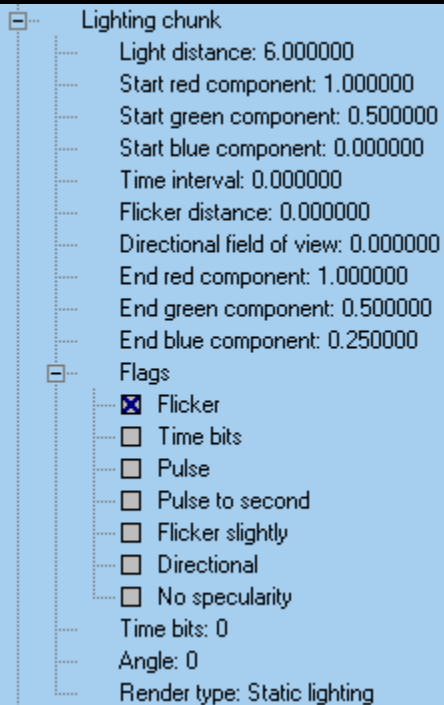
Set the rotation speed for each axis. Open to get access to X, Y and Z

Wiggle amplitude: Amount of this movement.

Wiggles per second: How many times per second.

Lighting chunk opened

Controls the light that an object emits. Example: a Cyclone shot down a dark tunnel. It lights the tunnel purple. Also for rotating lights.



Light distance: Distance that the light still reaches? (need to test)

Start red component: Amount of Red Light (0-1.0)

Start green light: Amount of Green Light (0-1.0)

Start blue lighting: Amount of Blue Light (0-1.0)

Time Interval:

Flicker Distance: Distance that flickering light still reaches? (need to test)

Directional field: Directed field of view: Light cone width 0 - 1.0 (cosine of the angle?):

End red component: ? (usually the same as begin)

End green component: ? (usually the same as begin)

End blue component: ? (usually the same as begin)

Flicker: Light flickers

Time bits:

Pulses: Light pulsates. Example emergency or warning lights

Pulse to second:

Directional: Light is a cone or ray, Directional field must be set

No specificity: Does not have specular light cast upon it?

Time Bits:

Angle: Unused??

Render Type: How the light should be rendered, i.e. how it spreads and appears in the level.

Back to the overview of section H

085 - A simple primary weapon <>

Papacat

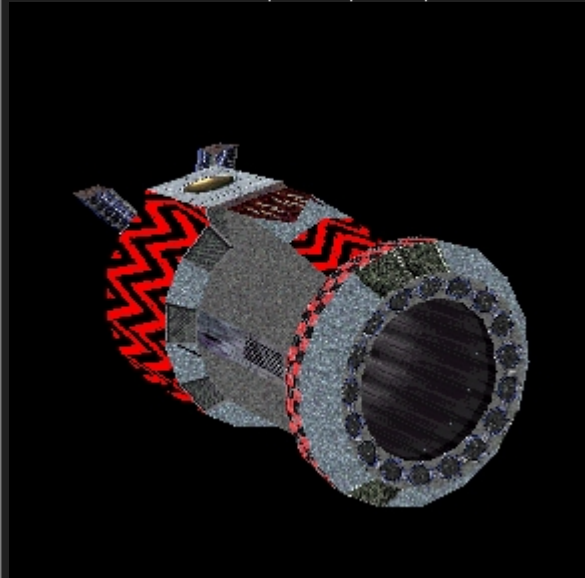
For our primary weapon we will be replacing the Plasma Cannon (weapon name: Plasma). The files required for this are in the.zip, so you save yourself having to build it yourself. This thing is called a ball lightning thrower.

When fired, you will see the ball lightning instead of the plasma blob. It is set for three bounces before it explodes when it hits a wall. When he hits the wall, he spawns another weapon, the Sparkspawn; we will set it so that three spawn. We will set the spark spawns to four bounces before they explode on a wall, or that they will time out after 2.5 seconds.

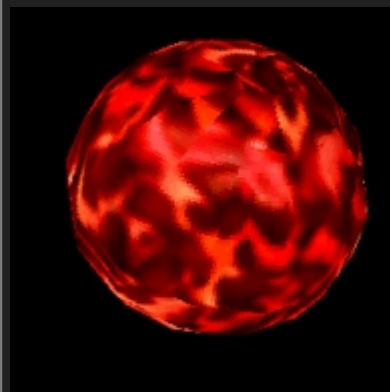
These are the five objects for the weapon:

Sparkballgun.oof

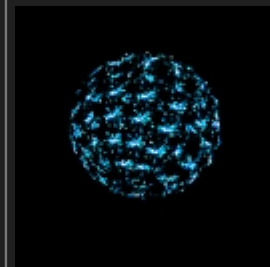
Will the Plasma Cannon powerup be replaced?



Sparkball.oof



Sparkyspawn.oof



SparkyHUD.ogf

64x64 image for the HUD image the weapon

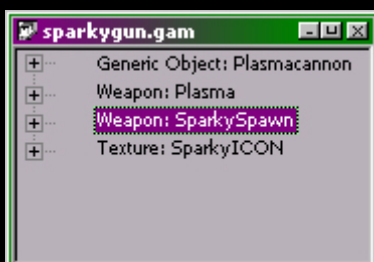


SparkyICON.ogf

32x32 image for the image of the Weapon without cockpit



We need to create four GAM pages:



Generic Object: Plasma Cannon: The powerup.

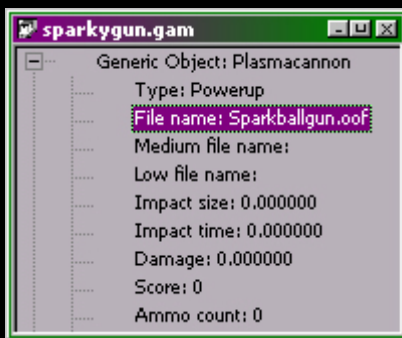
Weapon: Plasma: The weapon we will replace.

Weapon: SparkySpawn: The weapon to spawn. Note that since we don't shoot them from the Pyro, it can be any name (copy of frag particles).

Texture: SparkyICON1: The texture for display when the cockpit is deactivated

Note: In the screenshots here you can see the entry without the '1' everywhere; but then the whole thing doesn't work. So stick to the text!

Generic Object: Plasma Cannon



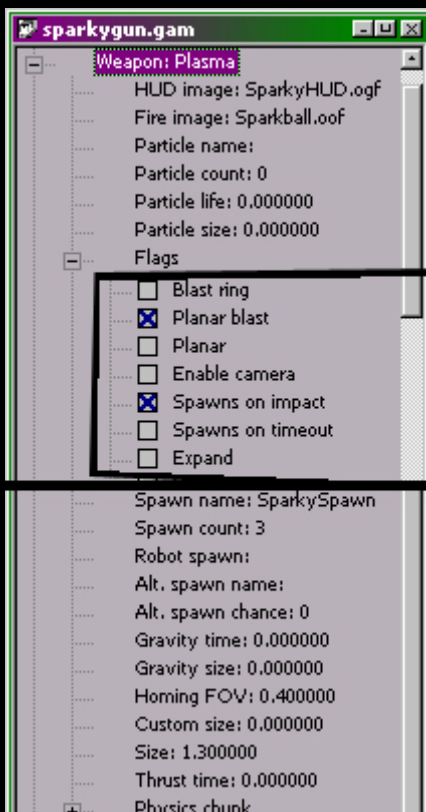
We need the plasma's powerup GAM on ours.oofchange.

I have 'Sparkballgun.oof' specified as the file name. Make sure you delete the Medium and Low file name entries.

Now we see the Ball Lightning Cannon instead of the Plasma Cannon powerup.

Weapon: Plasma

I have left out settings that we do not cover in these graphics in order to reduce their size.



JobHUD image on SparkyHUD.oof; that is the actual one.oif-File, no 'Texture'.

SetFire image on Sparkball.oof

TheParticle namedelete it (if not empty) so that there is no trace of sparks.

Planar Blastleave set

Spawns on impactturn on.

Put thatSpawn nameon Sparkyspawn. Note: I include the GAM name here, not the.oof!

SetSpawn countto 3. Three of the blue balls will spawn when the shot hits the wall and explodes.

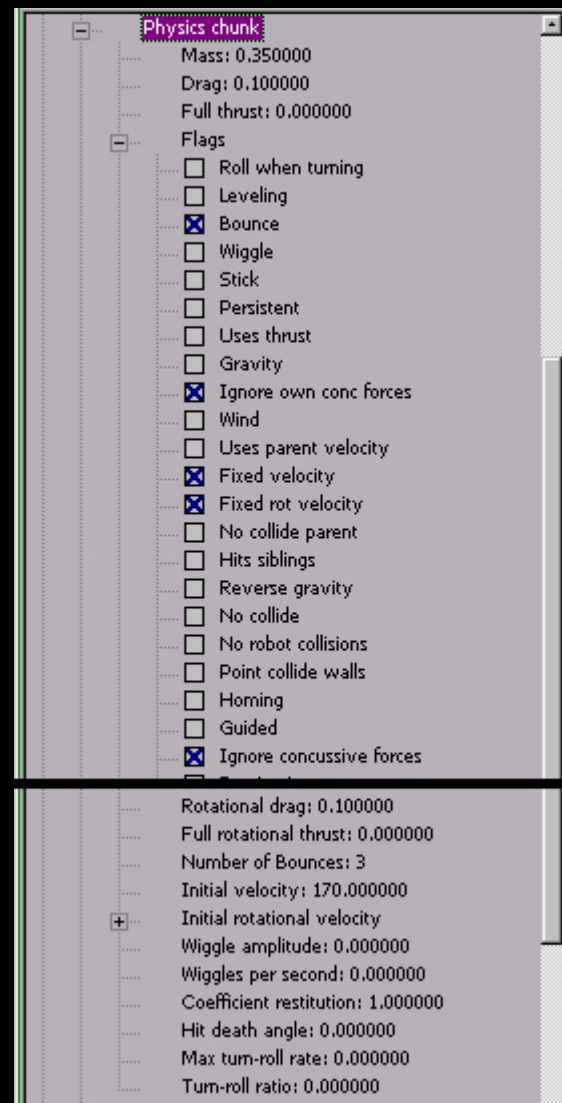
Bounce: The shot will ricochet off the wall a certain number of times before exploding when it hits the wall.

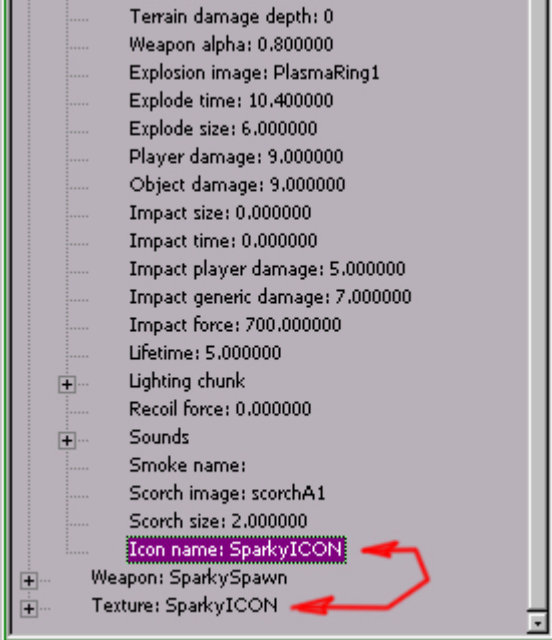
Fixed velocity: Speed remains constant.

Fixed red velocity: The rotation speed will not change.

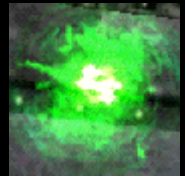
Number of bounces: How many times should it bounce off the wall. The fourth time you hit the wall it will explode.

Initial velocity: The speed at which the shot leaves the weapon.



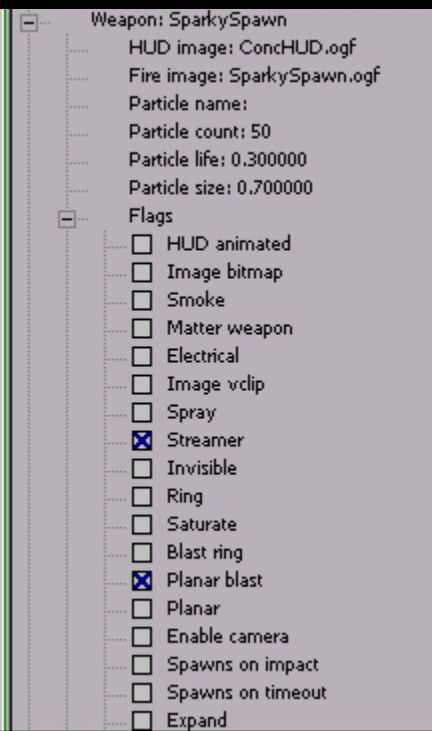


The `Explode time` increased to 10.4 for three reasons: First, you can see that a `Planar Blast` remains parallel to the wall. Second, how the setting keeps the exploded image visible. Third, like the flag `setExpand` the explosion image can be expanded.



The `Icon` name set to the GAM entry of your texture, `SparkyICON1` that's on the actual `.oif` shows (double arrow). You see them when your cockpit is switched off.

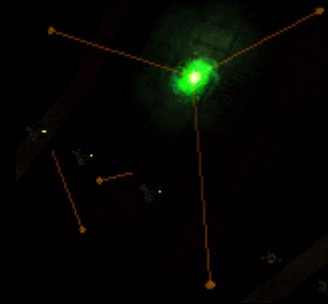
Weapon: SparkySpawn:



Change `Fire image` on `SparkySpawn.oof`

Delete `Particle name`. Without the particles, the streamers are obvious.

`Streamer-Set` flag. On the right you can see a stripe-shaped trace in the shot image, that is the streamer.



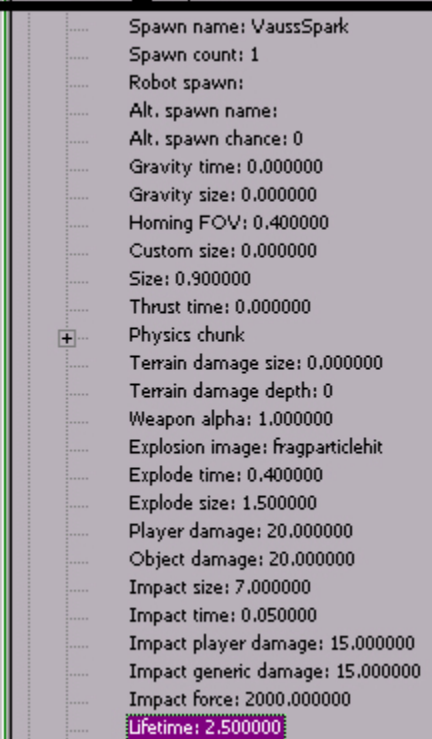
`Planar Blast` start. Note, the green explosion in the top right is the explosion of the plasma, not the `sparkyspawn`.

`Spawns on impact` and `Spawns on timeout` leave out, so no weapons more are generated.

Note that `Spawn name`! It doesn't matter here because none of the spawn flags are on.

(Not shown here) in `Physics Chunk` set the flag `Bounce` and `Number of Bounces` on 4.

Set the `Lifetime` to 2.5 so you have time to watch it bounce.

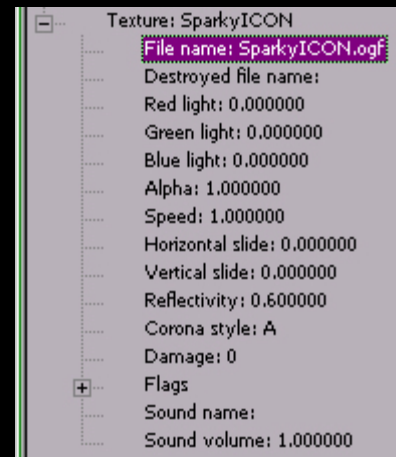


Texture: SparkyICON1:

Ultimately, to have a display on the HUD without a cockpit you have to create a texture.

Enter the name of the file name.ogf'sat.

Otherwise there is nothing to set here.



That's it. You can try the whole thing out by:gamand put the five objects into a level where there is a Plasma Cannon. Don't forget that for the....mn3and the.gam
The same name must be given. If the level already has its own.gamjust add the four GAM pages there.

Boom!

Of course you can now get lost in the details... create and integrate your own weapon sounds, tinker with the settings... etc...

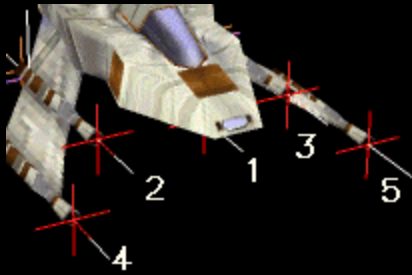
Back to the overview of section H

086 - The ship's GAM

Papacat

Now that we know how to control the weapon, we need to look at how to control its actual firing from the ship. The ship GAM pages are where we choose which gunpoint to fire from, the rate of fire, maximum payload, etc.

The weapon setups on ships are similar to those on robots. However there is a BIG difference, ships only use one battery, so everything is specified in the weapon entries.

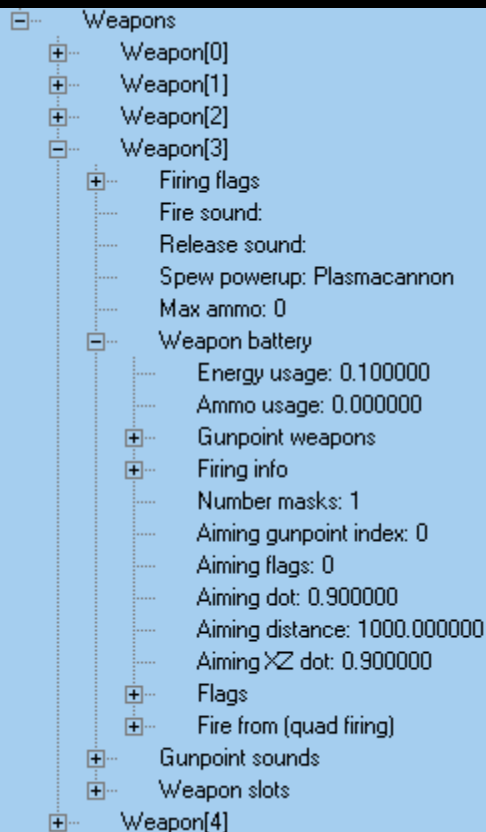


Since a lot of this relates to gunpoints, I included these diagrams of a Pyro.



Every weapon used by the ship is listed under Weapons.Weapon[0]untilWeapon[4]are the first five primary weapons,[5]...[9]their second selection (2=Vauss, 6=MD).[10]until[14]and[15]until[19]are the secondaries,[20]is not used. Countermeasures are scripted.

As you can see, not everyone uses weaponsFire sound. Most useGunpoint sound, which causes the sound when firing to come from the gunpoint (usually just one, even when firing from several).FireandRelease soundare used for effects such as loading and releasing the Fusion or locking and releasing the MD.



Firing flags: Open to make Fusion or MD special effects. Here you can also set that ammunition is counted in tenths.

Spew powerup: The powerup that the killed player gives off.

Max Ammo: Maximum ammunition load of the weapon (in the weapon battery [Vauss, MD, all missiles])

Energy Usage: How much energy the weapon draws from the player.

Ammo Usage: Ammunition consumption for weapons that use ammunition.

Gunpoint Weapons: Here you assign a weapon to a gunpoint by its index number. This makes them 'available' for use only.

Annotation! Wolf On Air has shown that this may be an obsolete area that does not need to be addressed. TESTING!

Firing info: Where the actual firing is set (See opened firing info).

Number masks: Think of masks as volleys. This sets the number of volleys that occur per repeated firing sequence.

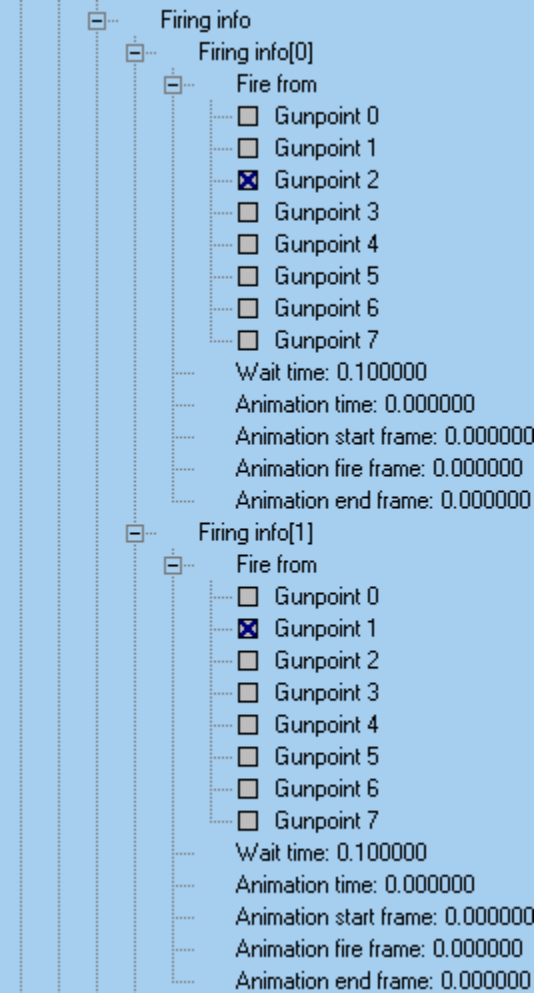
Aiming flags,-dot,-distance,-XZ dot: I do not know 😊

Flags: (see open flags)

Fire From (quad firing): Here you set where the fire was fired from when the QuadLaser was recorded. Concerns onlyWeapon[0]unless you're doing some heavy scripting.

Gunpoint sounds: Here you set the sound that a weapon makes

Firing from a specific gunpoint makes. If the same weapon is fired from two gunpoints at the same time, assign a sound to only one of the gunpoints.



Weapon slots: Here you assign a weapon to a gunpoint by name. This just makes them 'available' NECESSARY!
 Here you set the release properties for a mask (a salvo). You have to set a firing info for each mask. On the left are two firing info's that are used when firing the Vauss.

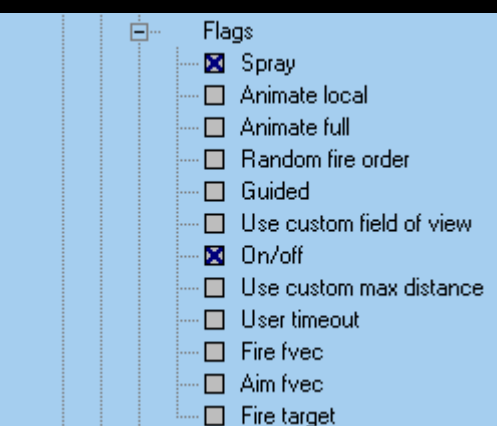
Fire from: Here you set which gunpoint this particular volley will come from.

Wait time: The amount of time that passes until the next volley is fired. If it is the last salvo/mask, it is the time until the sequence repeats. This also prevents the weapon from being fired again before the time is up (example: MD).

Animation...: Info for the animation of the ship. The animation time begins when the volley is fired. The animation fire frame (at which point in the animation the weapon is fired) does NOT work on ships (but does work on robots). No matter what you do (unless you script accordingly), it will ALWAYS fire at frame #1, even if frame #1 isn't in the animation frames at all.

Looking at the Vauss's fire info, we see that it fires from Gunpoint #2, waits 0.1 seconds, then from Gunpoint #1, then repeats.

This is how the Black Pyro gets its plasma 'bursts'. It uses eight masks with a shorter wait time than the standard Pyro. The last mask made the waiting time significantly longer. This causes him to fire eight quick volleys, pause, and eight quick volleys again.



I haven't tested this area of the Battery, but if you look into the napalm launcher or the Omega you'll get an idea of what it's all about. Most non-animated weapons do not use these flags, except, of course, the Guided Missile.

I hope these tuts have given you a good foundation for building your own custom weapons. There's more you can do with heavy scripting, but this is something I doubt I'll ever have time to explore.

Good luck!

Back to the overview of section H



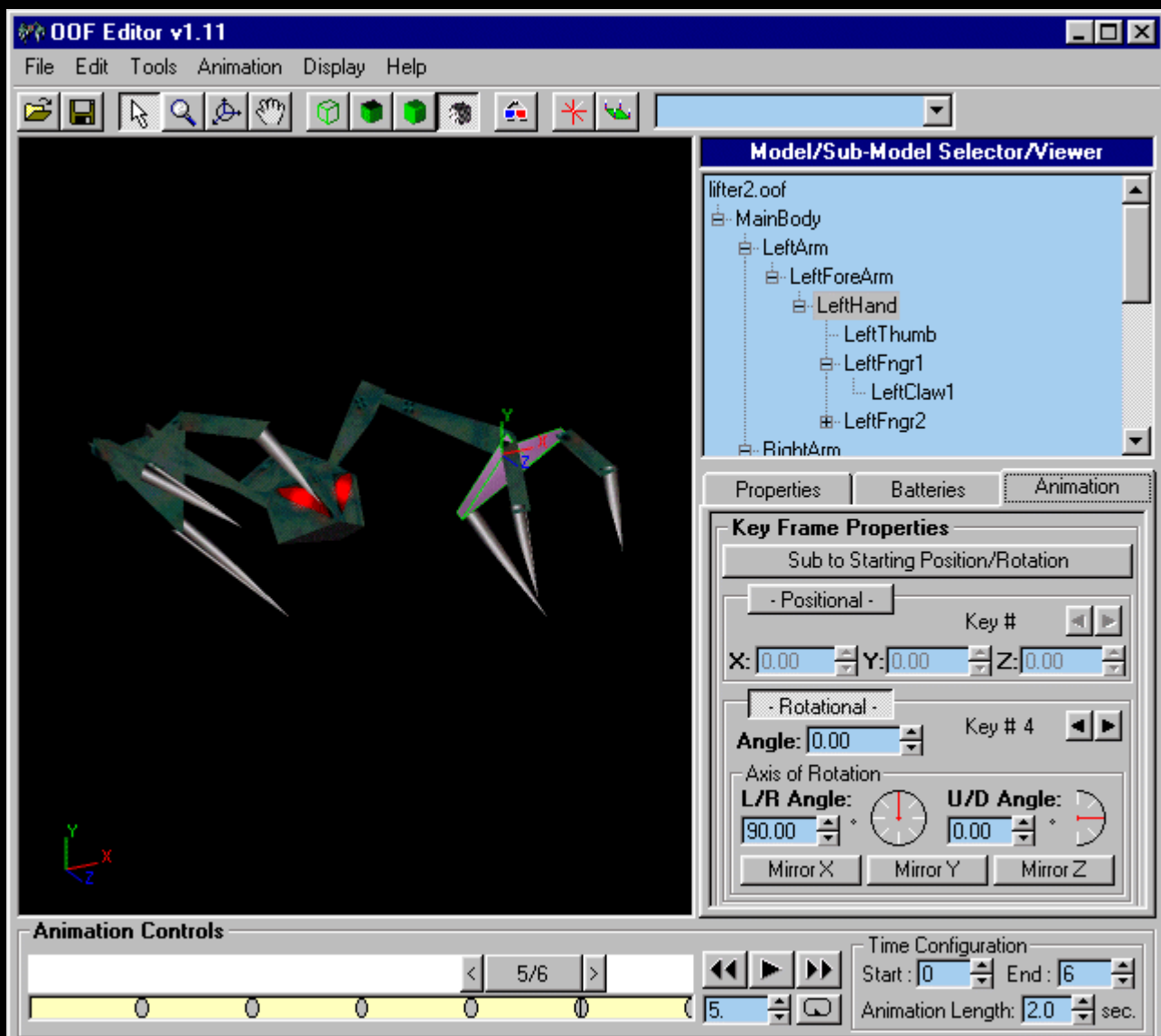
Excursus

By Papacat

OOFE Editor makes it possible to build complex objects, including sub-models, in D3Edit. You don't need an expensive modeling program. OOFE Edit converts that.oorf from D3Edit, where you create the complete model.oorf, complete with the submodels and in less than a minute (I stopped) you are ready to set properties, set gunpoints, add points, define weapon batteries and fully animate the .oorf. OOFE Edit shows you your .oorf also with custom and animated textures.

Not only does OOFE Editor give you the power to push level design to new limits, SuperSheep has also loaded this program with intuitive tools, making them easy to understand, learn and use. Remember, if you are new to 3d modeling/animation, there are concepts you will need to learn. Some of the more advanced animation concepts can be a little tricky. Luckily, OOFE Editor makes it easier to understand what you're doing and why.

Here is a screenie from OOFE Editor with my little animal, the lifter 'Max', when I animated him. SuperSheep was kind enough to use Max as the logo for the editor.



From the moment I met Max.^{orf}When I opened it in the OOFEditor, it took less than 20 seconds to completely structure the sub-model hierarchy. Less than two minutes to relocate all 16 pivot points exactly where I wanted them. The glowing eyes each took ten seconds - including color assignment and adjusting their radii.

I finished animating Max after 20 minutes (6 frames, 10 keyframes each).

After the installation, OOFedit only needs to know where you want it^{d3.hog}have lying. Once that's done, you can use the drop-down box at the top right of the program to do whatever you want.
^{oof}from D3. The program can be accessed directly from the^{d3.hog}and.^{mn3}-Read and extract files. No, you can't go straight into one.^{hog}or.^{mn3}save it back - you have to do a little work yourself.

We'll start with a simple one-model .oof, a powerup that the
\$rotation-Property used to rotate in D3. Then an animated switch. From there - robots with gunpoints, glowing eyes and (thrusters), and as much animation as D3's Fusion Engine will allow.

(Papacat recommends adding an .oof with SuperSheep's OOFtoORF.^{orf}to convert even though OOFedit is a.^{oofas}.^{orf}can export.)

087 - OOFEdit: Terminology

Papacat

OOF: Outrage Object File - This file contains all submodels, properties and Animation information for a complete model or 'scene' as some 3D programs call it.

Sub model: An individual part of a model, such as the handle of a switch or the hand on one arm. Note that a submodel can simply be a single face. In fact, glows and gunpoints are such; one-facial submodels with special properties assigned to them.

Pivot point: The origin point of a submodel. From this point the movement becomes determined when a translation (shift) or rotation key is assigned. When you bend one arm at the elbow, the elbow is the pivot point of the forearm. The wrist would be the pivot point of the hand. You can set the pivot point arbitrarily in relation to its sub-model.

Parents, Children and Syblings: This is how we refer to the relations among the sub-models are connected (logically, not physically) An arm is the simplest example: A hand is child to forearm. This one is Parent at hand. Fingers are individual children of the hand and are syllables to the other fingers of this hand (same parent).

frame: animation increment.

Key: A description of a movement that a sub-model performs within a frame. There are two types of keys: **Translation** and **rotation**. **Translation** is the movement from one X,Y,Z coordinate to another, somewhat like a piston moving upwards. It would still need a translation key to make it move back again. **Rotations**-Key rotates a sub-model around an axis by a certain amount of angular degrees. Examples would be a door that turns on its hinges, a rotating powerup, or an arm that bends at the elbow.

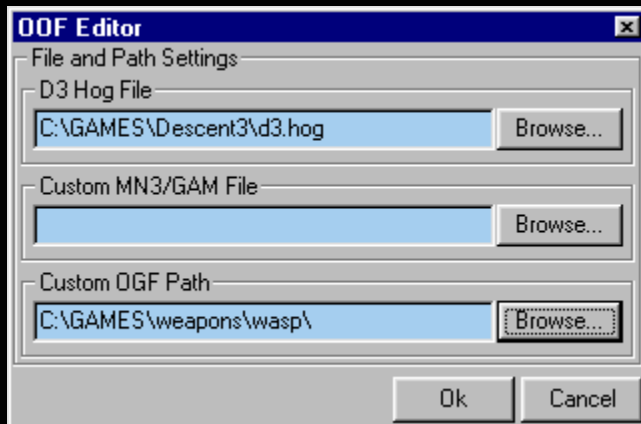
GunPoint: A point from which a weapon is fired or an object is spit out. It is created by converting a submodel that consists of only one face to a gunpoint.

Glow: A point from which a halo of light emanates.

Back to the overview of section H

088 - OOFedit: Features

Papacat

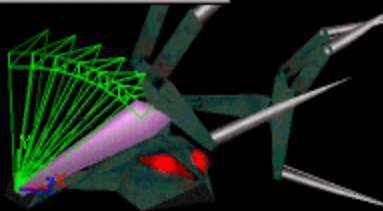
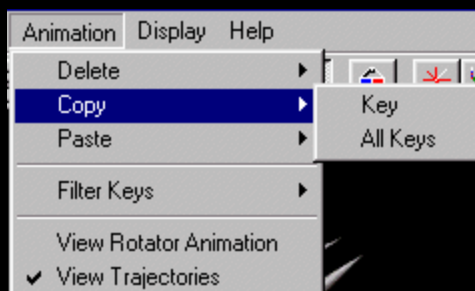


Once you show OOFedit where the d3.hog is located, it not only shows the contained objects with the D3 textures, but you can access every .oof from the d3.hog, play its animation or export it.

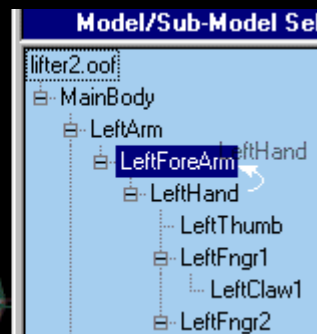
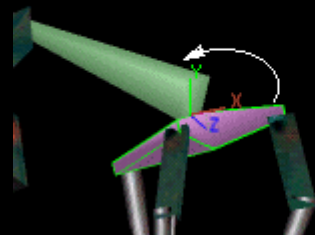
If you specify a Custom MN3 and a .gam, you can also access objects containing these files.

Or point to a folder that contains the .ogf's to have them available in the OOFeditor.

Quickly and easily prepare your .oof's animation by setting up the sub-model hierarchy using drag-n-drop. Simply drag the child to the intended parent. You can do this in the model/submodel selector/viewer, or directly in the display. Highlights make it easy to determine what you're doing. The selected child is purple. The potential parent is green.



used.

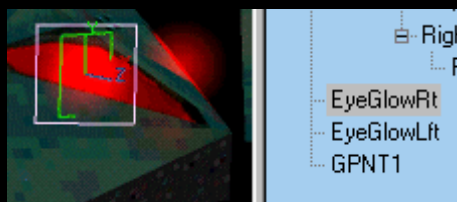


You can copy keys of a sub-model from one frame to another frame.

You can even copy all of a submodel's keys at once and paste them into another .oof.

Turn on View Rotator Animation to check rotation property settings. View Trajectories shows you the path that a sub-model takes in a frame.

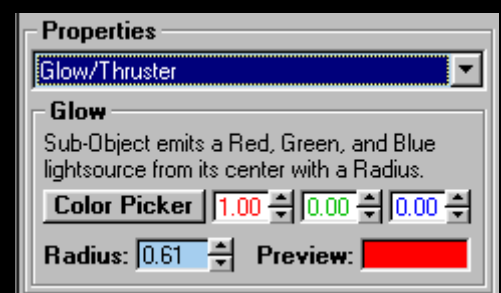
Here is the animation controls toolbar. Set Time Configuration for the frames you want to animate and press the play button. You also use them to add frames. The key indicator shows you whether the selected submodel has an animation key in this frame or not and is also used during the copy/paste process



Color picker select from a color palette. Use the radius to determine the size.

Glow's are just one-facial sub-models with the Glow property.

Select the Sub-Model and select Glow/Thruster from the Properties list. You can adjust the color by adjusting the RGB values, or via a



Back to the overview of section H

089 - OOFEdit: Getting started

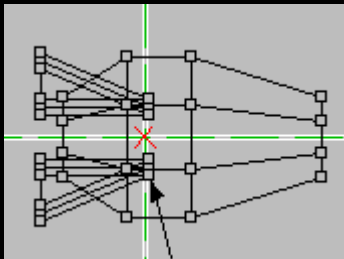
Papacat

You must build the ENTIRE model in D3Edit before importing it into OOFEdit. This includes faces that need to be converted to points (gunpoints, attach points, etc...) Aye, this means if you want to add something later, you have to do it in D3Edit and the entire thing.orfrre-import again. The good news is that OOFEdit is so simple that you can quickly get back to where you left off. A main point when using OOFEdit is good preparation in D3Edit.

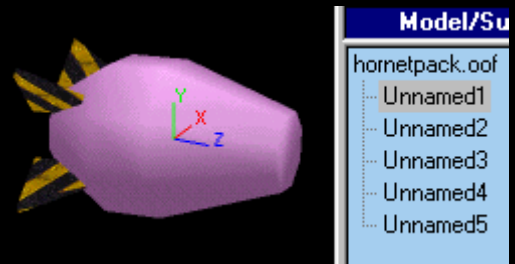
Prepare an ORF:

1.) OOFEditor determines which faces make up a single submodel based on the connection of vertices. This means that if faces are connected to vertices, they will be part of the same submodel. This makes it easy to construct submodels in D3Edit - just don't connect them. A good example is the Finns on my Hornet powerup. The fins were built sideways and then pushed through the main body. None of the fin verts are connected to any from the body, so I get a total of five submodels when I open it in OOFEdit. Don't worry - if you want them to be a model, you can pair them in the OOFEditor.

Here is the Hornet in D3Edit. Note that the fin's verts are not connected to the body's verts.

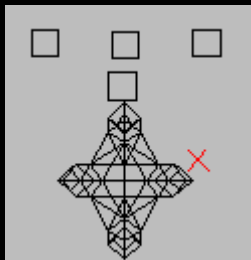


Here is the Hornet in the OOFEditor. You can see five sub-models in the model/sub-model tree.

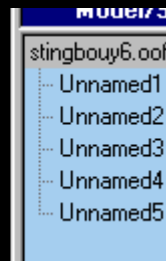


2.) Make sure you have yours for each GP, Glow, etc.oofshould contain, create independent faces.

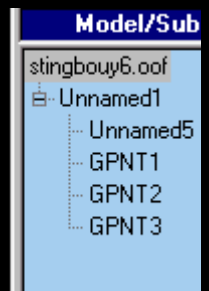
Here is the Wasp. I added one face for a glow and three for gunpoints.



Opened in OOFEdit. Note the five submodels.



Completed in OOFEdit. It took five minutes. I'll show you how to make gunpoints and glows later.



3.) Make sure ALL texturing and texture alignment is completed in D3Edit

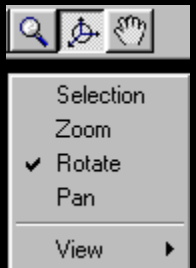
4.) Center the model at 0,0,0 as you want in OOFEdit. This affects the collision radius of the object. No problem if you forget it, you can always center it from within OOFEdit. You can also change the radius size once you're in OOFEdit.

Getting the .orf into the OOFEditor

- 1.) Start OOFEdit, go to File -> Open OOF/ORF and choose yours.orf.
 - 2.) Use File -> Save or File -> Save as around your space as .oof to save. Save creates one .oof-File with the same name as yours.orf in your folder. orf's Save As lets you give it a name and determine the storage location.
- That's it – Congrats – it's an OOF.

navigation

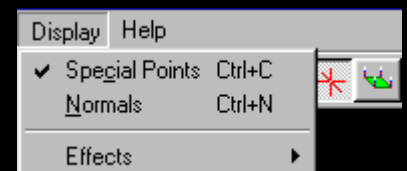
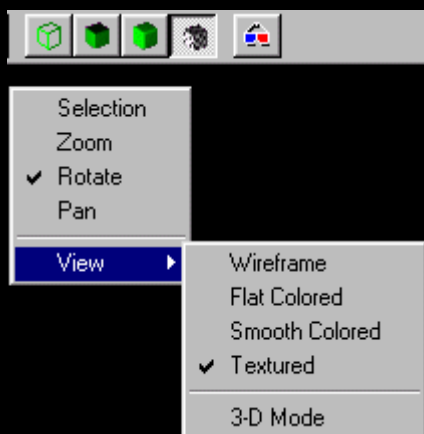
You can maneuver around with zoom, rotate and pan. You select it either from the toolbar or by right-clicking in the display and selecting it in the following pop-up menu (my preference). Both here on the right.



View options

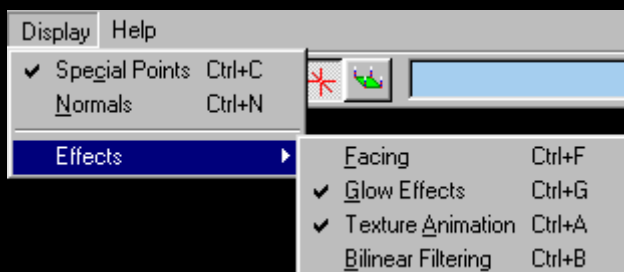
You can view the view as wireframe, solid, smoothed or Choose textured by pressing the corresponding button in the toolbar or the View-Select option in the context submenu. If you have anaglyph lenses, click on the glasses.

Turn 'Special Points' (Gunpoints, etc...) and Normals on or off using this display menu or use the toolbar.

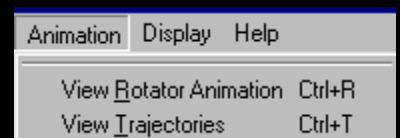


Effects

Activate their display by selecting below Display -> Effects.

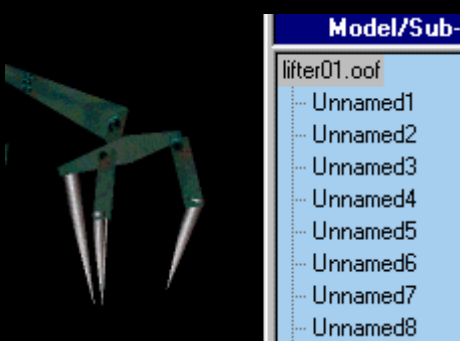


To see the rotation of objects using the \$rotation property go to Animation -> View Rotator animation.



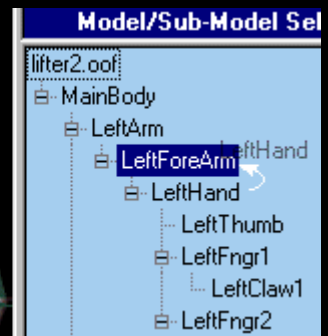
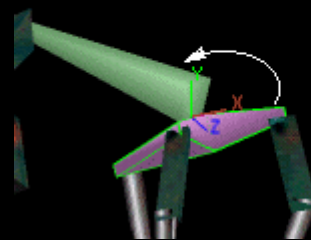
Parent/child relationships and union

Don't confuse these with each other. Although logically related, Parents and Children are still separate sub-models. When you combine two sub-models, they become one submodel. There is no undo for Combine, so save often.



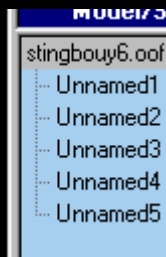
If the .orf was freshly imported, the sub-models are independent of each other.

Drag/drop a submodel you want to become a child onto the one you want it to become a child of. This works in the model tree as well as in the display. Highlights make it easy to control what you're doing. The chosen child is purple, the potential parent is green.



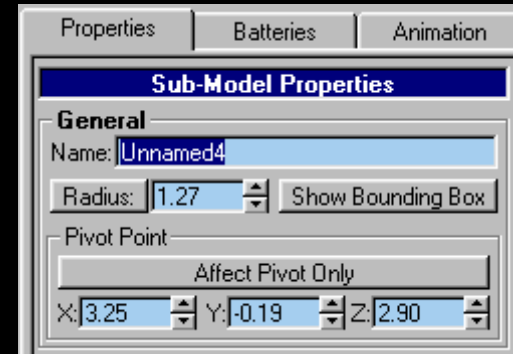
To merge submodels, hold Shift while dragging/dropping.

Gunpoints

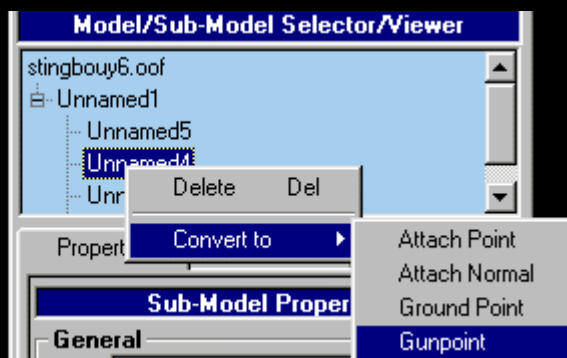


Give yours or independent faces to convert into gunpoints with.

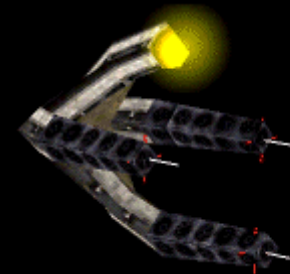
Use the X, Y and Z selectors to move them into position. However, you can also position them in D3Edit.



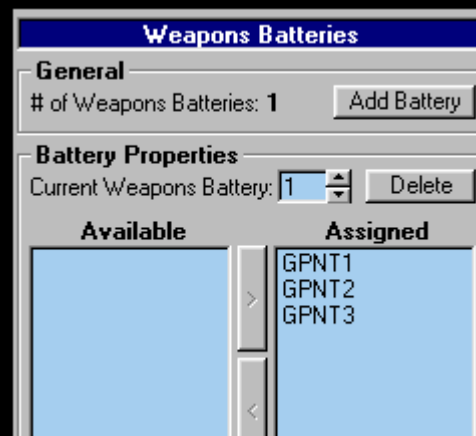
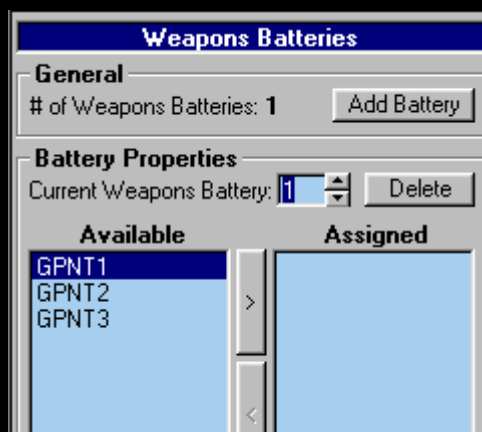
In order to be converted, a sub-Be someone else's child first. Use the method described previously for this. To convert a submodel, right-click on it in the tree view and select **Convert to -> Gunpoint** from the menu.



Complete.
As simple as that.



Don't forget that D3 can't use a Gunpoint until it belongs to a Battery. Go to the **batteries-Tab**. Click **Add Battery**, the available gunpoints become listed. Mark a gunpoint on the **Available**-page and click the **>** button. The same way you would make batteries for turrets.



Back to the overview of section H

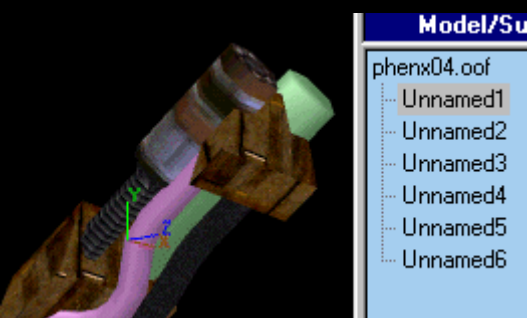
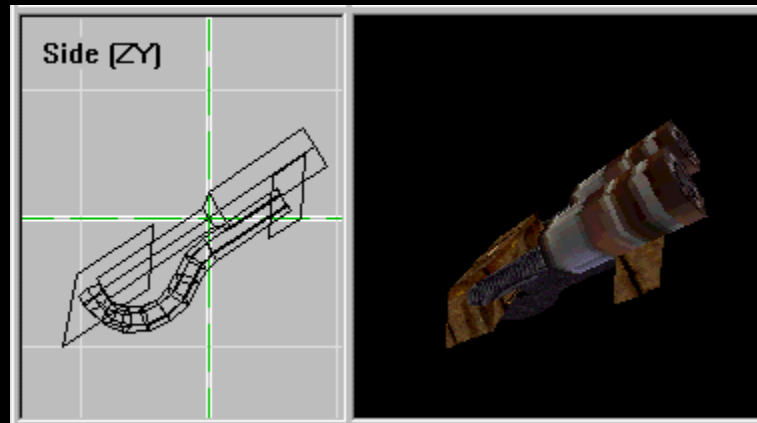
090 - OOFEdit: A powerup

Papacat

Here is a cannon that I use as a powerup.

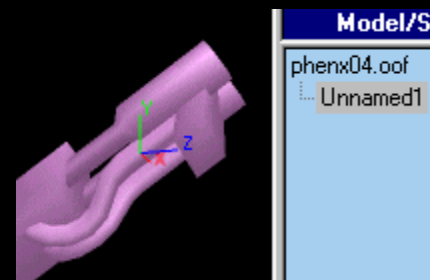
I completed the modeling work in D3Edit, texturing included. The model consists of six different parts that are brought together but not connected at verts.

I tilted it and centered it pretty much how I want it. I could have adjusted it in OOFEdit, but that works too.



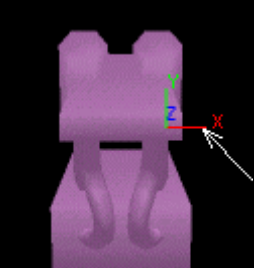
When I open it there are six sub-models. Hold **Shift** down while you set all submodels to 'Unnamed1' drag drop.

This unifies them into a single submodel. (right)



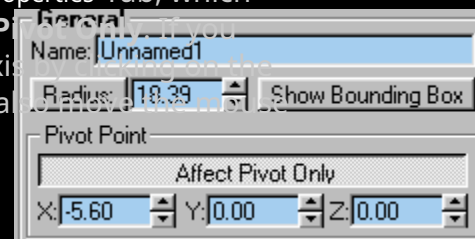
It's a good time now
SAVE. There is no undo. Your best defense

Self-inflicted injuries often require saving and making multiple copies.

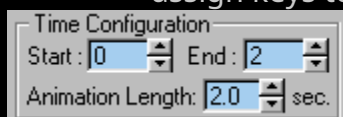


move.

I noticed that I ~~or~~ wasn't centered correctly before I imported it. On the left you can see the pivot point for the submodel, this is where the axes around which the object rotates originate. It's easy to move. Choose the one **Properties**-Tab, which shows you the sub-model properties dialog. Click on **Affect P** don't do this, the entire submodel will move. Adjust the X-axis spinner. You can click/hold one of the triangular arrows, or a



Once you have the pivot point where you want it, it's time to add frames and assign keys to them.

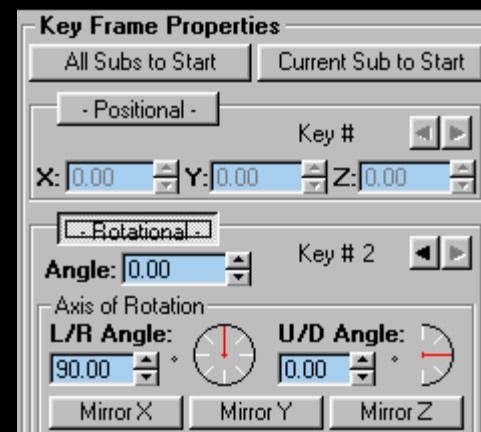


First we need frames which will contain the keys. In the lower right corner of the program you can see the **Time** configuration, the timing control. Leave Start at 0 and change End to 2. You will see that the frame marker now shows 0/2. 0 is the active frame, 2 is the total number

existing frames.

Now add animation keys to the frames. These tell you how the submodel should move. There are two types: rotational and positional. Also called Rotate and Translate. We will use rotational keys.

First, go after **Key Frame Properties** in the animation tab. Select the submodel while the frame marker is at 0. Click **Rotational**.



You will see an oval being inserted into the framebar. That is, the selected sub-model has an animation key in this frame. Move to frames 1 and 2 and add a rotational key to each frame.



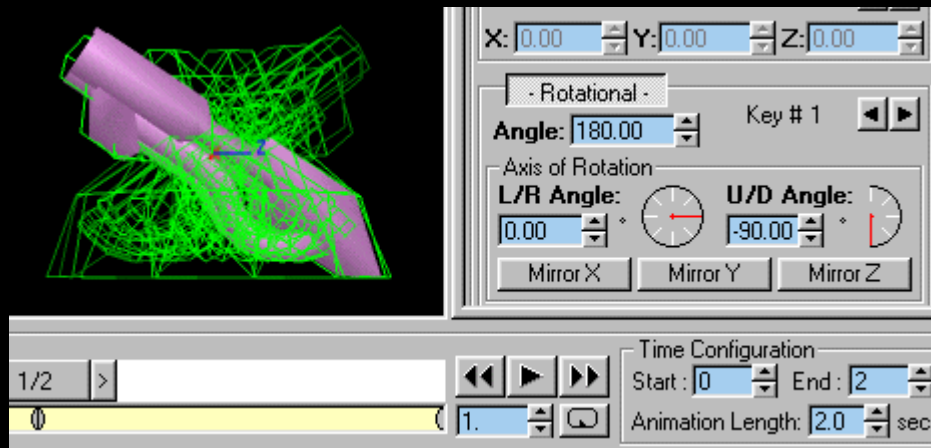
Before you do anything else, follow up animation in the program menu and switch View Trajectory. Set the frame marker to 1. The Angle (=angle) is the amount of rotation the object will perform. L/R (Left/Right) and U/D

(Up/Down) angles set the axis around which the object will rotate. You can change L/R and U/D either using the spinners or by dragging the angle indicators (red ones

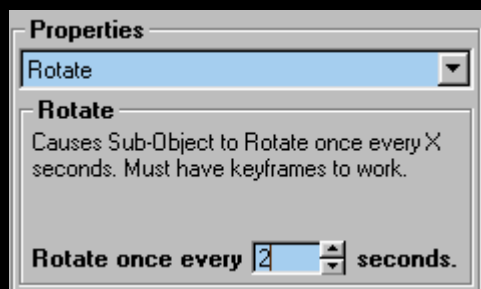
'pointer'). If you **Shift**

If you press it while pulling, they move in 45° increments. You can also enter a value directly. Set the angles as shown at right.

Move the frame marker further to frame 2 and specify the same settings. The green outline you see is the trajectory. It shows you the path the submodel will take when moving. Save so nothing is lost.



Let's play the animation to see what our .oof does so far. Press the button with the drawn loop (just to the left of the Time Configuration), this is the loop control. If you then press play, the animation will be repeated continuously. Press play. Your .oof should rotate while the trajectory trace is displayed. You can do this in the animation menu



switch off if you want.

With the submodel still selected, go to the Properties tab and select Rotate from the dropdown box. If you do this, you will see information about Rotate underneath and a place where you can You can adjust the rotation speed. Set it to two seconds.

Save

You're done.

Make one .gam who have this .oof used as a powerup. The Rotate property means you don't have to script DALLAS or Initial rotation in the .gam set. Just put that Use physics-Flag in the .gam. If we don't set the Rotate property, you would have to script an event to play the animation.

Back to the overview of section H

091 - OOFEdit: Turrets <>

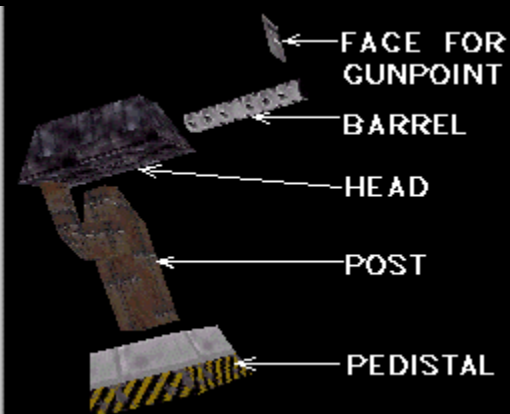
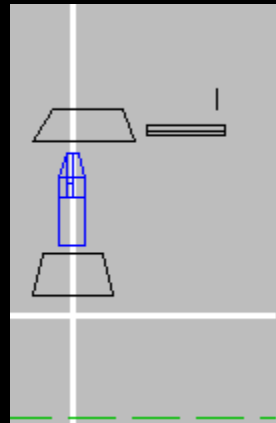
Papacat

Here too, the original .zip with the exercise files could no longer be found. The turret I recreated is a little different than the one shown here, but it all works the same.

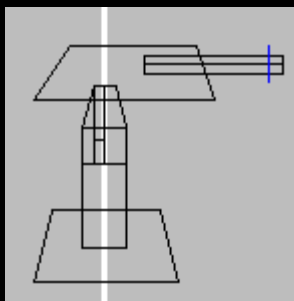
Turrets are the easiest 'animated bots' to build/animate. This is because you don't have to create the animation. All you have to do is thisTurret- Assign properties and set their settings. A few rules must be followed when configuring the submodels. Here you will learn how to use your.oofsuccesfully get them to do soTurretproperty to use.

Here is a simple turret built in D3Edit:

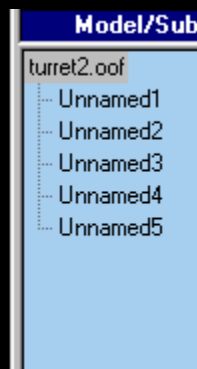
I have five separate pieces. The head and the stand will be the two moving parts. I'll keep the barrel separate in case I want to add an animation for the recoil (we won't do that here). Here is an independent face that I will convert to Gunpoint (Remember – plan ahead).



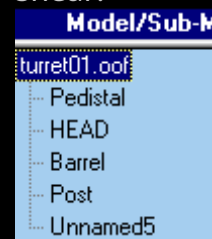
In D3Edit I make sure that the model is put together the way I want.



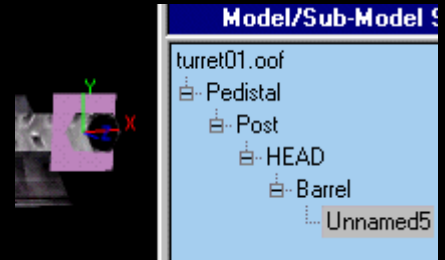
When you open it in OOFEdit you will see the five submodels in the tree.



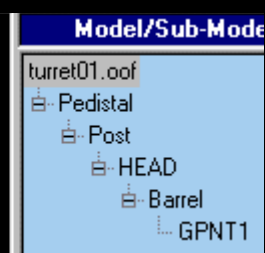
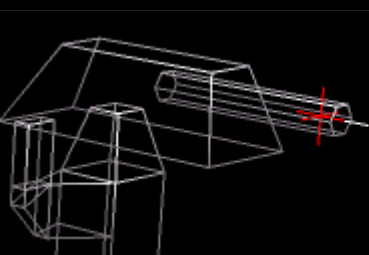
Name them submodels; around that that to Gunpoint will, Do you need not you shear.

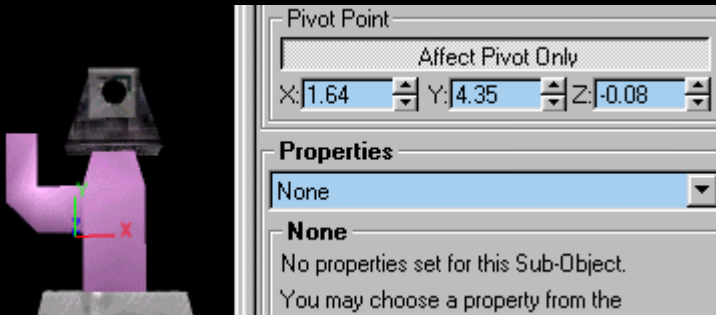


Configure the submodel hierarchy. Right click on the face and convert it to one Gunpoint. Gunpoints get an enumeration with GPNT#, in the order they are to be created.

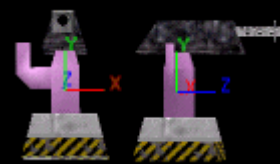


This tree shows one of the most important points when building a turret. GPNT1 is not only what makes the weapon fire, but also the 'Aiming Gunpoint' (in the.gamspecified.) It MUST be hierarchically below the submodels that move. When GPNT1 attempts to aim, it tells the submodels to move. He can be Child to Run or Head as long as he is subordinate to the Head.

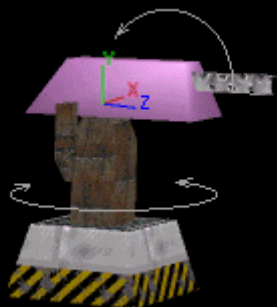




A sub-model rotates around individual axes that originate from its pivot point. Make sure those pivot points are there are where you want them. To move this select the submodel, click on **Affect Pivot Only** and adjust its X/Y/Z values.



At first I said that we don't have to do animation. By that I meant 'actual' movement. We will NOT tell the subs to move, we will just configure the keys for D3 and the turret property they use.



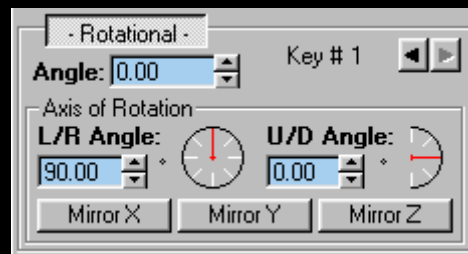
The head will rotate up/down (around the X axis). The stand will rotate left/right (around the Y axis). Around the Turret To use this property, you must have a rotation key in frames 0 and 1. D3 looks at frame 1 to find out which axis is being rotated.

Go to the animation Tab while the sub is highlighted and the frame marker is at 0. Set the **rotation** button. An oval will then appear in the animation bar. Do this for frames 0 and 1, for the head and the stand.



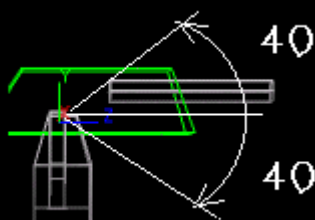
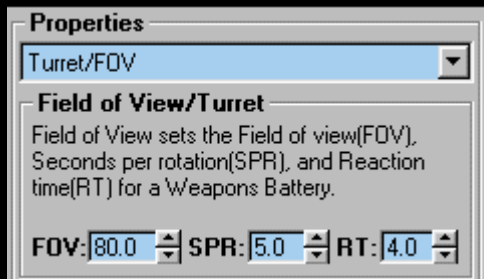
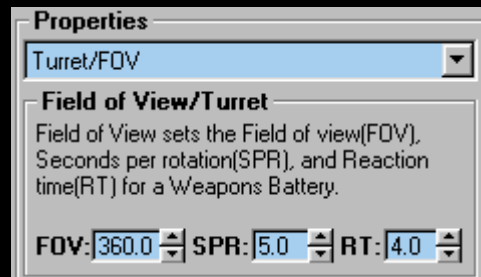
Select the stand and set the frame marker to 1/1. Set the rotation axis of the rotational key as shown below. Here you see it Turret-Property according to the axis. These settings make the axis vertical (=parallel to the Y-axis.)

Set the rotation key for the head as shown here. This makes its axis parallel to the X axis.



Note that we didn't touch Angle. If we did that, we would have defined an 'actual' movement.

While the stand is highlighted, go to the Properties tab. Set up its properties Turret/FOV. The FOV set to 360; that is actually the area of movement. By setting it to 360 it will be able to describe a complete circle. It will continue to rotate in the same way instead of having to turn back.

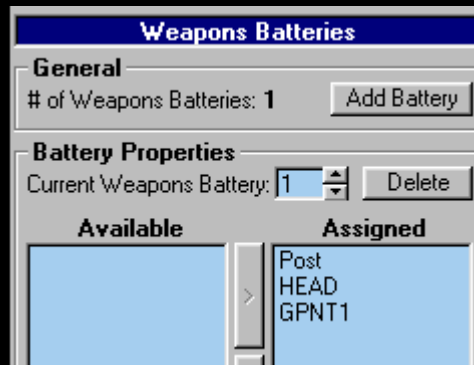


Set the properties for the head as shown on the left. The 80th FOV limit the range of motion to 80°, 40° in each direction.

Now the battery configuration:

Go to the batteries-Tab. Click **Add Battery** to create one. All available gunpoints and submodels are on the left.

While Current Weapons Battery is set to 1 select each submodel and the gunpoint and add them Assigned. The submodels and the gunpoint must all be in the same battery. Save and you are with the .oof complete.

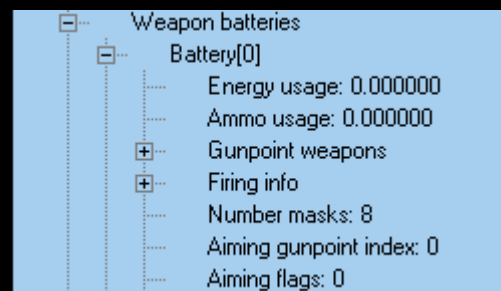


Now create one .gam-File. There are a bunch of settings with which you can control the behavior of the tower. I will highlight two main settings so that your Turret works properly. The first is under AI settings. Make sure there Type on Stationary Turret is set.

The other is found in Weapon batteries. Remember that the .gam starts numbering at 0, not 1. So go to the Battery[0].

Put that Aiming gunpoint index to 0. In the OOFEditor it was GPNT1. This is the gunpoint that will be used for aiming. He has to be part of the Battery and as already mentioned, it MUST be hierarchically below the submodels with the Turret property.

For more information on configuring a robot weapon battery, see 'Custom Robots Part 1'.



The easiest way to set up a .gam is to copy the GAM page of one of the original towers from D3. If you do this, there are a few things you need to watch out for. Mainly make sure that the GAM does not reference gunpoints that are in YOUR .oof are non-existent, D3 will crash when attempting to use a non-existent gunpoint. Go through that too Firing info yours Weapon battery and check that no animation is used, like the Lance Turret (the one with the homings).

That's it. Build the tower in yours.d3l, then do that.mn3 and don't forget those .gam and that .oof to incorporate.

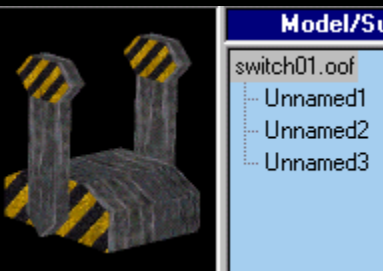
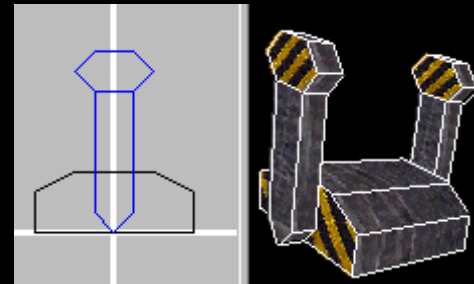
Thanks to Garner1 for helping me get started with Turrets.

Back to the overview of section H

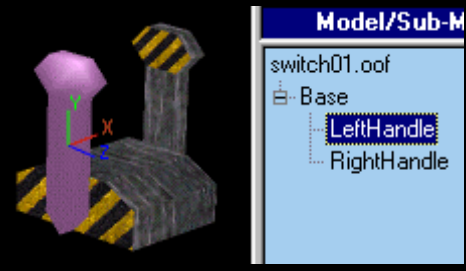
092 - OOFEdit: Animation <>

(Papacat)

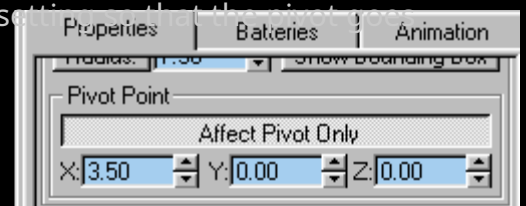
This object will be a switch with two movable handles (the switch1.orf is in the .zip). I made a base and two shift handles in D3Edit. Don't forget, you can't ADD parts in OOFEdit, so make sure everything you need is in .orf.



I open the .orf in OOFEditor and the three parts are listed. You can leave them unnamed, but it's easier if you name them. To do this, select it and change it Properties-Tab the value in Surname-Field. Then make the handles childs of the base. While it's not necessary for the tut, it's good practice.

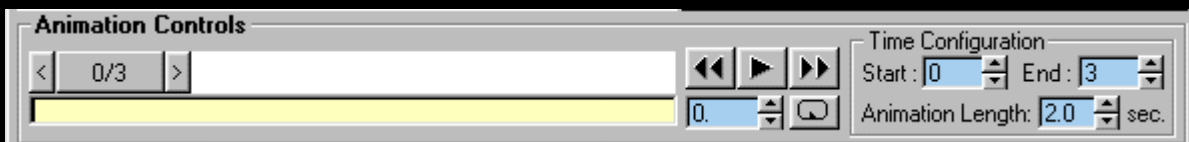


The coordinate cross is the pivot point. We need to move the LeftHandle's pivot point to where we want the rotation axis to originate. While LeftHandle is selected, go to the Properties-Tab and click **Affect Pivot Only**, then adjust the Y setting to where you want. Now is a good time to save.

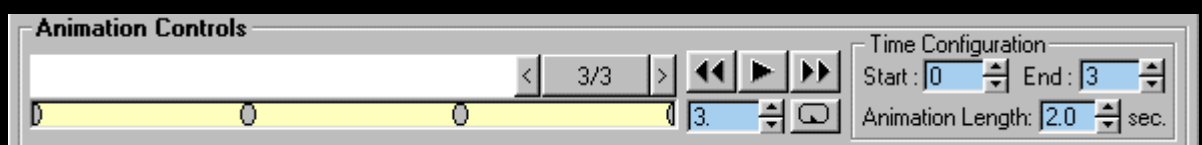


To understand the animation of an object you need to understand four terms: frame, key, rotation and translate. If necessary, look under 'field is not available'.

First we need frames where our animation can take place. Below in the right corner, near the Time configuration, let begin to 0 and set End to 3. You will see the frame marker change to 0/3. This means the marker is at 0 and the last available frame is 3. You might think that this makes four frames. Think of it this way: An equal amount of time exists between 0-1, 1-2 and 2-3. As we add more keys this will become clearer.



With the LeftHandle selected and the frame marker at 0, go to animation-Tab and select **Rotational**. This is what he says .oof, that this submodel has a rotation animation at frame 0. You'll see an oval appear in the animation bar on the left. This tells you that the selected submodel has an animation key here. Carry the frame marker with you > (In the animation-Tab) so that it now shows 1/3. Set a rotational key here. Complete the LeftHandle by also setting rotational keys in frames 2/3 and 3/3. With the LeftHandle selected, your animation control should now look like this:

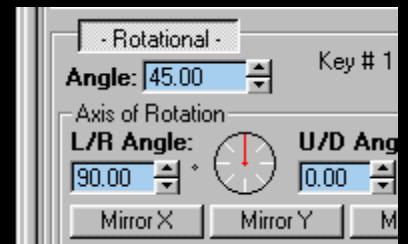


An important point is that a key only applies to one submodel. SelectRightHandle: all the ovals disappear. They represent the existing keys for the *active submodel*. Assign rotational keys to all four locations like we did forLeftHandlehave made. Save.

Time to move the submodels. First, so we can see the path the submodels will travel, turn on View Trajectory either via the animation-Tab or via the menu.

MarkLeftHandleand set the marker to 1/3. When you move a submodel at a keyframe, you tell it where it should have moved at the end of that frame. The movement starts at 0/3 and reaches the position we set at 1/3.

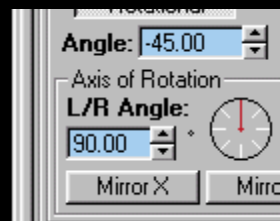
Angleis the amount of rotation in degrees by which you want the submodel to be rotated. UnderAnglesee YouAxis of rotation. By doing thatL/R angle (Left/Right) and theU/D Angle(Up/Down), you define an axis that originates from the pivot point. It is this defined axis, not the X, Y or Z axis that your submodel will rotate around.



AtL/R=0 andU/D=0 the rotation axis is on the Z axis.

We want the axis of rotation to be parallel to the X axis. Twisting it by 90° does this. Now rotate the model by 45°. Write it in andEnteror click the spinner arrows and drag with the mouse. If you use the dragging method, you will see it move and trace the trajectory. It should look like this above.

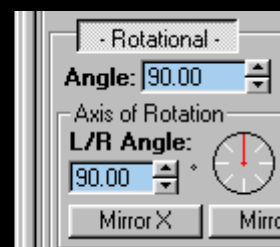
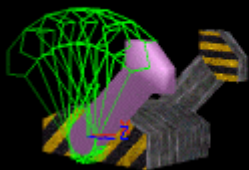
While the frame marker is still at 1/3 select the RightHandle and rotate it by -45° (minus 45°).



Move the frame marker to 2/3.

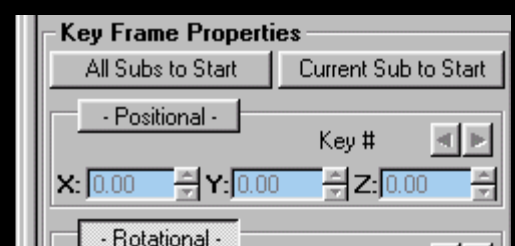
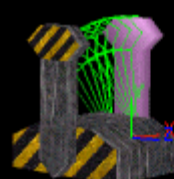
Select LeftHandle, rotate it at -90°.

Select RightHandle and rotate 90° (right).



Now we will move both handles back to their original positions. In this example it's pretty simple, but imagine we moved it three or four times and changed the axis angle each time. Do you think you could figure out the exact angle of the axis and the amount of rotation to get back to the beginning? Maybe after half an hour of mathematical projections. Luckily, SuperSheep has given OOFEditor a tool that does all the work for you.

With the frame marker at 3/3 click**All subs to start**. OOFEditor calculates all the necessary keys and applies them to the submodel to bring everything back to its original position. In and of itself we don't need to put keys in 3/3. The **All subs to start**does that too. Save!



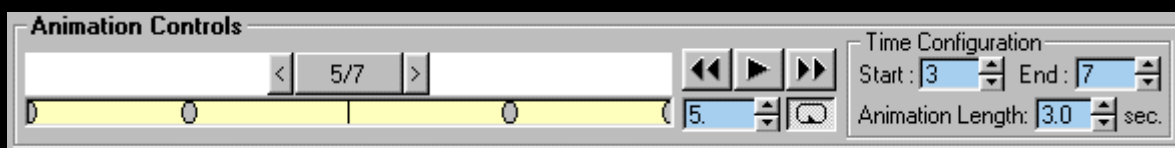
Let's test the animation. In the Time Configuration, set the animation length to three seconds. begin and End should still be 0 and 3. Move the frame marker back to 0 and click the play button.

Did you notice that the handles moved faster in the second frame (1-2)? All frames are the same length. In the second frame, the handles must rotate twice as many degrees in the same amount of time, so they rotate faster to complete their movement in the allotted time. The switch completed so far is switch01.oof.

Now we will configure the animation so that the handles move at a constant speed. At the same time, we will see how frames and keys relate to each other.

An obvious way would be to let the handles move to the first position, then back to the start, then to the second position: always 45° per frame. This would require a key in every frame, and since it's obvious you wouldn't learn anything. Let's try something else.

First go to the Time configuration and set begin to 3 and End to 7. If the frame marker is on the far left, it should say 3/7. Choose LeftHandle. You'll still see the marker at 3, far left. Add a rotational key for both handles in frames 4 and 6. NONE on 5. At 4, set the animations of both submodels to 45 and -45 degrees, the same as before. Skip 5. At 6, set both submodels to -90 and 90, same as before. At 7, where there are no keys yet, choose **All subs to start**. A key is added for both submodels and their animation is set by OOFEdit. Set your frame marker to 5 and your animation bar should look like this. There should NOT be a key at 5. Set the frame marker to 3, set the loop button and play the animation.



Now the speed of the handles no longer changes. You may find it easier to see if you turn View Trajectory off for the moment. The finished switch is switch02.oof.

This example shows us two things:

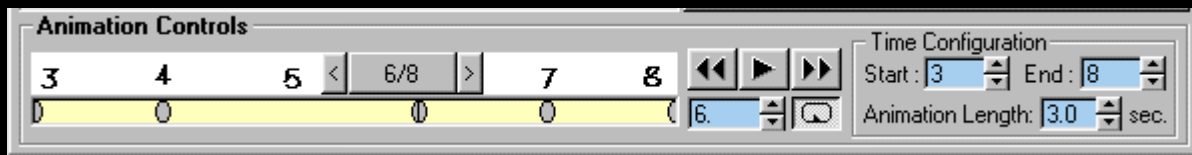
- 1) Animation is calculated key by key, not over a frame. This is just a time increment.
- 2) Animation and animation loops can start at any frame and end at any frame.

Here's the advantage we get from #1: Let's say you have a bot. You want its head to rotate 180°. As it rotates, you want the bot to extend its arm, grab something, and pull back. This happens during three frames (0 to 3). Since animation is calculated key by key, all we do for the head is keys at 0 and 3, at 3 set the rotation to 180°. Then we go back and set keys at 0, 1, 2 and 3 for the arm and set the movement for each one. If the animation were calculated every frame, we would need keys at frames 1, 2 and 3 for the head, rotating it 60° each.

That's why we need a key at the very beginning so that it knows when to start moving. We also need one to stop a submodel from moving. I don't mean that you need a key on each sub to make it stand still. What I mean is that you need a key to do it new to let it move (a little confusing).

Here is an example.

We'll use this animation bar as a reference:



From 3 to 4 our bot should turn its head by 30° . From 4 to 5 and 5 to 6 he should keep his head still. From 6 to 7 he should turn his head 120° . To achieve this, we set a rotational key at 6 and leave theAngle at 0. We set a key at 7 and set the angle to 120° . When the animation gets to 4, it looks at what it needs to do. It finds the key at 6 that tells her to rotate 0° , so the head doesn't move for two frames. When it comes to frame 6, she looks forward and finds frame 7, which tells her to rotate 120° and has a frame for it. Without the key on 6, the 120° rotation would have started at 4 and extended over three frames.

Danger! That's why you have to have a key that doesn't do anything. Without a starting point, nothing happens.

Now for #2: Animation and animation loops can start and end at any available frame.

This means that a bot can have a large repertoire of behaviors. Without that you would be different. You need for every behavior. Open this from 'Old Scratch' (scratch.oof), and let its animation run a few times. You will notice that it always returns to the starting position. Scratch has multiple behaviors that are executed on different occasions. To see how they are used, use gamTool to open its GAM page. search for Generic Object: Old Scratch.

If you are there, open its AI settings-branch and go down to Movement. Go to Movement[0], Animation[0] and you will see that Animation[0] runs from frame 0 to 11 over four seconds. Now look at these. Scratch begins executing a behavior at frame 0 and returns to its starting position at frame 11. Back in the GAM, take a look at movement[1], Animation[0] on - an animation that goes from 11 to 17. movement[1], Animation[1] goes from 17 to 21. Go to the.oof, and you can see the correlation.

Of course, unless you're a heavy-duty modder, you can't just add moves to the table and expect them to execute. However, you can take an existing bot that does things if you want and be yours. Replace with yours (make sure you set the start and stop frames accordingly). You can also script DALLAS to perform various behaviors. You can make a bot spank you if you don't have a key or make it bow to you if you do.

Sure, that was a lot of boring info but, hopefully, this information will help you plan your animation and take advantage of some of the basic concepts.

Back to the overview of section H

093 - Ship Tutorial

Darkside Heartless

I thought I wouldn't be able to do this, but I found it anyway. Unfortunately, there isn't that fancy ship that Julian built anywhere. I like this trilateral symmetry...

The required tools should have been installed long ago.

revised

Note: This tut assumes that you have already built a ship (regardless of the scale, as that can be easily changed), but not the necessary faces for gunpoints, attach points, etc.

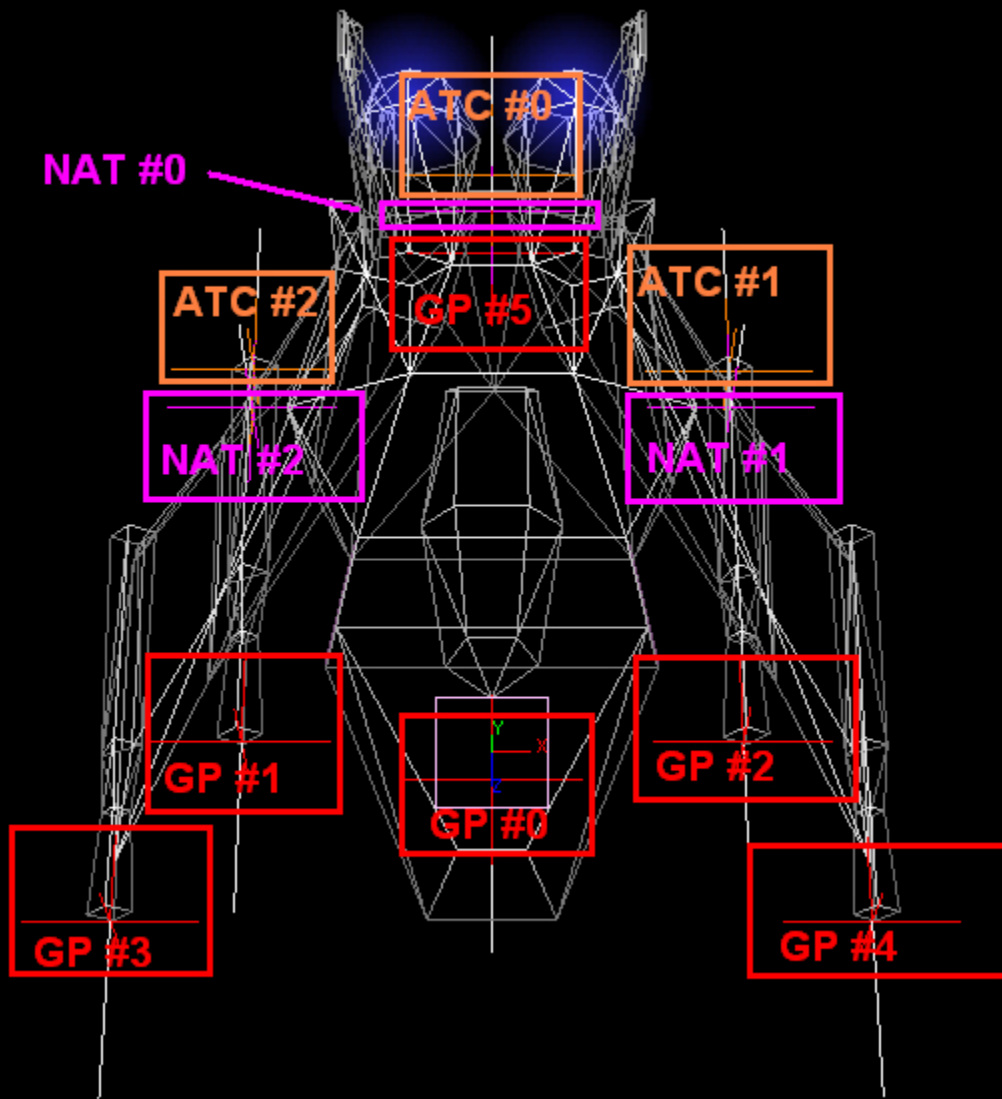
Open the OOFEditor and open your.oof with your ship.

Since we don't have any faces that can act as Attach Normal Point, Ship Logo, etc., we need to go back and add them. Here is the full list of all the previously mentioned things, all of which are necessary to have a good, fully functional ship model:



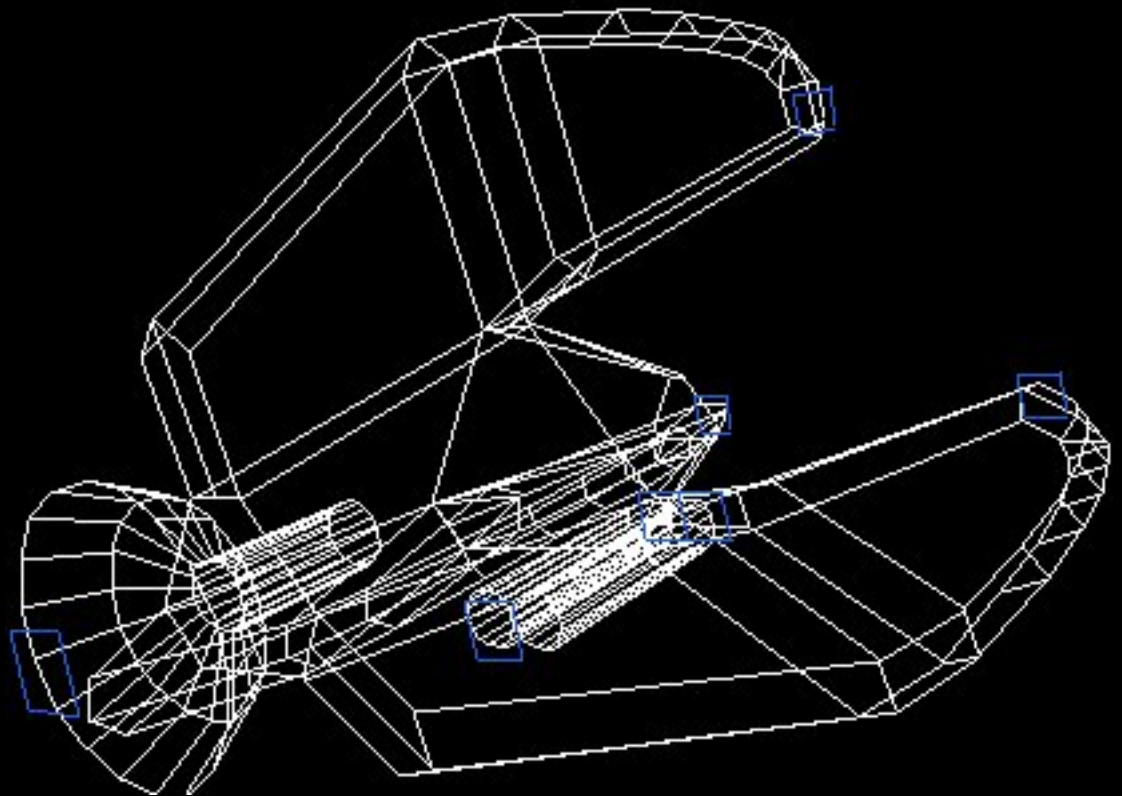
Surname	Description
Gunpoints (GPNT)	Maximum of eight pieces (numbered 0 to 7). To fire weapons. Note: After you have all available gunpoints in the.oof file, the remaining gunpoints are stored in the table file last available gunpoint fired. If you do just one, everything will work out - regardless of what's in the GAM stands - fired from this single gunpoint.
Attach Points (ATCH)	Used to attach objects to the ship, such as the flag in CTF.
Attach Normal Points (NATH)	Determine which direction is 'Up' for the corresponding ATCH.
Glow points	Create a glow, typical of the engine glow of ships.
Ship Logo Faces	They are used to display the multiplayer ship logos. Can also be more than one face. If so, they must be a submodel.

To get an idea of where you'll want your points to go, here's where they are on the Pyro-GL. The points are numbered in the order in which they were created:



When you create them, use small ones square or triangular faces. (I use 2x2 faces because they fit easier with gun barrels and Have motors aligned.)

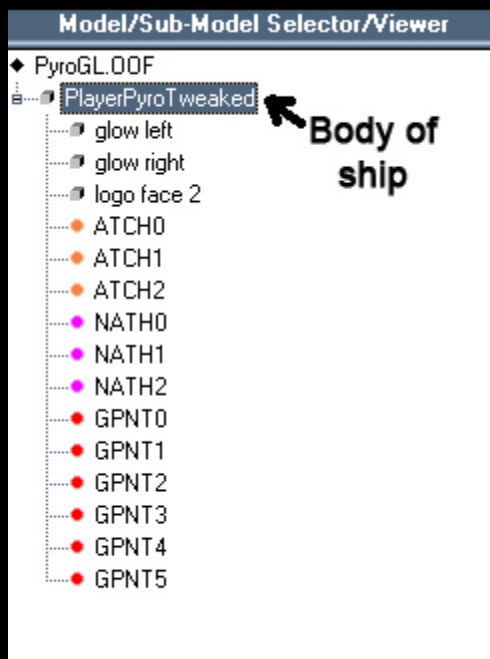
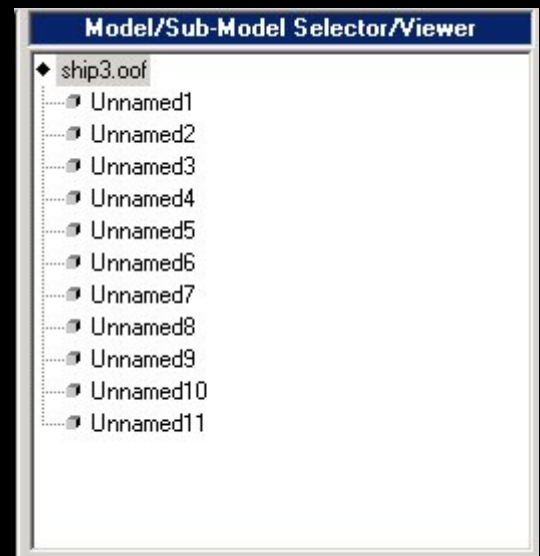
In this picture are the placed Gunpoint faces in Blue:



Go back to the OOFEditor and open your ship.oof. You can see the 'parts list' on the right:

Find the main body and name it accordingly. It's not necessary, but it's very practical to keep an overview.

Note: If you didn't make the ship in one piece, then combine all parts of the ship EXCEPT the future gun, attach, normal, glowpoints and logo faces by pressing Shift while creating the submodels pull together. (See the OOFEdit digression).



When you're done, there should only be one 'body' part left - the rest are dots, logos and glows. Here on the left is the Pyro-GL hierarchy as an example:

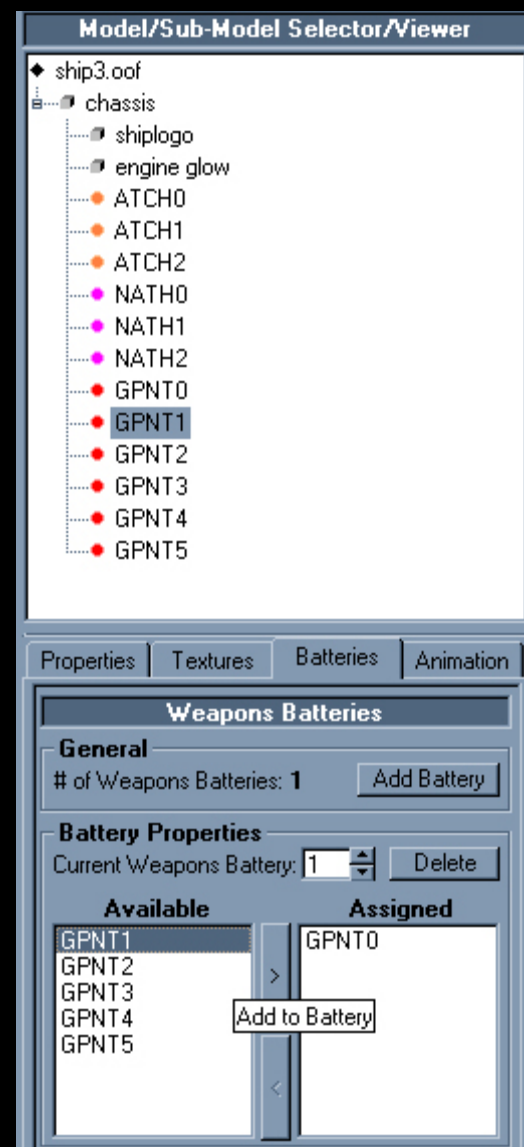
Back to your model; select one of the small cubes in the tree view and drag it onto the chassis. It will then appear under the chassis in the tree.

This is called assigning; do this with all faces that will later fulfill a function.

So to get started, click on one of the little cubes, the future oneGunpoint[0] should be on the right and select convert to -> Gunpoint. It is now shown in the view as a light yellow cross, with a white line pointing in the direction of fire (compare with the image above). If the gunpoint is deselected, it appears as a red cross (hehe) with white line. Do this for the other faces that will become gunpoints, then select the faces that will become your attach and attach normal points and convert them.

Then select one of the faces that is intended for the Glow engine and go to the properties tab and choose where 'None' pending Glow/Thruster. The point seems to disappear, but in the Properties-Tab a glow section has appeared. Choose the color and size you want and save the ship as.oofaway.

Then go into the batteries-Tab and click **Add Battery**; The available gunpoints are then shown in the Available-Box. Click on each one and then that one>-Button, the marked ones move into the Assigned-Box. Do this until everyone is in the Assigned box, then save.



Go to the Tools menu and select Scale Model..., there select Scale by Radius and enter a value around six, +/- 0.5 units. Values in this range are generally good.

Then open the gamTool and either open an existing or create a new GAM:Edit -> New -> Player ship. Open the branch with the + and place your ship-`oof` as a model. Pay attention to those `medium` and `Low` model names and delete them if you haven't made any corresponding models.

Then add the `.gam` and that `.oof`-Add a model to the level, note what needs to be taken into account (same name, etc.).

Back to the overview of section H

Section I–Scripting

In Descent 3 you can bring 'life' to objects in the game, and this is done using scripts. Programming is done correctly here, even if it doesn't look quite like it!

From here must Dallas run. If necessary, take a look at the relevant section: 'Anticipation: Getting DALLAS up and running'.

From here on out, it's also a good idea to have a pad and pen within easy reach



094	Scripting: Hello World	The first two scripts	Kyouryuu	343
095	Scripting: Introduction	DifferentOwner	Starkiller	346
096	New DALLAS Search functions	A scripting aid	Atan	348
097	make weather	Let there be rain!	Ragil Ral	356
098	Refined Player Respawn	Respawn on spew	MZero	358
099	Teleporter	From A to B in zero time	Fischlein	359
100	Create Lighting	Lightning, rays and other things	Fischlein	360
101	Wind scripts	Making wind by script	Papacat	362
102	Music regions	Event-driven soundtracks	Kyouryuu	363
103	Spew!	Flames, smoke, steam...	Starkiller	366
104	Broken glass with shards	Multiple conditions	Papacat	368
105	Scripting a Switch I	Insert a switch & use it	WillyP	372
106	Scripting a Switch II	Refine the switch script	WillyP	375
107	Alarm!	Event-controlled acoustics	Ragil Ral	378
108	Scripting an inventory Cloak	Switchable cloaking device	Starkiller	381
109	Reactor Gamma: Script Companion	Explanation of the script for the level	Kyouryuu	385

The Lost Tutorials				
Either the tutorial or the example level is missing here.				
	none does	exit sequence		
	no files	Hal9000 - scripting		

[Toc](#)

094 - Scripting: Hello World

Kyouryuu

The programmer's beginning is 'Hello World'. Anyone who has already had anything to do with programming will recognize this, and we will keep it that way here too.

DALLAS must be installed and set up, see 'Getting DALLAS up and running'.

Understanding the anatomy of a script

First a few words about file extensions. When you start working in Dallas, you generate two files - one.cpp- and a.msgfile (also a.o, but you can ignore them [here]). The.cppfile is a simple text file that contains all the program stuff that Dallas spits out. In fact, Dallas is a nice frontend that does all the coding for you. The.cppis the result. The.msg-File contains messages, more on that later. When you compile the script, you create a.dll-File. In reality, the final .mn3 of your level just needs to be the.dlland the.msgcontain. But most level authors throw those in too.cppin case the user wants to see how things are scripted or they want to compile it on another platform (e.g. Linux or Mac)

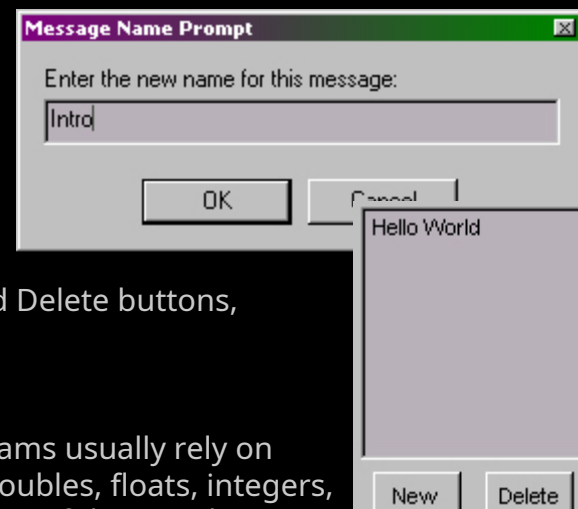
An example script

Let's assume you've built a level. I'm going to walk you through a very basic set of two event scripts. I want to plaster a message on the player HUD that appears three seconds after the level starts. Sounds banal and boring enough, right?

Okay, let's open D3Edit and load our level. In the world view go toFile -> Dallas Graphical Script Editor. Amazingly, a new window opens, filled with good things. The two error messages that say neither one.cppanother one.msgfile can be opened, you can simply click away - it doesn't exist yet.

The message list

First we need to create the message we want to put on the HUD. In the box with the headingMessage list click on the buttonNew. A small dialog box will appear asking you what the message should be called. It's not the actual message - just how Dallas responds to it will reference. GiveIntroa. In the box just above the New and Delete buttons, enter the text of the message. May I suggest,'Hello World'?



Create a user variable

If you are familiar with programming, you know that programs usually rely on variables that you declare at the beginning. These can be doubles, floats, integers, etc. Dallas has its own variable types and we will only use one of them in this script (aTimerID-Variable).

Click on the long button**User Types Workshop**. a narrow window opens. If you click on the pull-down menu above (WhereGoalIDyou can see a number of different variable types that can be declared. Different variables serve different purposes. For example, if I want a variable that is basically an integer that I can increment, decrement or whatever, I would have aUservarmake. In this case we want a timer variable, so we need oneTimerID -Use variable here. So chooseTimerID. Click on the button**Add New Value**and name your variable. You can name them whatever you want (I named mine**Message timer**), but try to keep it clear. Then click the button**Return to Dallas**.

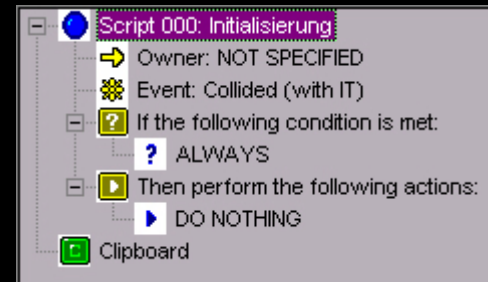
Generate script 1 of two

Well, let's make a script, shall we? Click on the New button under Scripts. A dialog opens asking you what you want to name the script. Again, choose something illustrative here. I called mine initialization.

Open it by pressing the button **Expand** click or use the +- Sign.

Let's take a look at what each of these fields means.

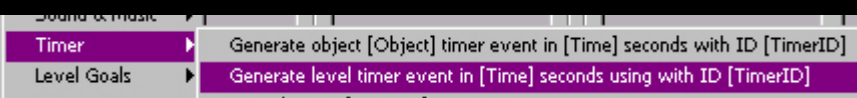
Owner-This refers to who or what the script concerns. For example, if I wanted to write a script where something related to a bot being destroyed, I would specify the name of the bot here. As with all fields, you can choose one option here **Right click**select on it. So right click on the owner. In this script we want it to start exactly with the level, so the level is theOwner.



event-What happens to the owner that allows Descent3 to continue with the script? If we're talking about our hypothetical bot, I would check if it was destroyed. But we are dealing with a level here. And the event is, as you might expect, is the start of the level. Choose Level start by clicking on this event-Right click on the field (if it is not already set on it). As you can see, there's a bunch of stuff that's grayed out - you can imagine the enormous possibilities.

If the following condition is met:-With this script you can do it on ALWAYS let. This is because we want it to run every time we start a level. We don't need any further conditions. But we will, in the second part of the script if we want to add anything.

Then perform the following actions: - Simply click on the right DO NOTHING and check out all the different things you can do here! Well, in our case we want the timer variable to count to three seconds. Right click on DO NOTHING and hold the mouse above Replace with a different action. Then go to the option timer, and then click Generate levels timer.... A few new fields appear (if not, expand the tree to see them). One



is Time, another is TimerID. For time specify three seconds by right-clicking on it, and Enter float value choose, enter and OK clicks.

That's how much we want it to be TimerID, which we will specify next, should count. So right click on TimerID and specify your variable. So now we have the following:

"When the level starts, leave the variable TimerIDCount to three seconds."

Aye, it may seem like a lot for a little, but it will make you faster.

Generate script 2 of two

So far we are actually showing the 'Intro'-Message not on HUD. That's what this script will do. It will also introduce you to something new. Click on New and enter a name. I named mine 'showmessage'.

Expand the whole thing and you'll notice that it's structured the same as the last script. Our Owner is still the level. No object or player action is involved here - just the level. Things are a little different under Event, we want Descent3 to only continue in this script when the intro timer has counted through the three seconds. So choose Timer (TIMER ID) went off.

Now, in the area If the following condition is met, we want to explain Descent 3 which TimerID must be checked for expiry. In other words, we want it to know that we are interested in when the message timer reached the third second and therefore stopped counting. To do this we will use a comparative statement - one that compares one thing with another. Right click on that ALWAYS and go over Replace with a New Condition and then click on Default Comparison Statement. A few new ones Fields should appear if not expanded.



Basically what we have here are two variables between which an operation is performed. It's not that difficult to understand - it's something like $A=B$ or $A>B$ or $A<B$. So what are A and B in this case? `Message timer` and `TIMER ID`. Perhaps at this point you are compelled to ask 'why `TIMER ID`?'. Well, think of it this way: news timer counts to three, but `TIMER ID` already know that it lasts until three. So by saying `Message timer=TIMER ID` Descent 3 will say "Hm, looks like `Message timer=3`, and `TIMER ID` said that it counts up to there, so the condition is correct!"

Now how do you get that to Dallas? Right click on the first literal and do `Replace with literal, Enumerated Types` and then `TimerID (USER TYPE)`. Now he should `If...-Area two TimerID-Compare variables`, but which ones are currently on (None) are set. Change that by right-clicking on the first one and `and Message timer indicate`. Next time assign `TIMER ID` by right clicking. So you have the statement "`If Message timer=TIMER ID`" generated.

Now for every 'if' there is a 'then'. `If Message timer=TIMER ID`, then we want our '`Intro`' message is displayed on the HUD. Click where `DO NOTHING` is written on the right and do `Insert new action (Instead of Replace-You'll see why later)`, then follow up `Misc` and then `Show HUD message [message]`. A new line will appear (otherwise expand) where you can specify the message. Right click on it and select the '`Intro`'-News.

Congratulations! You have now completed the actual creation of the script that we need here.

Compile the script and test the level

If your Dallas works properly, this should be a small matter. Just click that **Save** button and let Dallas do his thing. If the message box at the bottom of the Dallas window says 'Done', exit Dallas and save your level.

If you the `mn3` whether in Quicktest or `mn3Edit`, don't just include them `.d3l` but also those `.dll`, `.msg` and optionally also the `.cpp`. You can find this in the Osiris directory (you remember where that is, right?). Then save it `.mn3` and play your level.

Bonus!

Once you have the original two scripts functional in Descent 3, I have an extra task for you. Having the text displayed on the HUD like that is boring, right? How about we spice things up a bit with a sound effect? Go back and edit the second script to play a 2D sound at 100% volume. I'll leave you with this as a challenge!

Thereafter

Well, young hops, you're now well on your way to becoming Dallas script masters. There are still other things to learn (you've seen the grayed out features), but in due time you'll learn to master them! And then your levels will become more than just shells to fly around in. They will become living, thriving empires.

[Back to the overview of Section I](#)

095 - Scripting: Introduction

Starkiller

What is a script?

A script is a portion of a file used by Descent 3 that says "*If this happens, and that is so, do this.*" The '*When that happens*'-part becomes Owner and event called. The Owner is the thing that happens to, and the event is what happens to the thing. The condition is the '*that is like that*'-Part. There you set the more precise factors of the event. Such more specific factors are what creates the event and the definitions of some level events. The '*do this*' part is executing the following actions.

Make a level

For this exercise we will use a default room level. I expanded the space a bit by marking and moving a few verts. You can of course use more sophisticated levels in your experiments. Next add an object, a bot. Any bot will do, but I suggest you use a relatively simple one, you'll have to fight it. Place it about 30 units away from the player starting point

a. Then select and name it '**offered**' To name an object, select it. Then, in the upper part of the object panel, above the view of the object, there is a button where the type of the object is written; If you click on it, a dialog opens where you can assign a name. Name it and save the level.



Dallas

Just go to World View and choose File->Dallas Graphical Script Editor. Now you should get exactly two error messages, which say "Unable to open levelname.CPP" and "Unable to open levelname.MSG". That is normal, if the level hasn't been scripted yet, they basically say that the script files don't exist... because you haven't made one yet! From there create a new script. There is a button in scripts that says **New** is labeled. Name it with '**turnBotOff**'. Put that Owner by right clicking on level. Put that event on Level start. Set the condition (If the following condition is met) ON ALWAYS. Finally, get the action going Replace with a new one action -> AI on Turn [On/Off] AI for object [Object]. This action is pretty self-explanatory; if an action or query ('if that is the case') is a little harder to understand, take a look at the bottom of the Dallas editor. There is a box that contains information about the currently selected action or query. The action you just added should already be expanded. Right click on the variables that are inside. This is the default OFF and None. Change that by right-clicking None to offered (To change something, always right-click on it). This is your first script, what it does is it creates the AI for the offered disabled (The AI lets objects perform actions; all objects that are not powerups have AI, not just robots). Now the bot will just sit there and make noises, we need to find a way to 'revive' it.

Make another script. Name it '**turnBotOn**'. Put that Owner on offered, the event Collided (with IT). (IT) is a script-wide variable (it is used throughout the script, but not the entire file). (IT) can be set to receive actions and set so that only certain IT's work for the script. Set the condition with Replace with a new one

Condition -> Comparison Statement with Query -> Object ON [Object] is a player. Leave the default Settings. Note that this [Object] for the [IT] becomes. This means that if the object which with the Owner (I.T) collides with the object that the condition respects (a player), the '*if that is the case*'-part on '*that's so*' is set, and Descent 3 then looks into the action settings to find out what it has to do now. You will have two actions in this script.

Replace thatDO NOTHINGwithTurn [On/Off] AI for object [Object]. This is the same action you used in the other script, but this time we will change the variables. Put thatObjecton againoffered, but changeOFFtoON. Then right click on the actions header, not those Action yourself. Go afterAdd a new action -> Misc -> show Colored HUD message of color [Red] [Green] [Blue] saying [message]. Set the variable for red to over 200, leave the others at0. Then go to the Message List area and create with it**New**a new message; name them too**botON**. Now dive botONin the upper part, while there is nothing in the lower part. Write in there, pay attention to capitalization:**GUIDEBOT: Bot's AI has been reactivated!**Set the Message: variable in the message action to 'botON'. This script turns on the AI forofferedon when a player touches the robot.

Name your final script**endLevel**, put thatOwnerback up**offered**. Set up the event Destroyed. For this query we don't need to add any condition. Replace the action with Replace...>Mission->End level. This script will finish the level when you 'Robot'offered' you destroyed. Save your script, it will automatically compile for you.

What do I do now?

Make one.mn3-File as usual, but this time you have to add a few extra files. Under the **Add file(s)-Add** area<levelname>.dlland<levelname>.msgadded. If it is a final release of your level, add that too<level name>.oadded, for Linux servers. Play your level and touch the robot that should just sit there. He will then come back to life and will probably shoot you. You'll have to kill him; After it's destroyed, the level should end and you'll be back in the D3 main screen.

You just put three scripts into one level!

Here everyone again, the scripts opened:

The screenshot displays the game engine's script editor with three scripts defined:

- Script 000: turnBotOff**
 - Owner: Level
 - Event: Level Start
 - Condition: ALWAYS
 - Action: Turn OFF the AI for object bot
 - On/Off: OFF
 - Object: bot
- Script 001: turnBotOn**
 - Owner: bot (Object)
 - Event: Collided (with IT)
 - Condition: (IT is a player) = (TRUE)
 - Object: IT
 - Operation: =
 - Bool: TRUE
 - Action: Turn ON the AI for object bot
 - On/Off: ON
 - Object: bot
 - Action: Show Colored HUD message of color 212 0 0 saying botON
 - Red: 212
 - Green: 0
 - Blue: 0
 - Message: botON
- Script 002: endLevel**
 - Owner: bot (Object)
 - Event: Destroyed
 - Condition: ALWAYS
 - Action: End level

On the right, the **Message List** panel shows a message named **botON** with the text: **GUIDEBOT: Bot's AI has been reactivated!**

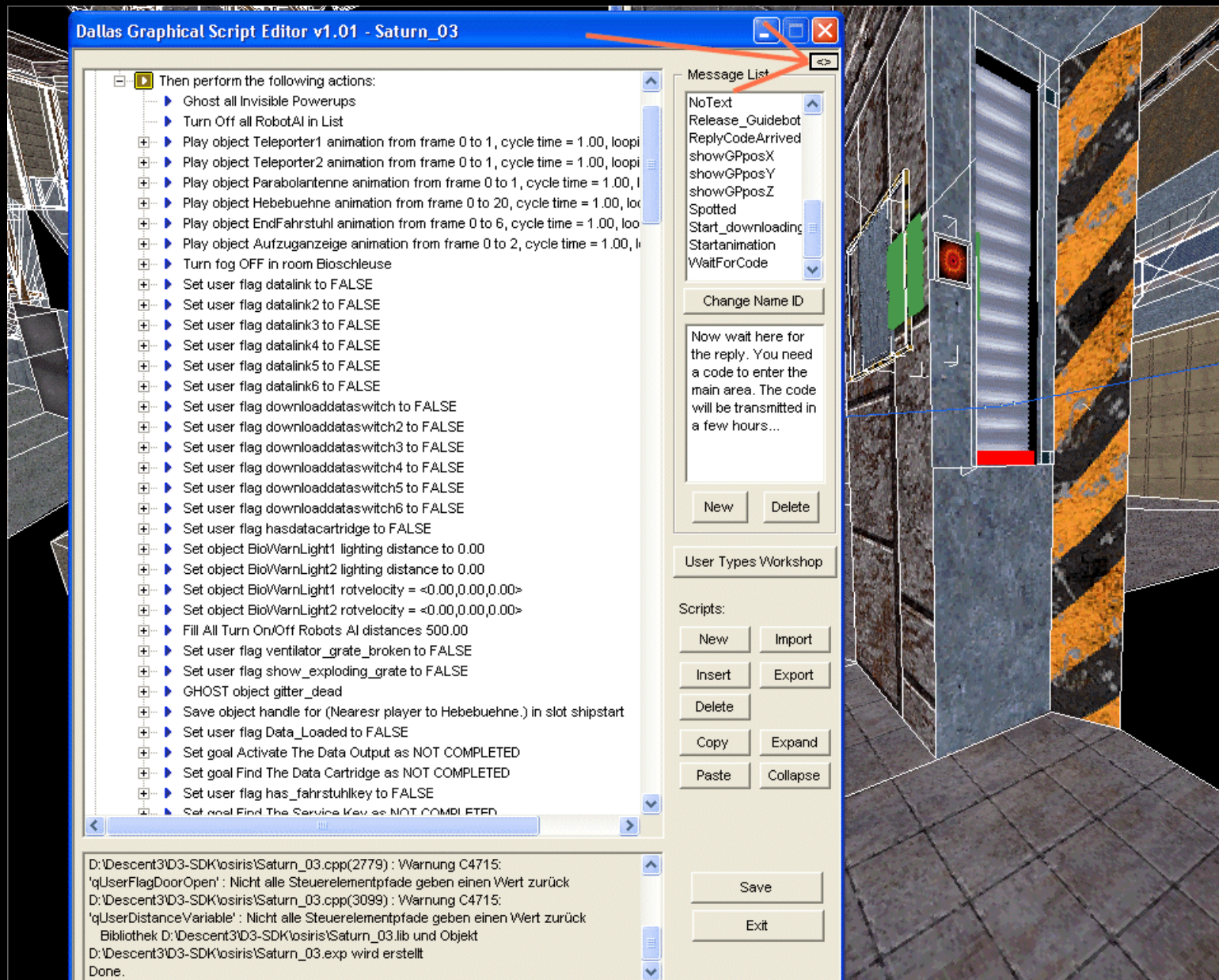
Back to
Overview of
Section I

096 - New DALLAS search functions

Atan

V1.0

The innovations can be switched on or off with a switch:



The dialogue expands to include a new area:

Dallas Graphical Script Editor v1.01 - Saturn_03

- Then perform the following actions:
- ▶ Ghost all Invisible Powerups
 - ▶ Turn Off all RobotAI in List
 - + ▶ Play object Teleporter1 animation from frame 0 to 1, cycle time = 1.00, loopi
 - + ▶ Play object Teleporter2 animation from frame 0 to 1, cycle time = 1.00, loopi
 - + ▶ Play object Parabolantenne animation from frame 0 to 1, cycle time = 1.00, l
 - + ▶ Play object Hebebuehne animation from frame 0 to 20, cycle time = 1.00, loo
 - + ▶ Play object EndFahrstuhl animation from frame 0 to 6, cycle time = 1.00, loo
 - + ▶ Play object Aufzuganzeige animation from frame 0 to 2, cycle time = 1.00, l
 - + ▶ Turn fog OFF in room Bioschleuse
 - + ▶ Set user flag datalink to FALSE
 - + ▶ Set user flag datalink2 to FALSE
 - + ▶ Set user flag datalink3 to FALSE
 - + ▶ Set user flag datalink4 to FALSE
 - + ▶ Set user flag datalink5 to FALSE
 - + ▶ Set user flag datalink6 to FALSE
 - + ▶ Set user flag downloaddataswitch to FALSE
 - + ▶ Set user flag downloaddataswitch2 to FALSE
 - + ▶ Set user flag downloaddataswitch3 to FALSE
 - + ▶ Set user flag downloaddataswitch4 to FALSE
 - + ▶ Set user flag downloaddataswitch5 to FALSE
 - + ▶ Set user flag downloaddataswitch6 to FALSE
 - + ▶ Set user flag hasdatacartridge to FALSE
 - + ▶ Set object BioWarnLight1 lighting distance to 0.00
 - + ▶ Set object BioWarnLight2 lighting distance to 0.00
 - + ▶ Set object BioWarnLight1 rotvelocity = <0.00,0.00,0.00>
 - + ▶ Set object BioWarnLight2 rotvelocity = <0.00,0.00,0.00>
 - + ▶ Fill All Turn On/Off Robots AI distances 500.00
 - + ▶ Set user flag ventilator_grate_broken to FALSE
 - + ▶ Set user flag show_exploding_grate to FALSE
 - + ▶ GHOST object gitter_dead
 - + ▶ Save object handle for (Nearest player to Hebebuehne.) in slot shipstart
 - + ▶ Set user flag Data_Loaded to FALSE
 - + ▶ Set goal Activate The Data Output as NOT COMPLETED
 - + ▶ Set goal Find The Data Cartridge as NOT COMPLETED
 - + ▶ Set user flag has_fahrstuhlkey to FALSE
 - + ▶ Set goal Find The Service Key as NOT COMPLETED

D:\Descent3D3-SDK\osiris\Saturn_03.cpp(2779) : Warnung C4715:
'qUserFlagDoorOpen': Nicht alle Steuerelementpfade geben einen Wert zurück
D:\Descent3D3-SDK\osiris\Saturn_03.cpp(3099) : Warnung C4715:
'qUserDistanceVariable': Nicht alle Steuerelementpfade geben einen Wert zurück
Bibliothek D:\Descent3D3-SDK\osiris\Saturn_03.lib und Objekt
D:\Descent3D3-SDK\osiris\Saturn_03.exp wird erstellt
Done.

Message List

NoText
Release_Guidebot
ReplyCodeArrived
showGPposX
showGPposY
showGPposZ
Spotted
Start_downloading
Startanimation
WaitForCode

Change Name ID

Now wait here for
the reply. You need
a code to enter the
main area. The code
will be transmitted in
a few hours...

New Delete

User Types Workshop

Scripts:

New Import

Insert Export

Delete

Copy Expand

Paste Collapse

Save

Exit

Search in Script for :

Flags

AIFlags
AIGoalFlags
CinematicFlags
DeathFlags

Enums

Activate/Deactivat
AIModeType
Axis
Can/Cannt

Objects

AIReflexschuss
AntennenDoor
Aufzuganzeige
BioDampf1

Doors

AntennenDoor
AufzugschachtDo
BioschleuseDoor
D CompRoom1

Rooms

Bad
Bioschleuse
CompRoom1
CompRoom2

Textures

Bluedatascrollup
CamTeleporter1
CamTeleporter2
CamTeleporter3

Sounds

AmbSwitch31
AmbVirusSwitch
BaseAlertedSiren
BeamOUT

Paths

AufzugCam
CrashPlage
Gittersprengung
P Teleporter1

Goals

Activate The Data Out
Find The Data Cartridg
Find The Service Key

Triggers

drop
MusikOn
NoMoney
T FahrstuhlStart1

AudioStreams

SpecNames

Antenne_ausfahre
buddyantivirus
chemical droid bar
Daten senden

Found :

Find text ☒ Exact

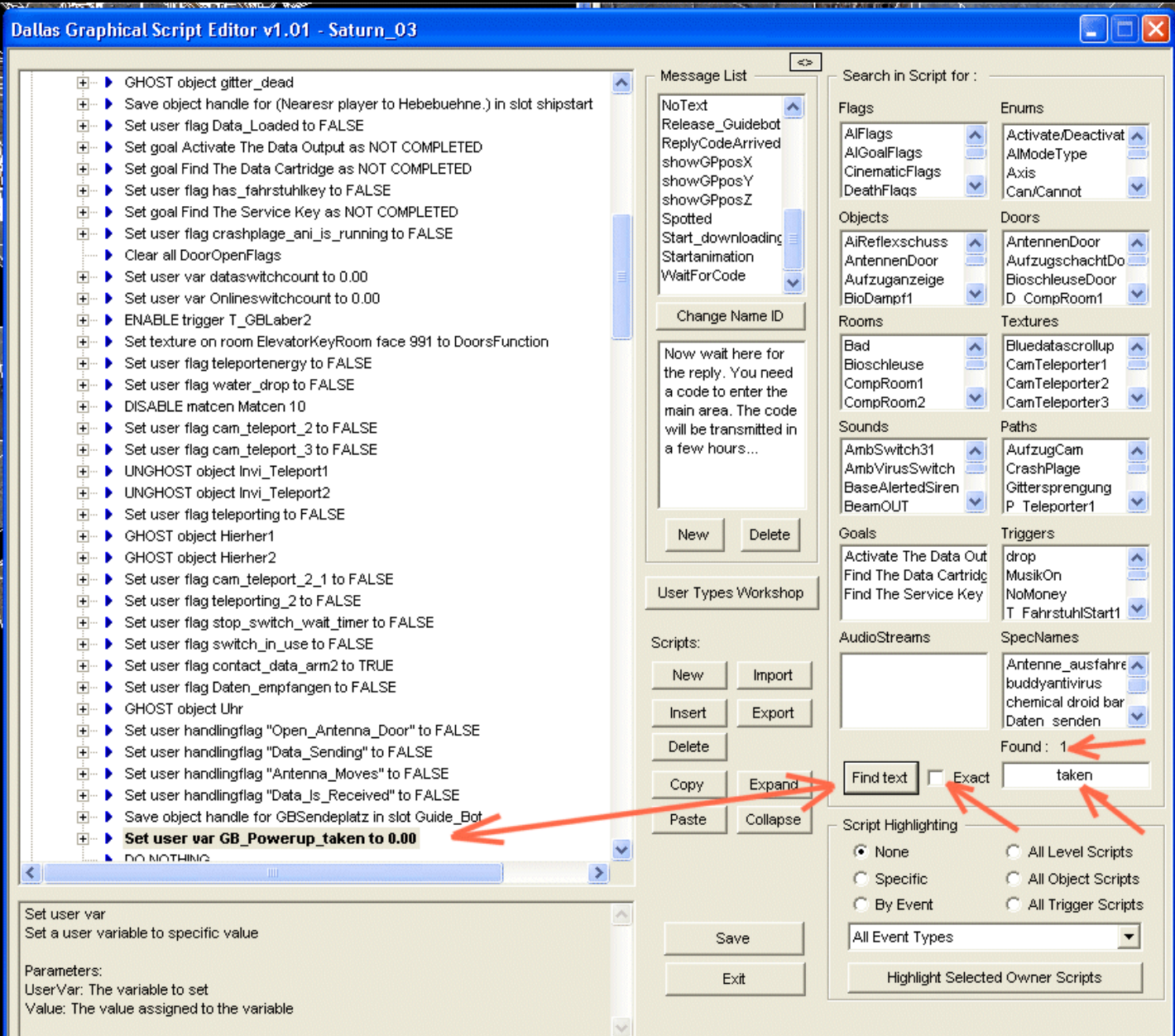
Script Highlighting

☒ None ☐ All Level Scripts
☐ Specific ☐ All Object Scripts
☐ By Event ☐ All Trigger Scripts

All Event Types

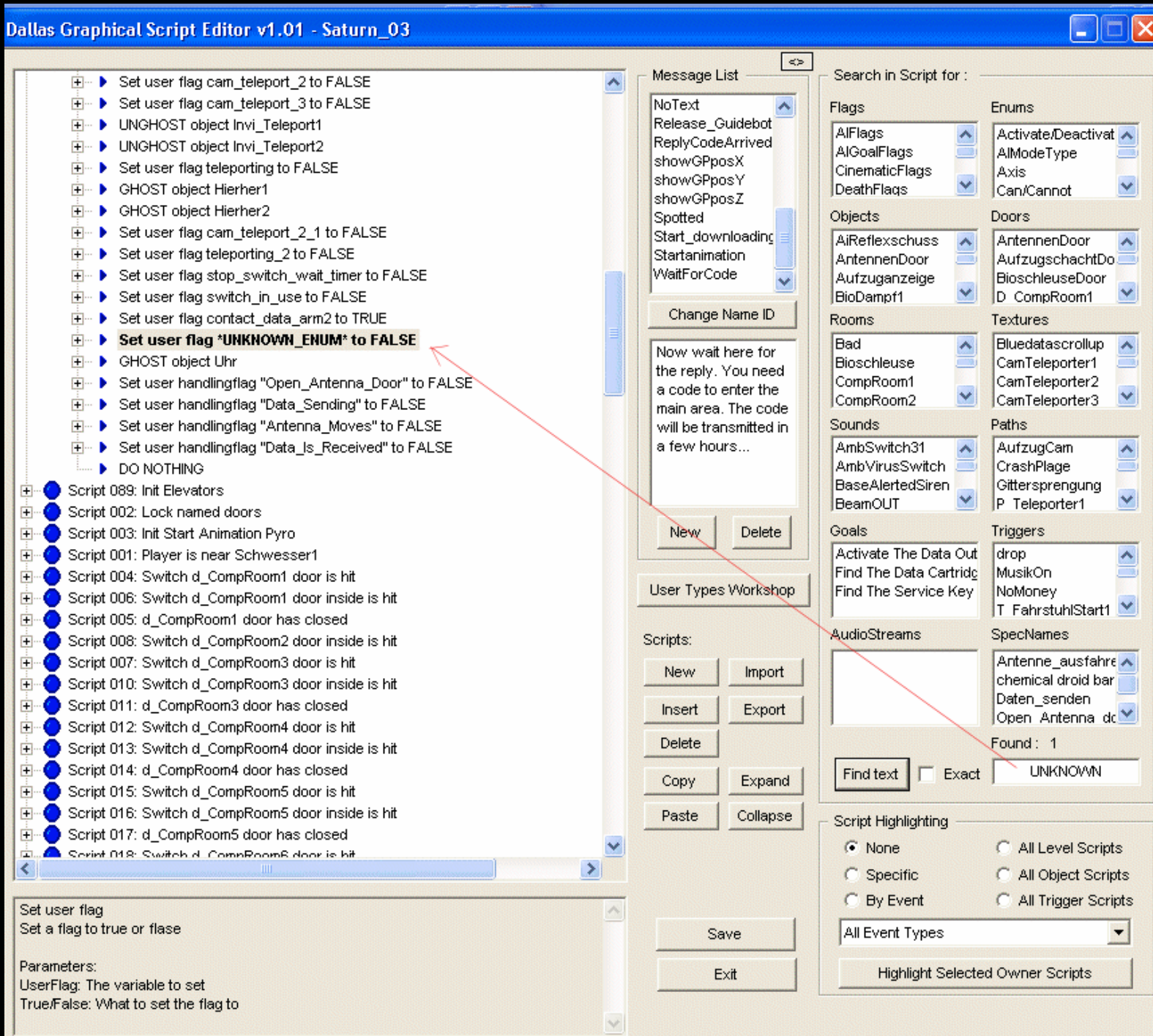
Highlight Selected Owner Scripts

It is now possible to search the script by mouse click or direct input and to highlight the locations found in the script lines. Here is a simple example:



It became the word **taken** entered directly into the search field, the checkbox 'Exactly' deselected and after pressing 'Find text' buttons we see that we have landed exactly one hit. The corresponding line is highlighted visually.

It's always annoying when DALLAS tells us that one or more script lines contain an 'unknown' object that needs to be changed. This example shows how the relevant location(s) can be found quickly:



The script opens automatically and the corresponding line is displayed. In new versions, DALLAS looks for such errors when saving and visually highlights them line by line.

DALLAS now fills corresponding list fields when starting. Left-click on the entry you are looking for and the search will start automatically. The searched text is entered into the input field, the number of hits found is displayed, all corresponding scripts open and the searched text is visually highlighted. You can use the scroll bar to bring additional scripts into view.

The screenshot displays the DALLAS software interface, which is used for managing and searching through a collection of scripts. The interface is divided into several main sections:

- Script List (Left Panel):** A list of scripts is shown, each with a blue circular icon and a plus sign. The scripts are numbered and describe actions related to doors and rooms. For example, "Script 004: Switch d_CompRoom1 door is hit". The list is scrollable, and a red arrow points to the scroll bar.
- Message List (Middle Panel):** This panel shows a list of messages or events. The first message is "NoText". Below it, a message is displayed: "Now wait here for the reply. You need a code to enter the main area. The code will be transmitted in a few hours...". There are buttons for "Change Name ID", "New", and "Delete".
- Search in Script for: (Right Panel):** This panel contains various search filters and options.
 - Flags:** A list of flags with checkboxes, including "AIFlags", "AIGoalFlags", "CinematicFlags", and "DeathFlags".
 - Enums:** A list of enum values with dropdown menus, including "Activate/Deactivat", "AIObjectType", "Axis", and "Can/Cannot".
 - Objects:** A list of objects with dropdown menus, including "AIReflexschuss", "AntennenDoor", "Aufzuganzeige", and "BioDampf1".
 - Rooms:** A list of rooms with dropdown menus, including "Bad", "Bioschleuse", "CompRoom1", and "CompRoom2".
 - Sounds:** A list of sounds with dropdown menus, including "AmbSwitch31", "AmbVirusSwitch", "BaseAlertedSiren", and "BeamOUT".
 - Goals:** A list of goals with dropdown menus, including "Activate The Data Out", "Find The Data Cartridge", and "Find The Service Key".
 - AudioStreams:** A list of audio streams with dropdown menus, including "Antenne_ausfahre", "chemical_droid bar", "Daten_senden", and "Open Antenna dc".
 - Trigger:** A list of triggers with dropdown menus, including "drop", "MusikOn", "NoMoney", and "T_FahrtstuhlStart1".
 - SpecNames:** A list of specific names with dropdown menus, including "Antenne_ausfahre", "chemical_droid bar", "Daten_senden", and "Open Antenna dc".
 - Found:** A text field showing the number of hits found (4) and the searched text ("D_CompRoom6").
 - Script Highlighting:** Radio buttons for "None", "Specific", "By Event", "All Level Scripts", "All Object Scripts", and "All Trigger Scripts". A dropdown menu for "All Event Types" and a button for "Highlight Selected Owner Scripts".
- Activate door (Bottom Panel):** A panel with a description of the "Activate door" action: "Activates (i.e. opens and possibly closes) the specified door". It also includes a "Parameters:" section with the text "Door: the object of the door to be activated".

Finding paths and text output (messages) becomes child's play.

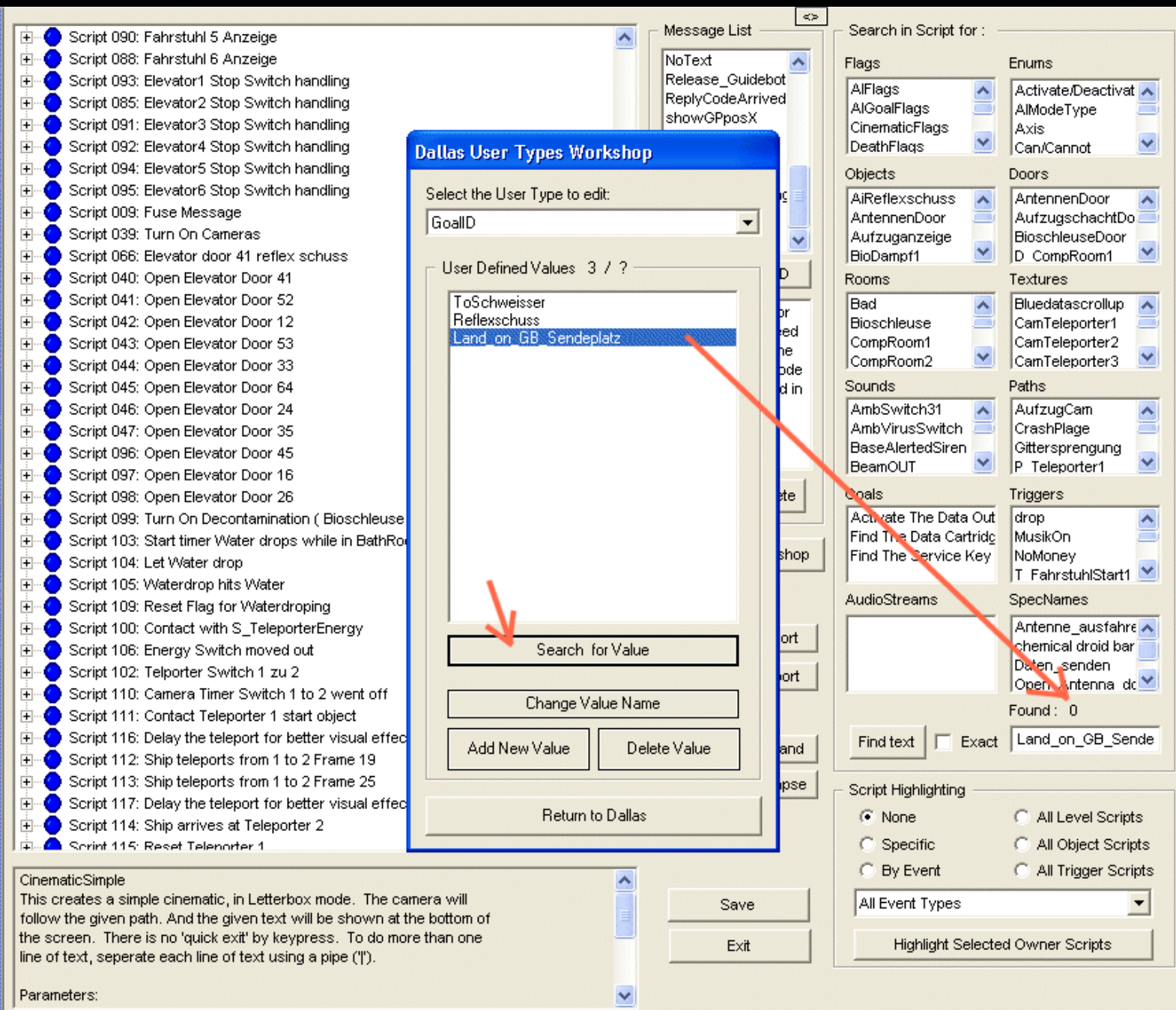
The screenshot displays a game development software interface with several panels:

- Script List:** A list of scripts on the left, including "Script 049: Track with Camera bot_46" through "Script 084: Fahrstuhl 3 Anzeige". A specific script is expanded, showing a sequence of actions: "Owner: Level", "Event: Timer (TIMER ID) Went Off", a conditional check "If the following condition is met:", and then "Then perform the following actions (only do once):". Under this, there is a "Simple letterbox format cinematic, using path AufzugCam, with text Elevator running" and "DO NOTHING". The path "AufzugCam" is highlighted in red.
- Message List:** A panel in the center-right showing a list of messages. The message "Your are spotted by a camera!" is selected. Below the list are buttons for "New", "Delete", "Change Name ID", and "User Types Workshop".
- Search in Script for:** A panel on the right with multiple sections for searching: "Flags", "Enums", "Objects", "Doors", "Rooms", "Textures", "Sounds", "Paths", "Goals", "Triggers", "AudioStreams", and "SpecNames". The "Paths" section is currently active, showing a list of paths including "AufzugCam", which is highlighted in blue. A red arrow points to "AufzugCam" in the "Paths" list.
- Script Highlighting:** A panel at the bottom right with radio buttons for "None", "Specific", "By Event", "All Level Scripts", "All Object Scripts", and "All Trigger Scripts". A dropdown menu shows "All Event Types" and a button "Highlight Selected Owner Scripts".
- CinematicSimple:** A panel at the bottom left with a description: "This creates a simple cinematic, in Letterbox mode. The camera will follow the given path. And the given text will be shown at the bottom of the screen. There is no 'quick exit' by keypress. To do more than one line of text, separate each line of text using a pipe ('|')." and a "Parameters:" section.

To search for the entries in the User Types Workshop, open it, select the entry and click on **'Search for Value'**.

This entry is not found in the scripts, so it is not used and can possibly be removed from the list.

You should always be very careful when deleting entries, as an error can easily creep in!



Another example:

Dallas Graphical Script Editor v1.01 - Saturn_03

Script 048: Track with Camera bot_43
Script 062: Make Hangturret Bot_59 precise firing
Script 055: Track with Camera bot_44
Script 063: Make Hangturret Bot_60 precise firing
Script 056: Track with Camera bot_45
Script 049: Track with Camera bot_46
Script 051: Make Hangturret Bot_61 precise firing
Script 054: Make Hangturret Bot_62 precise firing
Script 059: Track with Camera bot_49
Script 107: Track with Camera Teleport 1
Script 108: Track with Camera Teleport 2
Script 065: Make Hangturret Bot_87 precise firing
Script 064: Make Hangturret Bot_88 precise firing
Script 070: Show where guided keys should be used
Script 072: Check if all needed keys are switched to show grate animation
Script 068: Replace Grate Model
Script 071: Show exploding grate if enough keys switched on
Script 067: Do all necessary if player dies
Script 073: Find Service key
Script 074: WartungsDoor
Script 075: MatcenterOFF switch is hit
 Owner: S_MatcenAus (Object)
 Event: Collided (with IT)
 If the following condition is met:
 AND
 (IT is a player or player weapon) = (TRUE)
 (User Flag Matcenter_OFF) = (FALSE)
 Then perform the following actions:
 Play object OWNER animation from frame 0 to 1, cycle time = 2.50, looping -
 Set texture on room ElevatorKeyRoom face 738 to Offline
 Play Sound AmbSwitch31 from object OWNER, volume = 100.0%
 Set user flag Matcenter_OFF to TRUE
 Set texture on room ElevatorKeyRoom face 45 to ForceFieldsOff
 DISABLE matcen Matcen 7
 ENABLE matcen Matcen 10
 DO NOTHING
Script 077: Antose fuer brennenden Sicherungskasten

Set user flag
Set a flag to true or false

Parameters:
UserFlag: The variable to set
True/False: What to set the flag to

Dallas User Types Workshop

Select the User Type to edit:
UserFlag

User Defined Values 31 / 32

- downloadaddataswitch2
- downloadaddataswitch3
- downloadaddataswitch4
- downloadaddataswitch5
- downloadaddataswitch6
- teleporting_2
- hasdatacartridge
- cam_teleport_4
- ventilator_grate_broken
- stop_switch_wait_timer
- Data_Loaded
- show_exploding_grate
- Matcenter_OFF
- has_fahrstuhlkey
- switch_in_use
- Fahrstuhle_fahren
- contact_data_arm2
- daten_empfangen

Search for Value
Change Value Name
Add New Value
Delete Value
Return to Dallas

Enums
Activate/Deactivate
AI Mode Type
AtanUserVar
Axis

Doors
AntennenDoor
AufzugschachtDo
BioschleuseDoor
D_CompRoom1

Textures
Bluedatascrollup
CamTeleporter1
CamTeleporter2
CamTeleporter3

Paths
AufzugCam
CrashPlage
EndCamPath1
EndCamPath3

Triggers
drop
MusikOn
NoMoney
T_FahrstuhlStart1

SpecNames
chemical droid barrel

Found: 2
Matcenter_OFF

Copy
Expand
Find text
Exact
Paste
Collapse
Save
Exit
Highlight Selected Owner Scripts

This new search feature is very helpful for scripting. After you get used to it, you won't want to script without it. If used sensibly you can save a lot of time, try it out!

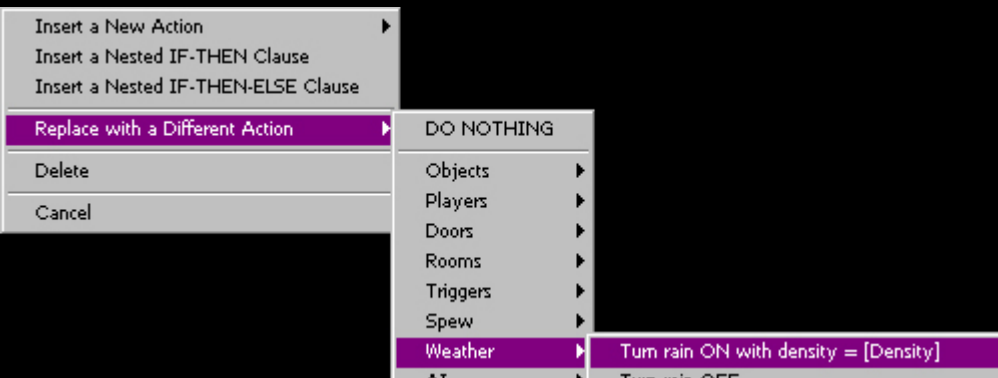
Back to the overview of Section I

097 - Make weather

Ragil Ral

This would actually belong in section E, but since it requires scripting it goes here.

It works like this: the weather is assigned to the level at the beginning by script. So build a script, set it Owner on level, the event on Level start and at the condition puts Always. Then - for rain - insert the following script:



At the Density indicates how heavy the rain should be.

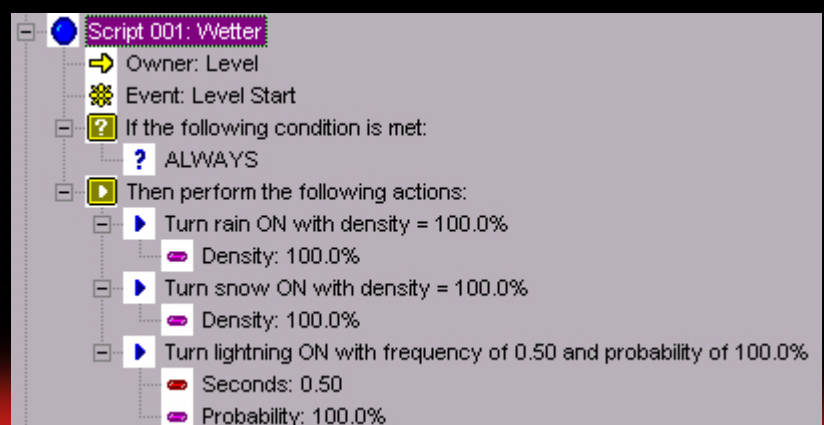
That was it. Just like rain works with snow, then you just have to use the action for snow, and of course you can use both at the same time.

There is also the option of making it flash Turn lightning ON with frequency of [Seconds] and probability of [Probability]. With the Frequency you specify which one intervals it should flash and with Probability what is the probability that lightning will actually occur?

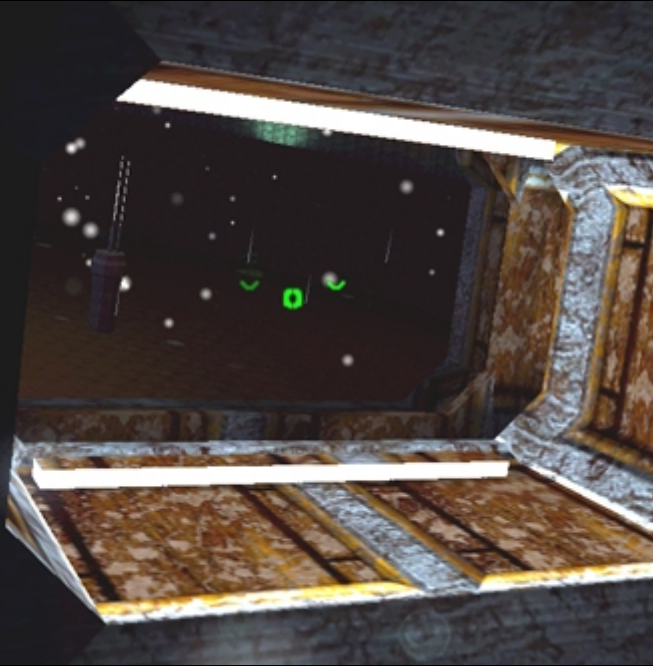
In this example script I make really bad weather:

You can then make the weather dynamic, make it rain more or less or change the frequency of lightning.

To do this, you simply have to assign new weather in a script-controlled manner, either based on the player's actions or time-controlled.



Unfortunately there are a few problems with the weather.



On the one hand, the weather is only simulated around the player and does not actually take place. Because weather one Property of the terrain, it is only perceptible on the terrain, except for the lightning. So if you are in an internal room with a good view of the terrain, you will not see rain or snow. Conversely, if you look from the terrain into an interior that is above the terrain, then there is 'weathering' in this room (left)

On the other hand, weather only takes place up to a certain distance from the player; Snow ends at around 200 and rain at around 700 units. When it rains, the additional problem is that it only opens the terrain or the graphical ones Impact effect produced, this also applies to hidden terrain cells (right); If it hits a polygon structure it goes through, just like snow.

You should keep these characteristics in mind when using weather so that it doesn't look too fake.



[Back to the overview of Section I](#)

098 - Refined player respawn

MZero

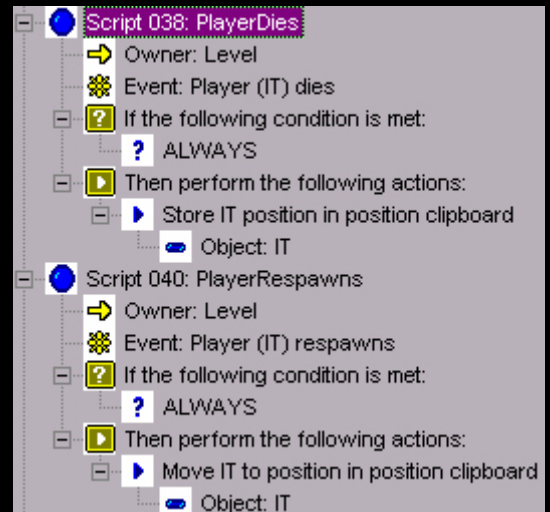
expanded

If the player is killed, he will respawn either at a player start or at a waypoint (see the Singleplayer section, LINK). But it can be helpful to change that.

As a D1/D2 dependent, when you die in PTMC mines you want to respawn in the same place, surrounded by your valuable weapons. It's easy to script.

Insert these two scripts into your level, you will see the necessary actionsObjects-Find menu:

The clipboard can only contain a single object position. Do not save any other object position between or within these two scripts and put the second script immediately after the first.



But you should think about where you use it; This can be useful in co-op missions, as the death of a player does not separate him from the group.

This script overrides waypoints and player starts, so you have to design the game flow accordingly. You could of course make the execution of the scripts dependent on a parameter, such as a flag, so that the respawn behavior changes over the course of the game.

Source:

<http://www.filelist.com/index.php?>

[option=com_content&view=article&id=13:mzero&catid=5:d3-construction&Itemid=4](http://www.filelist.com/index.php?option=com_content&view=article&id=13:mzero&catid=5:d3-construction&Itemid=4)

[Back to the overview of Section I](#)

099 - Teleporter <>

Fischlein, inspired by Starkiller

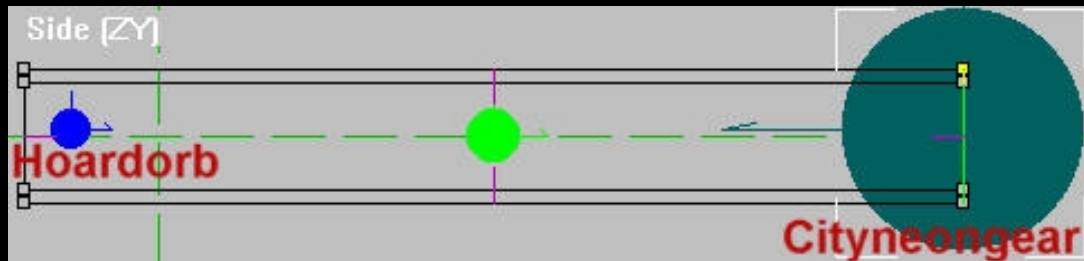
The files resulting from this tutorial are in the file.

We need D3Edit and Dallas Script Editor for this topic. First create a level. I have "New Level - Default Room" and stretched it 100 units.

The player starting point I positioned it in the middle of the room. I then use object mode on this wall the Clutter object "Cityneon Gear"

positioned, picture on the right

(you can also use any other object).



I now have this object "begin", you do this by clicking on the red-framed button in the picture on the left. So, now we need a "target powerup". I have positioned another powerup on the opposite wall, a Hoardorb (should be a powerup since you can choose the direction in which you exit). Starkiller points out that the two should be at least 50 units apart. Goal" called. Now save the.d3l.

We now need a script to bring the teleporter to life.

Switch to World View and select in the menu File/Dallas Graphical Script Editor, You can find out how this is set up further down. Click away the two information dialog boxes as usual. Once you have saved your script for the first time, it will no longer be displayed. In the picture on the right you can see what the script should look like. The explanation is in the boxes.

So that's it. You can also choose a portal as a teleporter entry (trigger).

Back to the overview of Section I

Scripts:

- Script 000: Start Level
 - Owner: Level
 - Event: Level Start
 - If the following condition is met:
 - ALWAYS
 - Then perform the following actions:
 - GHOST object Ziel
 - Ghost/Unghost: GHOST
 - Object: Ziel
- Script 001: Teleporter
 - Owner: Start (Object)
 - Event: Collided (with IT)
 - If the following condition is met:
 - (IT is a player) is TRUE
 - Bool: IT is a player
 - Object: IT
 - Operation: is TRUE
 - Then perform the following actions:
 - Store Ziel position in position clipboard
 - Object: Ziel
 - Play 2D Sound Extra life for player IT, volume = 100.0%
 - Sound: Extra life
 - PlayerObject: IT
 - Volume: 100.0%
 - Make object IT spark at a rate of 50.00 for 3.00 seconds
 - Object: IT
 - SparkRate: 50.00
 - Time: 3.00
 - Move IT to position in position clipboard
 - Object: IT

Clipboard

1 Says that this script will be executed at the start of the level. Once you have created the script, Owner says None. Right-click and select Level. Same with event.

2 Makes the "Target" object invisible. With the right click on "Then perform the following actions" click and then select "Add a New Action". Can then be found under Objects.

3 Here will be asked if "IT" player is. IT is one Variable.

4 You can find this function again under Objects. The object is placed in the "clipboard" delay.

5 Plays a sound when the teleporter is active. Can be found under Sound & Music.

6 Create lightning around your ship, similar to when you are low on shields.

7 Object is brought from the clipboard to the target. Can be found under Objects.

100 - Create Lighting <>

Fischlein

As always, the files for this are in the zip-File.

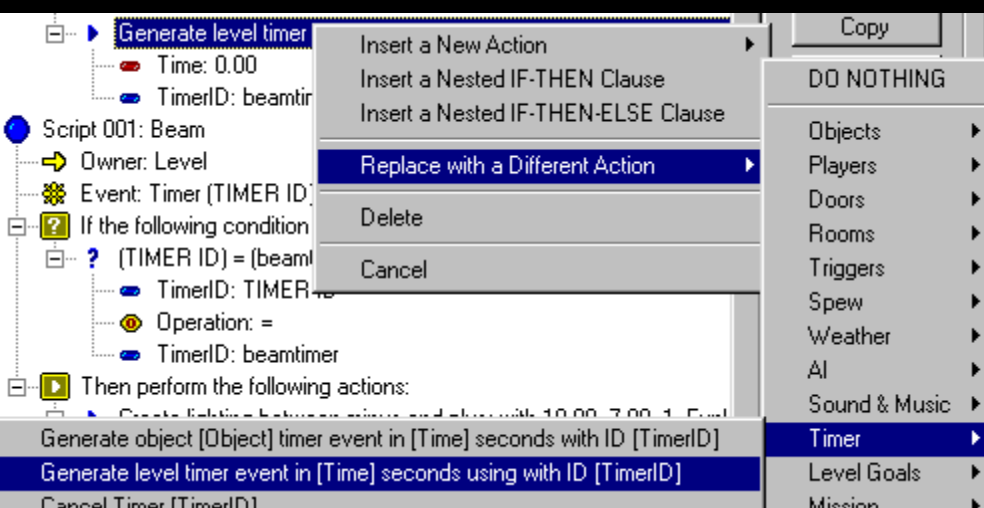
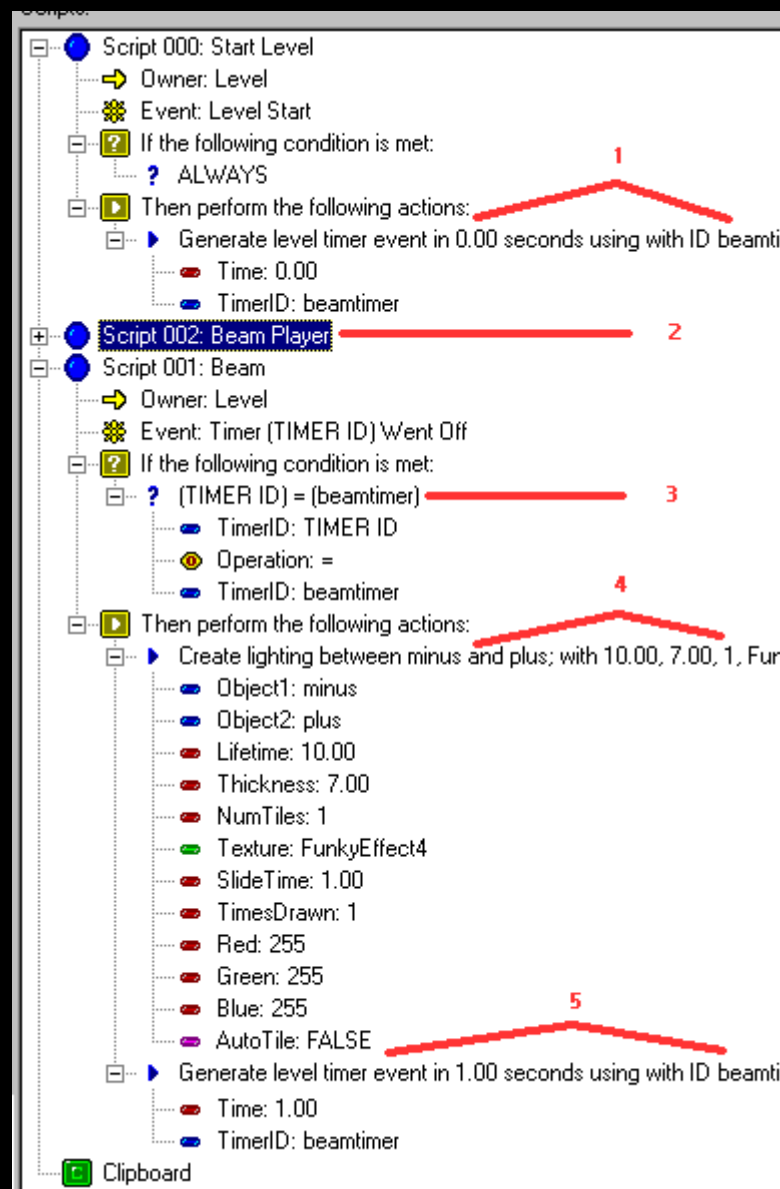
For this topic I assume that you have already dealt with scripting. If that's not the case, you can continue to add more information. Ok, first of all a picture on the right that shows you everything about the script. One more piece of information in advance: you need two objects (powerups) for this script, I have them

Invisiblepoweruptaken, and then have the one object "plus" and the other "minus" called.

Regarding point 1:

A level timer is initialized here. Now you might ask yourself 'what's the point of that', that's what I thought at first, but if you don't use the timer, you won't be able to keep the beam running all the time, because with "Lifetime" determines how long the beam can be seen and another effect is that you cannot see the beam in multiplayer (clients). I then asked the author of "**Botstample**" (**Bijwi**) who explained to me that you have to initialize the timer at the beginning of the level so that the beam can also be seen in multiplayer. It is simply set to zero, except that the variable is there for it (picture below).

Around the **TIMER ID** to specify this, we now have to create a variable, I have it "**official timer**" called. Click on "**Open the Users Types Workshop**", in the following dialog box you select "Select the User type to edit" from the list "TimerID" and now click on "**Add New Value**". Bear there "**official timer**" a.



To point 2:

Here I have once again shown another option that you can do with it. Take this. zipcome and take a look at it. Put a camera so you can see what I mean. In

the .zip are also all the files you need to view the script.

Regarding point 3:

It was a bit difficult for me to find, so I'll explain it again. First

the one shown in the picture on the right

use function; next right click on it. now you can "TimerID" and "official timer"

erate level timer event in 0.0

Time: 0.00

TimerID: beamtimer

Beam

Level

Timer (TIMER ID) Went Off

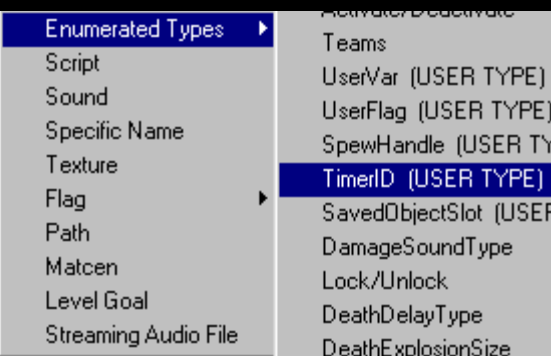
Following condition is met:

er Variable None) = (0.00)

Float: User Variable None

Replace with Query

Replace with Literal

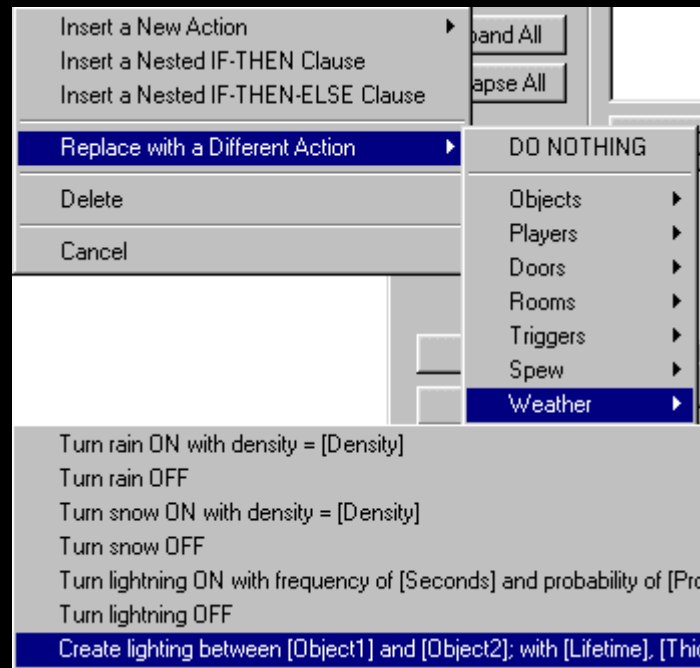


Regarding point 4:

Now comes the actual function that is what we are talking about here. You can see where you can find the function in the picture on the right.

I try to explain the parameters in the table. (I did the translation **Scorpio** helped a lot! **Many thanks again to Scorpio!**)

register. (Picture on the left).



Object1	The first object, in my case " minus "The
Object2	second object, with me " plus "
Lifetime	The time for how long the beam is displayed is specified here. I built in the timer so that it always runs.
Thickness	How wide/thick the beam/texture is displayed.
NumTiles	How many textures are displayed between objects (NumTiles =1 -> Texture , NumTiles =2 -> Texture Texture etc.)
Texture	The texture that should be displayed in my example " FunkyEffect4 "
SlideTime	I'm not entirely sure here, translated it means something like " <i>If you want the texture to move along the lightning surface? (or light area/surface) moves, this controls how fast it goes.</i> " But if I set the value to 100, it almost seems like the beam is standing still.
TimesDrawn	This value should always be 1, otherwise the frame rate will go to zero. This value is responsible for saturation. It will repeat the texture so many times. For a more blurry effect or to make it a little brighter, use a number larger than 1. (At 100 the frame collapses completely)
Red, Green, Blue	Are the normal RGB color values that are used to specify the light color of the texture's lighting.
AutoTile	If "True" is set, so many textures will be output between Object1 and Object2 fit. (does this overwrite NumTiles?)

You can play around with the parameters. **Regarding point 5:**

The level timer that we initialized at the beginning is set here. I set it to one second so that the beam can also be seen in multiplayer.

Back to the overview of Section I

101 - Wind Scripting

Papacat

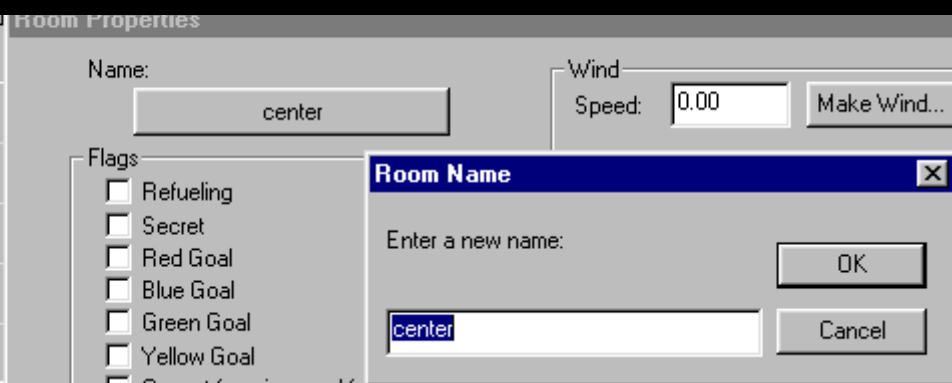
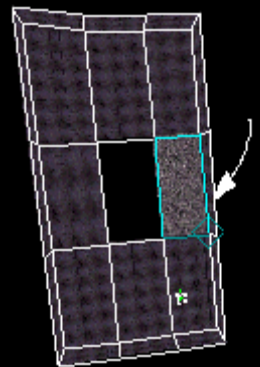
Here I will explain the use of wind and how to script it into the level. First, some rules about wind:

- 1) Wind is a spatial property and is applied to the entire room.
- 2) It blows in one direction per room, it doesn't make any turbulence.
- 3) You can script changes in wind direction or speed triggered by an event.
- 4) A speed of 10 is faster than a player ship.
- 5) You can set wind in the room properties (not scripted), but it is always on and cannot be changed.

Let's get started. First I make a small level. It consists of two rooms, one is U-shaped, the other is a simple cuboid. We will put the wind in these.

The room must have a name so that we can script to it. Make it the current room. Then go to the Current Room View Windows->Room

on, the **(none)** says. rush the Current Room

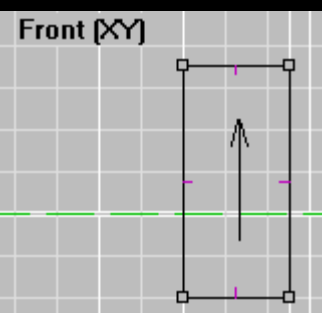


Save and start Dallas. We will set the wind at the start of the level. Go first Then Perform the following actions: and Add a New Action. Wind is a property of space, so investigate Rooms. Choose the one Set wind in [Room]...-Action; Below right is a picture of where it is located.

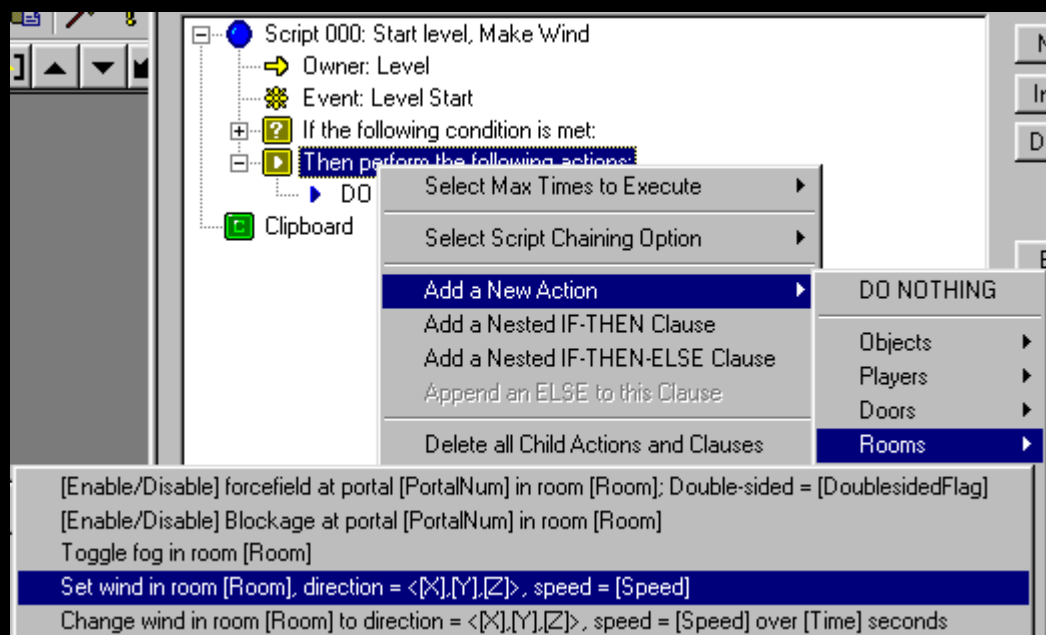
If necessary, open it Branch and right-click to set the room (bottom left).



x,y and z are your directions. We're going to set the wind as shown here by y to 1



place. Set the speed on 6.



That's it. Save the script, compile it and add it to your mission. Back to the overview of Section I

102 - Music regions

Kyouryuu

Learn how to.osf-Makes files and generates event-driven music for single-player levels. Parts of this tut were inspired by the work of Wolf on Air and Shoku in various DescentBB threads.

introduction

A trend in today's computer games is the use of dynamic, event-driven music. Basically this means that the music changes according to the mood of the level. For example, if we are strolling down an empty corridor, the music might be subtle and muted. Contrast that with an intense dogfight where trash metal and electric guitars scream from your speakers. The trick is to crossfade from one track to the next so that the effect isn't as abrupt as in an old Nintendo game. Descent 3 uses this method in all of its single-player levels. Each level's soundtrack is actually split into several fragments, which are organized with others in an internal playlist. Which part of this playlist is heard is determined by scripts in Dallas.

For you, the level designer, the effect is the same, but you have to do things a little differently. While all D3 playlists are hardcoded, you must have one.omf-Create file to use your own. You also need to get your tracks from.wavafter.osfconvert them, and then set up the Dallas scripting to use them. The process is simple if you know how, extremely unclear if you don't. Let's get started.

Quick reference

Music files for Descent 3 must be in.osfformat is available. MusicTester - in the D3Edit directory - converted.wavfollow up.osf.The .wav file(s) should have the following properties: 22050 Hz, mono, 8-bit, PCM. It was stated by pATChes11 that they do not have to be mono or 8-bit (that stereo and 16-bit also work). However, the 16-bit setting is subject to sample distortion. They can be converted using something as simple as MS-Sound Recorder. It should also be noted that heavy bass in the.osfformat sound terrible. Great care must be taken to ensure that the music does not 'pop' or create any undesirable effects in cheap speaker sets.

Rules and regulations

As I mentioned before, Descent 3 determines which fragments of a soundtrack to play based on a playlist called a.omffile is saved. In this you define the music regions. Such a region consists of two parts - an 'idle' theme and a 'death' theme. The Idle theme may be composed of several music fragments, each with its own number of repetitions. The optional Death Theme is heard when the player bites the dust. Let's say I have two music regions, one we'll call 'Upper' and one 'Lower'. An event, say the level start, sets the music region to 'Upper'. So the tracks in the upper region start playing when the level starts. Now let's assume that a player later sets the music region to 'Lower' by entering a room. The tracks defined in the .omf for 'Lower' will not start playing until the current track from the 'Upper' region has finished. The lesson here is to make the tracks short, around 30 seconds for each. The other consequence of this is that you can't make the music suddenly change. So, for example, if you want 'Boss music', you need to find a way to ensure that the main theme of the level is finished before the boss theme begins. Otherwise it will drag itself into the fight with the boss. As far as I know, you can't stop the music immediately. Descent 3 also doesn't crossfade tracks together, so it's up to you to make sure they transition smoothly from one to the next.

The .omf-Create file

The **OutrageMusicFile** is a small plain text file that contains a list of all available tracks and defines the music regions with names. You have to understand that the `....omf` exists solely for use in the game. D3Edit will not read them, more on that later. You can with the `MusicTester.omf`'s (in the D3Edit directory). He makes `.wav`->`.osf`-Conversions and also defines the music regions. Before you start the conversion, make sure your music files are in the `.wav` format are saved. Also pay attention to their properties (see above). Then go follow up `Actions` -> `Convert Files`, use the button **Add File(s)...** to add your files. Put that `Destination Folder` on the desired location, then click **Convert Files**. If the conversion of a file goes wrong, it was most likely saved in the wrong format. File conversion takes about the same time as the tracks are long. Click **Exit** when you're done.

Now we have a few `.osf` files to work with. The first thing we do in our `.osf` file before we add regions is to define what our stream samples are. Under `Actions` -> `Add a Group of Stream Samples` you can do all yours `.osf`-Select files that you have just converted. When you're done, you'll see that `Stream samples` branch in the main window area now has a list of `.osf` files.

At this point we will now define the music regions. Use the + and - next to the names to expand the branches. Expand the 'Default' region completely. You could click on "'Default' region" Double-click to rename it. Next, you should go to the area 'default_idle' Subtheme (Looped)'Samples to Play' see. Right click on it and do `Add New Subtheme sample`. You will see a pull-down menu of files that have been added previously, next to a box to specify how often they will be repeated. Please note that this entire region is looped continuously - the repetitions in this case only affect the respective region `.osf` in the respective region. This allows me to 'rgregion1' twice and 'rgregion2' let it loop once. All of this is looped - meaning that in two passes I 'rgregion1' four times and 'rgregion2' will hear twice. Repeat this as often as you like `.osf` files want to play in this region. You can do the same for the 'default_death' make subtopic. I suggest neither changing the names of the sub-themes nor adding new ones, as this will unnecessarily complicate things. Ignore those too `Rules`. We will define our rules in Dallas.

To add a new region, right-click an existing one and select `Insert New Region`. Note that the new region appears directly above the one you right-clicked on. This may not seem important, but it is. Because of a bug in `MusicTester`, it is crucial that the first region you define is the first one you play. For example, if this region is called 'Intro' for my intro sequence, I would need to make sure that 'Intro' is first in the list.

If you the `.omf` file, you must save it with the complete file name. Means that if you `gamma.omf` you want to name exactly `gamma.omf` you have to write in the save dialog. `MusicTester` does not append the file extension to the file name, another bug.

Getting Dallas to See the Music Regions

As mentioned before, Dallas is blind when it comes to music regions. It doesn't read any information from the `.omf` file. So when it comes time to add the script, you use `Set music Region to XXXX` for all `Players`-but wait, what is it? `XXXX`? Somehow we have to tell Dallas what the names of the music regions are. To do this you have to go through it manually `.cpp`-File your level digging, which is by far one of the most obscure facets of D3 level design.

Normally it will `.cpp`-File generated by Dallas in `/osiris`-Subdirectory of the installation location of the D3SDK package. Open it with Notepad or another text editor. By searching the document manually or by hand, look for:


```
// ===== // Start
of Custom Script Block - DO NOT EDIT ANYTHING BEFORE THIS //
===== /
**{CUSTOM_SCRIPT_BLOCK_START} **DO NOT EDIT! **/
```

In this area of the file you can manually enter the names of your regions, like this:

```
// ===== // Start
of Custom Script Block - DO NOT EDIT ANYTHING BEFORE THIS //
===== /
**{CUSTOM_SCRIPT_BLOCK_START} **DO NOT EDIT! **/
```

```
// Enter your custom script code here /*
```

```
$$ENUM region
0:Intro
1:Upper
2:Lower
3:Reactor
4:Escape
5:Ending
$$END
*/
```

I think you can imagine the pattern from here. If you want to add regions, just increment the number and add a name. If you want to remove regions, do so and make sure the numbers stay consecutive, starting with zero. However, you must have them in the same order as they are in the .omf file. Otherwise you'll get the wrong regions playing at the wrong times and - ayayay - a mess.

Music Regions in Dallas Scripts

That's the easy part. Create a standard Dallas trigger. In the Then perform the following actions area use Sound & Music -> Set music region to [Region] for all players. Specify the region. Complete. Here's an example, but something to think about. If you trigger the music region change by entering a room, you may want it to change back when you enter the previous room. In other words, let's say I have two buildings in my level. I should have defined a music region for building 1, which changes when I enter building 2. If I go from building 2 to 1, it would be nice if the music changed back accordingly. Just a suggestion.

Conclusion

Anyway, this music stuff - you do it once and you never forget how to do it again. It's not as difficult as it is opaque. I hope my tut has shed some light on this topic.

additional comments

pATChes11 adds: 'There is also a small issue when using 8-bit stereo audio where the OSF converter may crash, but there is a fix: *if* it crashes, open your.wavin a decent audio editor and delete a single audio sample, and try the conversion again (this has to be done because of the way the OSF compressor works)'.

Back to the overview of Section I

103 - Spew!

Starkiller

You'll learn how to insert Spew into a level and use its many fields for level effects.

A new level with the default room will be sufficient. Put an object of type `forcefieldnode` in the level and name it `spewer`.

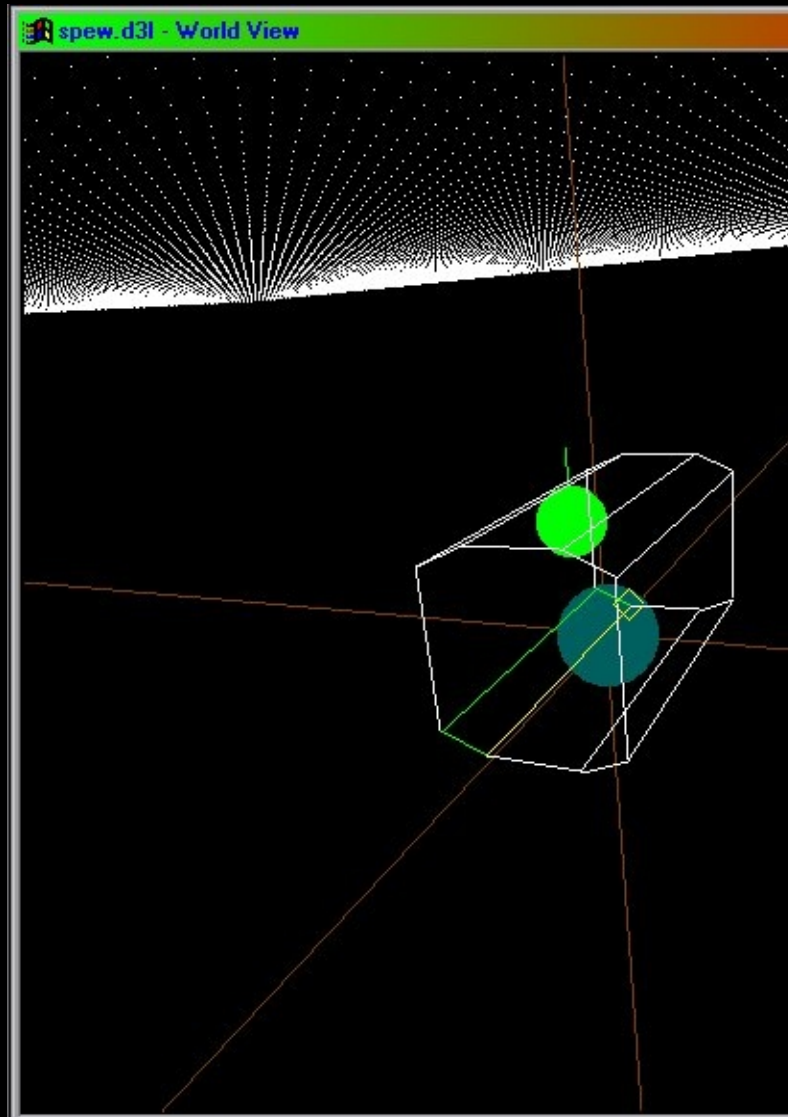
What is Spew anyway?

A spew is a collection of small 'blobs' that are... spit out. Each blob is basically an animation thrown into the air with some configurable variables.

Dallas

Make a script, set the `Owner` on `level`, the `event` on `Level Start`; the condition sets up `ALWAYS` and as the first event 'ghost' the object `spewer` (Objects -> [Ghost/Unghost] object [Object]).

Add a new action from the `Spew`-section, `Turn ON spew....` Set up the object you want to spew from `spewer`, the `GunNum` on 0, the `SpewType` on `Blue Fire`, the `SpewLife` on -1 and hook `everyonePhysicsFlags` away. Set `IsRealObject` on `TRUE`. Leave the other properties at their default settings. Now you have a pain-inflicting flame spear that is always on.



Dallas Graphical Script Editor v1.0 - spew

Scripts:

- Script 000: level start
 - Owner: Level
 - Event: Level Start
 - If the following condition is met:
 - Then perform the following actions:
 - GHOST object spewer
 - Turn ON spew from spewer at gunpoint 0;Blue Fire,0.00,0.00,[NO FL
 - Object: spewer
 - GunNum: 0
 - SpewType: Blue Fire
 - Mass: 0.00
 - Drag: 0.00
 - PhysicsFlags: [NO FLAGS SET]
 - IsRealObject: TRUE
 - BlobLifetime: 1.50
 - BlobInterval: 0.15
 - SpewLife: -1.00
 - BlobSize: 4.00
 - BlobSpeed: 20.00
 - Randomize: TRUE
 - SpewHandle: None

Clipboard

you can find some
in it
Spew
previous Spew
valon3. This is
possible.

eight? now to
complex part
all of these fields
applies...

Object	Simply the object that emits the spew blobs.	
GunNum	This is the gunpoint that expels the spew blobs. Note that not all objects have gunpoints, which is why I always use the forcefieldnode as a spewer. You can see if an object has gunpoints by inserting it into a level and viewing it in a textured view. If you see small orange balls on lines protruding from the object, it has them. The gunpoint balls are numbered so that you can see where the gunpoints are on the object.	
SpewType	Which animation is used for the blobs. Note, the entry 'napalm' listed below doesn't work; You have to instead Blue Fire use.	
Mass	The 'weight' assigned to the blobs. Usually left at 0; most spews don't have much physics. Note: If you use the flags 'Gravity' or 'Reverse Gravity' you use affects the Mass how much the blobs move according to gravity.	
Drag	The amount of 'stopping power' the blobs have. Higher Drag causes the blobs to slow down and even stop in mid-air. Usually leave it at 0.	
PhysicsFlags	These are settings that can be ticked/unchecked that help define how the blobs work and function in the game.	
	Fixed Velocity	This means that the blobs always move at the same speed throughout their lifespan.
	Gravity	Determines whether the blobs are affected by gravity. Note if you have no gravity to the level and no Mass to the blobs, setting this flag has no effect.
	No collision	Make sure the blobs don't hit anything. Setting this flag means that a Blue Fire-Spew cannot hit and damage a ship. Most often used for steam or black smoke that is for visual purposes only.
	Reverse Gravity	Like that Gravity-Flag, but it makes the blobs act in the opposite direction to gravity (rising steam.)
IsRealObject	This defines whether the blobs are real objects or just things to look at. Very similar to that No collision-flag. Put it on TRUE, if you want players and other objects to be hit by it.	
BlobLifetime	How long each blob 'lives'. $\text{BlobLifetime} \times \text{BlobSpeed} = \text{How far the blob moves.}$	
BlobInterval	How many times a Spewer spits a blob into the air. Higher values mean fewer blobs per second, lower values mean more. The unit of measurement for BlobInterval is 'distance between blobs in seconds'.	
SpewLife	The lifespan of the spew as a whole. Simply how long the spewer spews the blobs from its 'starting point', or when the script turns the spew on. If you put them SpewLife on - 1, the spewer never goes out.	
BlobSize	The size of the blobs, or how big the animation is that is being thrown into the air.	
BlobSpeed	The speed at which the blobs move.	
Randomize	Gives the spew a slight offset on the BlobInterval, which makes the spew more realistic. Always open TRUE let 😊	
SpewHandle	This is the name you can assign to the spew. This would be used by a script where a flame spew remains present until a switch is flipped to turn it off. For spews that do not shut off or are timer controlled, this can be left blank.	

104 - Broken glass with shards <>

Papacat

As a prerequisite you need a texture that Breakable flag has. To find out how to create something like this, see the textures section.

Krag showed that you don't necessarily have to script to break glass. I tested this; He's right, but you have to use 'Dirty Glass' or a custom texture with the flag Breakable have. Some people also had problems without scripting (don't know why).

First of all, (in case you didn't know) glass is a force field on a portal. The faces of both sides of the portal should be textured as glass. Let's get started.

Place a trigger on a portal

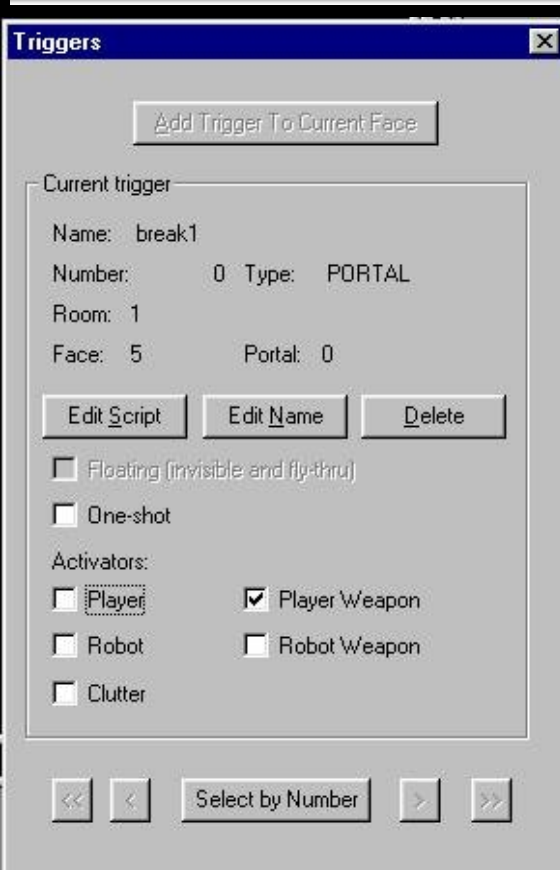


On the left is the example level. It consists of three rooms.

In the picture below left I have shown the portal number; This shows us that the current portal is #0 (lilac) and that the current room has a total of two portals. This is what you need to know when it comes time to script. Press P on your keyboard and cycle through the portals of the Current Room.

Make sure the textures on both sides of the portal are Dirty Glass (or another with the flag Breakable). It is also important that you use the space in the Room Properties name, I have mine centercalled.

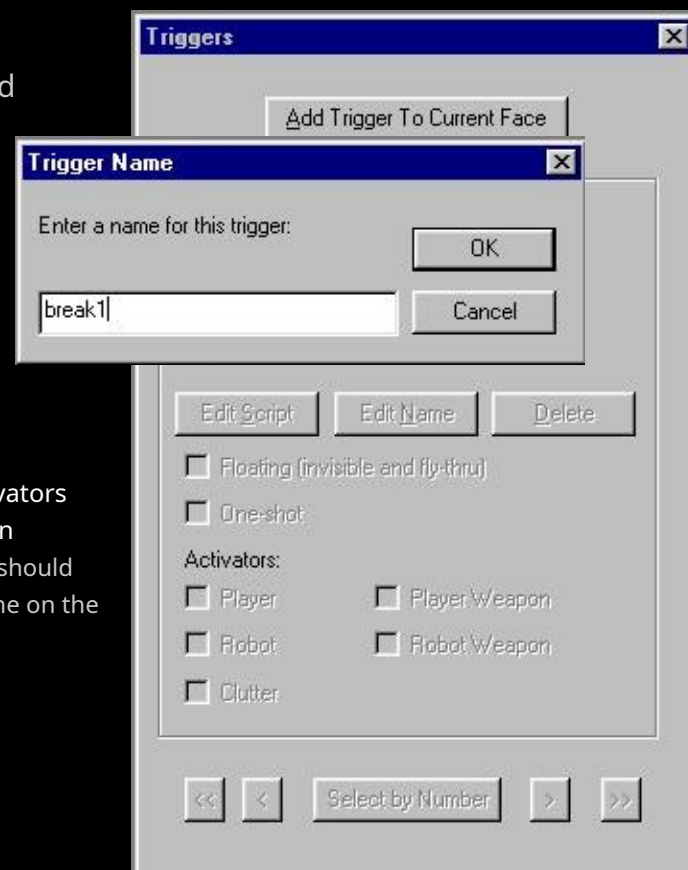
To break the glass we need a trigger. Go to Window -> Triggers, You will get the dialog shown below right.



With the portal face highlighted, click Add Trigger To Current

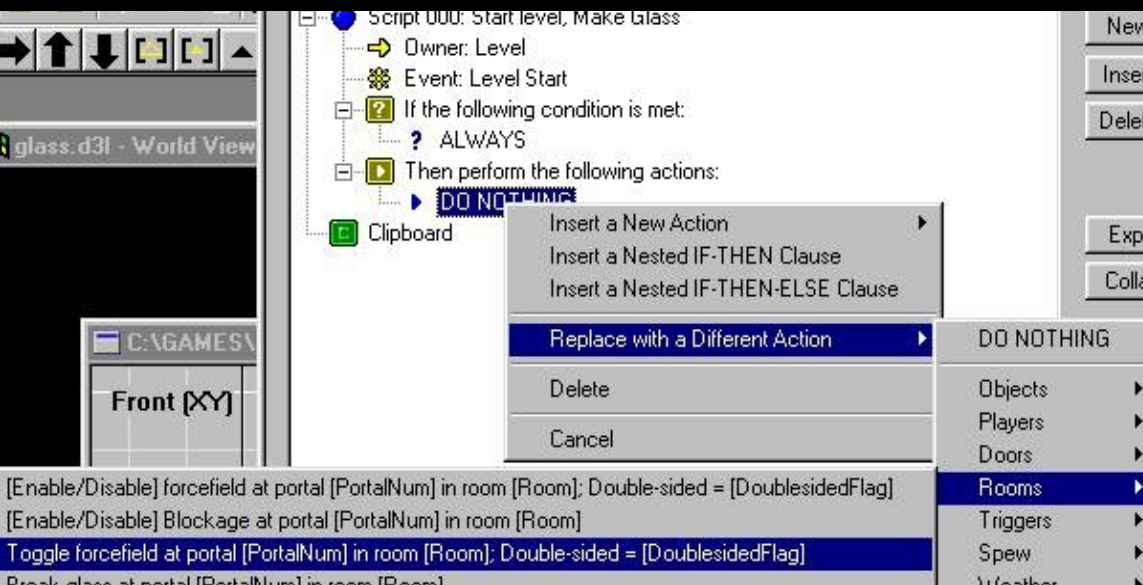
Face. In the subsequent Dialogue name the one Trigger, that's what you're going to do You while scripting the break reference:

Make sure that atActivators the hookPlayer Weapon is set. Your trigger dialog should look something like the one on the left.



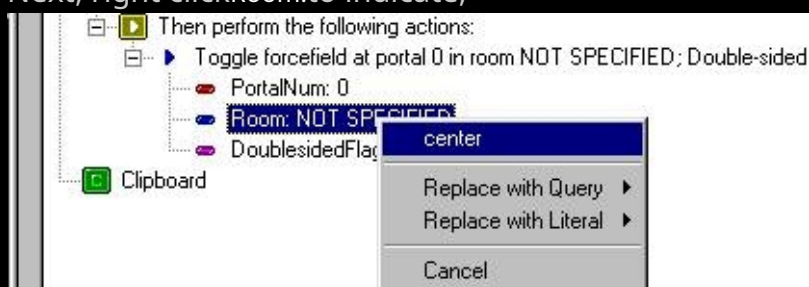
Activate the force field in the script = make the glass

Here we tell the scripting to put glass in the portal. An independent script can do this else want to set. w actionORReplace] forcefield...and



Toggle forcefield... and none Difference found. Important is itDoublesidedFlag to use and it onTRUEto set. Through this You only have to do this for one page of the portal make.

To get to the action's specifications, expand the branch. Right click on PortalNum, chooseEnter Integer Valueand set0a. Next, right clickRoom:to indicate, which room we are interested inPortalNum relate. In the menu that appears, all rooms are listed with assigned names. If you forgot to name the room, do that beforehand. The DoublesidedFlagstill set upTRUE. Your glass is now installed.



Script the break

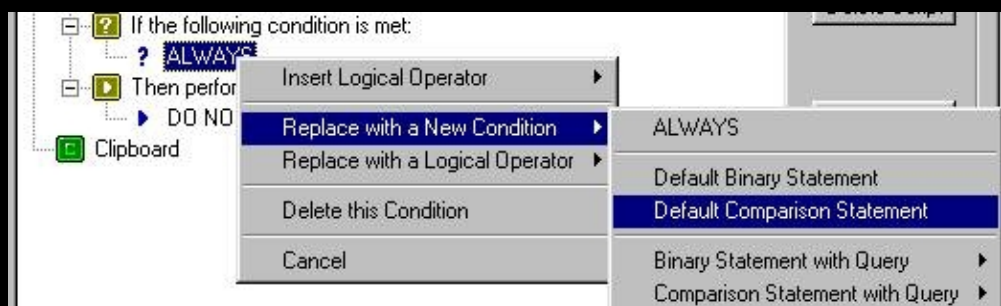
Make a new script and name it.

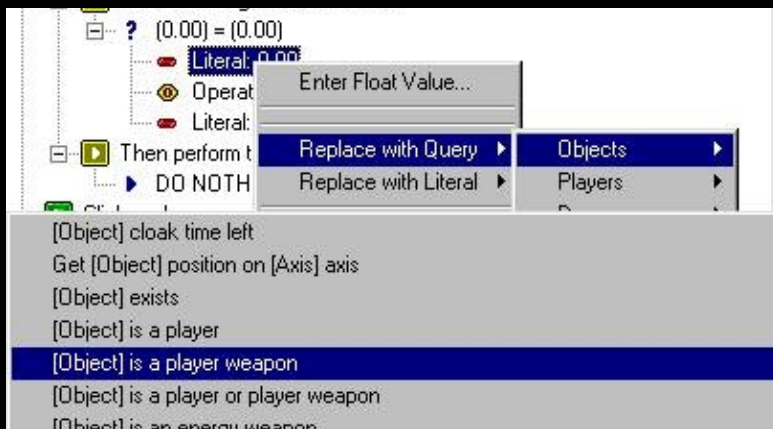


First we set thatOwnerof the script on the trigger. To do this, right-click on it and you will get a list of all triggers in your level. If youOwnersets to a trigger, the event automatically changes toActivated by (IT).

We want our glass to only break under certain circumstances. In this case that is: 1: it is a player weapon, and 2: it is NOT an energy weapon.

First we changeALWAYSto a comparison statement. We compare whether1.Tis a player weapon or not.



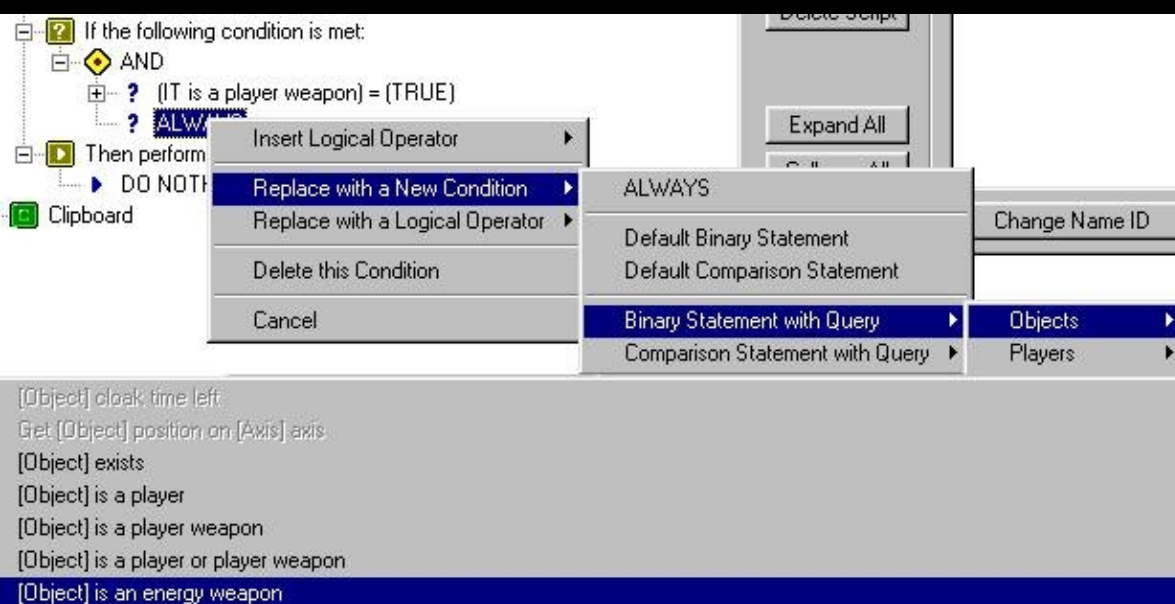


Right click on literal and put it according to the picture on the left to check whether it is a player weapon; it then changes to bool. Change that if necessary bool after surgery to TRUE. The result is (IT is a player weapon) = (TRUE).

Currently the glass would break if hit by any player weapon. We only want it to break non-energy weapons (Vauss, Concussions, etc...). Right click on (IT is a player...) and set one AND-Operator on.



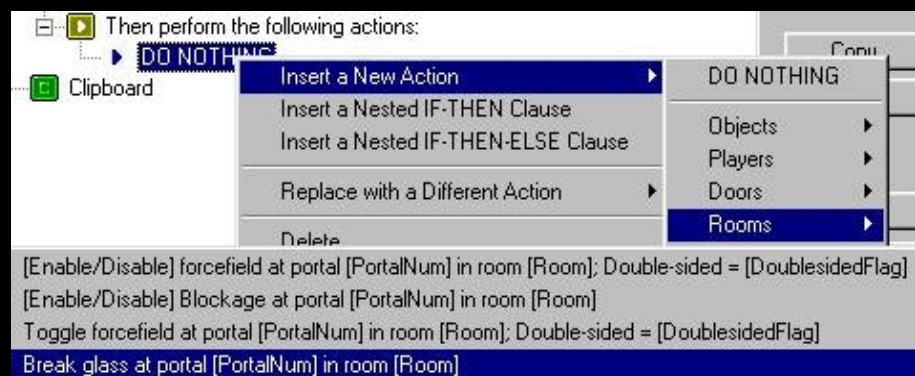
Below you can see how this works AND incorporated and the two conditions that must exist for this AND to fulfill. You can do that AND insert before specifying the conditions; it would just be two ALWAYS-Statements available for modification. If you want to use a third test condition, simply add more AND or OR instead of adding a condition, and it



branches out further.

Change that remaining ALWAYS after Binary statement with Query-> (Object) is on Energy Weapon like here on the left. Set the operation after FALSE.

All that's left is to break the glass. According to the picture on the right, set the action and set the [PortalNum] and [Room] in the same way as you did Toggle forcefield... you did above.



Your two scripts should now look like this on the right.

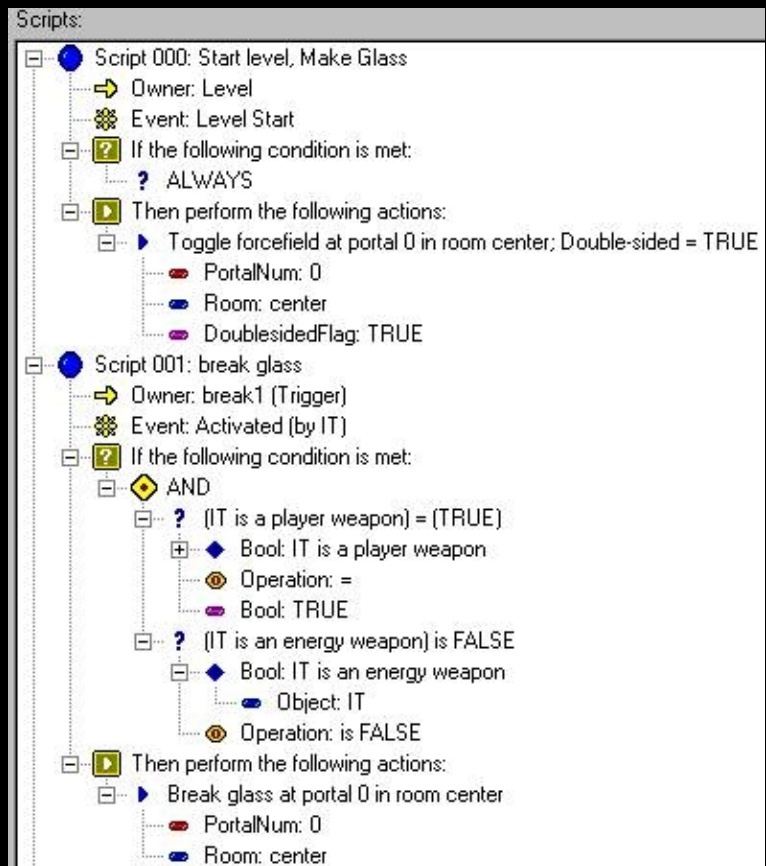
Save/compile it.

Although you can assign the mirror property to the scripted broken glass, this is not recommended. This creates an effect similar to having two faces on top of each other (=duplicate faces).

The glass breaks, but the mirror stays put.

Congrats! Now you have glass in your level that creates shards when broken.

[Back to the overview of Section I](#)



105 - Scripting a Switch I

WillyP

Credits

Atan wrote most of this tut in response to a question from Nightmare on Descentforum.de.
Thanks to both.

Let's get started

Do a simple level, two small rooms separated by a door. Install a switch somewhere in the first room, where the star player is. One of the handleswitch type would be best. Name him **MySwitch**. Verify your level and save it as **switch test**. Test fly your level and look for your switch. Shooting it doesn't do anything, it's just to make sure your level is ok. Also test the door. Go to World View and start Dallas.

Step 1 - First script

-> New script
Script000: name it **PlaySound**
Owner: MySwitch(Switch name, right click -> Objects -> Clutter -> MySwitch)
Event: Collided with (IT)
If The following Condition is met:
ALWAYS
Then perform the following actions:
Play 2d Sound Bell for Player object IT Volume 100%(Or any other sound except ENV sounds)
DO NOTHING

Dallas works until '**Done**' and - if everything went well - produces the files **switchtest.dll**, **- .cpp**, **- .o** and **- .msg**, if you have your level **switch test** mentioned (in the directory **<SDK>\osiris**). Include the script files in your **.mn3** and test fly your level. When you press or touch the switch you should hear the sound. This has to happen before we move on! Next we will animate the switch. I highly recommend taking a test flight after each step. Functions? Good! Further!

Step 2 - Animate the switch

We'll do that collision; add this action to the previous script:
Play object MySwitch animation from frame 0 to 20, cycle time = 1.00, looping=FALSE

Don't forget to save/compile. Use Quicktest for test flying, then you don't have to insert the script files manually every time.

A hit now produces sound and movement. Note that the animation speed in this action is with **cycle time** certainly; For example, increase it to 5.0 and see what it looks like.

Step 3 - Use a condition to limit the switch reaction to player weapon hits

You know the noisy effect when you fly onto a switch?

Replace that **ALWAYS** in the...following Condition...through:
(IT is a player weapon) = (TRUE)

Step 4 - Display a message

Create a new message in Dallas and name it **ShowHitText**. Write something like "Switch is Activated!" in the window below **ChangeNameID**.

Add the following action: Show HUD message **ShowHitText**

In addition to the sound and the ani, something is now also displayed on the HUD.

Note for people who want/need special characters like umlauts: Use ASCII code (e.g. Alt+246 (÷) for an 'ö')!

Step 5 - Use the switch to activate a door Add the following action:
Activate Door MyDoor
Now our switch serves a purpose.

Step 6 - Open the door when the switch is hit
If thatAuto Close-Flag of the door is set, remove this now. Change the previously added Action toSet door MyDoor position to 100.0%

Do you see the difference? Before the door opened/closed, now it just opens.

Step 7 - Lock the door using a script
How can we prevent the door from opening on direct hits so that only the switch can open it? Any ideas? Tip: We have to do something at the eventLevel startmake. Activateactivates the door, which automatically closes again when this happensAuto CloseFlag is set in the doors panel. (Note: do not use a 'blastable' door here.)Set position to 100% means that the door only opens. TheAuto Closemust be checked for this; otherwise it will close again without a new event. Make sure everything works as stated above and make adjustments if necessary. At Level Start we can set the behavior of the door. Add a new script:

Script001: Start
Owner: Level
Event: Level Start
If the following condition is met:
ALWAYS
Then perform the following actions:
LOCK door MyDoor

This should close the door for you via script. Try to open the door directly - it should be locked. Shoot the switch. In addition to sound, animation and the HUD message, the door will open and stay that way. If not, fix the problem before moving forward.

Step 8 - Use a timer to close
Now we will close the door via a timer event. Open thatUser Type Workshop-> TimerID->**Add a new Value**, name himCloseDoor.
Add another script:

Script002: Close Door
Owner: Level
Event Timer (TIMER ID went off) If
the following Condition is met
(CloseDoor) = (TIMER ID)(Accessible via Default comparison statement, Literal: Replace with literal ->enumerated types -> Timer ID; then right click to set CloseDoor and TIMER ID)
Then perform the following actions:
Set door MyDoor to 0.0%

Ok, this script looks for a level timer event calledCloseDoorand then sets up the door animation0.0(=close the door). But before that works, what do we have to do? Exactly, we need to start a level timer first,

Step 9 - Add a level timer
To do this, we add the following action to our well-known Script000:
Generate level timer event in 10 seconds using with ID Close Door.

The door should now open and close again after ten seconds. Does it work for you?

Well, what about multiple hits on the switch with something like Vauss or Laser? Could this jeopardize our idea? Hmm, what idea? Ok, the goal is that when the switch is shot, the door only opens once and then closes again. The next hits shouldn't open the door, you had your chance to get a goodie (or maybe get locked in the second room). Any idea how to achieve this? We will try this via two different methods this time. Does the timer work for you as described? If so, we can move on to limiting an action.

The simple first. Right click on Then perform the following actions, Choose Select max times to execute -> Once.

This does what we want. Where should we set this? Remember, the goal is for the door to only open once.

Second method, with a little more I/O this time, look closely!

Important: Change that Once back up again Run Infinitely!

Create a new message with the text %d Chances left. Name them Tell me. Use that **User Type Workshop** for a new one UserVar to create, name the count. Then add an action Script001 added:

Set user var Count to 5.0

make the following changes in Script000: two actions are added,

Decrement user var Count

Show HUD message TellMe (User Variable Count (Integer))

Ok, that's a little complicated, so I'll try to explain to you what should happen: At the beginning of the level we had a variable 5 preset. When we hit the switch, the script checks to see if it was a player weapon and the variable is greater than 0. If these conditions are met, the script starts the actions we know and decreases the variable (now 4), then shows the text '4 Chances left'. (the %d is a placeholder; %f would display a floating point value. See formatting information for C.)

This is a great way to display counting actions in D3! When our variable reaches zero, the script will show '0 chances left' but any subsequent switch hit will now be ignored by the script (since the condition no longer applies...)

Try it out until it works for you as described. Be careful to only hit the switch once to trigger it, i.e. use flares. Then try to see what happens when you fire the laser at the switch. What problems arise in this case? How can we solve this?



[Back to the overview of Section I](#)

106 - Scripting a Switch II

WillyP

We're left with a question... and a challenge!

Let's polish it all up!

Well, that was the end of the thread. But we still had a problem. When we test fly and fire flares - one after the other - our script works perfectly. But when we fire lasers, we use up all our chances with one quick burst of laser fire. The HUD message says 0 chances and after ten seconds the door is closed. Why is she doing this? Let's look at our script to find the answer.

In theScript000let's do a variety of checks, then we carry out a series of tasks. The script starts and runs every time a collision with a player weapon occurs until `count0` reached. So if we shoot quickly, the script will run five times in quick succession. Therefore, our supply of opportunities is quickly depleted. Is that the desired effect? Not really! We still have some work to do. Let's specify our goal: The door remains open for ten seconds per attempt. In Script000 we can check that the door is closed before allowing the script to run. Right click belowIf the following condition is meton theANDand add a new condition to check that the door position is 0.0%
(=closed):(Door MyDoor position) = (0.0%)

Now go test flying. We see that when we press the switch the door opens. If this is open, nothing happens when we shoot the switch. But if you hit the switch with both lasers at the same time, the script will run twice before the door opens. So we lose one of our attempts. The script runs in two instances, so close in time that before the first instance begins to open the door, the second is already past the point where it checks whether the door has begun to open. So we have to set a condition in the first instance that blocks the execution of the second one. Let's have oneUserFlag add. Go to the**User Types Workshop**and create oneUserFlagcalled**flagit**. In the Script000add the condition(User Flag flagit) = (FALSE). Now it's script000not executed, except **flagit**isFALSE. This new line should be the last one under the AND. Now add a new action under Then perform..., also in Script000. Can you guess what kind of action this is? We must**flagit**here onTRUEset:Set user flag flagit to TRUE. Drag it to the top of the list. Next go to Script002and add underThen perform...added a new action:set user flag flagit to FALSE. Also, I have the linesDecrement user var CountandShow HUD message TellMeafterScript002emotional. And I have the actionShow Timer CloseDoor on HUDin Script000added. This now looks like this:

```
Script000: PlaySound
  Owner: MySwitch
  Event: Collided (With IT)
  If the following condition is met:
    AND
      (IT is a player weapon)=(TRUE)
      (User Variable Count) > (0.0)
      (Door MyDoor position) = (0.0%)
      (user Flag flagit) = (FALSE)
  Then perform following actions:
    Set user flag flagit to TRUE
    Play 2D Sound Bell for player object IT Volume 100%
    Play object MySwitch animation from frame 0 to 20, cycle time = 5.00, looping=FALSE Set
    door MyDoor position to 100%
    Generate level timer event in 10 seconds using with ID CloseDoor
    Show Timer CloseDoor on HUD
```

Script001hasn't changed, andScript002:

Script002: Close Door

Owner: Level

Event: Timer(TIMER ID) Went Off If

the following condition is met:

(CloseDoor) = (TIMER ID)

Then perform the following actions:

Set door MyDoor position to 0.0%

Decrement user var Count

Show HUD message TellMe (User Variable Count (integer)) Set

user flag flagit to FALSE

Check your scripts carefully!

test fly. Our script works perfectly this time. Show Timer shows us the time until the door closes... we wouldn't want to be locked in the second room. I'll leave it to you to offer a reason why!

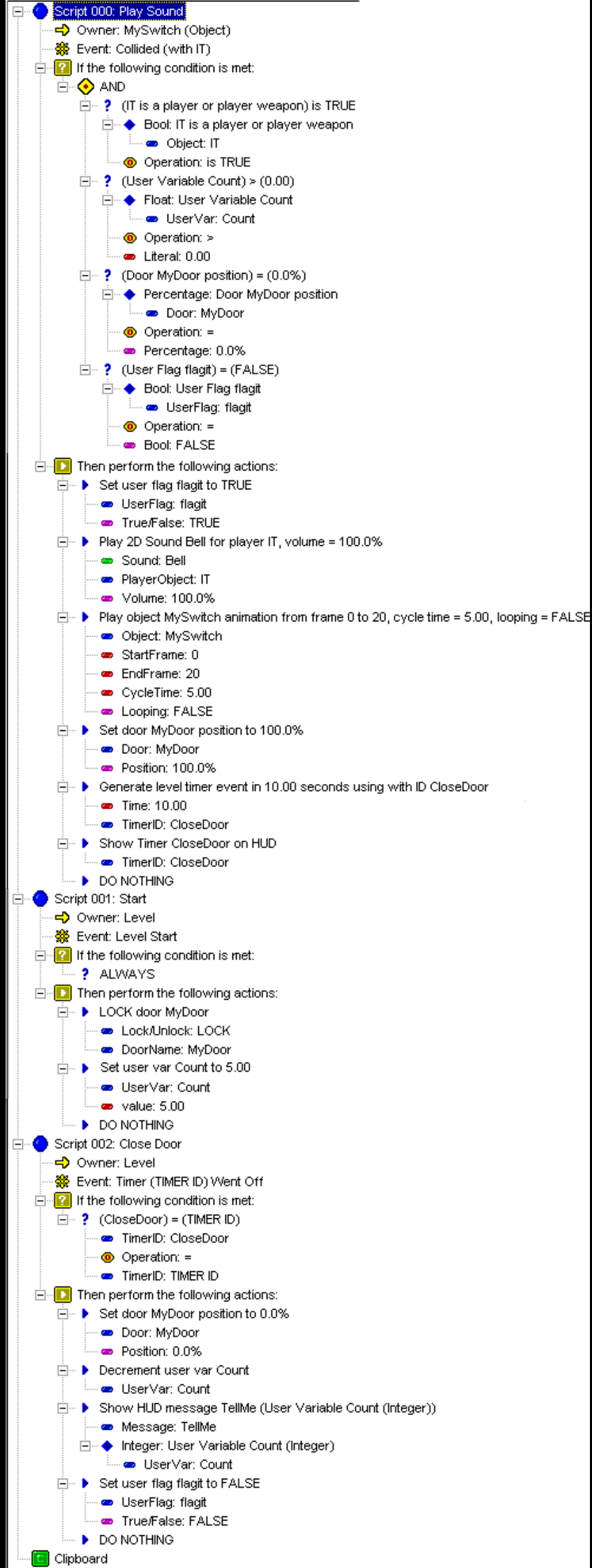
Troubleshooting

If Dallas is working but you are still having problems with your scripts, please go through the previous steps again. Any variation can produce unexpected results. After you get it working, you can go to Dallas and play around with it.

If everything's falling apart...

I suggest that this be the last resort since you won't learn anything by copying a script. But if all else fails to get the script to work, here is a screenshot of the script.

Back to the overview of Section I



107 - Alert!

Ragil Ral

With DALLAS you can create audio events, i.e. play sounds like you saw in the previous tutorial. I came across a peculiarity, so here's a chapter about it.

initial situation

The player has to carry out an action, which is evaluated by a script and, depending on the result, either a door should open or an alarm should go off (in addition to the obligatory Matcenters, which immediately spit out bots). The alarm should sound until the player carries out the correct action, the door opens and then they reach a switch with which they can turn off the alarm.

DALLAS offers these features for sounds:

Sound & Music

- Set music region to [Region] for player [PlayerObject]
- Set music region to [Region] for all players
- Play 2D Sound [Sound] for player [PlayerObject], volume = [Volume]
- Play 2D Sound [Sound] for all players, volume = [Volume]
- Play Sound [Sound] from object [Object], volume = [Volume]
- Play streaming sound [Sound] for all players, volume = [Volume]
- Play streaming sound [SoundName] for all players, volume = [Volume] (TEXT NAME VERSION)
- Play streaming sound [Sound] for player [PlayerObject], volume = [Volume]
- Set volume for object [Object] to [Volume]
- Stop sound for object [Object]

Play Sound from Object and Stop Sound for Object, a sound use the that Looped flag has and Done, very simple - that's what I thought at first.

Now it turned out that the sound was playing but couldn't be turned off; on the contrary, the alarm was triggered again when the corresponding action was carried out.

First try

In this example script the whole thing is put on a switch so you can try it out:



You need a named switch (here I have the switch (->Clutter) 'MatcenSwitches', with a name **alertTeschtSw**) in one named space and a user variable of type Flag (here 'alertTeschtFlag'). The you need to determine the status of the alarm.

Script 038: alertTescht

- Owner: alertTeschtSw (Object)
- Event: Collided (with IT)
- If the following condition is met:
 - (IT is a player or player weapon) is TRUE
- Then perform the following actions:
 - If the following condition is met:
 - (User Flag alertTeschtFlag) is TRUE
 - Then perform the following actions:
 - Stop sound for object alertTeschtSw
 - Set texture on room afterPin face 21 to P-NovStripeMin
 - Set user flag alertTeschtFlag to FALSE
 - Play object OWNER animation from frame 20 to 40, cycle time = 0.70, looping
 - Else, perform these actions:
 - Set user flag alertTeschtFlag to TRUE
 - Set texture on room afterPin face 21 to P-NovStripeMax
 - Play Sound SirenLooped from object alertTeschtSw, volume = 100.0%
 - Play object OWNER animation from frame 0 to 20, cycle time = 0.70, looping

What should happen? TheOwner(=the named switch) is triggered by the eventCollided (with IT) activated. TheIT is of course the player or his weapon. In theThen perform the following actions:- We first ask Block about the alarm status and carry out the appropriate actions accordingly. So here, when the switch is triggered and the alarm is on (alertTeschtFlag isTRUE) it is switched off and its status is set to off (alertTeschtFlagafterFALSE) and the sound stopped. Conversely, if the alarm is off, it should be turned on with the appropriate actions.

Make sure you are using a sound that has the looped and listener update flags set.

The Play Object OWNER animation...are used to animate the switch.

The actions for Set texture on room...are to check whether the script branches correctly when it evaluates the query for the flag (= status of the alarm).

Build an example level, put the script there and shoot the switch a few times. But turn down the volume first...

What happened? The script has properly turned on the sound and set the texture, but the sound won't and won't stop

Atan explained to me that a sound can be started with the looped flag, but cannot be turned off. His solution is to generate a timer event and use it to play a sound as long as a certain condition is met; The sound to be played no longer needs the looped flag, or may not have it at all.

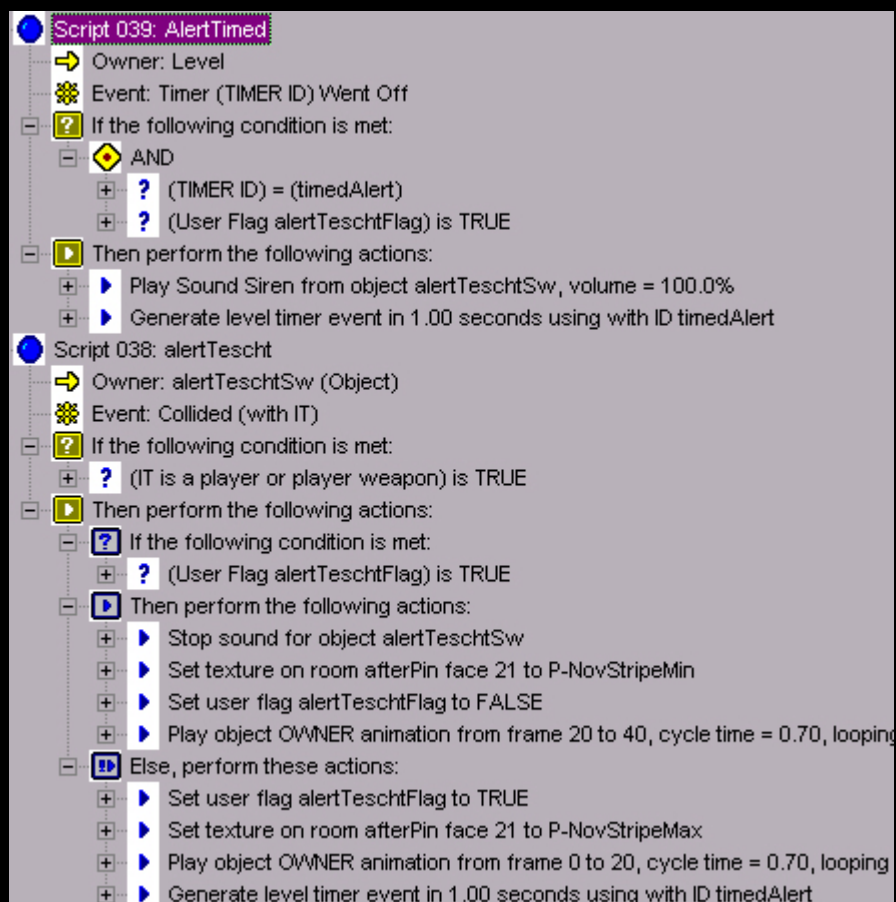
Timed alarm

How to realize it? First you also need a uservar of type TimerID (here `timedAlert`). This timer should play the sound when it expires. In order for it to do this repeatedly, the timer must be called again and again. So move the Play Sound [Sound] from Object [Object], volume = [Volume] -Action into a script that is controlled by a timer and calls this timer again in this script (script 039 in the screenshot). A function that calls itself is called 'recursive', and anyone who has been paying attention can see the problem: There must be a way to somehow end the script execution, otherwise it will call itself infinitely often. In order to have such an option to cancel, you now also have to ask for the alarm status, so that the sound is only played again and again if the alarm status is 'on', i.e. `alertTeschtFlag` is TRUE. If it is 'off', the timer will no longer be triggered.

In script 038 you only have to use the function for Play Sound [Sound] for Object [Object], volume = [percentage] replace with a call to the timer so that the timer-controlled function is triggered for the first time.

In this example, the alarm is triggered by a switch. The actual trigger is the timer, which is triggered by pressing the switch. It's up to your imagination what triggers an alarm, but there are undoubtedly suitable applications for such things, and DALLAS provides enough events to trigger them.

This script construction also offers the advantage of being able to use the alarm again and again, since only the timer (here `timedAlert`) must be started and the status flag must be set; you could then use the sound Play 2D Sound for [Player]...or Play 2D Sound for everyone Players...play so that it can be heard throughout the level. Furthermore, you no longer need to stop the sound, just turn on the alarm status flag FALSE sit and rest.



But it's also cheaper... Dark has found out that the flag is enough for the soundObject update to set, then it will work with that too. Stop Sound for Obj... from the first attempt because only the attempt makes a difference...

Well then hit the keys!

[Back to the overview of Section I](#)

108 - Scripting an inventory cloak <>

Starkiller

I've included the entire level for you. If you want to add more information or save yourself the trouble of building the objects yourself, extract them from the example level.

This script was not for this site (http://gameedit.net/custom/GameEdit_OG/index.html), but for Skylmian, for his project "Lunar Outpost MN0012" when he needed a cloaking device for the ship, like in that Novel "Descent" (although the Dallas screenies are from his script for his level "Space Station Kreathlan", where he introduced the use of the Cloaking Device when it failed to make it into "Lunar Outpost MN0012"). This script is very advanced and not in the least for beginners, as not everything is explained (assuming you can handle Dallas completely). Skylmian had a difficult time trying to duplicate this in Dallas when it was handed to him.

This so-called tutorial will show you what should be done to create a cloaking device that can be used by any ship and will generally explain what each script does. In total it should take between 30 minutes and an hour, depending on how comfortable you are with Dallas.

01 - User Types

The first thing you should do when the Dallas Graphical Script Editor is open, open the User Types Workshop and create the following user types:

```
SavedObjectSlot:Cloaked_Handle
TimerID:Cloak_Energy_Drain
UserFlag:Cloaked_Flag_1
UserFlag:Cloaked_Flag_2
UserVar:Useless
```

The Cloaked_Handle is the identification of the player ship that has the cloak, the Cloak_Energy_Drain is for the timer used to drain the ship's energy for the cloaking device, the Cloaked_Flag_# are essential for the cloaking states, and Useless is only necessary for to insert a space for the first script.

02 - News

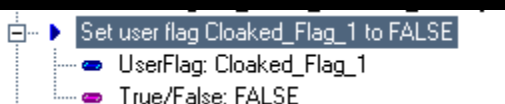
Generate these messages in the Message List (Surname->Text string):

```
Cloaked_No_Energy_Left->Not enough energy to use the Cloaking Device.
Cloaking_Device->Cloaking device
```

03 - The powerup

Create a powerup in the level called Cloaking_Device and place it close to, if not in, the first player starting point. Choose a good, solid object as players will see the model when the player who has the cloaking device is killed. Skylmian has chosen the data cartridge model for its version.

04 - Level start script



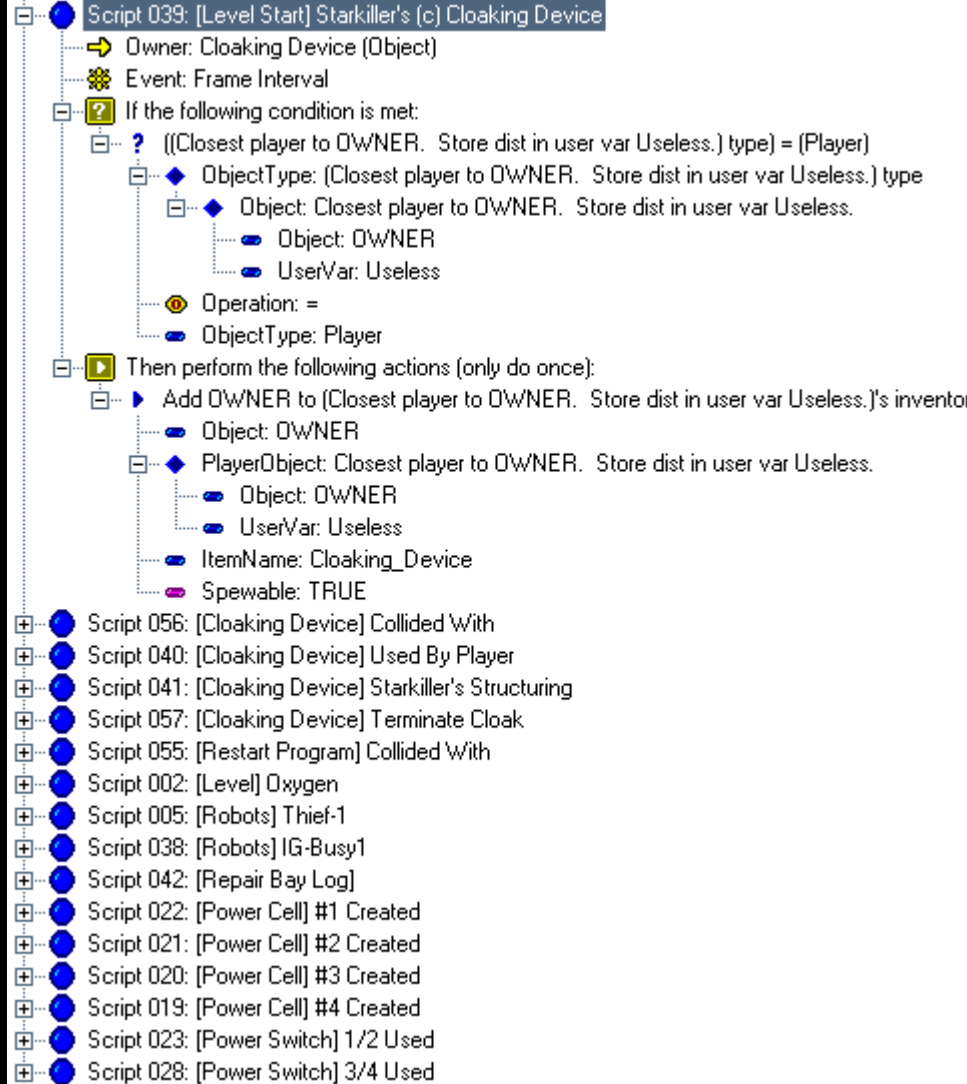
Create a script that will be executed at level start. Put in it theUserFlagCloaked_Flag_1onFALSE.

05 - First script

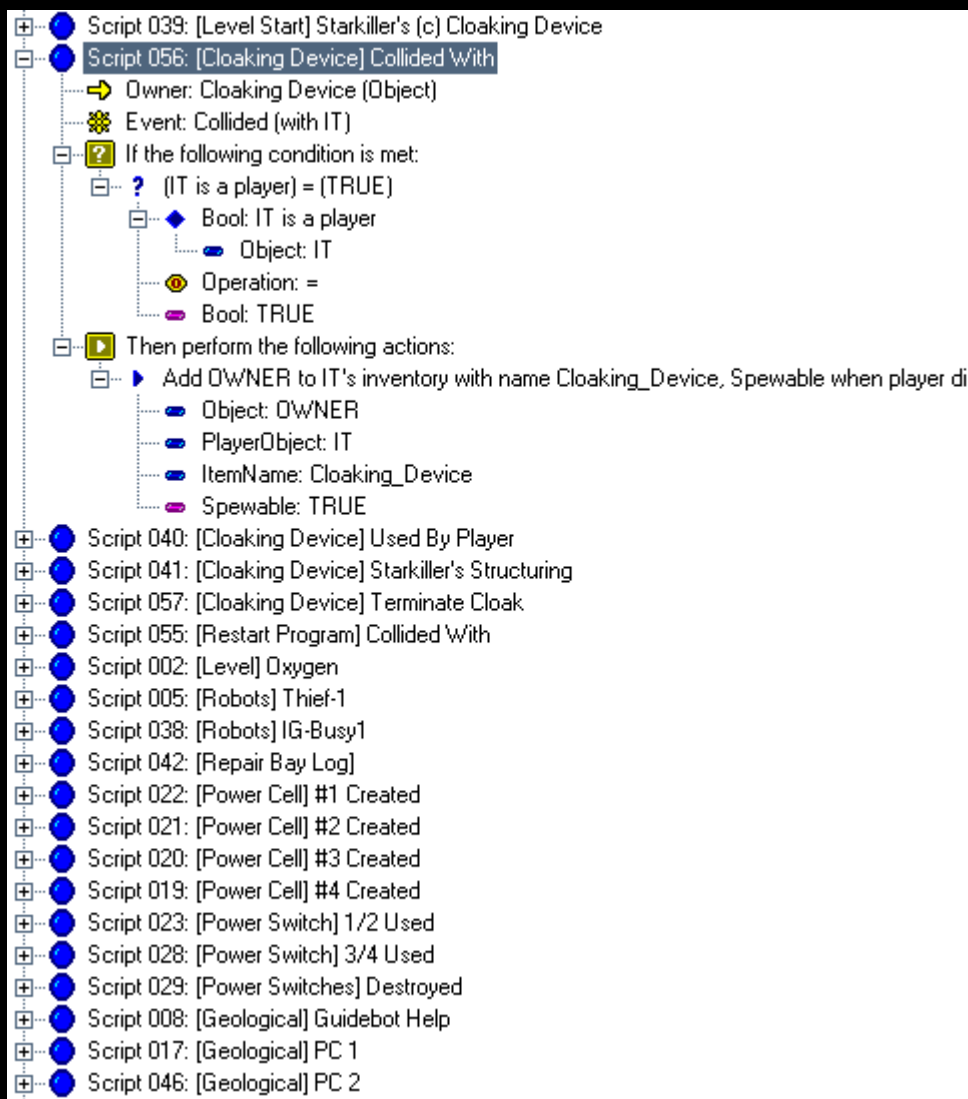
This script automatically names the object Cloaking device into the inventory of the player closest to him and [-uff- strange sentence... causes the object to be lost if the player dies while it is spewable]

Note: If you want, you can skip this step so that the player can see the object in the level can find/collect (in this case delete the UserVar Useless as it will not be needed).

Use this script to do this (Script039):



06 - Second script

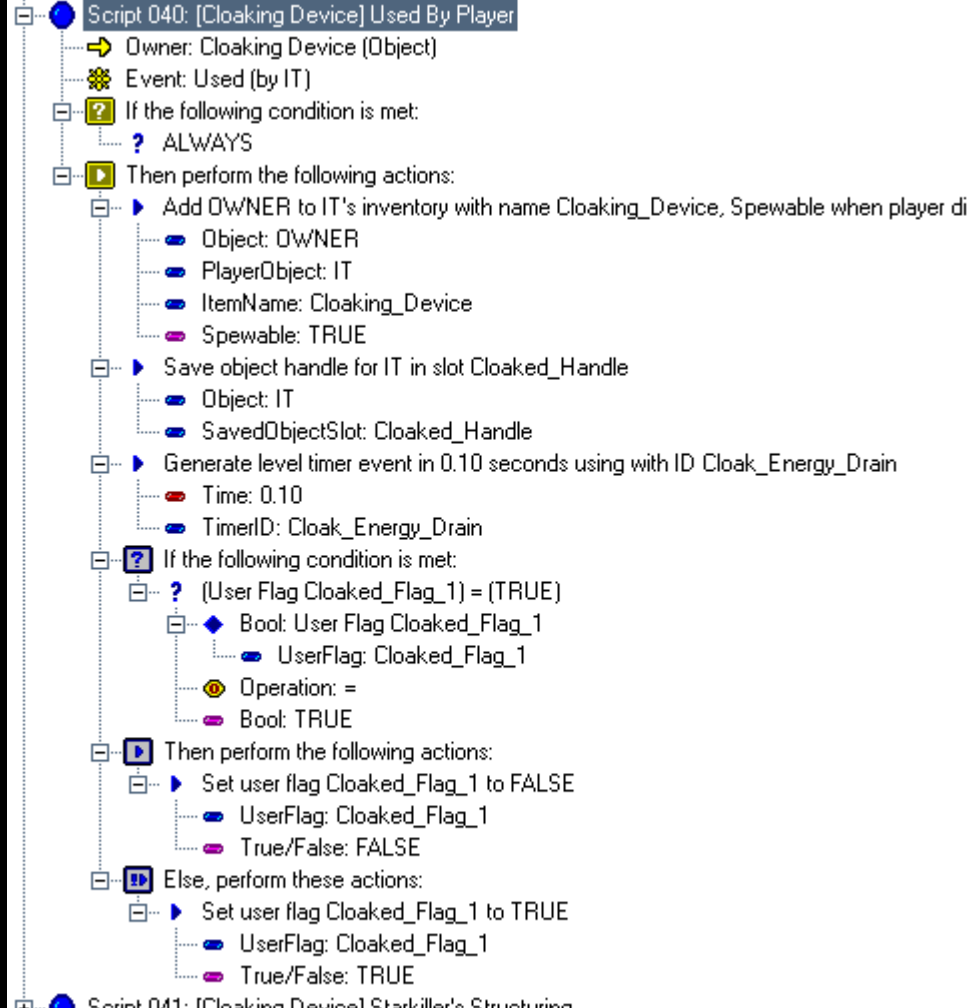


This script adds the Cloaking Device to the inventory of any player ship that touches it, with the attribute to be spat out upon player death. (duplicate Script056)

07 - Third script

Use for thisScript040. This script is for when the player activates the cloaking device. When executed, it adds the device back to the inventory, leaves it lost (?????) gives the player ship the

Cloaked_Handle-Identification, starts theEnergy_Drain-Timer since the device is now on and sets theCloaked_Flag_#'s the one for the Operation of the device is necessary.

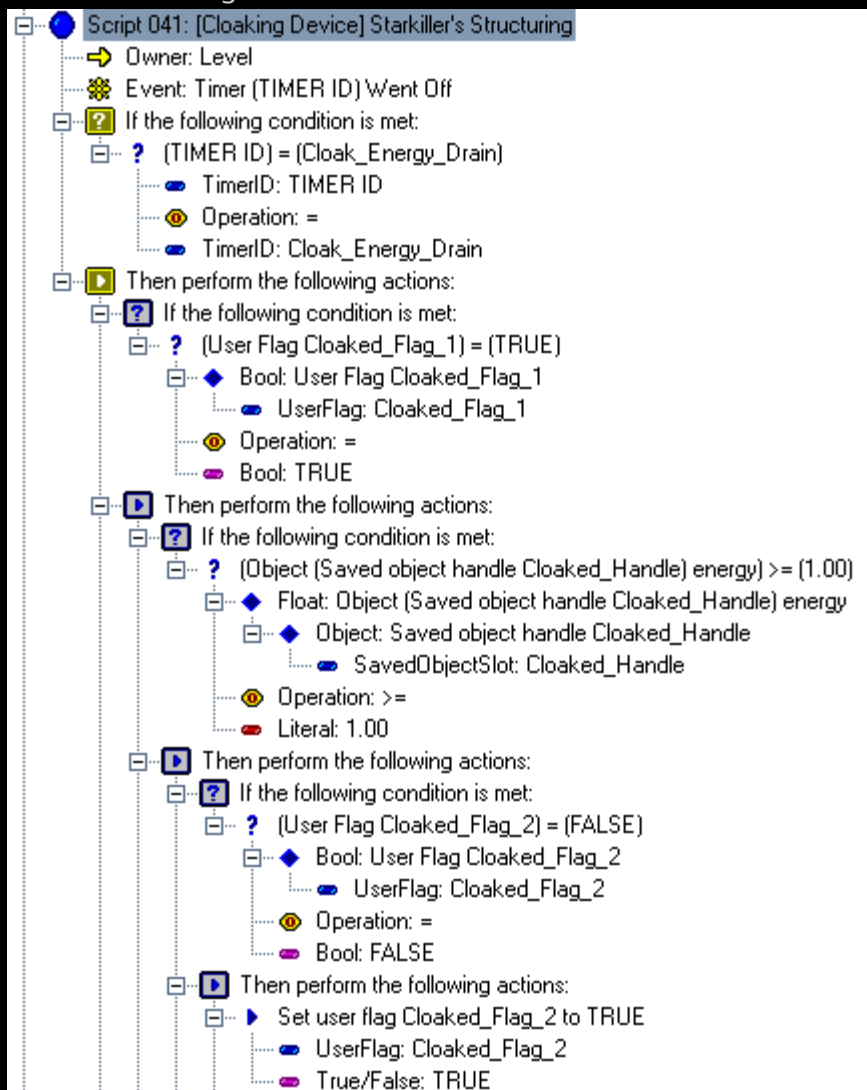


08 - Fourth script

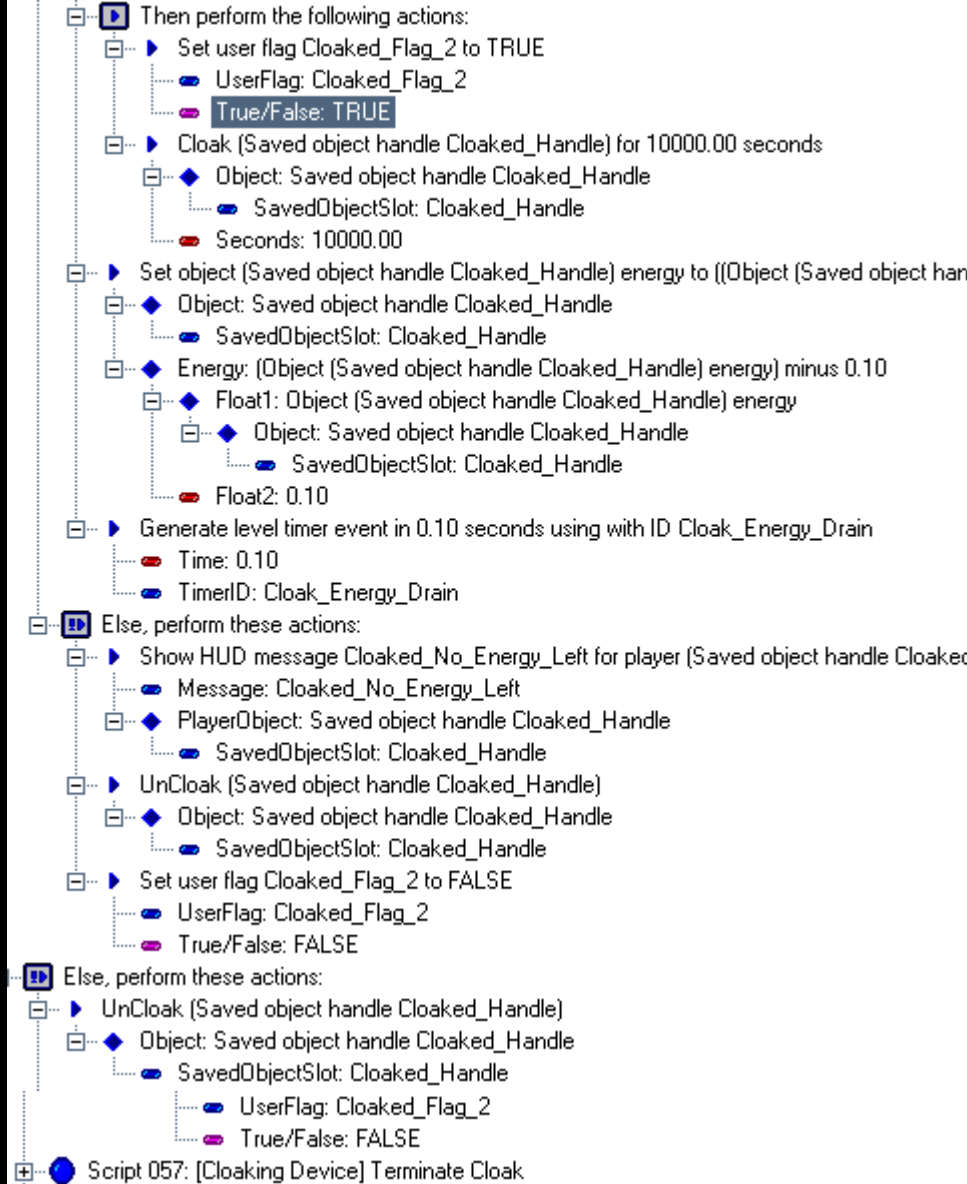
This is the most difficult of all as it makes the cloaking device work. It is divided into several

images because it is not on a single screenshot fits. In the first image the script title is highlighted. In the In the following images, the line that appeared last in the previous image is highlighted so that you can keep track. For example, in the second part of the picture is the Line "True/False: TRUE" highlighted because it was the last one in the first panel. Also, in the second panel, the window is scrolled slightly to the right to show what one of the longer lines in the script does, the one that drains 0.10 energy from the player ship. Although the first main line of the text is truncated (the top expandable tree line) it is still possible to know what it says after analyzing the script lines for it.

(The entire script image can be found in the .zip)



Second partial image:



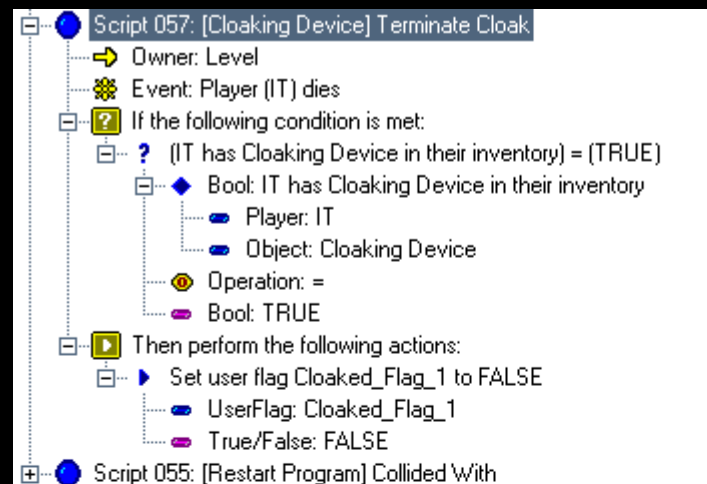
Third partial image:

09 - Final script

While the most difficult part has now hopefully been taken care of, there is still one more thing left. Replicate Script057; this switches off the cloaking device if the player dies.

10 - Done!

Six scripts in total, including the level start script, which is the important one. One thing you can do to refine the script is the value `Float: 0.10` to vary in the second part of the eighth step. The higher the value, the faster your ship will lose energy. If you set it to 0 no energy will be drained, but that would render the timer pointless. If you know how, you can extend the timer from this script so that the cloaking device simply switches on/off without any loss of energy or unnecessary script code. Although, hey, no one would ever turn the stupid thing off...



► CLOAKING DEVICE

I have modified the script so that the 'wave' effect no longer occurs. Versu starts when there is not enough energy!

[Back to the overview of Section I](#)

109 - Reactor Gamma: Script Companion

Kyouryuu

The accompanying guide to `gamma.mn3` file containing Reactor Gamma in the `mn3` is enclosed

Extract the file(s) from 'gamma.mn3' so that you can follow this tutorial. I won't provide them here as I will respect Kyouryuu's work; I also don't have any screenshots of it here. If necessary, get the level from one of the relevant level sites. Load the level into D3Edit and open the script file with Dallas. Don't forget to set the table file first!

Intro

When I was scripting my Reactor Gamma level for Descent 3 a while ago, I was sometimes a bit lost when it came to tutorials on the subject. Although much of Dallas runs logically, there are a few obscure elements that any level designer is likely to come across. That's why I have `gamma.cpp` file that can be used by Dallas in the `mn3` leave. I did the same thing in an earlier SP level, Centroid Military Base, and it came in handy when I had to use Dallas again for Reactor Gamma a year later. This guide for intermediate to very advanced users of Dallas will walk you through `gamma.cpp`-Run script. Stand up `gamma.cpp`, `-msg`, `-dll` and `-oin` in your Osiris subdirectory of your SDK installation.

Intro cinematic

Script 000: Level Start FF AND REACDOOR TOGGLE!

Perhaps the largest and most varied script in the set, it is executed at level start. First, it turns on all Spewers in the level and causes them to produce gray smoke. Note that Invisible Powerups, which for `OutdoorSpew1`, `OutdoorSpew2`, `OutdoorSpew3`, `OutdoorSpewTall` and many others use the `gunpoint-1` must be set to function correctly. Others, like `ShaftSpewerX` and `ReacSpewerX` work with `Gunpoint0` set. Also note that the `ReacSpewers` the physics flag `Gravity` have set. This causes them to move in an arc 'down' as they fly through the air. `Script000` also activates a variety of force fields. Note that only one of these is actually a force field - the one to the lower corridor, which is the portal 3 of the Central Junction. The others actually 'activate' glass when `Upper Core` and the `Tower-Portals`. The doors to the generator rooms are locked and many variables are involved `doorFALSE` initialized. The pumps are set to animate from frames one to three, and many objects - mostly invisible powerups for the spears - are hidden. Hiding an object makes it completely invisible and flyable. The `FireGadgets` that appear in the final cutscene are ghosted just like that `HiddenLaser` (A bot). And `ReactorGamma` gets his shields up `10000.00` set. The `Thiefbot` will also appear `1000.00` Pimped up sign. I turn on the AI for the `spyhuntersSpy1` and `Spy2` off, just like for them `CorSwatters`. More on that later. I put that too `EntryDoor` on open for the following cinematic. Rain and thunder are also activated. You also see an unusual command for `ENABLE forcefield` at portal 0 in room 2 `WithDoublesidedFlag=FALSE`. The reason for this will also be in the Intro cutscene script discussed.

Script 017: Intro Cinematic 1

This is also triggered at the start of the level. I set the Music Region for all players and tell the CED Asagiri to move along her path waypoints for the introductory cinematography. I start a simple letterbox format cinematography using the CED Asagiri as the focus point for the camera. You don't have to align the path nodes because the camera is told to always look at the Asagiri. I also generate a level timer with the intro voice. After three seconds the next event is triggered.

Script 048: intro cinematic voice

The trigger for this script is inIntro Cinematic 1. The Introvoice timer will run this event three seconds later after Intro Cinematic 1 runs. All this event does is play the sound 2d sound Gamma intro, which is the opening narration for Reactor Gamma.

Script018: Intro Cinematic 2

This script involves a bit of trickery. Its purpose is to execute the second cutscene immediately following the end of the first. To achieve this is theOwneron the camera's targetIntro Cinematic 1set what the CED Asagiri was. The event trigger is AI-Movie Ended. So when the movie involving IBD Asagiri ends, this event begins.

[Back to the overview of Section I](#)

Section J–Designing

These are not tutorials in the strictest sense, but rather philosophical design considerations for the different game modes in order to best serve the respective game variant.

110	Level Design Rules		Dark Wolf	388
111	Single player level building		Hydra	391
112	Single player		Robot	393
113	Anarchy		Hydra	395
114	Anarchy		Robot	397
115	Team games		Hydra	399
116	When Great Levels Die		Schplurg	401
117	Eye candy		Robot	408
118	Weapon balance		Robot	411
119	Why build missions?		Robot	413

[Toc](#)

110 - Level Design Rules

Dark Wolf

I built a lot of levels for Descent, Duke3d and also a few for Shadow Warrior as well as a few quests for Zelda Classic. And I've learned a few key points in level design, and I thought it would be helpful to share some of these points with aspiring level designers who might take them seriously and use them. These are design points that I came up with myself and that worked best for me. They're aimed at level design for Descent, but they can be applied to any game out there. Have fun.



The main theme:

This is the most fundamental rule in level design, the very first thing before you even start putting pencil to paper or fingering the mouse, think of a theme for your level. Is it supposed to be an alien world? A lush green mine with lots of waterfalls and bodies of water or a fiery inferno with lots of lava? What colors and textures will you want to use? It's best to start with a central level theme and build all your thoughts around it. Your level will most likely get much better as a result.



Frame rate:

Although this isn't a big deal for most level designers out there due to the ever-increasing number of 'super' computers on the market, I decided to write something about it anyway. It used to be an important thing in Descent that it was best if the player didn't see more than 40 cubes on the screen at the same time. The same goes for Descent 3, try not to have more than a few rooms in direct line of sight or the game will noticeably slow down, especially when playing online. Try using walls or obstacles, or even doors, to block out the invisible boundaries between cubes or rooms that cause the frame rate to drop. A better frame rate means a more playable level, meaning it's more likely to be played more.



Continuity:

This is directly related to the main theme and has almost as much influence on the level as the main theme itself. Several times I've seen levels that started with a single theme and immediately transported you somewhere else. Having a Seoul-esque city contrasted with volcanic mines isn't going to help create a cohesive level. Your level must seem like a believable place, and it won't be fragmented if there's a volcano in the center of a Korean city. Well, this will work if you're going for the twisted and extreme, but for most level design themes it won't work and will alienate the player. Decide on a set of textures and structures you want to use and stick to them throughout the level.



Details and realism:

The player is notoriously eager to catch small details in the level, although you are limited in what you can do in Descent 1 and 2, Descent 3 gives you immense freedom to work with details in your level. If you think a particular area is bare, add more details. The two basic details are shading and matching textures. Once these are applied and the room/level is still bare, try inserting objects and/or hand-created structures into the level to improve the realism. In my level Chocobo Word, for example, I included wheelbarrows, hoes and haystacks as details in the barn.



Carrot and stick:

A good level should have a series of challenges and rewards. The challenge can be before or after a reward. I often saw levels with goodies and baddies scattered around without any rhythm or reason. Make sure your challenges and rewards are related and you will most likely achieve better player satisfaction. For example, a few Class 1 Heavy Drillers guarding a Smart Missile or two. Or Thresher guarding one of the extremely valuable Omega Cannons. If possible, try to relate the goodies and objects you use in your level to the main theme.



Choose your audience and build for them:

You have to decide whether you are building for a single player or multiplayer audience. This will dictate the flow of how you build the level. A good single player level tends to be large and contain lots of areas to explore. You want your single-player audience to have a lot of fun searching for and chasing wildly fighting hordes of bots or solving tricky puzzles. A good multiplayer level, on the other hand, must be more open and significantly smaller, with a design where one player can get to all locations equally quickly and find the other player within a minute or two at most. Sometimes you can cater to both types of audiences with one level, denying single player access to areas with keys or scripted items, and areas that can be opened with multiplayer blowouts or scripts.



Player interaction:

This is a very important factor in many levels where Descent is a bit lacking, but you can still incorporate interaction into your levels, you just have to be innovative. Also try to use as many things as possible that the player can change or destroy. Things like lights that you can shoot out, boxes and barrels that can explode and switch panels that you can operate. Also make as many secret areas as possible, forcing the player to look at their surroundings and be more interactive with their environment.



A well balanced level:

Make sure your weapon and robot levels are balanced. In single player there should be just enough ammo and shield to compensate for what the bots take away from you. Too few goodies means the level cannot be completed and too many means the level is child's play. You also have to think about the different difficulty levels in the game, where four Fusion Hulks seem pretty easy in trainee mode but are a deadly barricade in Insane. Don't forget that your audience won't be familiar with your level and therefore won't know what to expect, so pack in 'a little more' goodies than you would need. A good multiplayer balance would be for the goodies you place throughout the level to encourage the player to follow your 'intended' path through the level.



Push the limits:

In contrast to realism and FPS, it's always good to push the limits sometimes and see the limitations of the engine and what it can achieve. Despite everything, the game is only a virtual reality and you can amaze the player with amazing rooms and views in the game.



Pattern:

Players, whether they know it or not, love patterns. They help the player to solve the entire level. If there is a switch on one side of the level, there will probably be a corresponding switch on the other side. If you destroy one reactor, another one in a similar-looking region will have to be destroyed. Also think in terms of timing when placing enemies in certain areas - old scratch, pest, plague, old scratch, plague, plague, etc. Don't be too monotonous, mix it up a bit to make the player feel a little uncomfortable.



Links:

This is doubly important if you want to use the 'best' secrets within the level. If you can, give the player a view through windows or grilles of areas they may be able to reach and give them access to those areas later. It's also satisfying to see a powerup behind a gate and ultimately find your way to the other side and grab the goodie. There's something wonderfully enjoyable about seeing something early in the level and gaining access to it later.



Puzzle:

This is where most of the levels in the Descent 3 world fall short. There is a time and place for a guided missile puzzle, but that is NOT a required level objective! Always make guided puzzles optional and only for the purpose of gaining access to secret areas. It's OK to have a few panel puzzles in the level, but keep them for the majority of the area in the side objectives. Descent is about flying around and destroying bots, puzzles should only be secondary. If you absolutely must have a puzzle with Guideds, place a few Guided packets right in front of the puzzle to direct the player toward the solution. Likewise, in Descent 3 it's best if this is a recurring rocket pack.



'The Fiddler's Roof':

Don't constantly force the player to take a linear approach to the level. Your level shouldn't just have 'one way' to be solved. Although harder to do in Descent 3 than in D1 or D2, it is best to give the player a choice of multiple routes to reach a specific goal. Sometimes let players go where they aren't supposed to go and greet them with enemies and threats they aren't yet equipped to deal with.



'Practice What You Preach':

Always test your level in the game! Make sure you've checked everything possible at your level, you should be absolutely sick and exhausted of your level by the time you're done testing and ready to move on to the next venture. Regardless of whether it's for single or multiplayer, test it under exactly these conditions. Follow these design guidelines and little can go wrong.

[Back to the overview of Section J](#)

111 - Single Player Level Building

Hydra

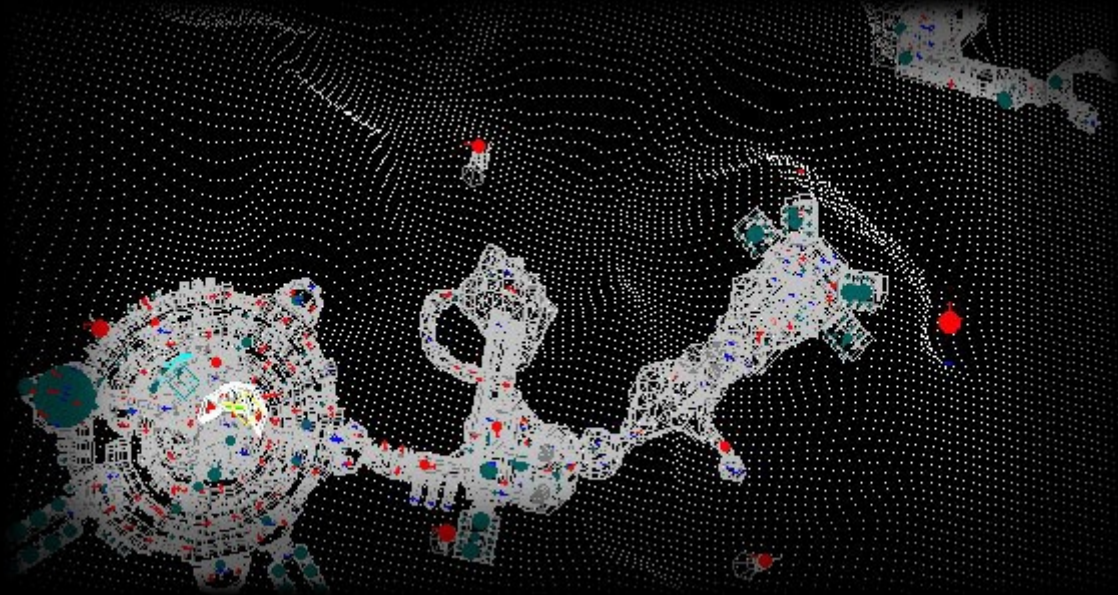


Image taken from level 2 of the level set "Descent 3: Retribution".

Single player levels are the hardest to build. You have to put a lot of thought into the level structure, much more than any multiplayer level. SP levels are also much larger on average, as most multiplayer levels are small enough to breeze through in around 2 minutes. There is also a lot more emphasis on scripting. While scripting doesn't have much effect in MP, in SP levels it is the 'color' of the levels themselves that makes the levels more interesting.

A long, long time ago I heard a phrase about building single-player levels for any game: 'Make the level like you hate the players'. Although I don't completely agree with this, I think it makes a good point. If you want an action-packed level, you can't make the game slack by making it relatively easy to pass with one or two lives. This balance is difficult to achieve, it takes a lot of playtesting to make sure it's relatively easy as a trainee, but not too impossible as an Insane. This is where other people (β-testers) come into play, as they most likely have different descent skills than you. Plus, when you play a level that you built, you know where all the secrets and traps are, which makes the level a little easier.

Architecture is much more important in SP levels than in any other game mode. In multiplayer, you're always too busy running from or chasing someone to notice the wonderful 'wall effects', which really don't have much use in the game other than adding to the atmosphere. On the other hand, in single player, you could go through the same room multiple times depending on the level layout, and I (for me) always like to stop and admire the scenery after 'cleaning' a room. A final important tip for room creation for single players is to use the interior space. Having a large room with bare walls can look cheap and thrown together unless it fits the atmosphere of the level. If you find a room that looks too bare, use Extrude or Lathe to add a few things to hide in. Just remember not to crowd him, the bots won't run into the walls as often as the players.

Level layout is another essential thing when building. Although it is easy, and sometimes necessary, to create a 'one-way' level where you can only progress in one direction, it is not always as fun or exciting. The one-way levels seem to be more of a gauntlet of rooms filled with bots are until you reach the end where you have a boss/reactor/etc. destroy. When you do levels in a different way, you make players fly all over the level looking for a key, with traps and matcens going off and spawning more bots. Another big thing is symmetry. Unlike others, I think that symmetry in levels is good as long as it is not exact. If you're building it symmetrically, you'll want to add certain things that make each side unique in its own way, like a shattered pillar on one side while the others are solid. When you make a level, try not to give it that 'copy-paste look', that can get boring.

Robot and weapon placement is a must for traps. Place a weak bot or strong weapon in the middle of a room and a few foolish players will rush in and destroy/capture it. Then all the bots hidden in the shadows rush up and attack the player, or you can let the Matcens in the room jump at you. The amount of weapons in the level should directly correlate with the amount and difficulty of bots. It's hard to beat a boss when all the level provides is a Vulcan and some Ammoclips. On the other hand, it's too easy to destroy a horde of weak bots with a Blackshark and Omega. Much like the difficulty level of the level, it is very difficult to achieve the right balance between weapons and bots. This is where the beta testers come into play again, everyone has different skills in Descent.

Well, I think I've covered all the important things about creating a good SP level. All it takes is creativity, time and a whole lot of patience. If I think I've forgotten something here, I'm sure I'll add another paragraph or two.

[Back to the overview of Section J](#)

112 - Single player

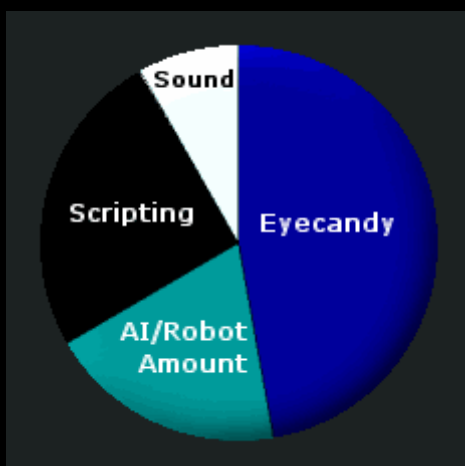
Robot

Single player is the basis of all games. Basically you are on your own. In Descent 3 this means destroying robots, completing objectives and escaping the mine. Single-player levels are by far the hardest to build. This tut might make it **easier** for you...

BlueFlames really took the words out of my mouth (or the gamepad you might say) with their soupe du jour on atmosphere. I quote:

InSingle playera fair amount of planning has to go into what a level will look like,**if it is given that there is a backstory to the level's existence. Mines need deep shafts and rocky surfaces, command centers need to appear efficient and designed around the site's operators, and so on and so forth. The level must look so that it seems functional in some way (or not, depending on the backstory), or the player will lose the atmosphere.**This is obviously not the top priority in a multiplayer level. File size, game balance and overall efficiency are clearly the most necessary components of a multiplayer level, but making it seem like it has a 'purpose' shouldn't be completely forgotten. It really sucks when you have to fly around in a quiet moment in a level that looks like... well... corridors with randomly applied textures on every wall.

And how right he is. Unlike most multiplayer levels, the opinions, gameplay and feel of the level itself are directly influenced by the atmosphere and structure of the level. Even now, after 17 (!) replays of the single player missions included in D3, I still look at the levels and say "Wow...". This is every level designer's dream and goal, or at least it should be. Without atmosphere your level is, how shall I put it... dead.



So what exactly is atmosphere? Well, you could say it's the amount of features and characteristics of your level. Look at this pie chart. I would estimate that 45% of the atmosphere depends on the eye candy and geometry in your level. 20% from the amount of robots, 25% from the scripting and 10% from the sound.

I deliberately gave Eyecandy such a high proportion because it was the biggest deciding factor^{is}, in every level. Like me in mine[Tutorial](#)via Eyecandy, without it your level will be blank walls and not interesting. You have to have eye candy! 😊

Unlike in multiplayer, you're not flying around in a total frenzy; You'll probably take your time solving all the puzzles and roasting the bots. As you fly around you will also have time to look around, as everyone needs to know where to go - and everyone needs 'landmarks' to determine their position. There's probably so much I could say, I could go on for four pages - but I won't. I'll just leave you with the important points:

- Make sure you build very attractive eyecandy, [this tut] might help.
- Try to balance the atmosphere of your level, not too much sound or scripting, but not too little.
- Your level, if done correctly, should appeal to all types of D3 pilots - aim for that.
- Many people take their time when going through single player levels, you should do the same when creating them.
- Make sure you publish the level to some good archives, like PlanetDescent, GameEdit.com or Levels4You.com (now only with registration)!
- If this information helped you in any way - send me a message and I will give you a shout out 😊

Seems a little short, I may have forgotten a few things, I'll add them as I remember
Good luck with the levels! 😊

[Back to the overview of Section J](#)

113 - Anarchy

Hydra

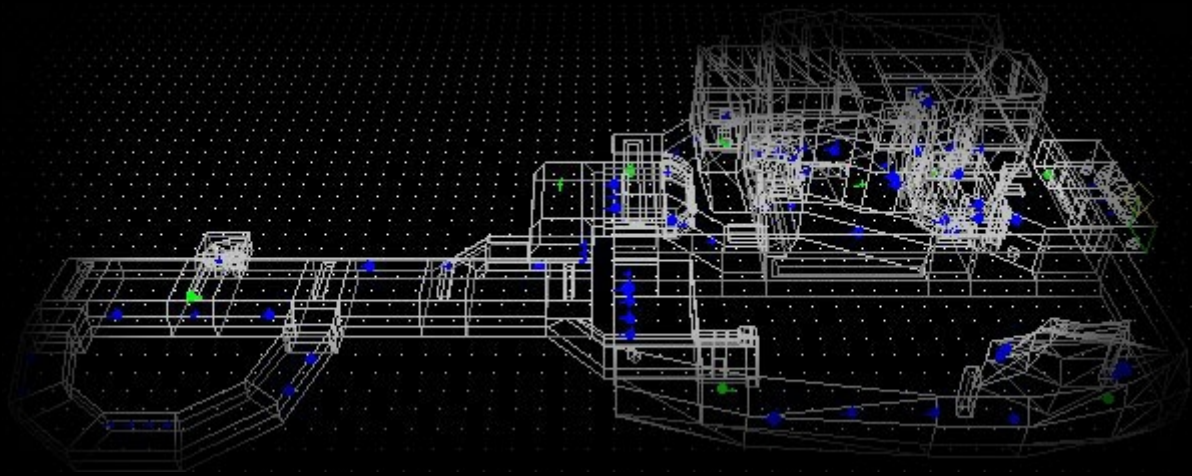


Image taken from the level "Ask Tag".

Anarchy is possibly the most played mode in Descent3. Anarchy may also be the most diverse level type there is. Some large anarchy levels can consist of one or two large rooms, while others are good with honeycombs or tunnels. Each has its strong and weak points for gameplay. Scripting isn't a big deal in Anarchy levels, as most players are too busy either running or chasing someone to notice nice scripting or elaborate structural design. In Anarchy, the focus is on the playability of the level rather than how complex the level design is.

Ambience is a big thing in Anarchy Levels. Not just the sound atmosphere, but the additional effects like fog. These give the player more of the feeling of really being in the Pyro without reducing the frame rate or lagging. I recommend using ambience as needed, I wouldn't dress up every room with a sound or mist colors, it would detract from the gameplay. An atmospheric sound for the entire level is a nice touch, especially if there is a specific theme or motif in your level. Applying mist is always a good thing if used moderately. Don't forget that you have to see your opponent before you shoot him. Shooting blindly sometimes gets you a kill, but drains your energy much quicker.

When you're in a firefight, the last thing you want to do is ram into wall bulges. As you level, look out for things in your level that interrupt the 'flow' of the level. All free-standing obstacles in the level should be a good distance of two or three ship lengths apart (around 50-70 units) to make navigation between them easy. Of course, sometimes having a narrow corridor or passageway that really challenges your skills in a hunt can be really pulse-pounding.

Placement is another key point when creating an Anarchy level. Mainly the placement of energy stations. The amount of stations should depend on one thing, the average amount of players that level will accommodate. When placing energy stations, pay attention to the amount of energy weapons, the amount of players it will contain, and the size of your level. In my experience, most players will not share an energy center if they are aware of it. If the level is designed for 1vs1, one energy center should be sufficient. Otherwise you should have at least two.

Weapon balance should be planned very carefully in your level depending on your level structure. In open levels, weapons like EMD, Homers, MD and Omega are very effective and should be placed sparsely unless you want a high kill level. When creating tight, close-range levels, weapons like Fusion, Frags, Vauss, and Napalm will wreak havoc when used correctly. The amount of weapons can make or break a level. Too few will result in one or two players who have picked up a bunch of weapons simply slaughtering all the other unarmed players. While too many heavy weapons just kill everyone all the time.

[Back to the overview of Section J](#)

114 - Anarchy

Robot

Anarchy is the game mode where you are on your own. You collect weapons, kill other people and avoid being killed yourself. This leads to many tactics such as doubling back, out-turning and numerous weapon tactics. But the trick to getting people to play Anarchy on your level is to map out the level first.

Tactics are everything in an anarchy game. Without tactics you stay in newbie mode, and you will be stuck with 'newbie' for many games to come if you don't get any kills :)

If you don't act lethally, quickly and efficiently, it's fate that the poor guy down there will fall to you. Doubt at your own risk!



But you can't fully use your tactics in a level that suits you

- sometimes in more than one way. Have you ever had the experience of holding up fairly well in a level and then after you get to a level you...

Played for the first time and found yourself stranded with confusing thoughts?

For example:

- Where am I?
- Where is this energy center?
- What, that gun AGAIN?
- Stupid pipe, get out of my way! Argh!
- Doh! Where am I flying to now?
- What the hell was that!?!?

Thought #1: Where am I?

The problem - confusing tunnels. I've played levels where it's almost impossible to tell where you are, and trust me, that thought echoes through my head. These levels are like mazes. There are many solutions to this problem, and you can try changing your rooms or adding colorful lights. What people need in a complex level is a reference point to tell where they are.

Thought #2: Where is this energy center?

As I said in my Energy Center tutorial, these are an essential part of a level. And that means people must be able to find it without much difficulty (not always the case, hehe). It should be easy to locate and well placed tactically.

Thought #3: What, that gun AGAIN?

Weapon balance is also an essential part of the level. The last thing you want is for your level to become known for fusion scrubs or easy kills with megas. Anarchy can have a mix of weapons, good or bad, it just depends on the type of level you build. If you have narrow tunnels, you won't want (many) Homings or Smarties, but maybe Plasma or Superlasers (and just a few Frags to create a bit of chaos).

Thought #4: Stupid pipe, get out of my way! Argh!

Eyecandy is good, but excessive amounts of it can be fatal for your players. I can't tell you how many times I've darted around to throw napalm into someone's windshield and hit a nearby pipe or beam. Dammit - I'm burning!



Sure, add a few beams or light hangers, but don't block the players' line of sight or flight path.

Thought #5: Doh! Where am I flying to now?

Dead ends: one-way tickets to the path of doom, my friend. Big mistake! At least have two exits or you'll get some nasty pressure!

Thought #6: What the hell was that!?!?

Don't scare your players' brains out, it'll make a terrible mess! Falling, flashing lights, explosions (Kool! What's that?... Boom!) and fog can frighten, irritate, annoy or even enhance gameplay... you never know...

I hope this was helpful for you, now build these Anarchy levels!

[Back to the overview of Section J](#)

115 - Team Games

Hydra

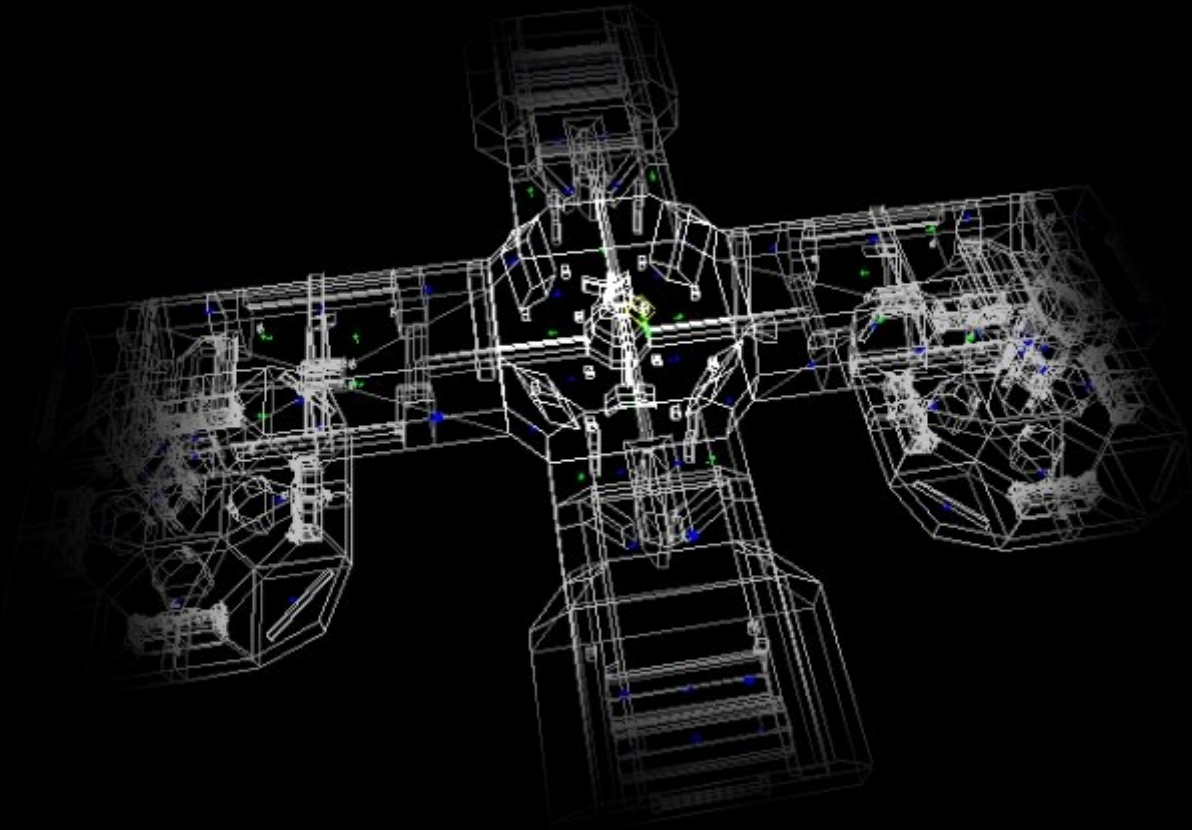


Image taken from the level "No Man's Land".

Symmetry is key for team games. Wait, let me put it better, **BALANCE** is the key for team games. Making symmetry isn't always a good thing because if you know your base inside and out, you know your opponent's too. Keeping the team areas balanced is the most important thing to keep in mind when making a team game level. One team might have structures in their base to make nice offensive attacks, while on the other hand the other team's base might be good at defending their position. How you set up the bases determines the level.

Access to the bases is also an important thing. More often than not you won't just make an entrance to a team base. This would allow campers to sit at the opening and pin down anyone coming in or out of the base. Typically there should be two or three entrance holes, which you can be creative with (the main gate and a secret passage, etc...)

From my perspective there are two different 'styles' of team games. One where you have a 'base' that is only attached with a few entrances. The other is more of a labyrinth of tunnels and these tunnels connect to a larger space with the flag/entropy goals. Each of these styles has its own strengths and weaknesses. The 'base' style is nice because it's generally easier to defend, but usually the area between the two bases is sort of a big 'no man's land' (see where I got the name of my level? :) which means most battles happen here be beaten. It's hard to sneak into a base unless there's a tunnel leading almost directly into it. The 'Labyrinth' style has its own strengths, for one thing it is easier to sneak to the enemy's base as there are many more openings for them to defend and chances are that they won't defend all of them. Although one of the biggest weaknesses is that if you have no sense of direction, like me, it's easy to get lost in the tunnels and fly in circles.

Weapon placement should be balanced for both teams. There should be the same amount of weapon spawn points per team, and many more in no man's land, or maze, or wherever. Depending on the level size, there should be two of each weapon, so that if one team starts with one weapon somewhere in the other team base, the others start with the same weapon. Of course, if the level is medium sized, two Blacksharks wouldn't be the best idea, so use your common sense when placing weapons too.

[Back to the overview of Section J](#)

116 - When Great Levels Die

Schplurg

Part I

Why doesn't anyone play my level?

It's happened to most of us - you spend months with D3Edit to build what you think is the "Ultimate Level". Custom textures, custom ships, some outrageous new weapons, great architecture so cool you'll think a real place should be modeled after it! After spending days checking and rechecking every detail, you package it up and post it to the level pages for the world to enjoy.

The tension rises after you wait as long as possible before checking online to see how many servers are running your recently published masterpiece. You see that your level isn't playing... "Oh, well, it might take another day or two...". Incorrect. After days of searching, your level is nowhere to be found. Maybe you saw a server that had it running for a moment, or so you think. But then again, your level seems to be ignored by the masses. Your level has been sucked into the black hole of levels, never to be seen or played by anyone. All that hard work for nothing... well THAT'S a sad story, my friend.

So why does this great level remain unplayed while other levels look like they were cobbled together in an evening and are listed on dozens of different servers? What exactly is missing from the new level? What about the other, shabby looking level that makes it so popular on PXO, HEAT or elsewhere? Why doesn't anyone play my level?!?!?

I don't know exactly, unfortunately. But I have many ideas and suggestions that will give your level a real chance at immortality in the gaming community.

Layout and balance

Most of the layout information covered is specifically relevant to Anarchy levels. The design layout is the most important factor in level building. If your rooms and tunnels are too small or too large, or if the layout of the rooms is not designed to promote a consistent 'flow' through the level, no amount of cool weapons, lighting, or snazzy architecture can save the level. At the same time, a perfectly laid out level will suffer from poor lighting, poor weapon balance, or ugly texturing. But adding or subtracting weapons or changing a few lights or textures is much easier than increasing the size of all the rooms in a level. You **MUST** have a well thought out layout for the level to have a chance of it being playable, fun and therefore successful.

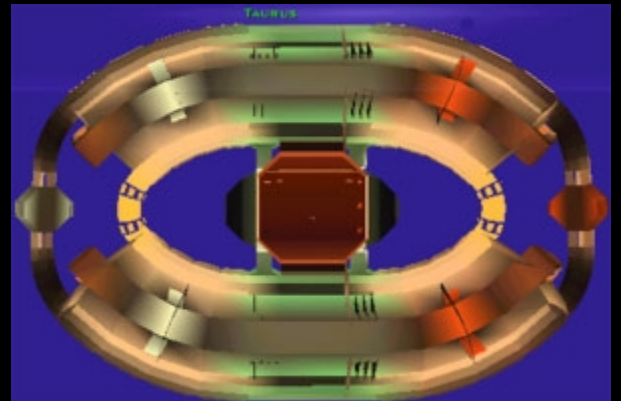
So what makes a good level layout? I was afraid you would ask that, it's a good question. One way to find the answer is to examine some of the most popular levels and figure out what makes them run.

For Descent 3, some of the levels that have stood the test of time being online are **Burning Indica 3**, **Pyroglyphic**, the levels from the **Fury-Set** (from Outrage), **Skybox** and even my own **Earthshaker 2001** has at times as many as six servers at a time (and as few as one or two) What is it about these levels that 🤔 makes them so popular?

The donut theory

Considering the first three levels mentioned above, **Burning Indica 3**, **Pyroglyphic** and the levels in **Fury-Set** reveals an interesting discovery (specifically the Fury levels) I believe that what I call the 'Donut Theory' is responsible for most of the playability and fun factor of some of these levels. The levels don't exactly look like big donuts, but think about it... in the Fury set, levels 1, 3, and 5 all have a central space for combat, and a series of outlying passages surrounding it. Pyroglyphic is similar (although more sophisticated and less symmetrical) in that there is a series of passages that more or less surround one or two central areas. Burning Indica is also a little less donut-like, although it also somehow fits into this theory. I don't want to offend anyone when I refer to a level as a donut, as none of these levels are that simple! Hey, any level that somehow fits the following description fits my donut theory.

— **The Donut Theory™** - A level with a central battle area more or less surrounded by a series of smaller passages with a certain level of complexity. Taurus, the Level 5 from Outrage's Fury series, is a prime example. The reddish central area has a simple ring of tunnels encircling it.



Now, I'm not a psychologist so I can't say exactly why this theory works (if it really works). But I can advise 🤔

- It's easy to get from one side of the level to the other via the central area.
- Players can move in more than one direction and still reach the same destination (via the outer 'circle'). Navigating requires less thinking, in fact you can turn in the same direction anywhere in the level... more brainpower can be used for actual fighting.
- The level is easy to navigate - you're never far from the main area.
- The level is more action-packed because it is more compact. Compare how long it takes to get across the level in Fury Level 5 (Taurus) with the time it takes to get through Level 2 (Halfpipe), which is a long, straight level. Getting to the action is quicker in a level like Taurus, which is actually shaped like a donut 🤔
- Appeals to both dogfighters and tunnel rats.

The Donut Theory is a personal theory I have about level design. Of course, there are other popular levels out there that don't fit into this design. But the donut seems so obvious to me that it deserves its own section here.

Behind the donut

So ok, donuts may or may not be a good idea for Anarchy levels. However, we don't want all of our levels to remind us of the local donut shop! So what else can we do to make our levels playable and interesting enough to keep people coming back for more?

Let's stick with the layout thing. Most of us are familiar with Pyroglyphic by Skyrat. A great level. Pretty textures and perfect lighting give the level a great 'mood'. But there are other levels that look just as good or maybe even better (I won't mention those levels here or say anything negative about ANY level) but they don't get played. It's not looks alone that make a level great. Burning Indika 3 uses a mix of non-custom textures, D2-like architecture, not much eyecandy in the way, and yet it is probably the most played multiplayer level on PXO! So what do Pyroglyphic and Burning Indika have in common that makes them so damn popular?

The levels are compact and a player can move quickly from one area to another... The action is therefore fast. There is more than one way in and out for each room, and usually more. There are places where a player can take cover behind a corner during a fight and quickly appear behind the opponent. If you see a player coming around a corner, he/she could retreat or take an alternative route to circle around you and appear behind you. You just never know where to look.

The tunnels are large enough to fight in, but small enough that frag missiles and concussion weapons are effective and difficult to avoid. It's usually easier to hit a player with a laser in a tunnel because players have less room to dodge. Sometimes I just scatter my quadlasers down a hallway, knowing that at least some of the shots will hit their target. Dogfighting in a large room or outdoor area can be almost impossible with a slow connection. A bunch of players with analogue modems hang out in the tunnels for this reason. Very narrow areas can give the player a claustrophobic feeling, plus sometimes the walls are so close that you can't easily see the way out.

The open areas of these levels are large enough for dogfighting, but not so large that it is difficult for players to hit each other. Having both dogfighting areas and tunnels in the level will attract players of both types. Keeping the polygon count down while having impressive architecture is very possible. It's much nicer to play a well-built level that also appeals to the eye.

aesthetics

The way a level looks can affect its playability. Using texture/lighting combinations that make it difficult to see other ships, or are simply hard on the eye, will seriously detract from the level. Using lots of flashing lights, animated textures, or lights so bright that even Ray Charles would have to wink should be avoided. That doesn't mean you shouldn't have things like this in your level, just don't overdo it... Pilots can tire more quickly from too much eye candy.

The right texture/lighting combination will be easier on the eyes, player ships will be easier to see (if wanted), and people will like you more. One way to keep things under control is to texture the level as if it were a real place. You won't go to an industrial place and see flashing green lights, a yellow ceiling, blue stripes on the floor, and other flashy stuff. It will be rather less spectacular, with simple, metallic textures and a fairly bland color scheme. But maybe you're thinking of a different industrial facility than I am. Be creative, of course... it IS a game after all. 😊

Change is good... sometimes

Ok, so now you have a well laid out level, it's lit and textured the way you want, no one has complained about having seizures from looking at some crazy textures for extended periods of time, what else?

One cool thing about game editing is that you have the freedom to do almost anything you want with the game. You can make stronger, or even totally customized weapons. You can make your own bots and player ships! You can script your own custom events, such as a switch that fills a room with fire to fry opponents. Your first impulse may be to use this 'god'-like power to develop the biggest bomb, fastest ship, and most scripts you can. However, before you make any changes, there are a few things you should consider.

Don't try to be different just for the sake of being different. People don't necessarily like change. Only add major changes if you think they really add to the gaming experience. Make sure you think through your ideas for custom ships and weapons very carefully. You don't want to create a ship that's so fast that the other two or three don't have a chance. And, if it's too different from what people are used to flying, they might not bother to learn. Especially since the ship is only available in ONE LEVEL, namely yours! Including a custom ship may or may not help the success of a level. People usually like their favorite ship to remain mostly unchanged... "Hey, why is my Pheenie so slow?!"

The same applies to weapons. I've seen some pretty crazy weapons people come up with. Some of them are promising, others are either grossly overpowered or just plain silly. I remember testing an impact mortar that kept bouncing until it finally hit someone.

The Earthshaker we have developed seems to be maintaining its popularity, or at least the level it is at. This weapon was successful for certain reasons. The main reason is that many players still know the Earthshakers from Descent 2. The shaker levels were very popular online, especially with high-ping connections, since accuracy wasn't a big factor with these rockets... dodging them was the point. A lot of people missed the Shakers... anyone who redesigned them well for D3 would be in for a possible success.

But if it had been done badly, it would have failed miserably. Having a website like GameEdit to promote it didn't hurt the shaker either.

Through a lot of testing and troubleshooting we were able to come up with a passable design. There are a few small 'issues' with the rocket, but overall it is well balanced for all three ships. Each ship's capacity for this missile has been altered to balance the speed and strength for each ship. The slowest ship has the highest Shaker capacity, while the fastest can only carry two Shakers. Players can succeed in any ship.

The level itself isn't perfect... There are a few misaligned textures and a few other things that I forgot to fix. I still stuck to my principles for texturing and lighting...smooth textures that provide a nice backdrop for you to see your opponent in, lighting that isn't too bright or too dark, although there is a wide variety of lighting throughout the level. There's a mix of room sizes and designs... a few spots to hide in, wide tunnels, narrow tunnels. It's interesting to see how certain players thrive on certain areas of the level depending on their play style.

A well-designed custom weapon can make a level successful... 15 flashy but hastily designed and unbalanced ones can ruin it. Think through your ideas very carefully.

Promote your level

So now you've built the perfect Anarchy level... some well thought out custom weapons, a nice, compact level that looks like it was built by a professional. This probably won't go wrong, right? So...

In order for people to play your level, they have to know it exists. The first, obvious thing is to submit the level to all the level archives you can think of. Some will post about your level in their news page, others will simply add it to their archives without any mention at all. This will get you some publicity, but maybe not enough. If you want your level to be popular, you need to do a little more self-promotion.

A good way to promote your level is to announce it when it's almost finished. Let people know ahead of time that your fantastic new creation will soon be ready. You want people to know about your level before it's even released. The more people see and hear about it, the more they will recognize it and be interested in when it's ready. Just don't overdo it.

You can start your own little website or 'project page' that provides screenshots and descriptions of your level. It's a good idea to wait until your level is almost complete before starting your promotional 'campaign'. People get sick of hearing 'Coming soon...' after a while. Send press releases to gaming sites (like GameEdit.com), perhaps linking to your site to show screenshots of your soon-to-be-released work. If you don't have a site, send images to all the sites that will post their news.

Go to the bulletin boards and post news about your upcoming level, with links to screenshots. You might also get some valuable advice from people this way. When you're ready to beta test it, invite all your buddies to join and their buddies. You'll get useful advice and criticism, plus a little more promotion for your level.

Finally, this could be the most important way to promote your level. If it's close to your level's release date, try to schedule time on a good dedicated server or two. It's crucial that your level is seen hosted in the server list so that people can easily join and autownload it even if they've never heard of it. Try to get as much time on a server as possible. When people see your level hosted on a popular server, they may think 'Well, if server xyz is hosting the level, it must be pretty good'. DwnUnder was kind enough to host Earthshaker for a week or two. By the time he removed it, other servers had caught up and were hosting it. If it wasn't on its server it might have died... I can't emphasize enough how important a dedicated server is to the success of a level. The server admins are the gods of online gaming, especially for D3... kiss their ass accordingly. Most of the people I know are really cool people.

Ready...?

Ok, here is the checklist:

- ◆ Level layout is good and fun, texturing and lighting are perfectly balanced. A professional would be jealous.
- ◆ Custom weapons are well thought out, not overdone, and add something to the game.
- ◆ Custom ships are well modeled and textured, and well balanced with each other.
- ◆ I have a website, and all the news sites and web boards know I'm coming!
- ◆ I have a fast dedicated server ready to host my level for a week.

I'm ready to release my level! Is there anything else I should do first?

If you want to be picky, here are a few other ideas:

- ◆ Release your level on or near a weekend. Not everyone is like you and plays Descent 3 every single day. A lot of people have their free time on the weekend and that's when they play, then they search the sites for news and new levels (I have stats to prove that). If your level is at the top of all of these lists on Saturdays, there's a better chance that more people will see it. If you release a level on Monday, you could already be pushed off the 'new releases' lists on Saturday.
- ◆ Make sure it's ready. There's nothing more irritating than having to download new versions of the same level every few weeks. Try to avoid version 1.1, 1.2, updates, etc., just build a new level instead :)
- ◆ Be nice to the level archive people (like me :)). Most of us want all the news we can get. Instead of emailing us 'here's my new level, it's great, post it', try giving a few details in your email and text file about what makes your level great. If you have a custom weapon, describe your thoughts behind it and explain why you think people should try it. If you can do that, chances are you have a good weapon. If people have hundreds or even thousands of levels to sift through, any information you can provide will make their decision easier.
- ◆ Call your grandmother, you know she'll be happy to hear from you.

Good luck

Even the best level can be ignored and lost in a level graveyard. Luck also plays a role here. However, if you cover all possibilities, it will have a chance of achieving the ultimate goal of level immortality. If he doesn't get it right, don't get too frustrated. Learn from every criticism you receive and build another one. I don't know what else to add. End.

[Back to the overview of Section J](#)

117 - Eye candy

Robot

Can you imagine a level with bare walls, poor texturing and a bland look that immediately gets a good review? Nope. Can you imagine PXO taking over and becoming the next level of hit? Unlikely. Can you imagine that you like that yourself? 'Yeah.' (Damn, now you've ruined the joke) 🤖

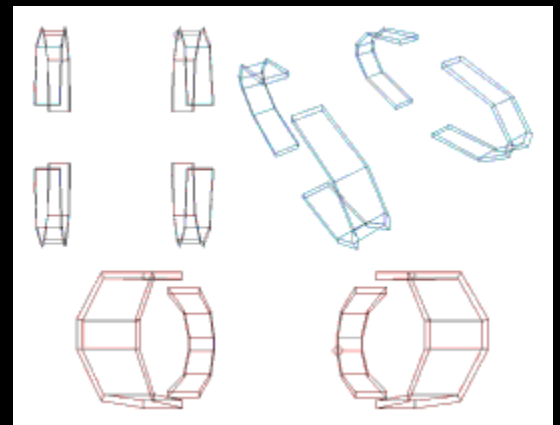
Without good views, yours will likely be sucked into the bottomless pit of lost levels... (aka my first four levels). If your level has nice sights, you can certainly expect it to achieve a certain level of success if you release it correctly of course ;) What this will do is show you a few examples of 'Eyecandy'... The expression of the for Things that you create in your level that make it look so good or make it so unique are used. A little eyecandy is fine, a little eyecandy is bad, and too much eyecandy can send you to the optician with partial blindness 🤖

Let's get started...

The wall bracing:

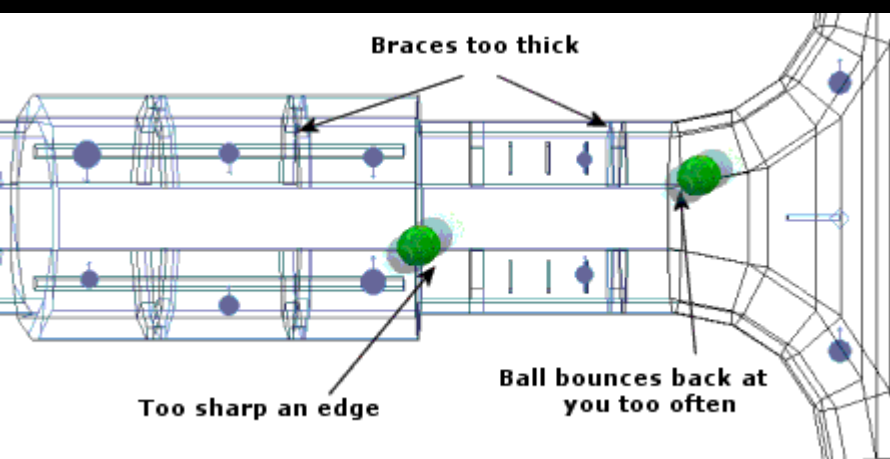
Much used in Dodgeball and improved in Dodgeball Enhanced, also used in all the Outrage levels, with different designs. Wall braces are a very effective way to make a level look good by anyone's standards if you do them correctly.

If too thin, they are simple pieces of eye candy. Too thick and they'll get in the way, but at some point they'll be perfect, I promise!



They are both a blessing and a curse for level designers. They're good for hiding behind and deflecting weapon fire, but in Monster Ball they can cause serious problems. As shown below:

(Dodgeball Enhanced new wall braces)



The pipe:

One of the most effective and simple eyecandy designs. The pipe has the power to make your level look strangely realistic. I'm thinking of Level 2, the Novak prison Complex. The acid-filled mine has broken and cracked pipes cutting through the rock, making for a neat scene.

They are also more flexible than wall braces. You can bend and twist them more easily, and you can also use indentations. You can create a glass section on a tube with textured faces inside, giving an acid-filled tube effect. You can even lath complex tube shapes, like the tubes (pictured) with depressions in them. The possibilities are in the thousands! 😊

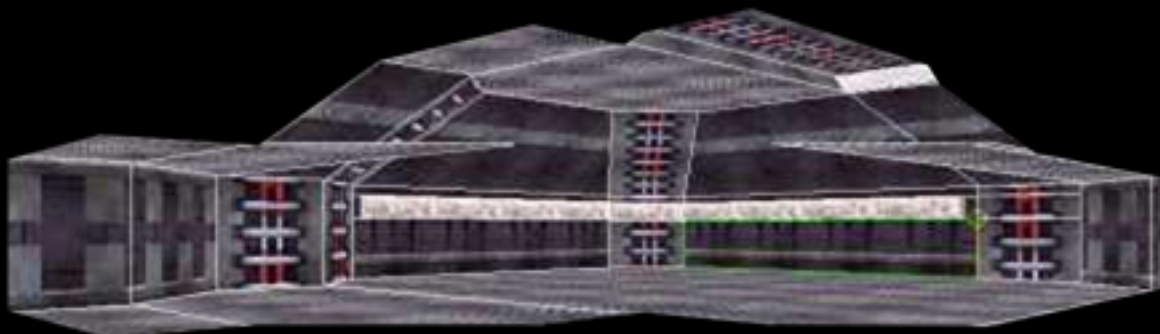
But again, be careful, they can really get on people's nerves if you have too many. Do you want to dodge tubes in addition to napalm blobs? I would try it 😊



Animated textures:


They can bring a sense of complexity to a level, which saves you from creating so much eye candy. But they can also work together. The room (picture) From the Outrage level 'The Core' (well, I actually built it again from scratch when I was bored, perfect copy - hehe) has a moving texture in a kind of 'wall recess' which is ' makes it look more technical. To top it off, they added a halo around the texture. Fantastic work.

I like using animated textures to create space age themes. I do energy beam reactors, lighting, teleporters, even toasters. Just try to use them to your advantage depending on what type of level you're building. 😊



The Spew (say 'eeeeewwww'):

Ahem. Spew is emitted by spew emitters (O_o). A spew blob can be an energy particle, fire, water, steam, smoke or even snow. The spew in the picture (picture) fits the scene nicely... A large battle room with lots of frag turrets looking at you and fire coming out of the walls - perfect!

All I have to say is that the Pyro isn't me, as I wouldn't dream of flying in that direction and I'm flying a Skipnix anyway (Robo doesn't get  kicked in the cross by Turrets, unlike this guy...)



[Back to the overview of Section J](#)

118 - Weapon Balance

Robot

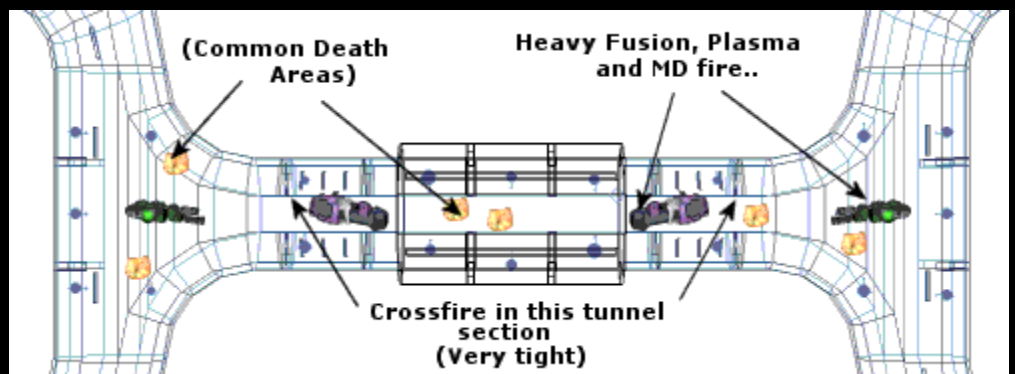
Weapon balance can be the difference between a great level with fantastic gameplay and top reviews and one with poor gameplay and reviews you'd rather not see. I'm not joking (or am I? *ahem*) 😊



I experienced this problem first hand in one of my levels. It's called Dodgeball, I'm sure you've seen something related to it somewhere around here. It was very serious Weapon balance issues causing easy kills in the main corridor connecting each side of the level. It was actually a real mess but it was too easy for some 😊

Here is a sort of 'battle plan' of the problem areas, the problem weapons and the place where you will most likely be sent to hell. Nicely illustrated 🤖

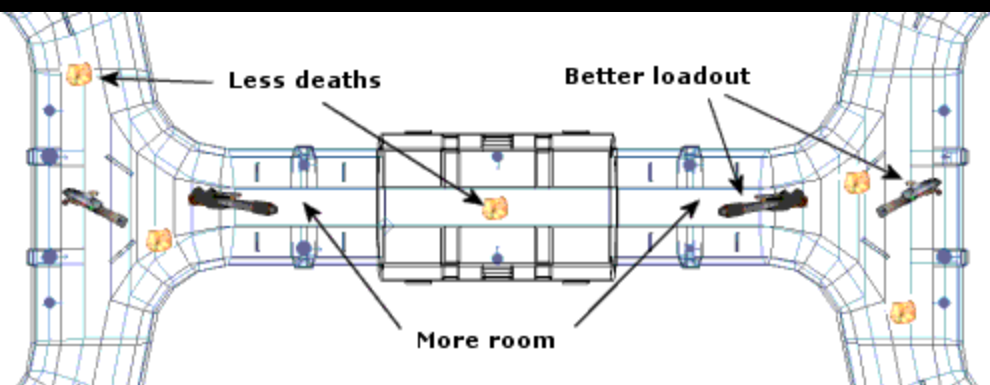
You can easily see the reasons. Firstly, the tunnel is extremely narrow, secondly, there are too many 'spray & pray' weapons and thirdly, there are too many weapon spawn points around around this area.



I call it the one **Bottleneck theory**:)

The bottleneck theory comes into play when you have a narrow tunnel and it is the only way across the level. Weapons are also piling up on both sides, making the area a major battleground. If you make it through the main tunnel in Dodgeball during an intense 4v4 game called 'Throw the Mortar' with more than **10% shield** 'When you get out, you'll get a medal from me and I'll announce it' 🤖

The bottleneck theory is still part of the revision, Dodgeball Enhanced, but not as prominent. There are no longer any places to cover and snipe, there are fewer weapon spawn points, and weapon loadouts have been changed. Instead of an abundant supply



of Smarties, Plasma, Homings and Fusion, you now have a new load of Vauss, MD, Frags, Super Laser and Concussions. However, the geometry is the same, see left:

Ultimately, the game mode and the characteristics of the level decide which weapons are best. It's best to maybe put a few frags in a tight level, but maybe no smarties. It's a bad idea to add a plasma to an outdoor level, you won't hit anything. MD or Vauss would be a far better choice.

Here is a list of recommended weapon loadouts for some game modes:

- **Anarchy:** Everything, it depends on your level. Just don't take too much of anything.
- **Team Anarchy:** Same, really 😊
- **Capture The Flag:** Give lots of secondaries in the areas around the bases, but not too many - and a good amount of primaries in the dogfighting areas.
- **Monster ball:** Strange. Lots of stuff and MD can do the level a favor - or just ruin it. Super Lasers and Concussions are also my favorites in Monster Ball levels.

Short and sweet but acceptable 😊

Just try to apply the bottleneck theory, knowing it will help the gameplay of your level. Beta testing will probably iron out most of your bugs



[Back to the overview of Section J](#)

119 - Why build missions?

Robot

A lot of people ask me, why are you still building D3 levels after almost four years? Well, the answer is as simple as "I like it simple." or "fun"; In fact, it's a relatively complicated answer... There are many different reasons why I and many others still do this, and now I'll give you a few examples...

During my D3 life I tend to use the quote "You don't get guaranteed success, you fight for yourself.", and that works very well for me. Translated into level design terms, this means that you are not guaranteed success with an easy level, just a simple, flat, boring level... You have to try harder every time, improve through your mistakes, find tips and tricks, your Optimize your own work schedule to be more efficient. If you don't think you can, after all it took me twelve levels and a year and a half to get it right



1.

I think this might be the most important of all my arguments, so it comes first. All games are slowly dying away, you have to realize that one day. Although you and I don't want to believe it, Descent3 could be long gone in ten or 15 years, forgotten by everyone except fanatics like you and me



The way I see it, to keep people interested in D3, you have to keep the community active. Can you tell how many people left D3 in its box because they found little activity in the community and then forgot about it? Maybe many. My theory is that in order to sustain a shrinking community for as long as possible, people need to keep building levels. This doesn't have much of an effect in certain respects, but it delays the death of a community enough.

This is one of the main reasons why I made "Robo's D3Edit Tutorials". Trying to get people more excited about learning D3Edit and building those all-important levels. I have to say, if D3Edit were any other program, Descent3 would get a bunch of levels every week. The simple fact is, it is not an easy program to learn. A lot of things are not obvious and difficult to explore.

2.

My next reason is to help people. I started out three years ago with a bunch of boxes, with random textures, gaping holes in the walls, and I didn't know how to test anything I made. Then I discovered tutorials and started building my levels. Over three months I created four levels (officially three) and thought I'm not proud of them, you can say they are special to me :) Then after a year a level called "Dodgeball" was born. This gave me great confidence and gave me even more experience. I then learned to learn even more from my mistakes than before. After a month, my finest work, "Dodgeball Enhanced", was released. People liked it (and still like it), and it's sometimes hosted by Lawrence 'DwnUndr' Britt ;)

Now I'm a level designer for Orbital Design Studios. Work with a program that is MUCH more complex than D3Edit. In a year or two I may be rolling in money, but even if we fail (and we won't :) I'll keep this on my resume for future jobs. I could get a modeling job or join a big software company - you never know!

My perspective is, look where I started and look where I am today. Thanks to a few simple D3Edit tutorials, leveling experience and help from my buddies, I now work for a company 😊

Where could it take you?

3.

pleasure and experience. Perhaps the most important of all reasons is pleasure and experience. This ties into #2 in many ways. Level designing, although it seems difficult, complicated and impossible from the outside, is actually (almost) quite simple and enjoyable. It's just like creating your own world, being 'God' in a way. You say where everything goes, what to do, and you can destroy your subjects at will 🤖

4.

I think my final point is to look cool, but I'm sure how you do that 😊

Inspire people, learn the ways of D3 and D3Edit, don't be afraid to push the boundaries of conventional thinking, and most of all - be creative! And good luck!

[Back to the overview of Section J](#)

Section K–Single player

What else you need to do exciting solo missions

120	Level goals	Orders in SP	Blackshark	416
121	Robotic centers	There will be bots!	Blackshark	417
122	Build according to plan	Build from a template	Ragil Ral	419
123	Paths	Send bots where	(LL)Atan	421
124	BNode tutorial	So that the Guidebot doesn't always Keeps hanging	Kyouryuu	425
125	Set BNodes		(LL)Dark	427
126	Player respawn points	So that you don't always have to start from the beginning must	Ragil Ral	431
127	Include load screen	Replace the default loading image	Robot	432
128	Level briefing	Build a story into the level	RagilRal	434
129	Choice of ship in Single player	More ships for single players	Ragil Ral	439

[Toc](#)

120 - level goals

Blackshark

Hello, I'll now explain to you how to incorporate goals into the level. Because we don't want to go through the hassle of creating a new room, click on the blank sheet at the top left of the menu (File/New...) and up "Default room". Now go to World View and set "Textured with outline". The current face is labeled "Outrage Test". Choose a different texture, I have the "**Brown Volcano Rock**" taken.

Press on **Insert** on the keyboard. Now click on "Room Properties" in the menu "Window" and presses the button that says "None" it says. We call the room "Room1".

Current Goal

Number: 1 of 1

Goal Name: Offener Raum1

Location/Object Name: Raum1

Goal Description: Fliegen Sie in den Raum1!

Priority: 0

Goal List: 0

☐ Secondary

☒ Objective

☒ Enabled

☐ Completed

☐ GB Doesn't Know Location

☐ Not Location Based

Current Goal Item

Number: 1 of 1

☒ Room: Raum1

☐ Object: <none>

☐ Trigger: <none>

Completion Rule

☐ On Activate (triggers)

☒ On Enter (rooms)

☐ On Pickup/Destroy (objects)

☐ Player Collide

☐ Player or Player Weapon Collide

☐ DALLAS Controlled

Annotation: Around objects or Robot too name, you click once on it and look at the top right in the current level Object window (Object Bar), what is selected. Press the button on which the name of the object is written and give to that object, or Robot one names. (Objects to name)

Now that opens Goal window in Menu "Window" under "Goals",

presses "Add Goal" and fills in the two columns and the empty text field like in the picture. The mission goal should be to reach or enter a room. So, what has to be clicked on??? It's clear, "On Enter (rooms)".

Saves the level and calculates the lights. Then save it as an MN3 file and start Descent3. When you play the level and enter the room, you will see what happens! Anyway, I love this!!! If you have any questions or problems, send me an email and I'll try to answer your questions.

I will also explain goals in a little more detail, for example "**Not Location Based**" means. Write to me and the answer will reach you soon.

Bye

Your Manuel

Back to the overview of section K

121 - Robot centers

Blackshark

revised

Hello! Here I will explain to you how to build robot centers. So, click on "New" and "Default room". Goes Go to the World View and select any texture you want. Then press **Insert** on the keyboard or in Menu "Room", "Add room at current room". Now go in the menu "Window" on "Room Properties" and names the room as Room1 (button "none" Click). To enter "Matzen" (this is how a (called Robot Center in the editor language) again press the menu "Window" and then "Matzen".Squeezes"New Matzen" and fills everything in as you can see in the picture above.

Under the "Attach type" sets in which room, at which point or objects robots should be built:

Room and faces: Here you have to specify a named room and a face of it.

Object and gunpoints: Here you enter a named object and the number of it **GPNT**.

Unassigned: hmm....

First we have to specify what and how much robots or powerups are produced shall be.

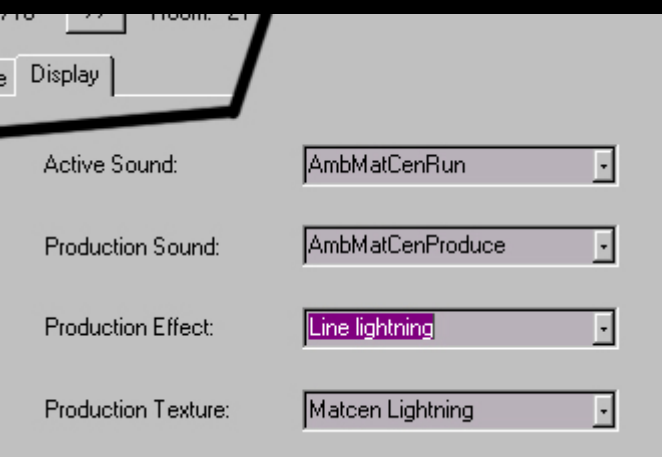
Switches to tab

"Produce" and fills everything in, like in the picture on the right.

So that you also have your own Robotic centers can create, so Not like this, I will now explain a few terms:

Quantity	This allows you to adjust how much of each type of robot or one Powerups are to be produced.
Seconds	The amount of time used to produce the object. The object appears in the middle of this period. Integer values only!
Priority	(Probably comes into play with random production)
Random/Ordered Production	Determines whether production is carried out sequentially or mixed up.
Enabled	This flag determines whether matcen is active or not
No Hurt Players	Setting this flag prevents the player from taking damage when in the Matcen production area.
Number of object types	There you can set how much and what different types of robots should be produced.
While player visible	This means that the robots should form while you are visible.
While player nearby	The robot center will start producing when you are close to the source.
After player visible and After players nearby	This means that after the ship is visible or after it is close to the Source was on, the production of the objects begins.
Script controlled	You have to be able to use the DALLAS Script Editor if you want to specify, for example, that production should start when a door is opened, or other.
Maximum objects produced	There you can set how many objects will be produced TOTAL should. If you leave 0 here, the Matcen won't produce anything.
Production rate multiplier	has no effect... [factor for seconds?]
Maximum alive children	determines how many objects produced by the Matcen are in the level at the same time can be located. -1 here means "Maximum objects produced", i.e. all of them.
Pre-production time	Time before production begins
Post production time	Time waiting after production

A full production cycle consists of Pre-production->Seconds->Post-production, so the total length is the sum of all the times that you specify here.



You can now set the following under the map display:

Active sound is the sound that indicates whether the materialization center is active (=ready for production).
 Production sound is played back when the object is created.
 Production effect is the graphic effect displayed in production, and
 Production Texture is the texture used for the graphic effect.

Finally calculate the level and save it as MN3 file. Complete!!

Have fun building!

[Back to the overview of section K](#)

122 - Building according to plan

Ragil Ral

A self-respecting level builder makes sketches and drafts before he starts crafting, if only to avoid major repair modifications, but also because you can be creative in a different way on paper - it usually starts with a few lines and after an hour there is a finished draft. But what if you want to build as closely as possible according to the design, or want to recreate a given floor plan exactly?

Transferring dimensions and ratios from a plan or sketch to D3Edit is quite laborious, especially if the plan/sketch is only available digitally.

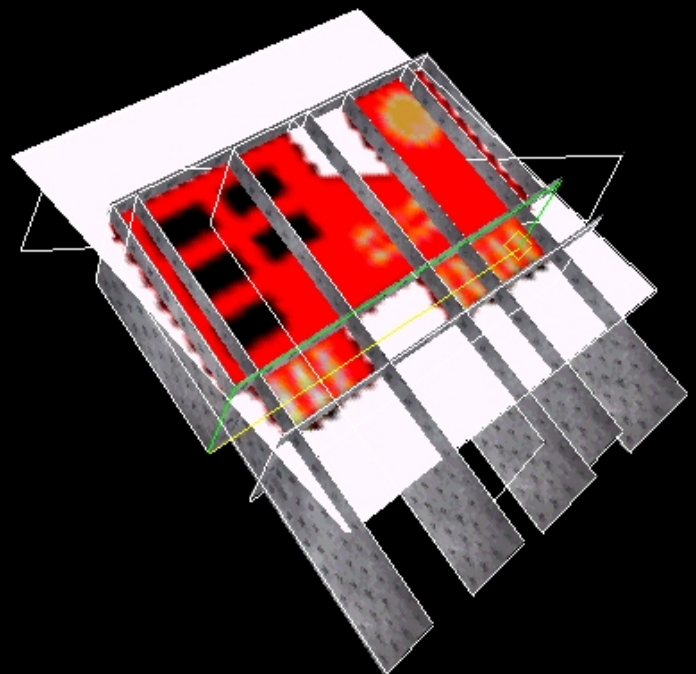
My solution is now the following:

If the underlying images are already available as data, great. Otherwise, you will have to scan or photograph the templates in order to have them as a file. This can now be transformed into a D3 texture, into a customgamLoad and use in D3Edit. You stretch the texture onto a simple square and position the new faces at right angles to it.

Then it's easy to follow the given lines - you activate the face interface **Z** aBdffering, so textures to be displayed exactly and recreate the template.

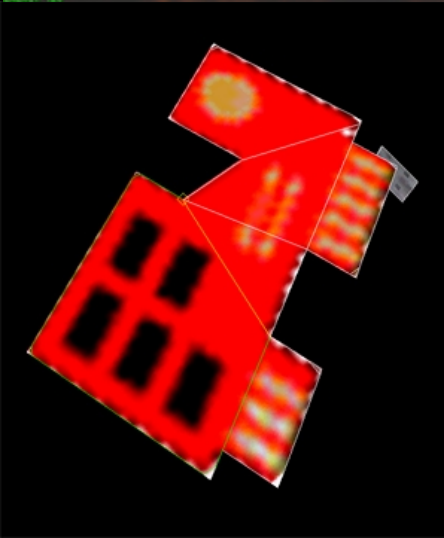
If the building is to be larger, it is advisable to divide the template into several textures so that the representation or resolution of the template in D3Edit is still sufficiently good. It is also recommended to work with 256 textures, which display even finer.

It started with me wanting the original D1/D2 keycards (bottom left)...



Face with template as texture, with some positioned faces (top right)...

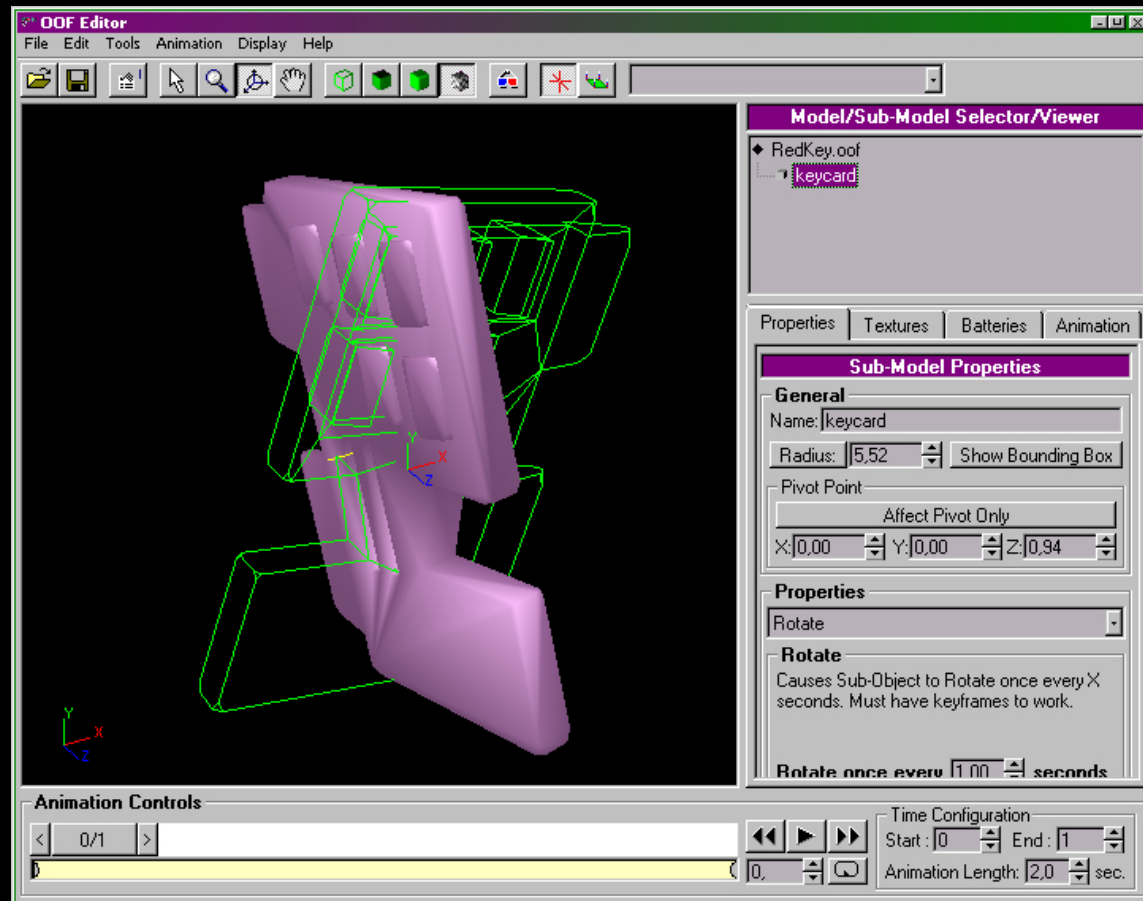
On the left is the cut front of the keycard.





The finished map...

... convert into an object.



the Redkey ingame.



Back
to overview
from section K

123 - Paths

(LL)Atan

These instructions show how to create paths with D3Edit. -

Open your level and switch to the current room view. (Current Room View)

 Descent 3 Level Editor - - Current Room View - "<none>" - [- Current Room View - "<none>"]

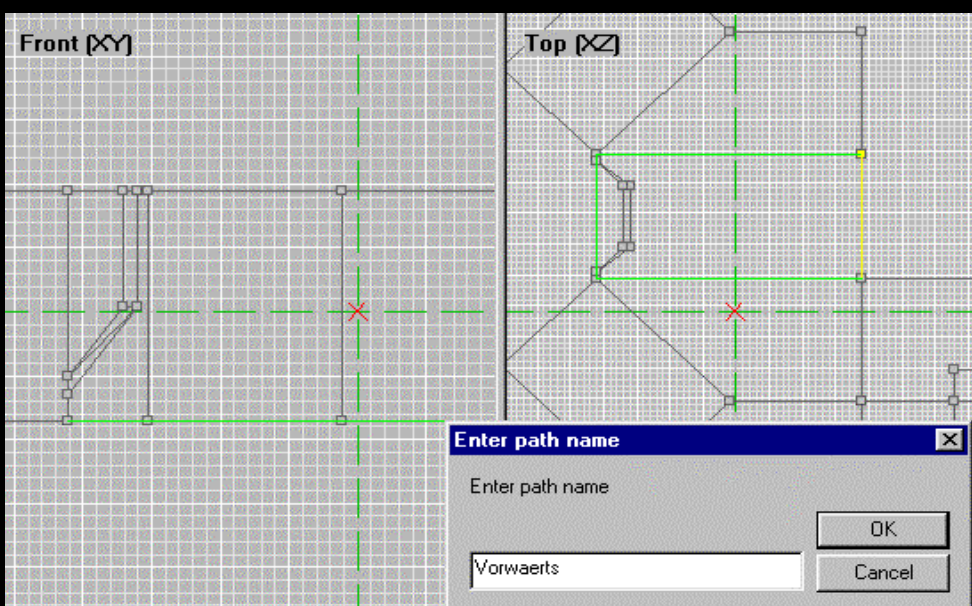
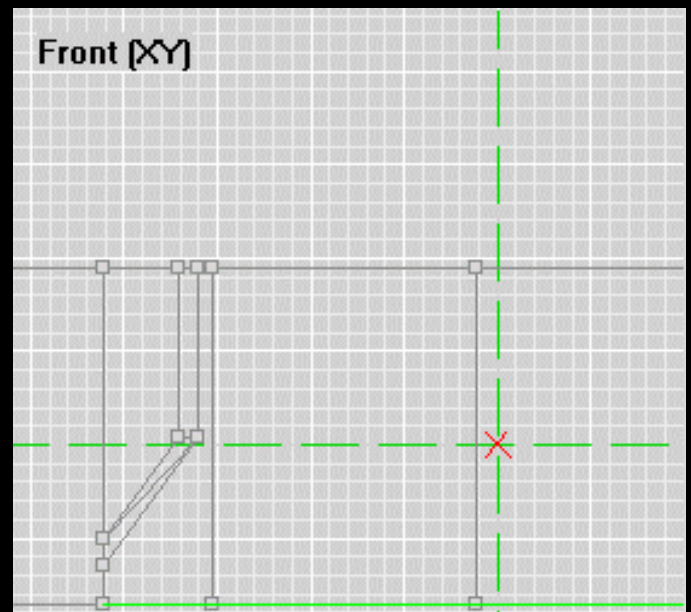
Toggles 'Textured View' or 'Textured with Outline View' view.

Now switch on path mode (**Ctrl-H**):

R: 0/1 F: 0/138 -1/0 E: 0/4 V (idx): 0/4 (1/151) Grid: 2 1896, -82, 2048 1998, -70, 2068 **Path mode** Marked: V: 0 F: 0

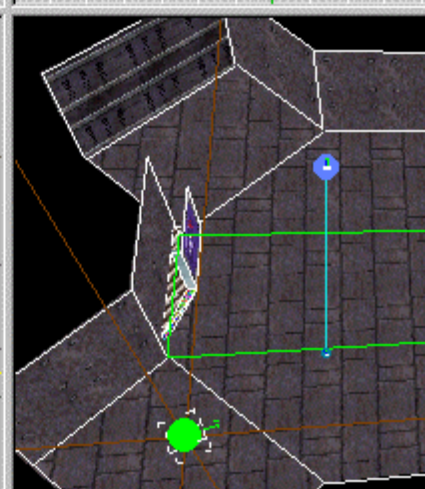
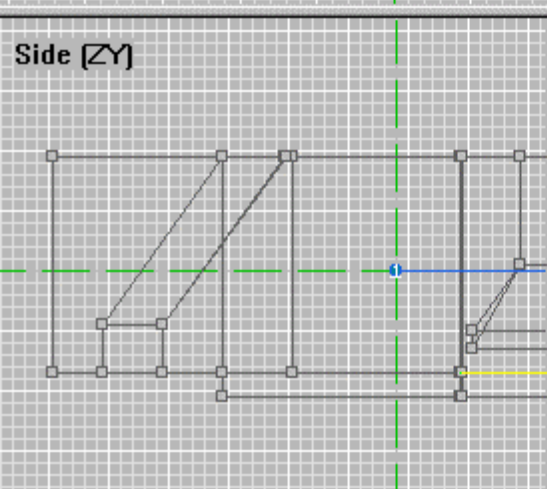
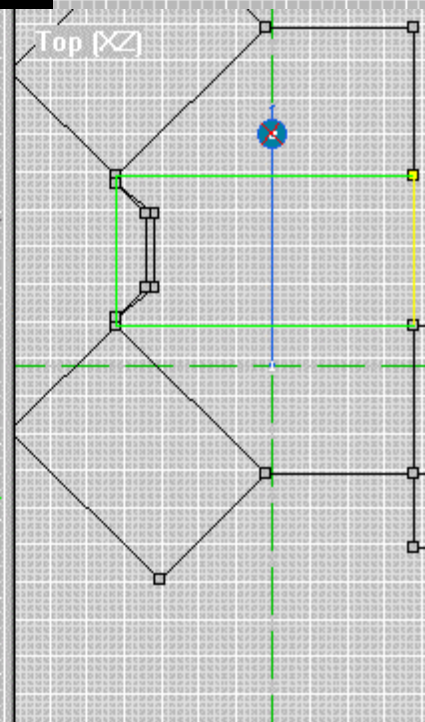
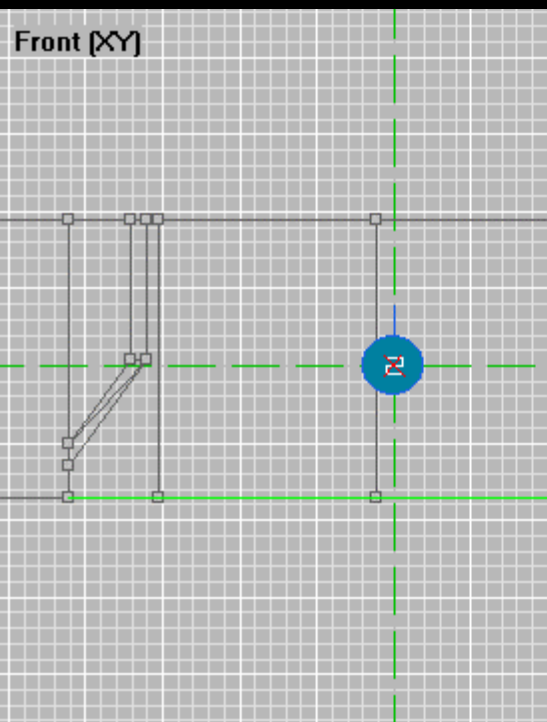
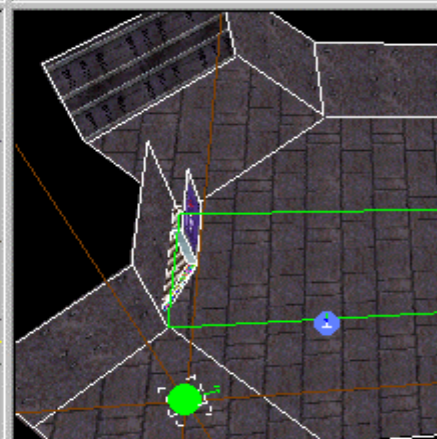
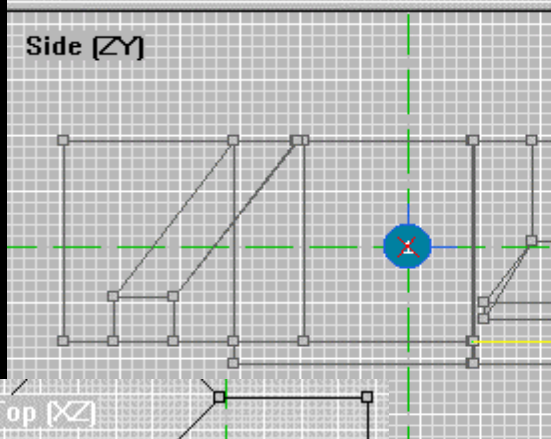
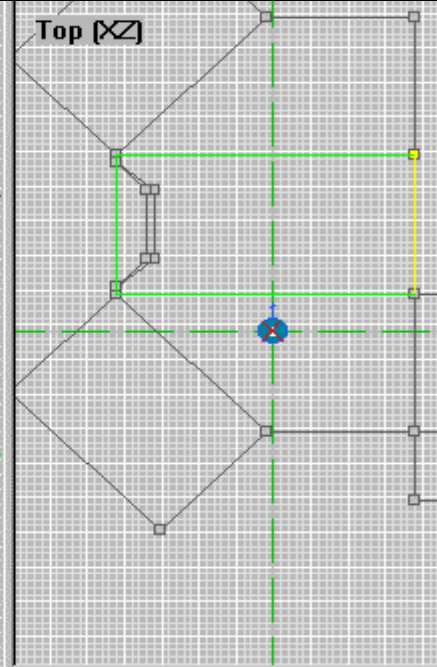
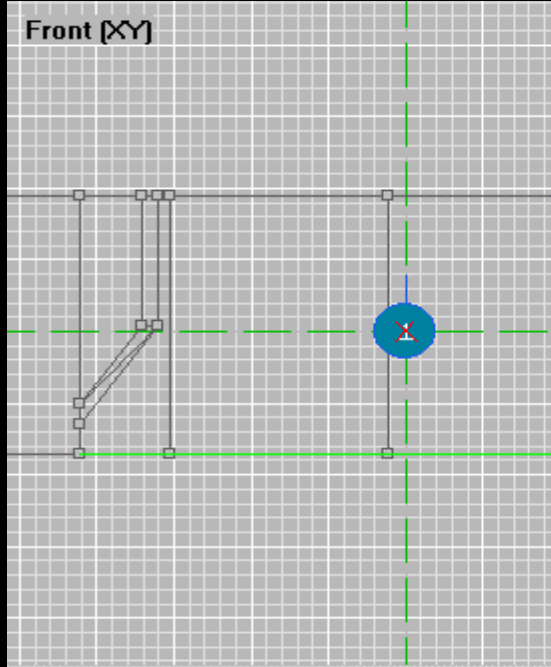
The **X** with **Ctrl + arrow keys** bring it to the desired position.

One of the front, top and side windows must remain active.



SHIFT+Insert around one
Create path.
In the input field that
now appears
desired path name
input. (Left)

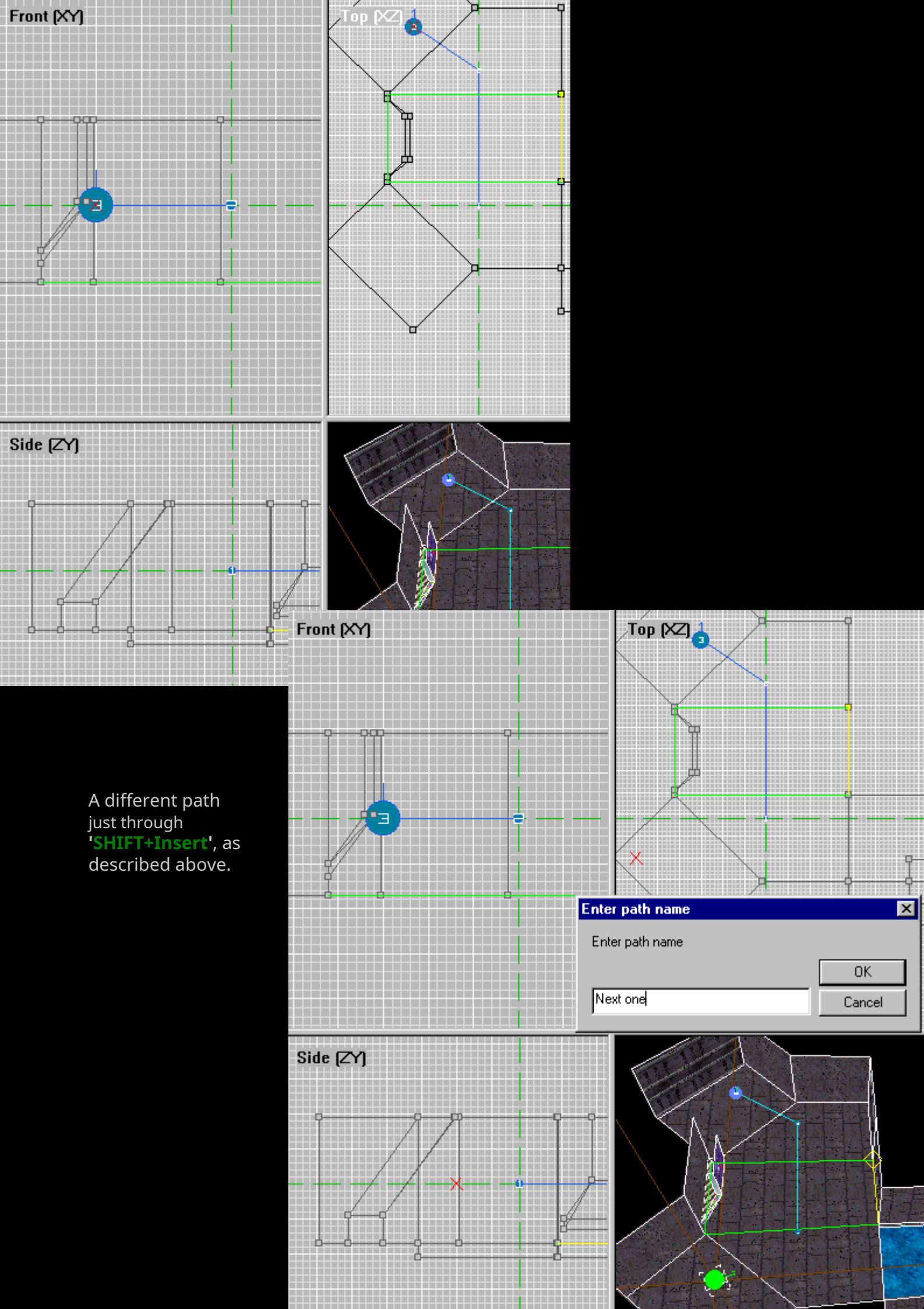
Once entered, the path will appear with a single Path node (No.1) at the target position (X) generated.

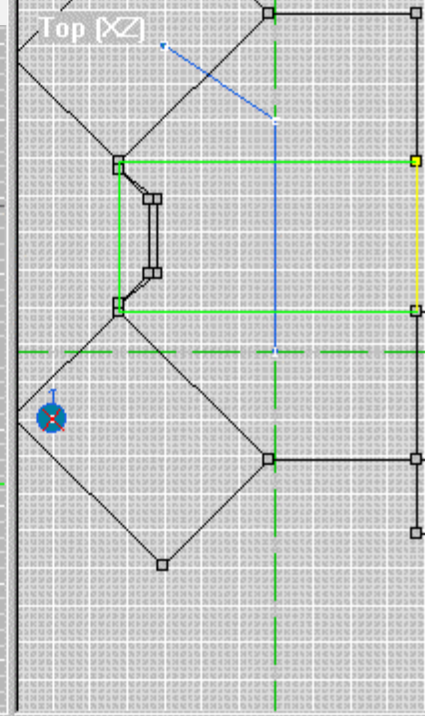
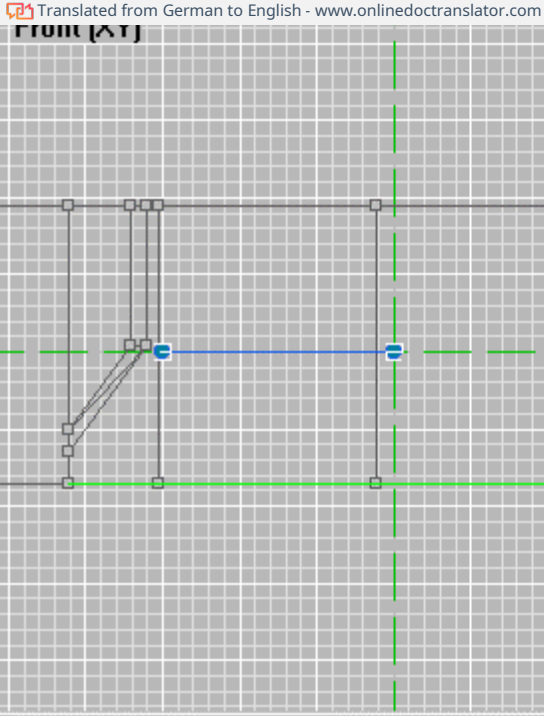


The new pathKPkt is each **after** inserted into the current pathKPkt. (For this, a path KPkt must first be active (blue circle)).

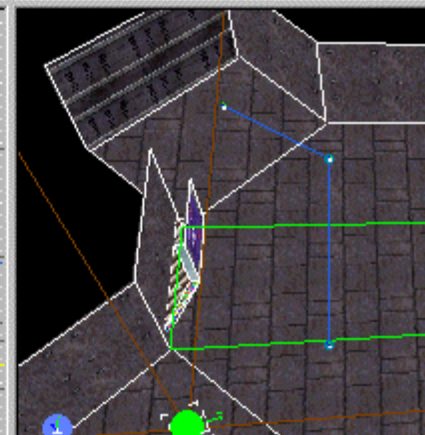
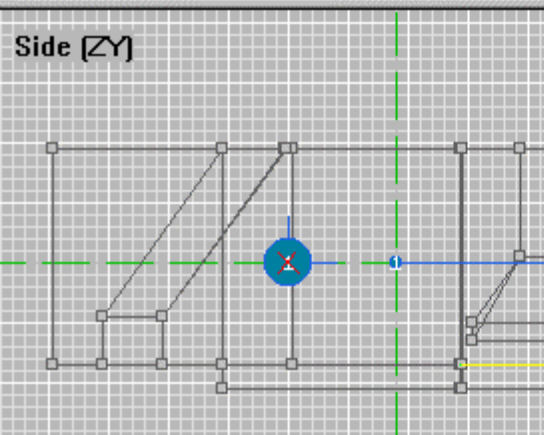
To create a pathKPkt to **current** To make PathKPkt, just get close to the click desired or with '**N**' and '**SHIFT+N**' Go through the path points to select the correct one. (For this a pathKPkt must first be active (blue circle)).

The '**X**' with '**Ctrl**' + **arrow keys** move to the next desired position. With '**Insert**' you can now enter additional path nodes. (No...2...3...etc.)





If there are several paths in the level, you can use 'H' and '**SHIFT+H**' select the paths.



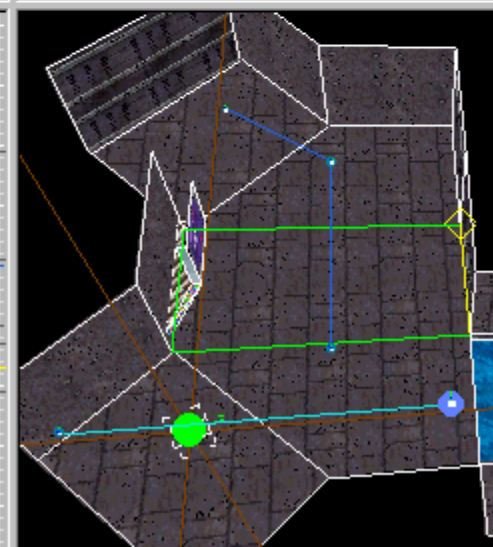
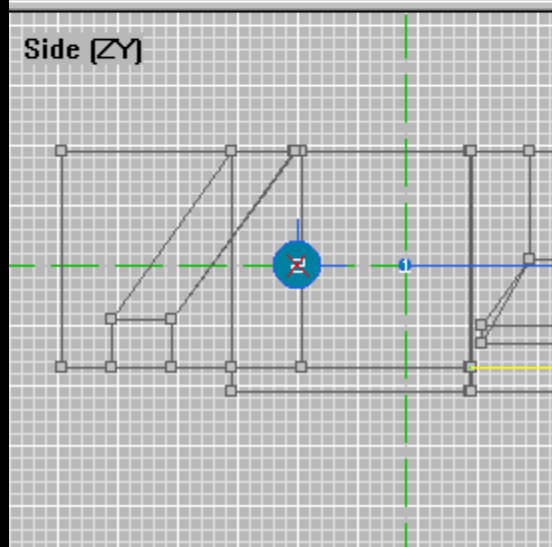
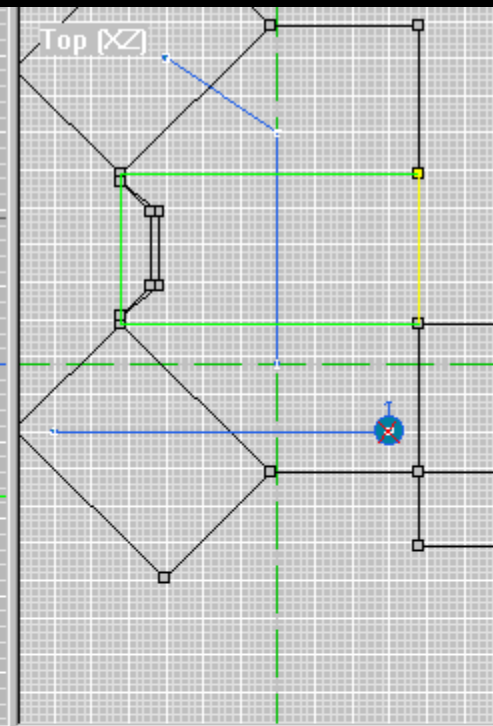
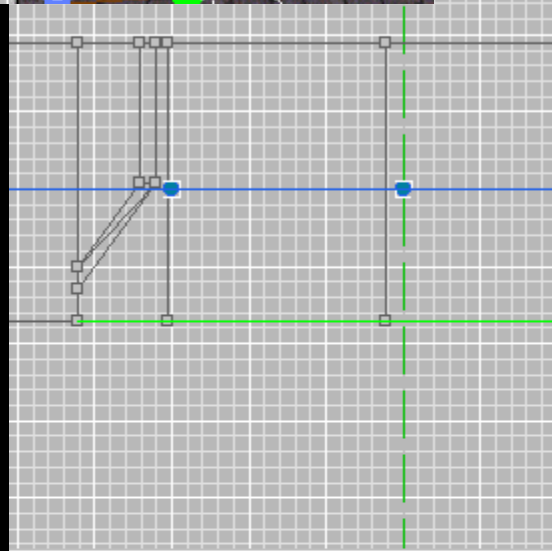
Message window (1.1 Beta 8) you can see which path, Point you have just selected.

Path 1, Node 2 selected. Path Name = Next one

You can edit the path points in path mode and then as usual with the number pad keys. One of the 2d windows must be active for this.

Deleting the path components is done with '**Remove**' possible. If the last pathKpkt of a path is reached, the editor asks whether the path should be deleted.

Good luck
(LL)ATAN
Back to the overview of section K



124 - BNode Tutorial

Kyouryuu

revised

Learn what the hell BNodes are and how to use them here. This tut was revised in January 2003.

Introduction

Back in D1 and D2, when everything was made of cubes, navigating the levels wasn't difficult, even for the robots. Place the GuideBot in a level and it automatically knows where to go. Put a Bulk Destroyer in a level and set its behavior to Snipe, and it will find the best places to do it. Don't get me started on Thiefbot. There was practically nothing you had to tell the robots where to go.

In Descent 3 things are more complicated. The AI of the game's enemies and the GuideBot might be smarter once they encounter you - but to navigate these huge mines, even the bots need a map. That's what the BNodes are actually for. BNodes are a guidance system for the AI - a series of waypoints that the AI can follow in its quest to get from point A to point B.

The AI is very accurate with the BNodes. If there aren't any in the mine, chances are your bots will act like complete idiots. What's more, the GuideBot won't understand how to get from one objective to the next. BNodes are essential for single player but serve no purpose in multiplayer levels.

Before you start

Depending on the size of your level, BNodes can be quite a time-consuming task, especially since D3Edit seems to crash regularly when you manipulate them. Save often! Before you get started, there are a few scenarios that should be covered in detail to make BNodes as easy as possible, and also a few warnings.

- ◆ First, never have portals that are inaccessible. D3Edit sets a BNode to every portal! The AI is not smart enough to understand that a portal is inaccessible and will persistently try to ram into the obstacle rather than find a way around it.
- ◆ In D3Edit there are a few unavoidable situations where the Guidebot just behaves stupidly. Especially if you have more than one portal in a room, but only one can be passed through. For example, let's say we have a room with six portals, but five of them are scripted to be blocked with glass. Instead of looking for the free portal, the Guidebot will ram into the glazed ones. Sometimes he gets it when you shoot at him or make him follow you to the free portal. Then the Guidebot will combine the correct series of BNodes and fly off from there. Of course, both of these negate the point of having the Guidebot from the start. I'm not sure, but I have a feeling that Outrage had more sophisticated intentions with the BNodes when they designed the original D3 levels than the dull every-portal-gets-a-BNode strategy used.
- ◆ Rooms should have all of their portals fairly accessible from other portals. That means don't build a 360° ring room. Optimize for two 180° spaces or four 90° spaces. That's smart level optimization anyway.

Use BNodes

Back to the task at hand. Let's assume you have a single player level and are ready to use your BNodes. You should do this when your level is almost finished - don't do it while you're still building it. Click to open the Path panel. We want to manipulate the BNodes, so click on that AI Paths (BNodes)-Radio button.

You might ask, what is the other path type used for? These game paths are used for activities such as guiding robots on specific paths through the mine, where I want them to consistently follow a very well thought out path. They are also used to guide the camera for in-game cinematics.

Click Create **AI Path Nodes**. A small dialog should then open (if not, the button opens it **Verify Nodes**). The dialog tells you what's wrong with how things ended after D3Edit auto-inserted the BNodes. Ideally you want a 'VERIFY OK!' message here, but more likely than not you will get a list of problem spaces and the BNodes in question that are having problems. The key is that you have to make sure that each BNode can reach at least one other in the mine. You can't just leave any of them lying around, locked away from the system.

Finally, the basic strategy is to either move existing BNodes in the room view so that they reach the others, or create new BNodes in the same way. After you do one of these, click on the Auto Edge Room button (To update) to see if everything worked. If not, repeat. However, there are a few precautions:

To select a BNode, you have to go to the Room View Textured with outline have set and click on the BNode with the mouse. You also have to be in path mode (**Ctrl-H** or click on the button in the toolbar). BNodes cannot be selected via the 2d views. However, once selected they can be moved around.

To move a BNode, select it and use the keys on the numeric keypad to move it.

Don't use more BNodes than the bare minimum. Otherwise the AI will be confused.

Keep this in mind and you will be a BNode master before long!

Back to the overview of section K

125 - Set BNodes

(LL)Dark

The editor tries on every room portal and in the virtual Put a BNode in the middle of the room and then connect them together.

In a round or rectangular room this isn't a big problem. However, as soon as the room has a somewhat unusual shape or has built-ins, the editor regularly fails. In addition, it may not be desirable to have a bnode on every portal because the portal, for example. A window is or leads into a room that the GuideBot shouldn't find, or, or, or....

In all these cases you have to intervene manually.

How this is best done will be explained here.

rooms

The important things first:

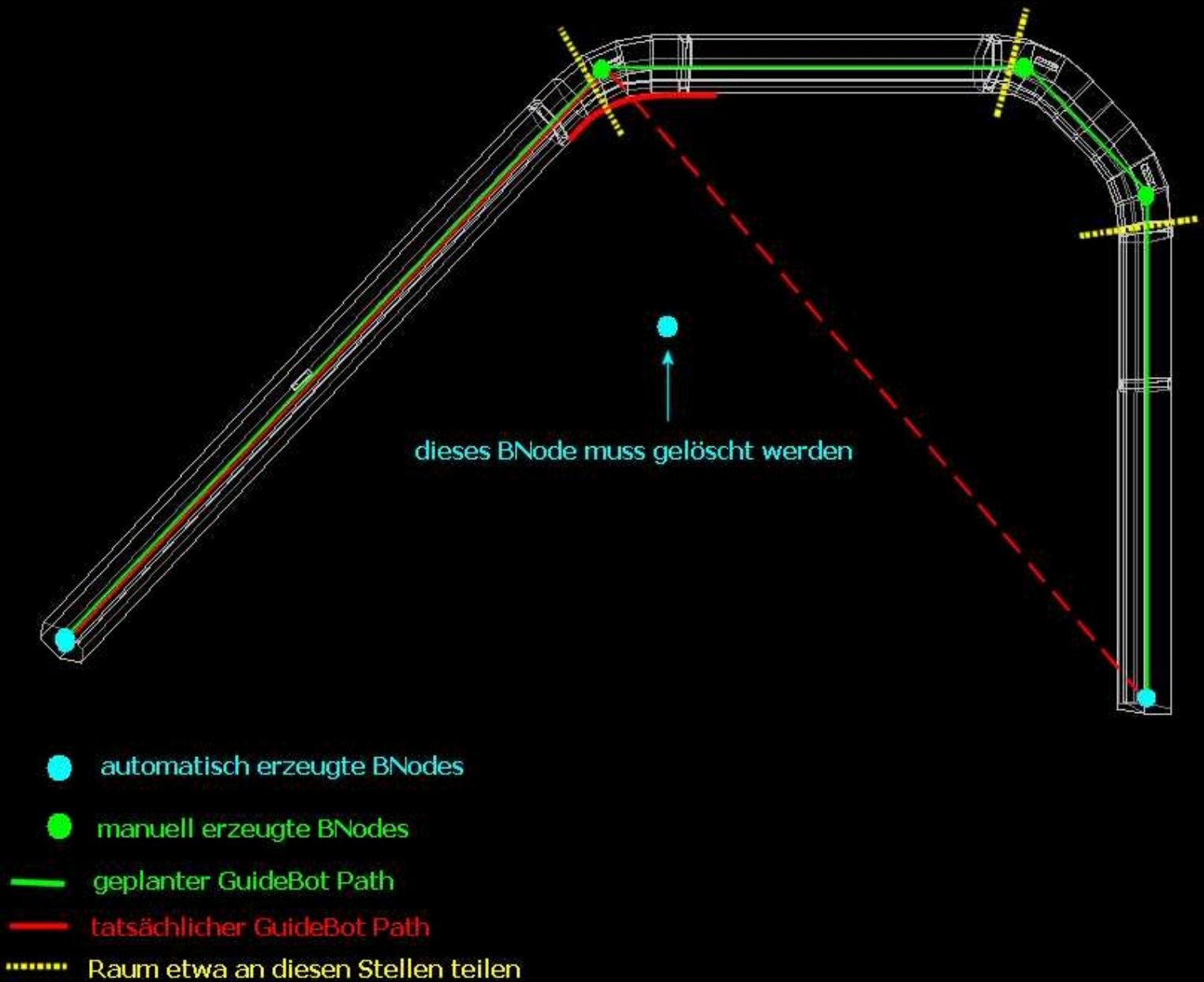
It is of the utmost importance to build the rooms in such a way that they do not have any angles, deeper bulges or larger curvatures. If you have such spatial structures, you have to break them down into individual rooms which are then connected to each other using portals.

Why ?

The GuideBot was actually supposed to move through the room from node to node. Ideally it does. It flies from the entry portal node to the node in the middle of the room and from there to the exit portal node. From there it goes into the next room. As I said, this is the ideal case - unfortunately it rarely happens.

As a rule, you have several nodes distributed throughout the room and perhaps more than just two portals. Then a rather unpleasant feature of the GuideBot comes to light. You notice that the GB is a lazy piece. It tries to find the shortest path to the nearest node and doesn't care whether these nodes are directly connected to each other by edges. (Edges are the connections between the nodes) It doesn't matter to the GB whether there is a wall between these nodes. The result is that the GB is stuck stubbornly in front of a wall and can't get any further.

Here's a picture to illustrate:



As you can see, a BNode is created in the virtual center of the room. The editor is not interested in whether the center of the room is inside or outside the actual room. BNodes that lie outside the level structure must be deleted. Next, add new BNodes by hand. This always happens so that there is a direct line of sight to another bnode. In the image, these BNodes are shown in green. You would now have created a continuous AI node path through the entire room and you could be satisfied. Unfortunately, the GuideBot would take the path marked in red and thus get stuck on the wall marked in red.

The only solution here is to divide the room into 4 individual rooms along the yellow dashed lines and then connect them again using portals. If you then run the automatic AI node creation, hardly any subsequent manual work will be necessary and the GuideBot will also find its way without any problems.

Note:

The curvature of a corridor should never be greater than 45°.

In very wide aisles it can be a maximum of 90°. (but not recommended)

Avoid deep room bulges. It is better to add such bulges as a separate room.

Portals

By default, the editor tries to set a BNode on each portal. **However, setting the BNodes depends on the texture of the portal!**

Even if you don't see the texture on a portal, it's still there. (select a portal and look in the texture bar) In order for the editor to set a BNode on the portal, this portal must have the texture *Palmleaf1* have. (In Atan's last editor version the portal texture is automatically changed accordingly) If a portal has a different texture, then NO BNode is set. This means you can easily influence whether a BNode is set on a portal or not. This also works afterwards. If a BNode is set, it is always in the middle of the portal.

Execution step by step

Create levels and insert all objects.

1. Check all rooms to see if they meet the above requirements. If a room had to be split and/or a new room had to be added after the AI node system was created, all BNodes of the level would be deleted and you would have to start from the beginning!
2. Check ALL portals to see whether they meet the requirements described above.
3. With older editor versions you have to make sure that all portals have the texture *Palmleaf1* (except those that shouldn't get BNodes).
4. In the latest editor version it is the other way around, here you have to make sure that all portals that should not get a BNode do NOT have this texture. (since the portal textures are created automatically when the portal is created *Palmleaf1* were set)
5. Now create the AI node system! This happens as follows:
 - a. The *Path Bar* open and there AI Paths (BNodes) choose.
 - b. Now click on *Create AI Path Nodes*.
 - c. BNodes are now automatically created in all rooms throughout the level and attempts are made to connect them with each other using edges. You will then be shown a list showing which rooms this did not work and which BNodes are not connected to each other there.
 - d. In any case, you now have to take action in these rooms yourself. However, it is STRONGLY recommended to check ALL other rooms as well.

Select the room you want to work on and go to Room View.

If you don't see any AI Paths or BNodes, click the **right** Mouse click in the window and click on *Display AI nodes* a tick.

Change with **Ctrl+H** into Path Mode or press the button



Now you can come with me **N** switch through the nodes individually (with **Shift+N** it goes backwards).

Of course you can also use it in the *Path Bar* using the arrow buttons below BNodes switch through.

This also shows you which node is currently active.

In the Room View, the active node is enlarged and displayed yellow in the coordinate windows. Now first check whether one or more nodes are outside the room. Since they are sometimes difficult to find, simply switch through all nodes for this purpose. If you have found such a node, delete it by typing in the *Path Bar* on **Delete** clicks. Also delete nodes that are placed too close to a wall. (or move them) Such nodes usually cannot be connected to the other nodes through edges. After removing all such nodes, click in the *Path Bar* on **Edge cRoom Nodes**.

Now an attempt is made to connect all remaining nodes in the room with each other. If that works, then this room is finished and you can move on to the next one. If it doesn't work, or you are not yet satisfied with the arrangement of the nodes, then it is probably necessary to add more nodes.

To do this, set the dashed green zero line in the XY or ZY coordinate window **Ctrl+left mouse** to the room height at which the node will later stand.

Then set the red one in the XZ coordinate window (top view) with a mouse click **X** to the place where the node should be inserted and then press on the keyboard **Insert** or click in the *Path Baron* **Insert**.

A query may now appear as to whether the new node should be integrated into the current edge (which is yellow) or not. Unless you want to cleverly redirect an Edge, you can always answer this query with no.

Now insert your new nodes so that they are always in line of sight with another node. Also make sure that they are not placed too close to a face (5 units apart is usually enough).

The nodes must be arranged in such a way that a continuous connection from one portal to the other can be established from node to node.

In addition, additional nodes can of course be arranged in the room to bypass installations or to achieve smaller indentations.

The nodes can also be moved by selecting them in a coordinate window and then using NumBlock **8th/6/4/2** moves. They cannot be pushed through faces. When you think all nodes are correctly placed, press again *Edge cRoom Nodes*.

Repeat the above until you are happy with the result and the room is no longer listed as defective.

A notice:

If your level has a terrain and there are external rooms there, the verify list will probably show BNode errors for rooms that don't exist in your area.

(at least you won't find them)

In Atan's newer editor versions it says there *No path from x to y in terrain regions* You can safely ignore this error message.

Sometimes nodes on portals simply cannot be connected.

Simply move these nodes 2-3 units away from the portal. (continue if necessary) After that it should work.

If you have parts of the level that are separated from one another by terrain, then make sure that there is a direct line of sight. Otherwise the GB will not find the way.

Unfortunately, AI paths in the terrain cannot yet be created with the last editor version I know of (V1.1 Beta9 Atan 0.3_3_N - no public). However, this function is already in preparation and may be available at some point.

(this works now)

Have fun level building
(LL) Dark

Back to the overview of section K

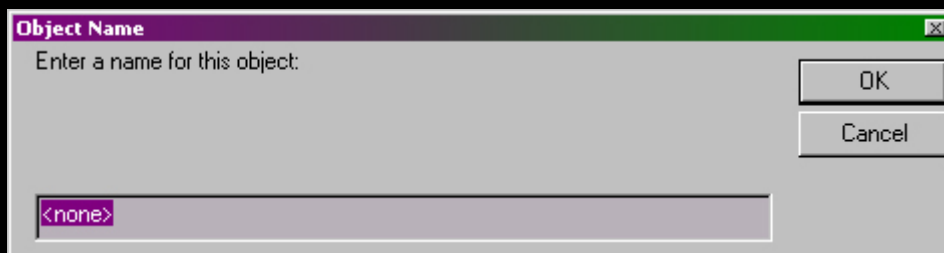
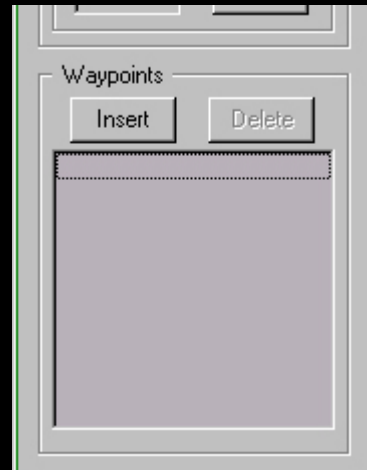
126 - Player respawn points

Ragil Ral

If the player runs out of ammunition or other cataclysmic events occur, he may lose a life. So that the player doesn't have to move through the entire level to the point where he left his spew, there are waypoints.

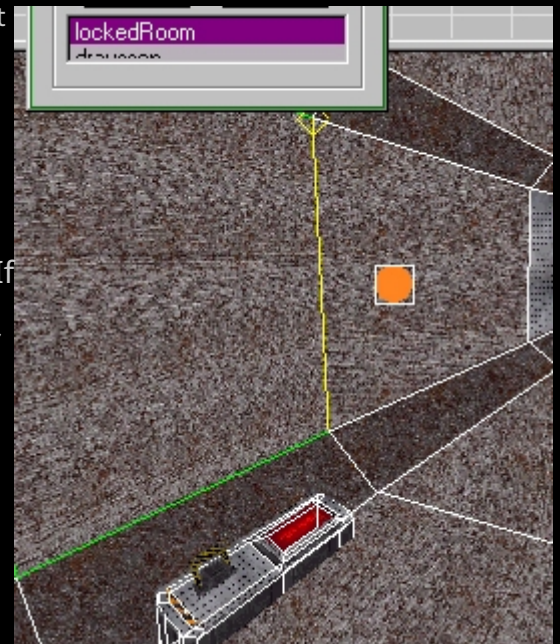
These are located in the **S** the soundbar and are inserted in the room view (right).

So you go into a room, do a face current and add over it **Insert** -Button enter a waypoint. A dialog opens in which you can name the waypoint.



The waypoint appears as an orange ball, you can view it in Object Mode (**Ctrl-G**) can also be moved and handled via the Object Bar () - so **O** works in a very similar way to the sound sources.

You can't place route points directly on the terrain, and they don't work in external spaces, you end up in HOM. If you need a waypoint on the terrain, add an internal room, insert the waypoint into it and connect all faces of this room to the outside world (bottom left), similar to sound sources on the terrain:



The waypoint is activated as soon as the player enters the room in which the waypoint is located.

There is nothing more...

Back to the overview of section K

127 - Include load screen

Robot

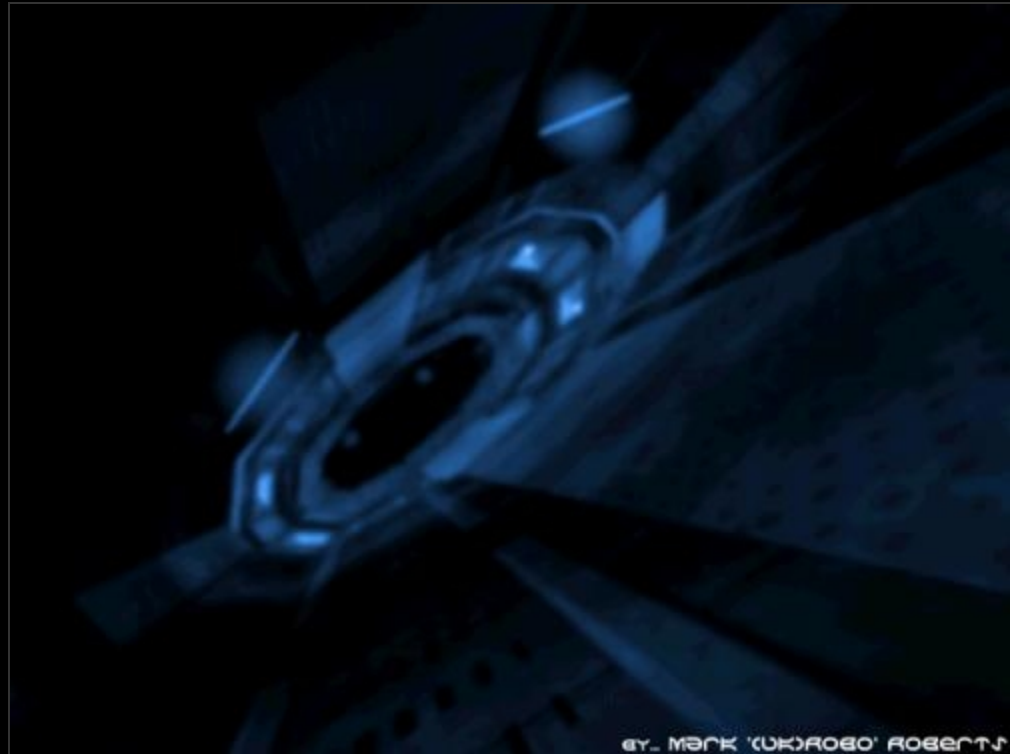
revised

When you select a level it begins to load. You may see some image when this happens, which is a load screen. They usually put me in a trance, which just shows how easily impressed I am 😊

Creating a load screen is one of the most enjoyable parts of level creation for me. I like to be creative, add nice shades of color and cool effects. And I always include a name. Check out this load screen:

I am here 😊 This Loadscreen is the result of 30 different ones Files, that's a lot of colors. The level doesn't look like that, of course, it's just a little art. What you **not** You can see in this picture what Descent 3 does with it. I applied a Gaussian blur. On the outside

Rand sees it as a simple and normal one Soft focus, but in D3 the colors become a nice effect split up.



This is a kind of 'Outrage-style' loadscreen, because all Outrage levels have this effect in their loadscreens.

I'm working in Paint Shop Pro since I don't have Photoshop, but anything that can do a Gaussian blur will work. Your image must be a 640x480 bitmap. Once you have completed the creative process, apply the Gaussian Blur.



You're lucky if you can change the strength of the blur. An image may not look right in D3 with a setting that is too high/too low. Make the values higher if the image has light colors, lower if it is dark.

Then add your name. We're doing this now because we don't want it to be blurred. When you're done, save the image as **uncompressed 24-bit .tga**. Now to convert.



Descent 3 itself can't.tga-Use files for the load screens, you have to put them in.oif-Convert format. You can do thatOGF tool(with D3Edit included) or the fantastic program calledD3 Image Tool(see Tools section).

Find your file and set the options as in the picture on the left, set your own output directory.

Finally, we need to specify where your loadscreen is in themn3file is. Use thatLevel propertiesdialog in the MN3 editor, it should then look something like this:

```
Lvl: C:\Games\Descent3\missions\D3-Edit\Dodgeball Enhanced\dodgeballenh.d3l
... Loadscreen: C:\Games\Descent3\missions\D3-Edit\Dodgeball Enhanced\loadscreen.ogf
... Briefing: (No File)
... Music: (No File)
... Branch to: (Not defined)
... End mission: No
```

If you use the Quicktest, first include your load screen in the file list. Then double-click at the bottom of QuicktestLevel setup/ordering:onLoading;and select your load screen.

Save yoursmn3... and test it!

Back to the overview of section K

128 - Level briefing

RagilRal

A single player level can be further enhanced with a level briefing. Here you can give the player additional atmosphere by presenting them with a context in which the mission takes place.

You can create level briefings with the Briefs32 tool, which is part of the Descent Manager but can still be installed individually; see section 'List of all D3 tools'.

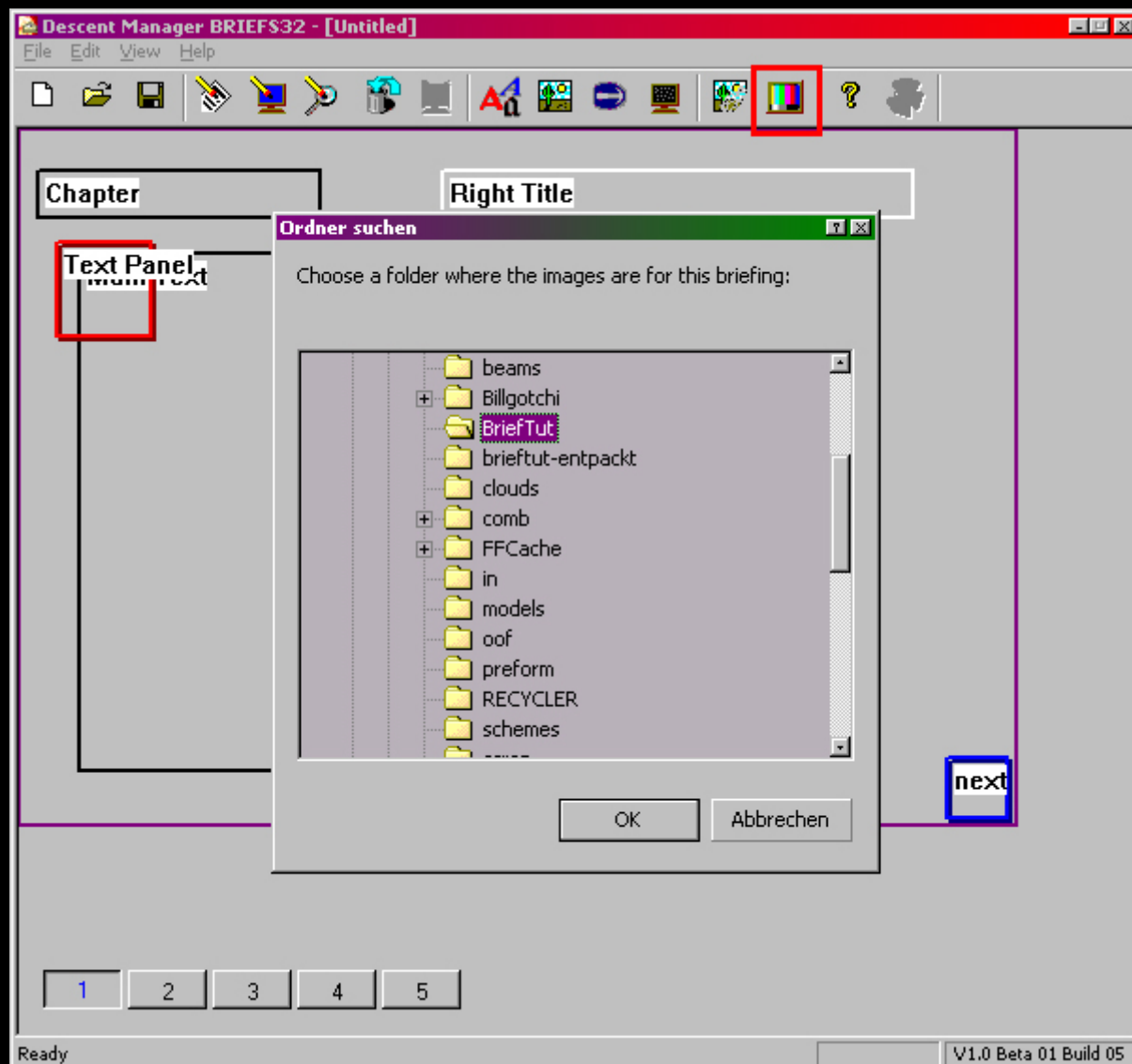
Set up Briefs32

Install it and then unzip itbrieftut.exe, either with Winrar or by running it.

Start Briefs32 and open tutorialsample.brffrom the unpacked.rar: You will be asked where the pictures for the briefing are. Unfortunately, you have to do this every time, regardless of whether you open a briefing or new created.

Do you have that set, changes into the opinion first of all nothing. You must nor the red one framed click button, then everything will be shown.

The numbered Buttons in lower area of the window represent the single ones Screens of the briefing.



It is better to always confirm the dialog boxes by clicking on the OK button; the Enter key is not always used 😊

The practice

If you hover your mouse over a button, you will see a short description; Most of it is actually quite self-explanatory.

But, who would have thought, it's not that easy

☺ who does the graphics for

If you don't want to create the briefings completely yourself, you have to do it from thed3.hog first the graphics

extract that are needed. At this point, most of you will already have a folder somewhere with the oif's from thed3.hog and this is where the next difficulty arises: unfortunately you can only specify one directory for the images. This means that you either put your pictures in this directory or you get them oif's for buttons, backgrounds and frames in the directory where you put together your briefing. I have a list of the original oif -Compiled files that are relevant for a briefing. This is the 'infrastructure' of a briefing, i.e. the frames, buttons, etc. mentioned above. I hope it is complete

Back.if
BackFocus.ogf
BackGlow.ogf
BackGlowFocus.ogf
bluepanel.ogf
bluepanel2.ogf
bluepanel3.ogf
botarrow.ogf
botarrowglow.ogf
botarrowglowleft.ogf
botarrowleft.ogf
BriefingLayout.ogf
BriefingLayout_GiantTitle.ogf
BriefingLayout_GiantTitle2.ogf
briefinglayout_largealpha.ogf
briefinglayout_largealpha.ogf
Briefsideswoosh.ogf
Forward.if
ForwardFocus.ogf
ForwardGlow.ogf
ForwardGlowFocus.ogf
Quit if necessary
QuitFocus.ogf
Quitglow.ogf
QuitglowFocus.ogf
recon.ogf
reconglow.ogf
robots.ogf
robotsglow.ogf
Secobjlayout.ogf

The sample file included with Briefs32 only uses self-created images.

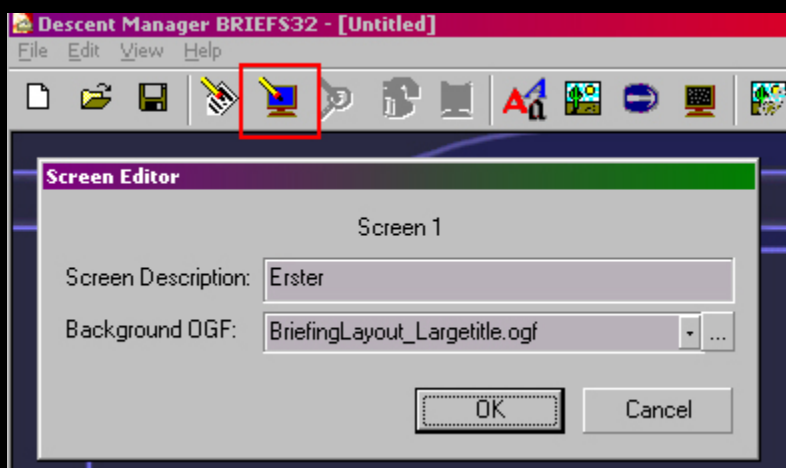
The graphics used

These must be available and you create them tga's with an alpha channel, where you make roundings or similar transparent - especially with buttons - so that the background is not cut off at the graphic border. But you can also use 'normal' ogf images.

You have 530 by 370 pixels of space to design your briefings, this is because the briefing is embedded in the surface of D3 and the frame also needs its space.

Set background

First of all, you need a background in which to put the briefing; You set this by clicking on the button framed in red and entering a title and a background graphic in the dialog box that appears.

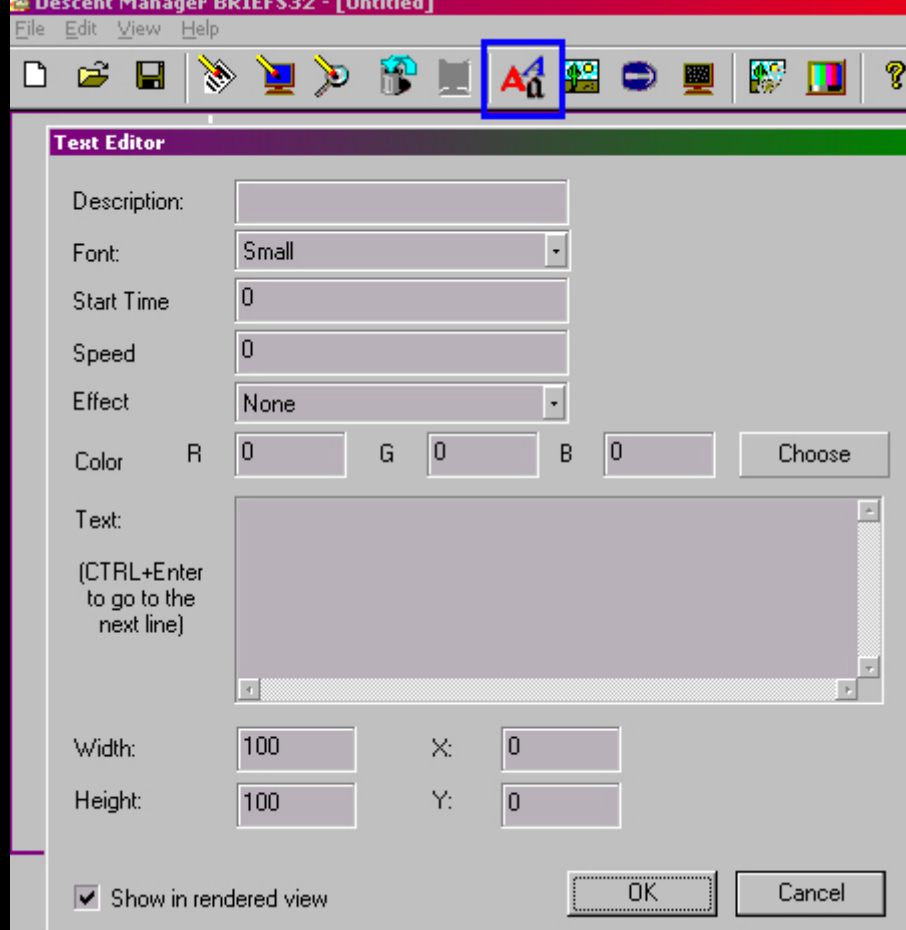


Insert texts

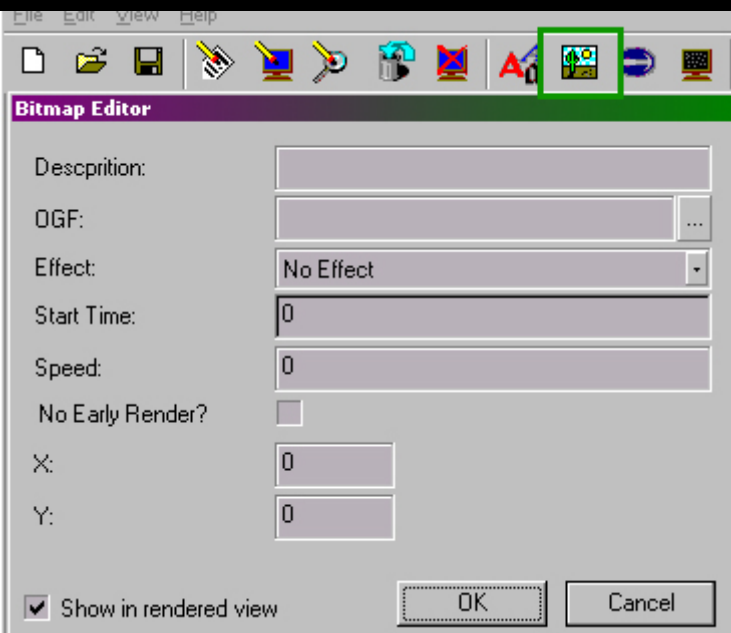
By clicking on the blue-framed button you create a text field. Here you can set a few things in addition to the actual text, the important thing is width/height. This can only be set here, in contrast to the position, which can also be set using drag & drop.

In addition to formatting the text, you also specify here

(break down individual parameters)



Install graphics



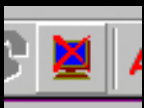
You click the button framed in green and you get the following dialog box:

Under OGF: you not only have to select the file using the button with the three dots, but you also have to enter the file name of your image by hand - otherwise it won't work!

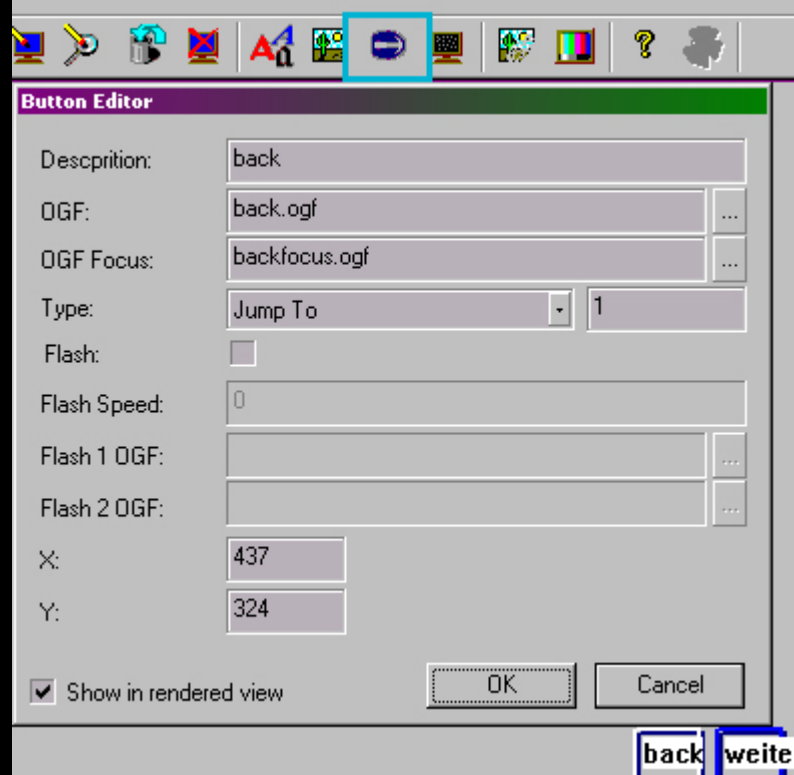
When the graphic is loaded, the frame is adjusted to the size of the image.

New page

If you are satisfied with the layout of the first briefing page, add a new 'screen' using the button shown on the right, and again there are pitfalls... You should be clear about what you want, because in Briefs32 you can only add the last page! However, you can delete it



Of course, every page (button on the left) is different, but its content is then gone; So if you decide that you still want a page between screen numbers 3 and 4, pages 4 and following are lost. So, first sit down and figure out what the whole thing should actually look like.

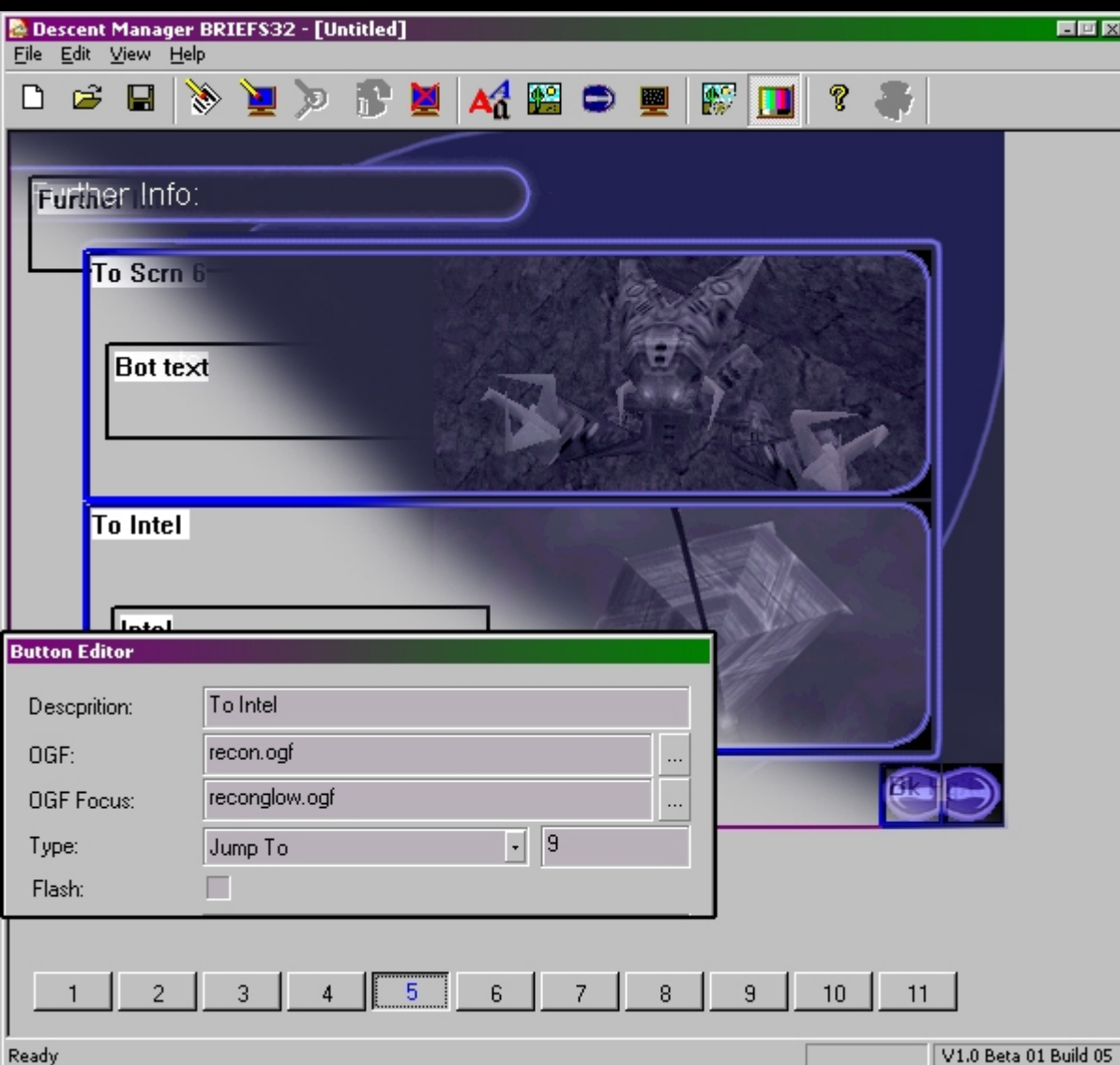


navigation

You also have to be able to move in the briefing, for this purpose you create buttons. If you click the button framed in turquoise...

(explain parameters)

Under Type you specify where the button leads. You have to have a 'back' functionJump To realize, you then simply refer to the previous page.

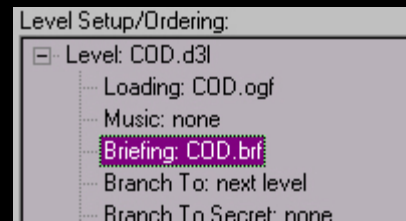


The mission relevant information in the originals Briefings are via buttons structured, so that the Player targeted select something can.

Once the briefing is complete, there is only one thing missing:

Include briefing in the level

It's incredibly easy. First of all, you have to add the briefing files to the file list in the quick test.brfincluding all associated graphics so that they do not come from the original collection. then open the branch for your level in the lower area of the quick tester, right click onBriefing: none, choosesEditand then choose your briefing. Do you have that?.brfnot in the file list, you can't set anything here!



Well, that's pretty much it.

The trick now is to use what is available, so get to work



Back to the overview of section K

129 - Ship choice in single player <>

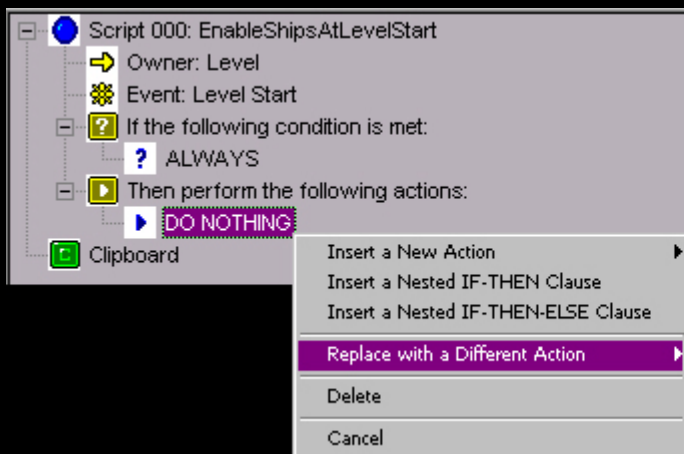
Ragil Ral

Of course, it's a big advantage if you can cruise around the area with your preferred ship. How to make this possible for a level is now described here. The suggestion came from the discussion in this thread on the Descentforum:

www.descentforum.de/forum/viewtopic.php?t=4261

1. Prepare the level

Anyone who has already rummaged around a bit in the GAM pages for ships will love the flag Allow by default to be noticed; This flag is only set for PyroGL. In order for a player ship to be selected, it must first be approved, and this is done via Dallas. The script must be executed at the start of the level, so it incorporates the following construct:



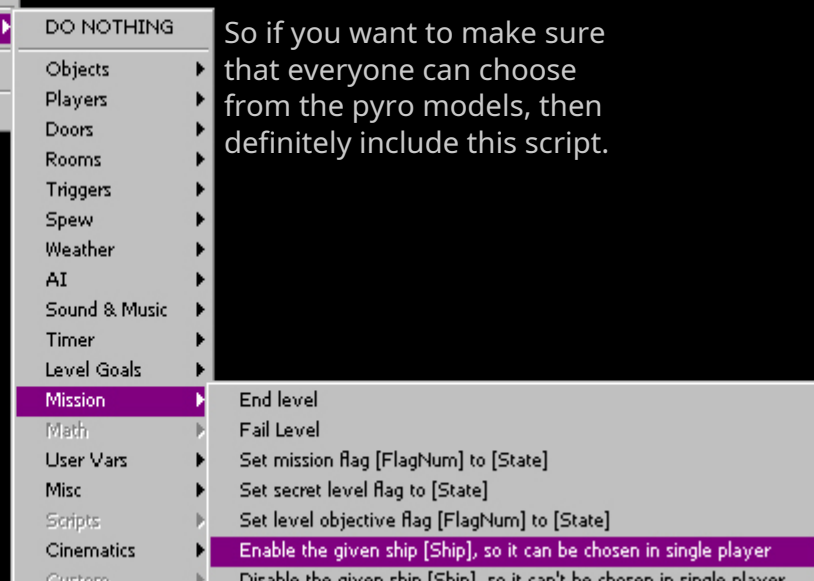
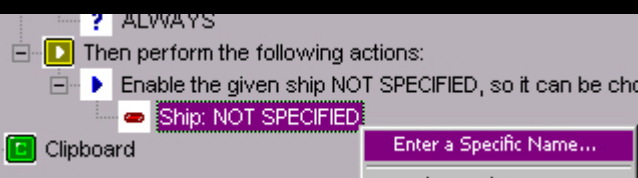
This tells D3 that a specific ship is selectable.

However, if a player has already completed the original levels, they can choose from the different models without explicitly allowing ships.

So if you want to make sure that everyone can choose from the pyro models, then definitely include this script.

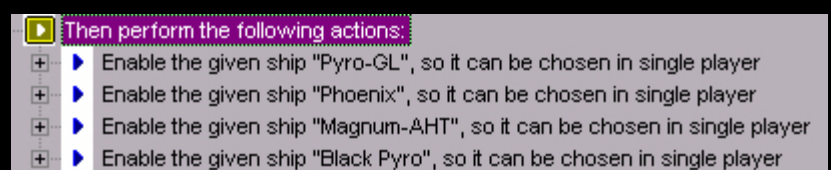
For this action you have to specify which ship is meant as it is not possible to choose from a list.

To do this, open the action and right-click on the part where the ship is NOT SPECIFIED is set:



Now write the name of the ship there, as it appears in the D3 game file, with upper/lower case.

For each selectable ship you need a corresponding action, which looks like this:



If you also want to have the Black Pyro inside, you have to integrate it first; more on that...later.

2. Activate ship selection

If you have saved/compiled the DALLAS script and now start the level, the ship selection will not be triggered; A briefing is necessary for this. In a single player mission, a briefing is necessary anyway, unless the mission is so simple (that doesn't necessarily mean easy) that a briefing is not necessary to solve the mission. But since you need one to trigger the ship selection, you have to fool D3.

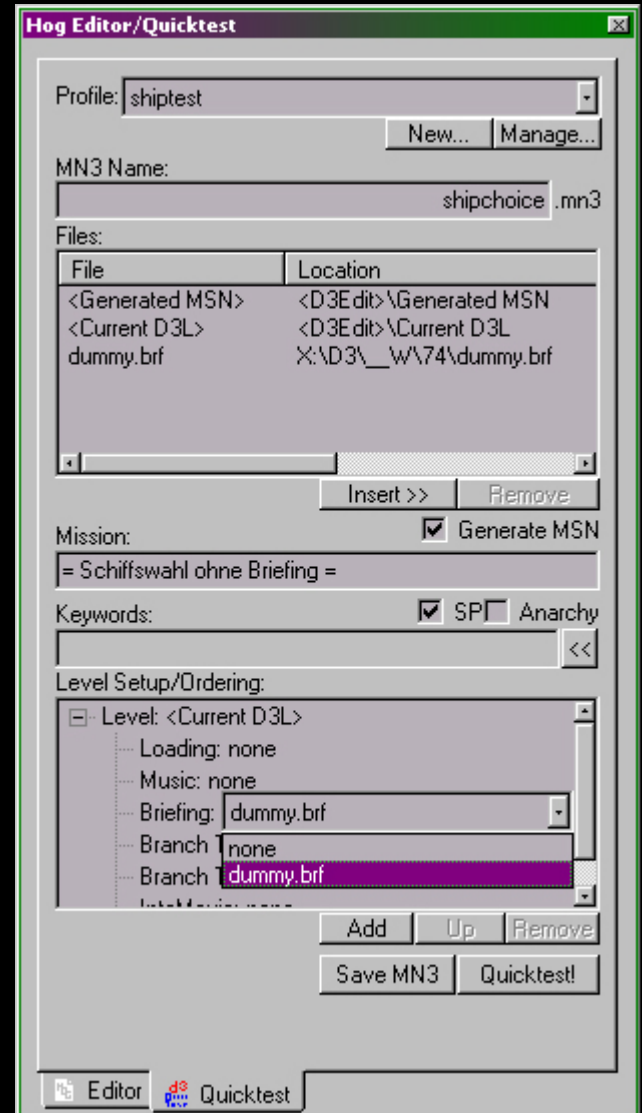
To do this, simply create an empty text file and name it with the correct ending. This.brfyou then bind it in a: ■

The fact that the briefing file has a size of zero doesn't bother D3.

At the start of the level, the briefing initialization screen appears, and if you then press a button to move forward, you can select the ships for which you made the selection possible in DALLAS; at least if you haven't already unlocked the ships through the original missions.

That was it.

Back to the overview of section K



Section L–Masterclass

Anyone who has gotten this far will appreciate the following information

130	Optimize doors	About dealing with defective doors	Ragil Ral	442
131	Dynamic light	Broken lights!	Ragil Ral	447
132	Group files	CopyPaste on a large scale	Ragil Ral	450
133	Customs, again	. . . but final	Ragil Ral	454
134	Building with objects	This makes it easier	Ragil Ral	455
135	Mercenary bots use	Bees and Troopers	Ragil Ral	460
129	Choice of ship in	Single player with your favorite ship	Ragil Ral	439
136	Single player Scripting & Matcen	On events from Matcenters via Script react toprog	Ragil Ral	463
137	Some rooms scale	If only a part can be enlarged should	Ragil Ral	467
138	Compounds Objects	Attach objects to each other	Ragil Ral	469
139	Recycling	Work more economically	Ragil Ral	475
140	Open Mercenary levels	How did they...?	Ragil Ral	478

130 - Optimize doors

Ragil Ral

You now know the problem: if you install a door somewhere, the face where it is attached is completely cut up. Although minimal new faces are created, there are an unnecessary number of verts. Furthermore, it is not easy to properly texture a door space because the door object usually blocks your view.

Not to mention that adding stock doors usually introduces T-joints and other niceties into the level... so what do you do?

Think beforehand

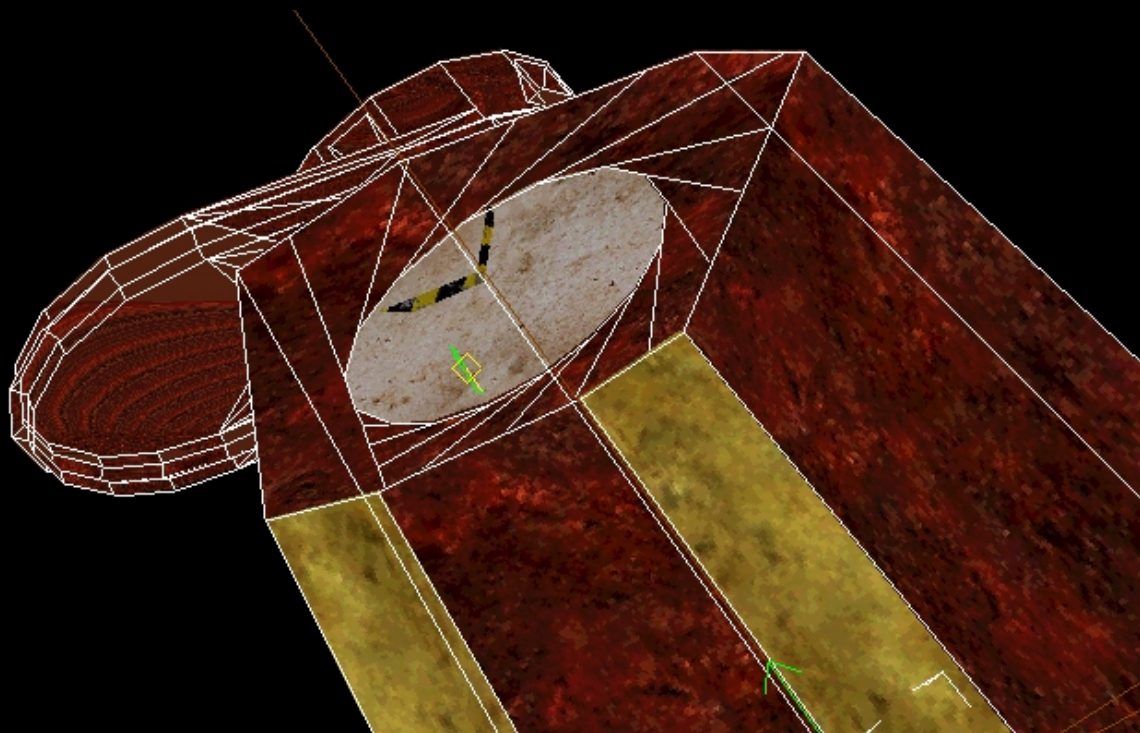
... is of course always helpful. If you want a door somewhere in the level, align the geometry or architecture accordingly. Sirian's method of enclosing faces to which doors are attached in such a way that the cutting during insertion does not influence the surroundings too much is always a good idea, but requires that you know in advance where you are going to install a door. But often planning is just replacing chance with error...

In retrospect

Of course, it can always happen that you only realize in the course of building the level that you still need a door somewhere, be it because you forgot about it while planning or because you build it improvised. There are two possibilities:

- Thanks to the face cut function, you can cut a polygon where a door should go before inserting it in such a way that the fragmentation when attaching the door does not completely transform the surrounding area.
- If you have simply adapted the door, you have to optimize the faces that surround it, so to speak, put them together; How to solve it depends on the door. The more verts the front face of an installed door has, the more 'splinters' result when inserted; Think about the 'PTMC Covert H' door, which has eight edges on the FF, or the Secret Doors, which are quite round and even have 16 edges. I will use this door as an example because it causes major problems when installing.

Begins with **Ctrl-N** and creates a standard level with standard space. Enlarges this standard space a little. Then insert a 'seoulsecretdoorround' (it's exactly the last one in the list). You will then get something like this:



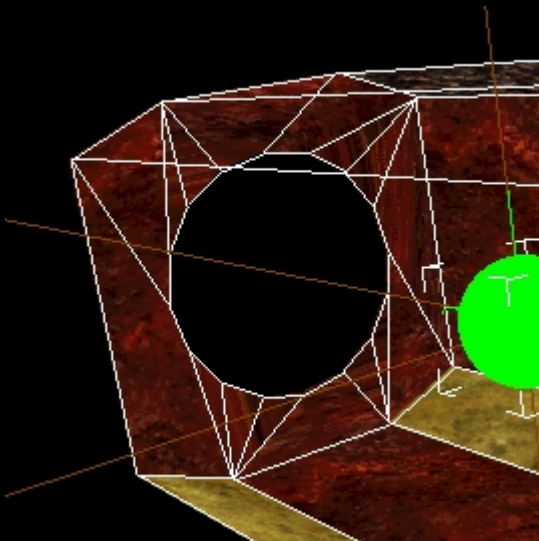
Inserting the door completely shredded the face it was attached to. Then leads
Verify Minefrom: You get 16 duplicate or unused verts and three T-joints, the shell is of course
criticized as 'bad'. The UV values of the door space texturing are *somehow*.

1.: Optimization of the cultivation site

Again you have two options here, you can either use the Snap Vert function to combine the verts, which however destroys the texture alignment, or you can split the faces using the Split Current Face function and then snap them together **Ctrl-Shift-LClick** together again (optimized), the texture alignment is retained if you had already finished texturing the front room. Unfortunately, the latter is not always possible and is also more time-consuming.

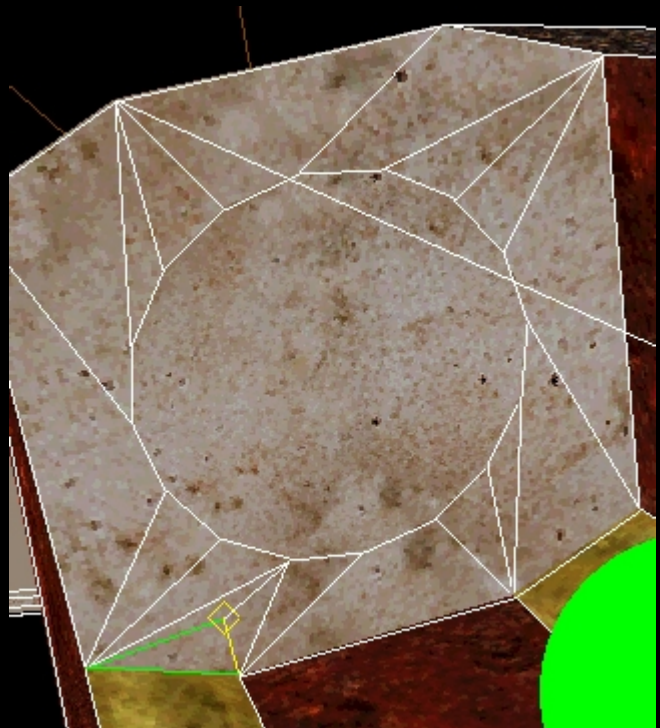
Summarize Verts:

Let the verts of the splitter faces snap to the verts of the adjacent faces and do Remove Extra Verts. But as you can see, the UV values are now quite distorted.



Splitting and reorganizing faces:

This requires a bit of brainpower, but preserves the texture alignment. You have to think about which direction you start from. After the work is done, run the Remove Verts magnet.

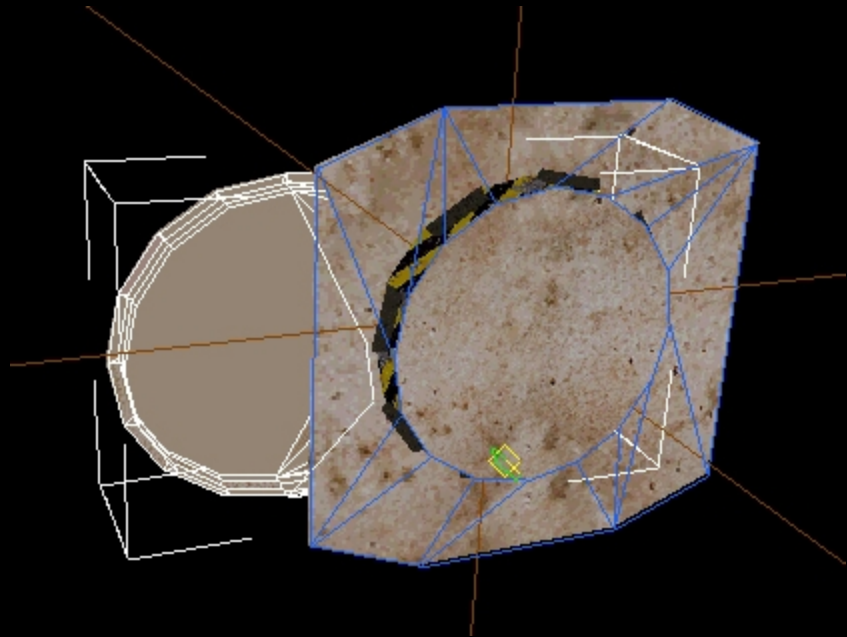
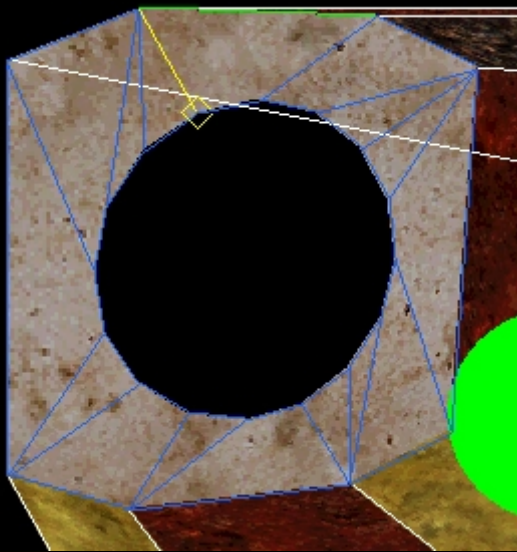


As you can see, with the second method you end up at a dead point; the area around the current face can no longer be further optimized without snap verts. However, since most of the surrounding faces are already optimized, you can easily work with snap verts a few times here **Ctrl-Lclick** the UV can be taken over by the neighbors.

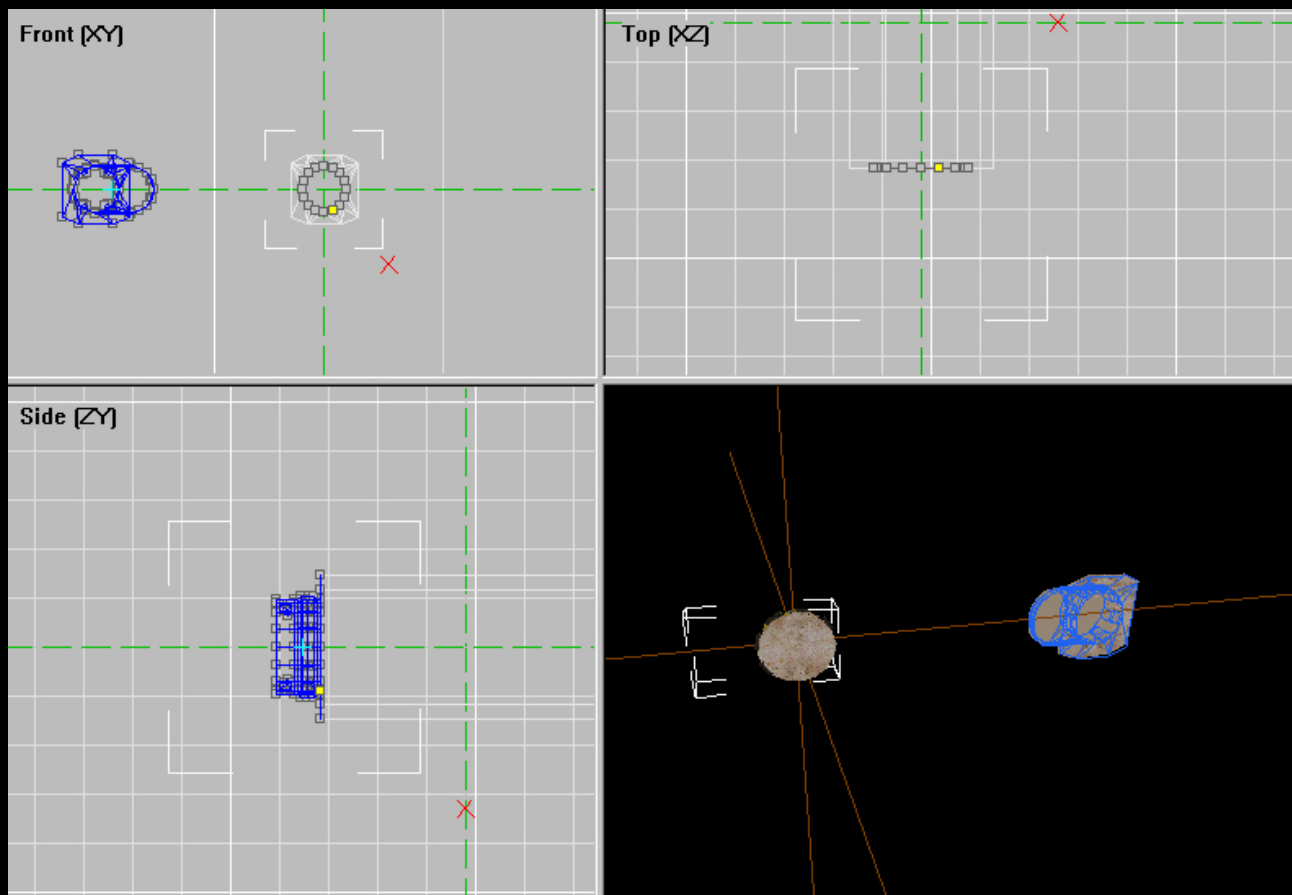
2.: Optimization & texture alignment of the door space

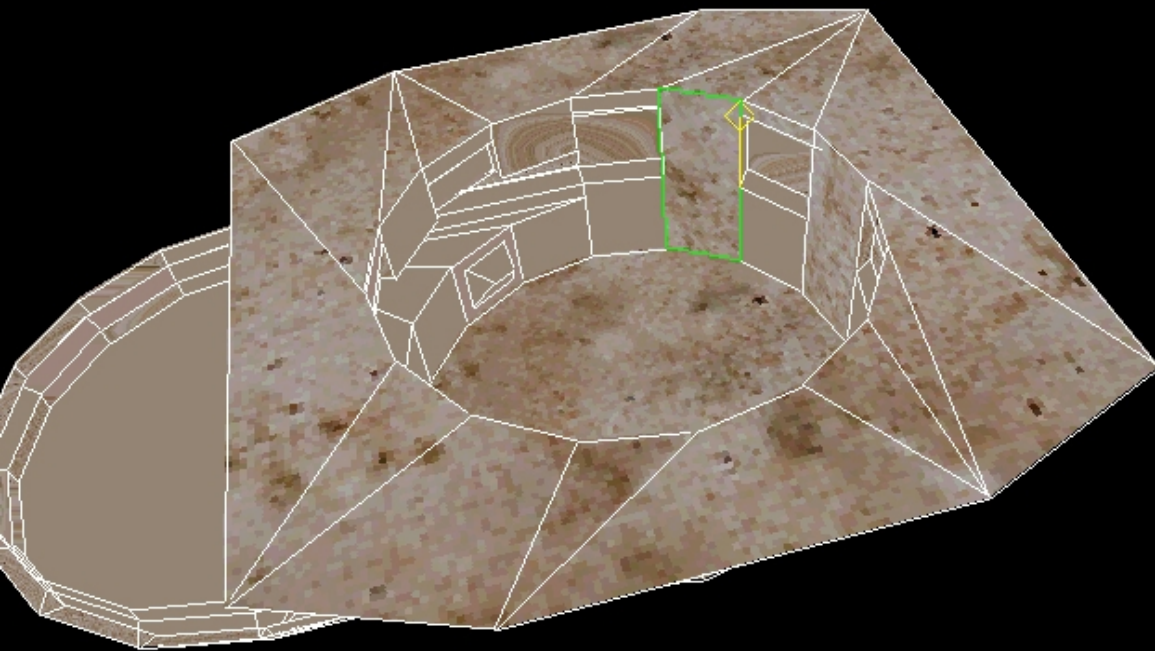
First you would be in the world view **Ctrl-Lclick** bring the orientation of a face into the door space. Here - in the world view - you would have to zoom in so that you are 'inside' the door object in order to reach one of the adjacent faces. But I find that impractical, which is why I prefer a different method - you simply do the following:

a) Go into the anteroom and copy the - aligned - environmental faces of the door. Then go into the door space to be aligned and add the faces there **Ctrl-Shift-Va**. Then Remove Extra verts out.



b) You now have the faces with the desired UV orientation in the door space, but once again the door object is in the way. Now mark all faces except the portal(s), cut them out and add them back **Ctrl-Shift-V** in place. Then go to one of the three ortho views and set the grid **7** (In the number row in the alpha block) to 100 units and move the faces you just inserted in any direction:





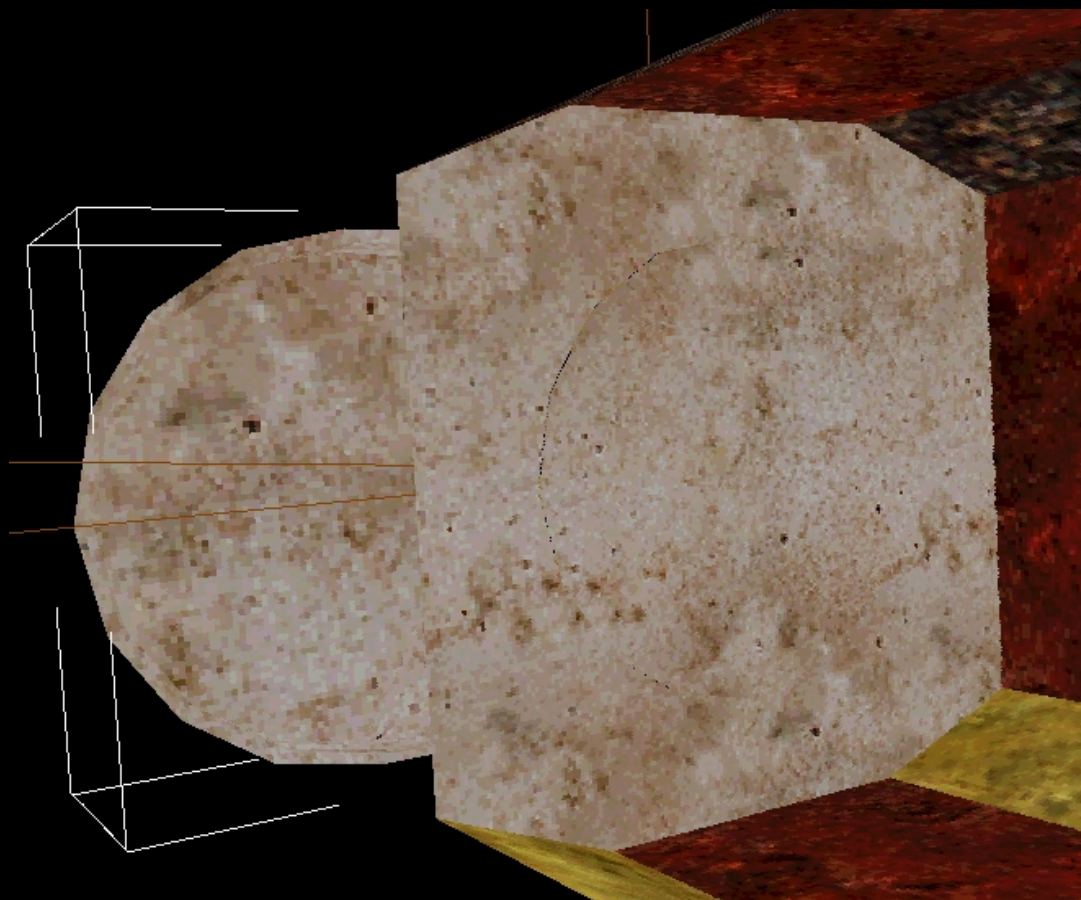
can you
relaxed
alignment
textures
hmm.

c) When you're done, delete the faces that come from the anteroom. The door shell from the example here still needs to be repaired, as it still contains T-joints. Optimization is also required. So removes all imperfections.

d) Then mark the entire face group (forget the internals not if you moved them too) and move them back the 100 units. Thereafter Remove again Perform extra verts. Verify should no longer report errors for the door space.

That was it.

If you build your own doors, it's a good idea - especially if the front face has a lot of verts - to make a frame. (compare Multiple Layers of Sirian)



As some of you may have already discovered, some of the 'included' doors in D3 are, well, ahem, let's just say not entirely flawless. If you don't want to fix the same door every time you install it, you can do the following:

First, extract the affected door and convert it into one.`orfb`ack.

Then repairs the door and creates one again from the repaired door.`oorf`; But you have to make sure that everything is exactly the same as with the original door, but you can open more than one instance of OOFedit so that shouldn't be that difficult.

You place the repaired door in your `model-Directory` (see Customs, again), and from now on instead of the door from the `hog` used the ones you repaired - every time you take the door out of the `jects`.

[Back to the overview of section L](#)

131 - Dynamic light

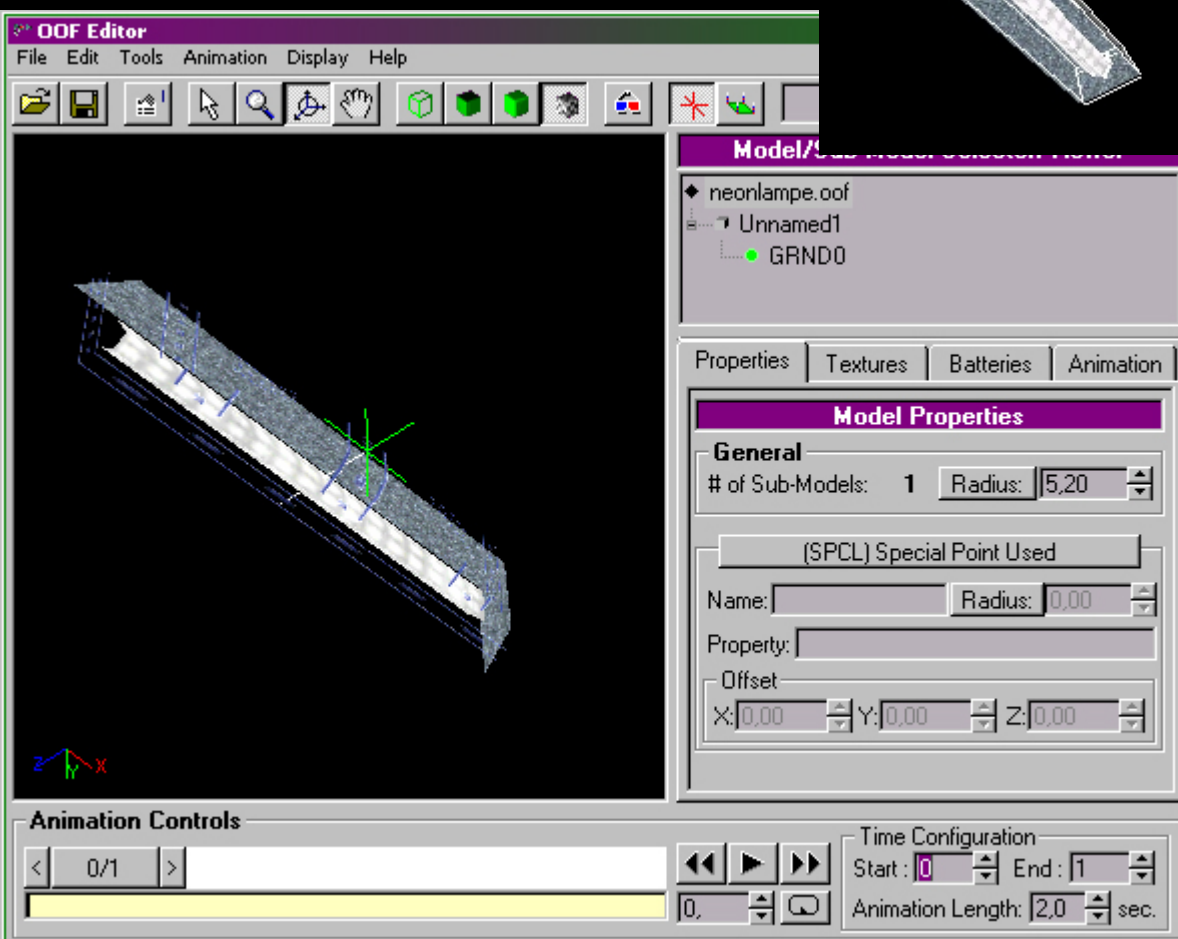
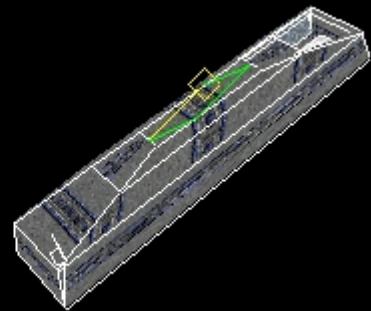
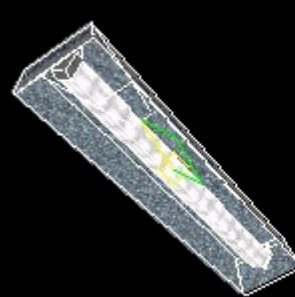
Ragil Ral

Exciting light and shadow games can be organized inside the mines; but the outside world is only illuminated by the satellites. If you try to put lights somewhere on the terrain, they will not emit any light and will not cast any shadows unless the terrain radiosity is 100%, which will usually be the case.

But there is a workaround!

Descent 3 allows objects to emit light. This is done in the level.gamfixed; So you need an object that should glow and a levelgamto determine the lighting conditions of the object.

So first create a light object; builds it in D3Edit, inserts a face that should then become the GRND (the current face in the right picture), and converts it with the OOF

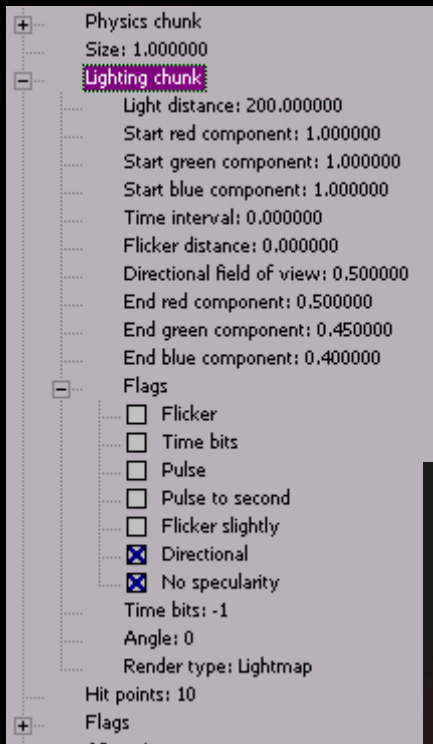


Editor in one object around.

little more complicated.

Now it will be one

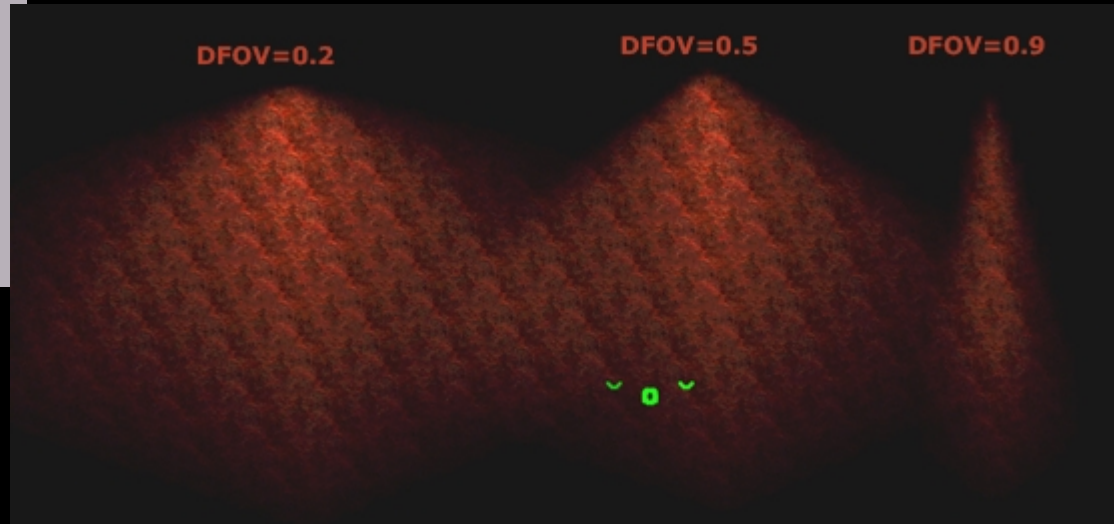
Every object has in its.gam-Entry a 'Lighting Chunk', which determines the lighting conditions. All relevant values are stated there, such as the range of the light, color and whether it shines in a certain direction. The origin of the light is the point at which the axes originate in D3Edit. The most important parameters are:



Light distance: Range of light in units.

Start red/green/blue component: light color at origin; in proportions, between 0 and 1.

Directional field of view: If the light should have a direction, that area is specified here
{toprog - find out what exactly is stated here. Papacat speaks of cosin.}

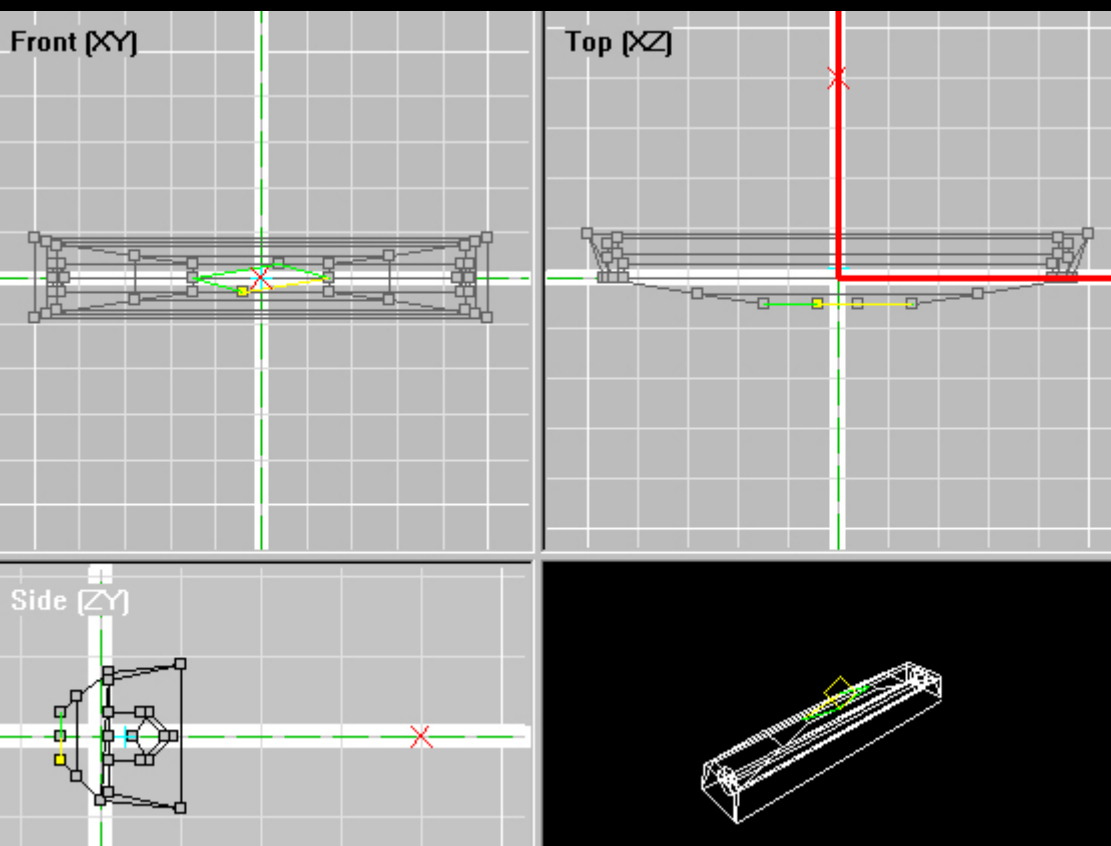


. If you specify something here, the flag must be set.

As you can see on the right, the edge of the light cone becomes softer with smaller DFOV values.

End red/green/blue component: light color at the end of the Light distance. These values must also be stated in proportions.

Flags:



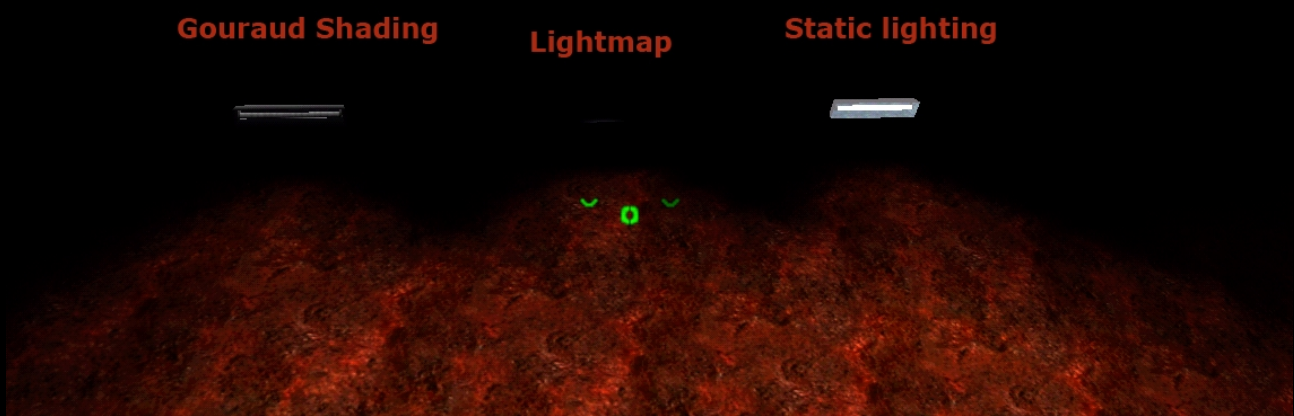
Flicker: Makes the light flicker.

Directional: If you a directional light
If you want it, you have to set this flag. The Direction of radiation Light is parallel to the Z axis, from negative after positive; So you have to create your object in D3Edit accordingly align. (left is

You can set this flag under Directional field of view (so) determine what angle the light cone should have.

Pulses Let the light pulsate, you then have to go under Time intervals specify the rhythm in seconds.

Flicker slightly caused no more light to be emitted.



Render type: Specifies how the light object should be drawn in game.

I haven't dealt sufficiently with the other flags yet; Some pretty strange things happen there. If you are in Lighting Chunk If you enter something invalid, the object will either not be displayed in-game or it will not emit any light.

But you can do something completely different with dynamic light: lights that you



In the image above you can see a dynamic light on the terrain, in an area that is shaded by a hill in the terrain; left before, right after the light object is destroyed.

However, there is also a catch, because such light-casting objects are real FPS killers. I put 18 of these things in a level to try out and consistently, everywhere in the level, I had around 20...30 FPS. After I shot out all the lights there were a decent 200 or more again. However, my computer is a bit older.

Back to the overview of section L

132 - Group files

Ragil Ral

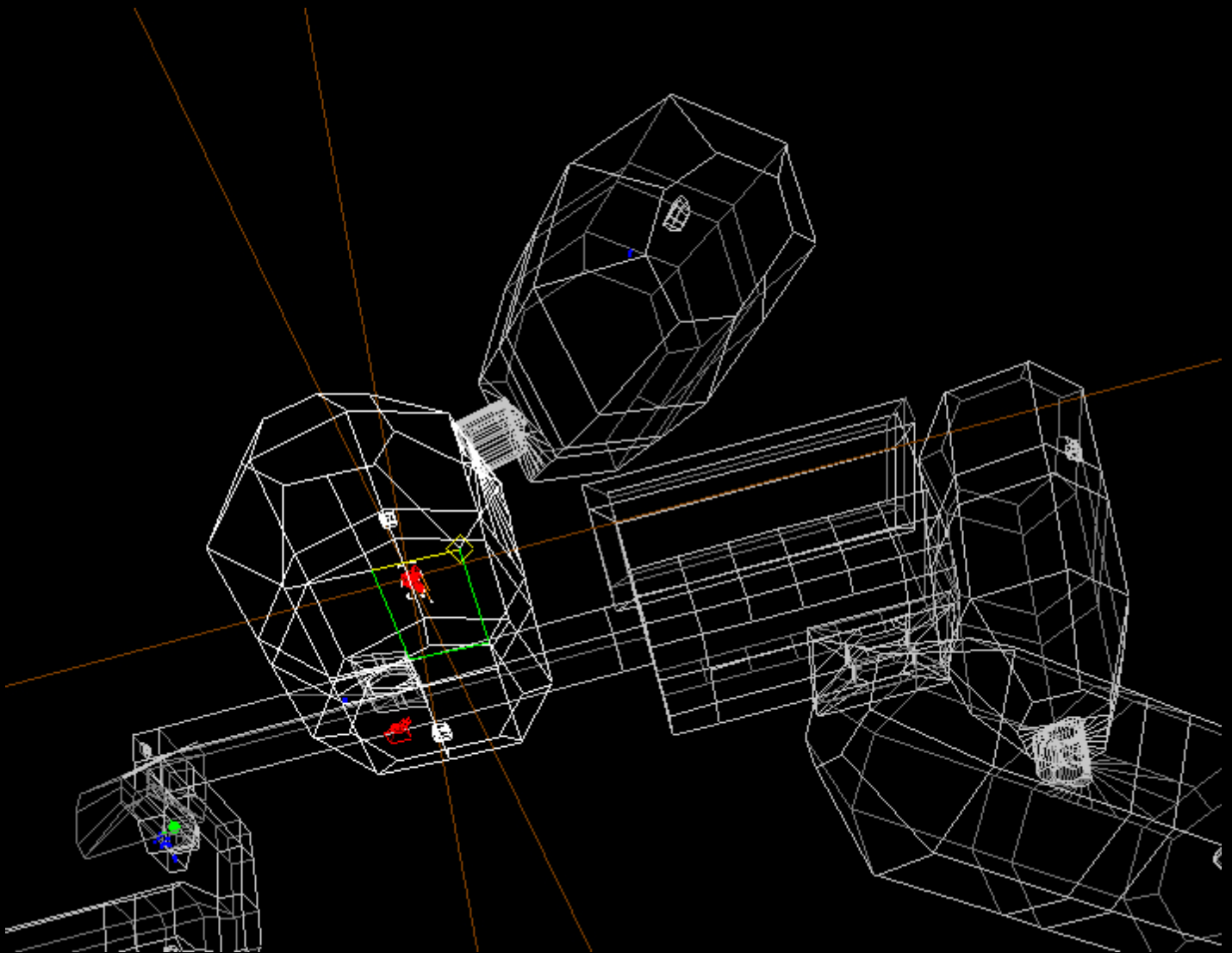
Perhaps it has already happened to you that you wanted part of the mine somewhere else or that you built it out of the Valid MP area.

With older D3Edit versions you had to delete these rooms, possibly save them beforehand because you had tinkered with them, insert them again and then insert all the objects and doors again. In this case, there is now the option to export groups of rooms to a file and attach them like a room, including all objects and doors. This is also suitable for attaching constructions from several rooms multiple times.

Sounds complicated, but Atan has done a sensational job here.

Initial situation

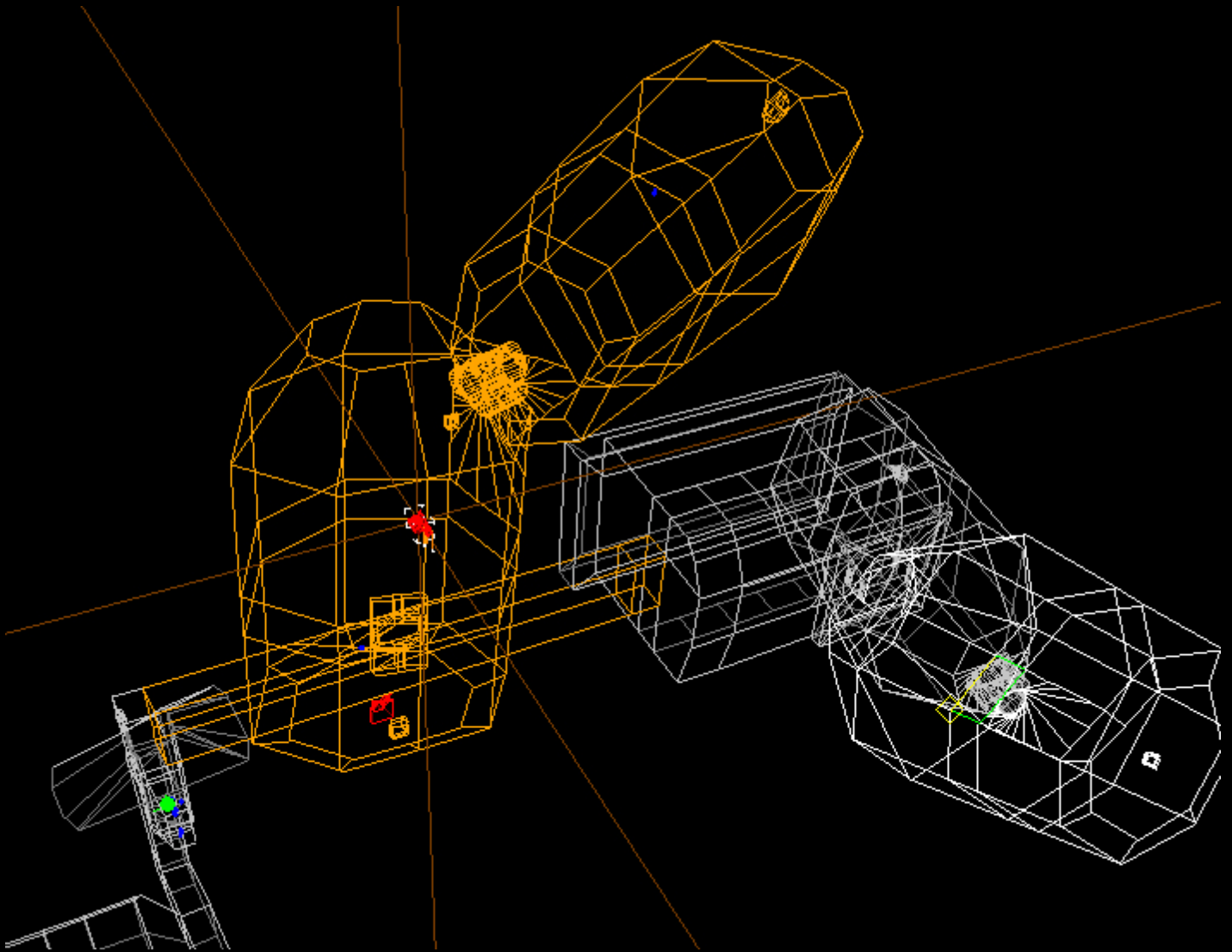
So you have a level where you have several rooms somewhere else, or want to use these rooms several times.



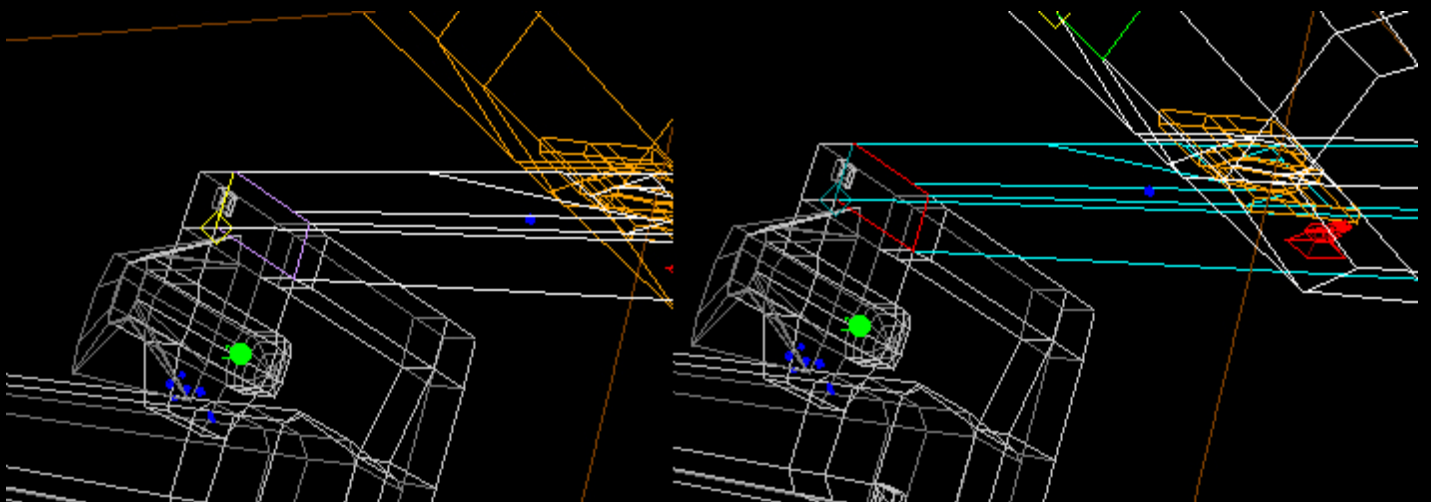
As you can see, the rooms already contain objects and are connected via doors. And now comes trick grp:

Mark the desired rooms

Stay in the world view and click on one of the rooms. Then press **Space**: the status line then says Room xx selected. Do this for all the rooms you are interested in. Selected rooms are then drawn in orange (unless a selected room is active):



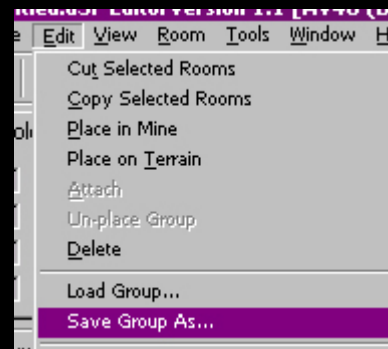
You then have to decide which face the group should be attached with. Move to this face, click on it and press - still in the world view - **M**: The status line then says Room:Face:Edge:Vertex x:x:x:x marked. This face can also be a portal.



In the picture above, the room in which the face - here a portal - was marked is active on the left, and another room on the right. Pay attention to the color coding: The room with the marked face is shown in turquoise, the marked face is shown in red if it is not the active one. Also on the right, the active room is one of the selected ones and is shown in white.

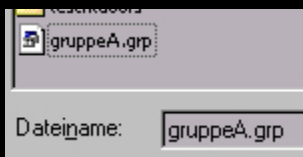
Copy or cut

Now you have to decide whether you want the rooms somewhere else or again. If you want to transplant them, press **Ctrl-X**, otherwise **Ctrl-C** - still in the World View: The status bar reports *x* rooms copied to group (if you copy them) or *x* rooms cut to group when you cut them out. Then you go **Edit -> Save Group As...** and give the thing a name, click **Save**. It will be a file with the extension *.grp* written.



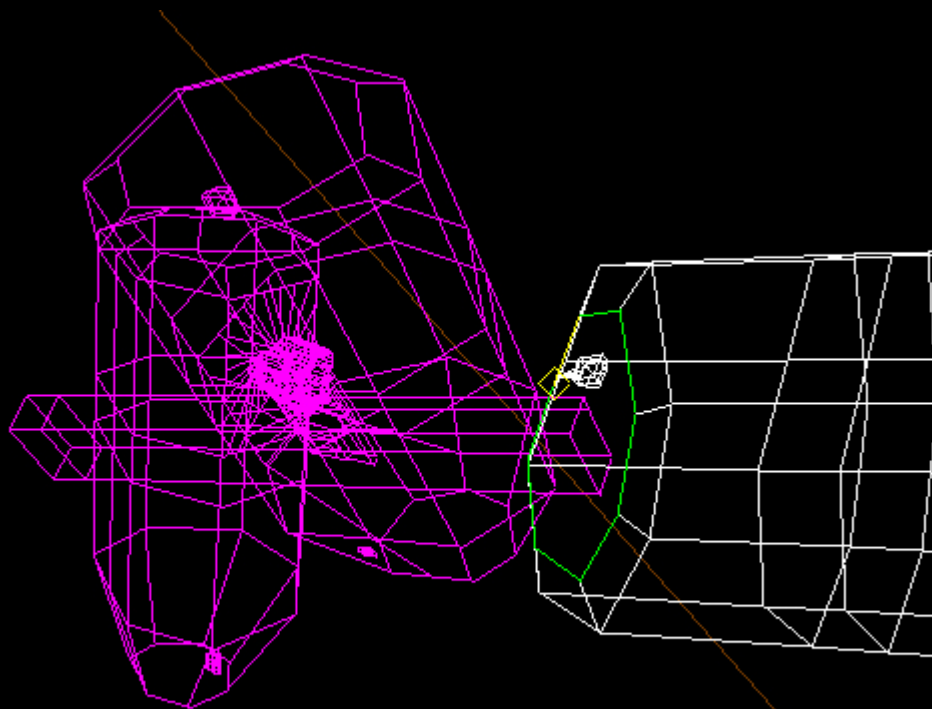
Attach the group

Here you proceed as you are used to with rooms: first do a face current where you are group wanted to have. But then you rise **Edit->Load Group...**, and select the group you just saved. But nothing is happening yet!

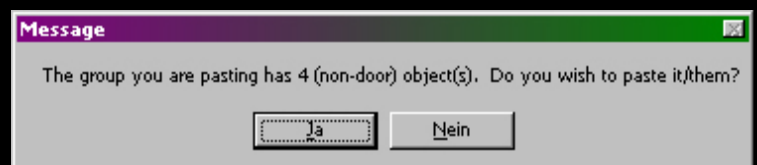


Then you go up **Edit->Place Mine** or does **Ctrl-V**, and group is sent into the mine

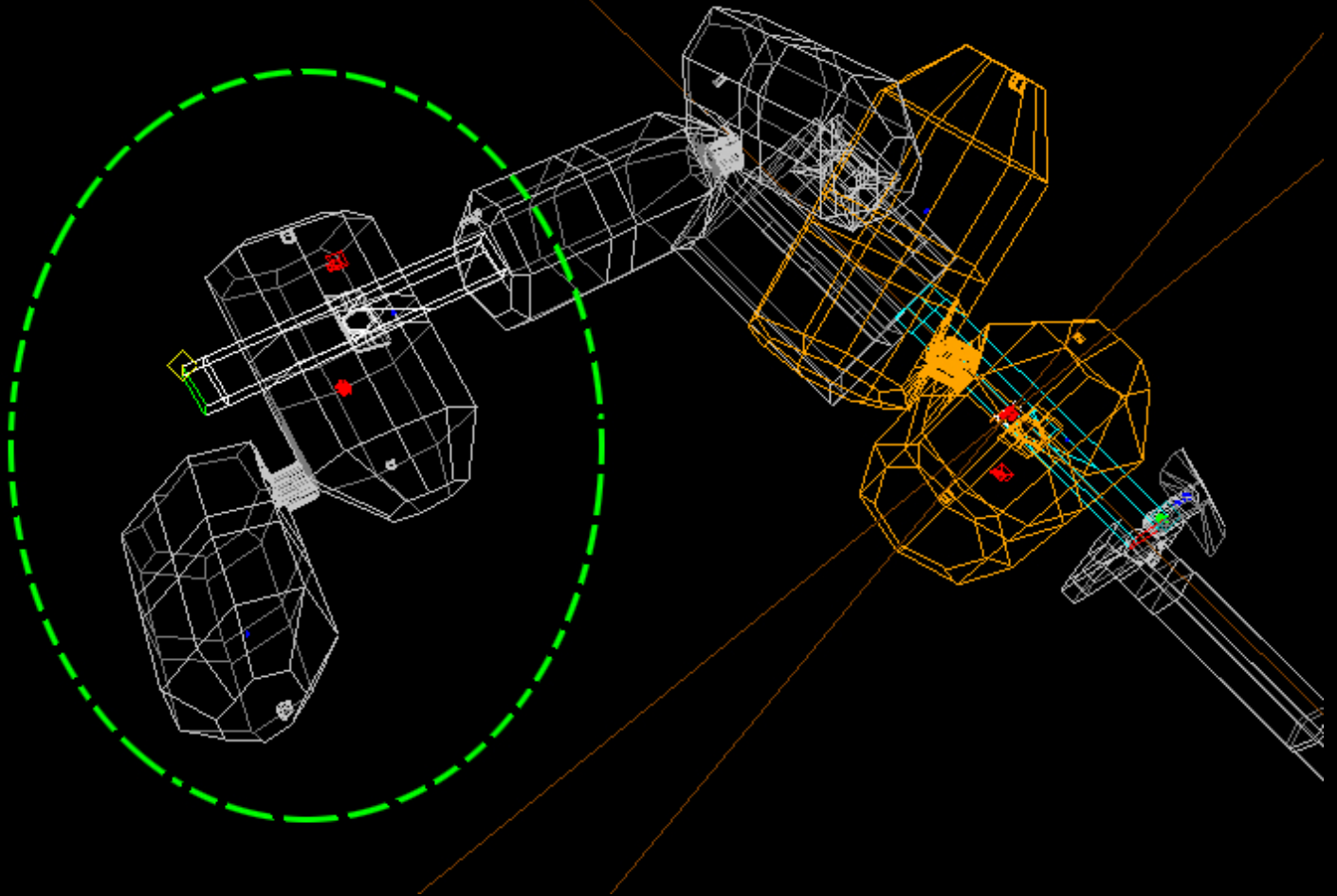
You can now position the group.



Then you do it **Edit->Attach** and - if you have objects in the group - you get this dialog:



So you can still decide whether you want to include the objects. Et voila...



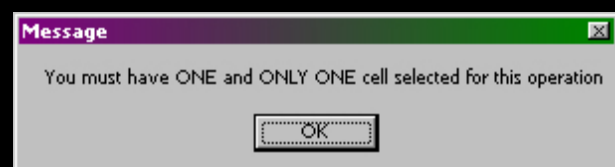
... great thing, right? Doors, objects, everything is there.

And another trick

Since the group feature allows you to cut out several rooms and then remove them from the mine, you can delete several rooms in one go this way - you don't have to save the group. However, it remains in the clipboard until you copy or cut something else.

And another one

Anyone who has looked closely will notice Edit->Place on Terrain. You can also place groups on the terrain; to do this, a terrain cell must be selected; The marked face of the group is then aligned anti-plane-parallel to the terrain cell, including the group. But make sure to only mark one terrain cell, otherwise:



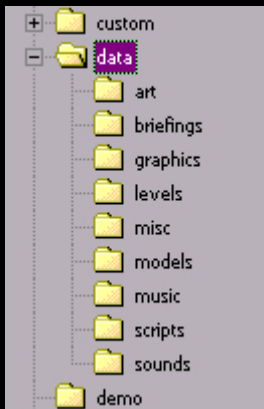
Back to the overview of section L

133 - Customs, again

Ragil Ral

Sections F and H talked a lot about custom textures and objects, their .gam's and their use is gossiped about. Several ways to handle customs have been shown, but another one - by far the most convenient - will be discussed here.

Atan discovered that the developers of Descent worked with a folder structure that we level crafters can take advantage of.



Create the following folder structure in the D3 folder:

- ?
- ...
- here you present your .oif's and .oaf's in
- ?
- here come the .oof's in
- Music files
- Dallas scripts
- come here .wav's purely for sound effects

If you put your customs there, you will always have them visible in the editor and save yourself the baggage.hog-Files. In addition, you then have a 'central collection point' and don't always have to collect everything to pack up the level. Furthermore, if you place something here that has the same name as an original D3 component, it will be used when building instead of the one that is in the d3.hog located. In this way you can, for example, repair the original doors; You grab the door out of the d3.hog, repairs it and puts it under models from - this will be used from now on. You then only have to include the repaired door with your level and don't need an extra one.gam-Entry for it.

Back to the overview of section L

134 - Building with objects

Ragil Ral

The method described here comes from Atan and saves you a lot of work.

The problem

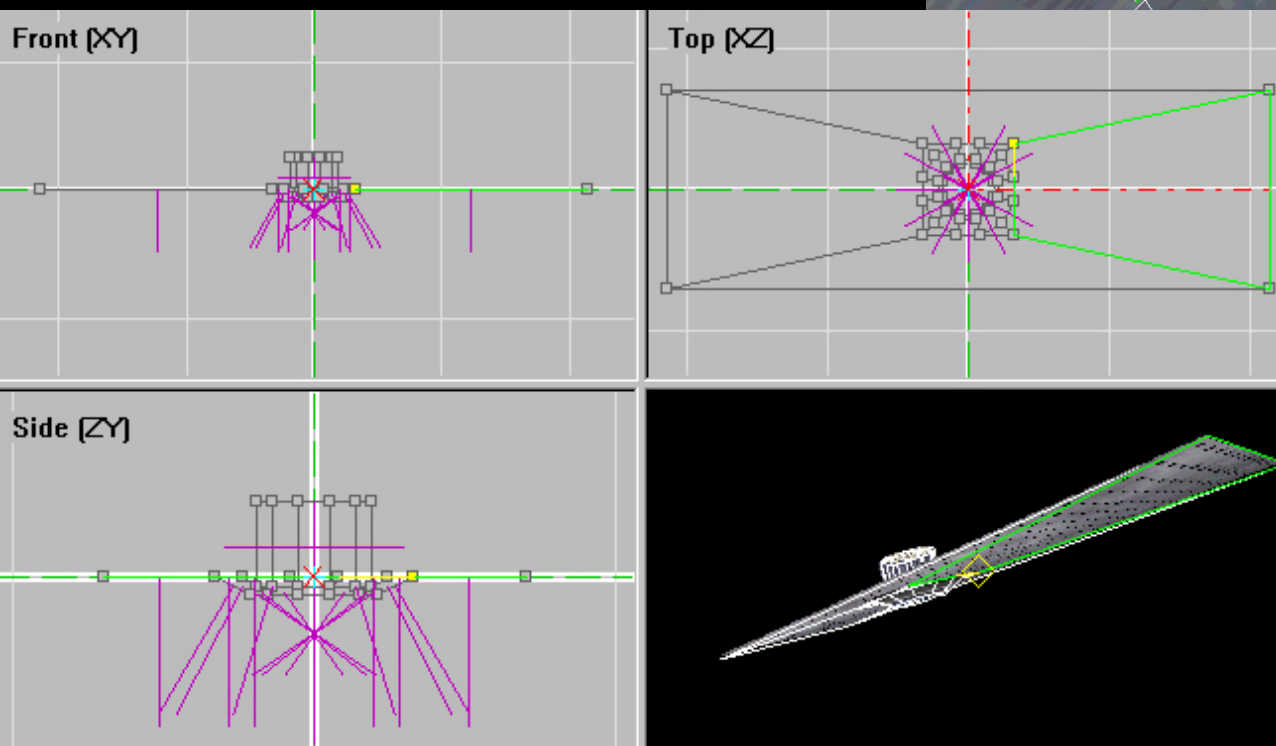
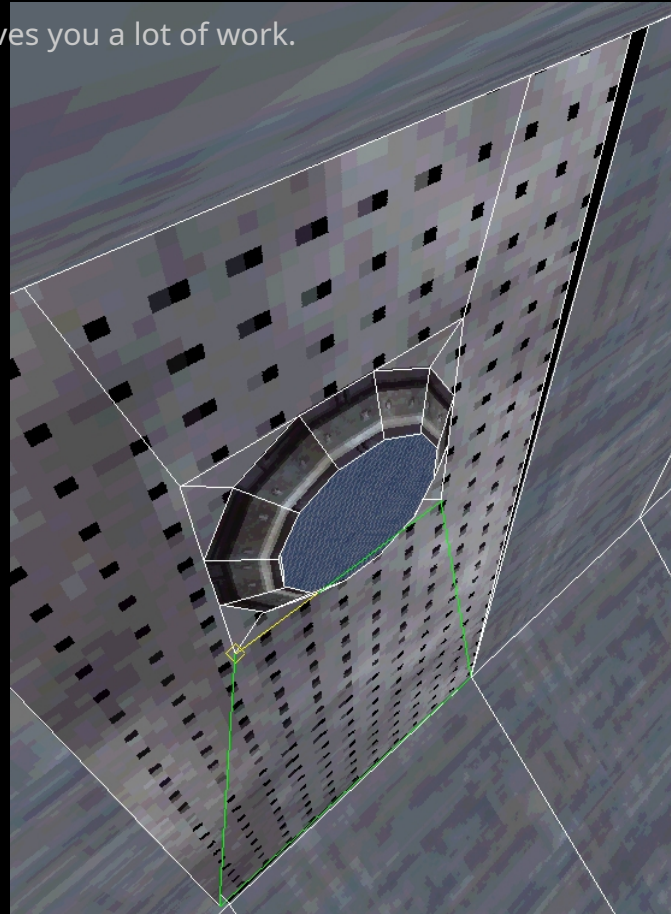
You've probably had the problem of aligning a face construct to something before, but since D3Edit performs angular operations in discrete steps - a minimum of 1.4063° - sometimes the alignment simply isn't possible. On the right you can see an example like this, the lamp cannot be fitted exactly, gaps remain. (Update!)

There is a workaround!

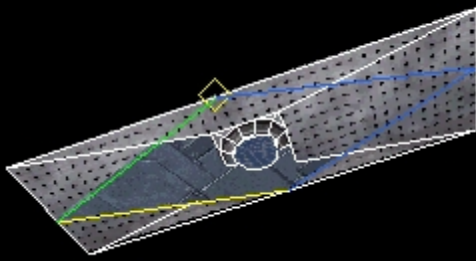
In short: The construct that you have to use more often and align again and again is transformed into an object with a ground point and can thus be used over the game integrated into a level. If it is then inserted, it is plane-parallel. Then you can merge it into the level.

The object of desire

First you create a room where you put the object:



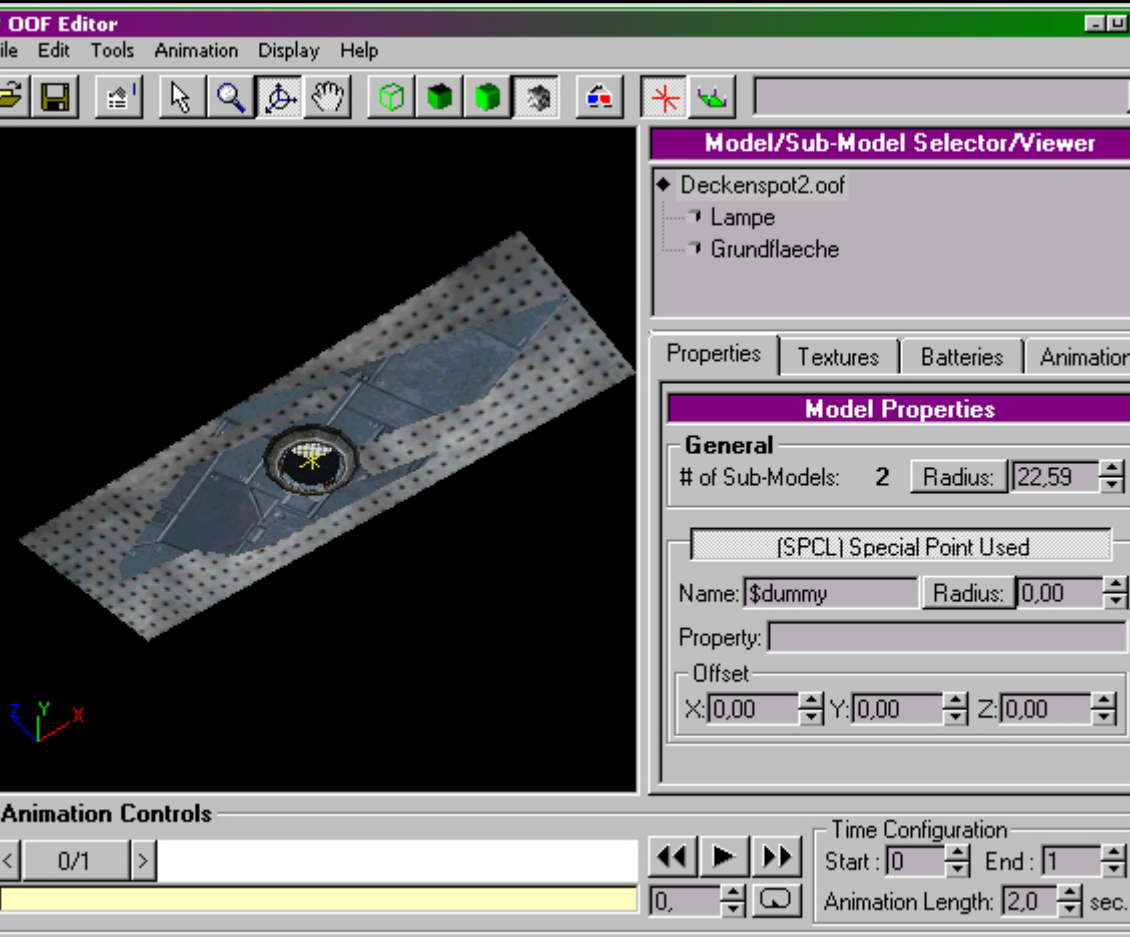
Now you need a face that will then be converted to the ground point; this is then used to align the object. For the lamp above, the four surrounding faces should be level with the one where you install them; So add a face in such a way that it corresponds to your ideas.



It doesn't matter what texture the face has. The normal determines the direction in which the object looks after installation, so it must look in the same direction as the face to which the object is to be attached or installed.

It is advantageous to make the construct as central as possible to the coordinate origin. Save the room and start OOFEdit; load your room with the object there.

From space to object



Forgive in OOFEdit I always thatSpecial Point Usedwith the names\$dummy; I've never had one without it tried 😊

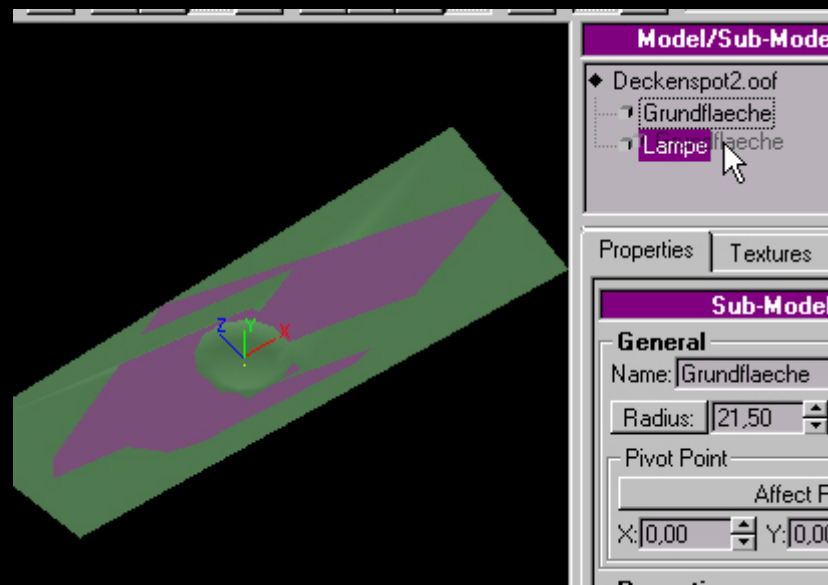
The Designation are not mandatory necessary, only serve for clarity here.

The pivot points must be set to 0 unless you want it otherwise; after merging

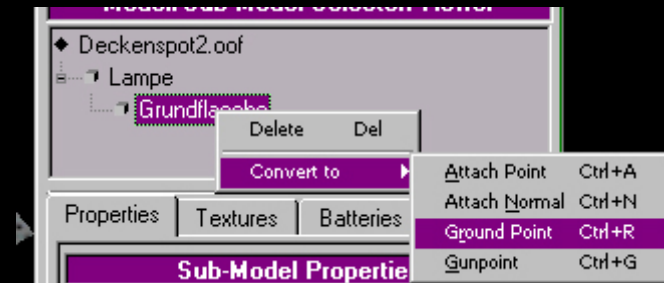
becomes the construct namely with his Pivot point to that Face aligned.

Next, make the face, which will become the base, a submodel of your construct by left-clicking it and dragging it onto the object:

You can't go wrong here



Then right click on the future base area and select Convert to->Ground Point; the face disappears and is replaced by a yellow spatial cross that has a longer white line (below): the ground point.



As you can see, the naming has also been changed. Now save the object as.oofaway.

This was one part.

In order to use the object, it must have a.gam-have entry.

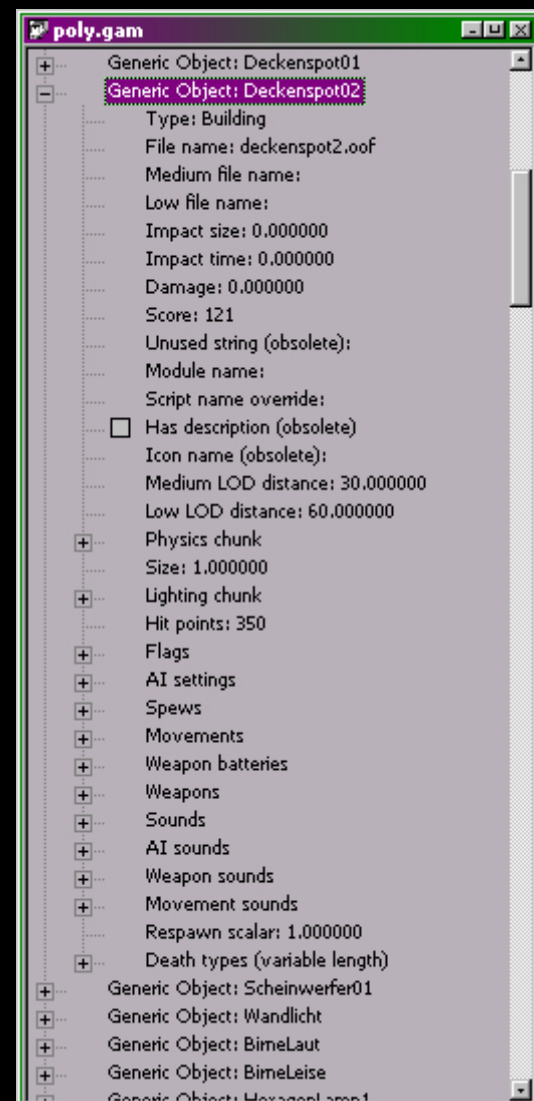
Include the object

So start the Gamtool, open or create one.gamfile, so create or copy one.gam-Entry for oneGeneric Object. Name it, wear it. oofone, you only need to set the rest if you want to use the object as such; If you're going to merge it, you can leave it as it is.

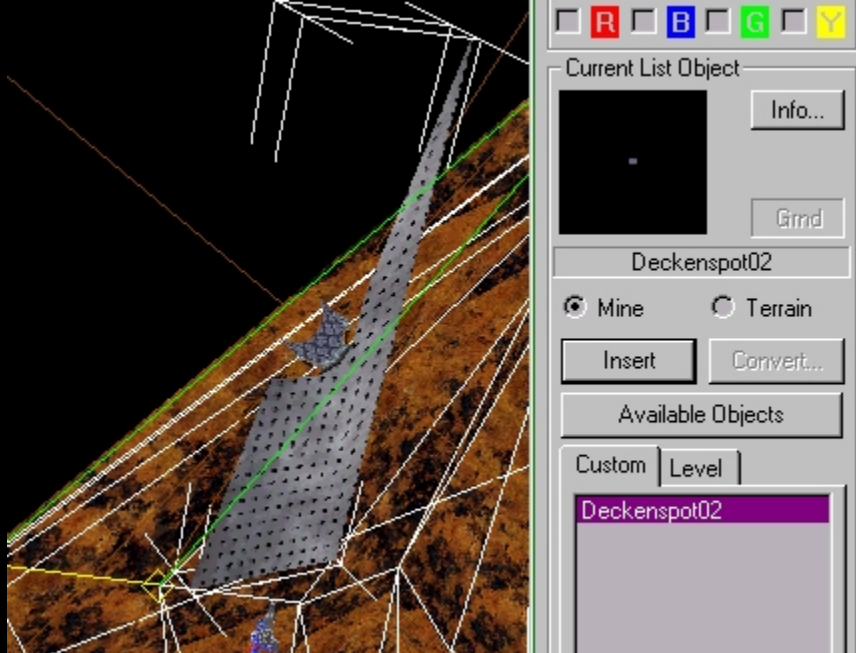
Either assign it to the TypeClutterorBuildingYes, I prefer the latter (it's supposed to be a component, plus there are fewer in the list, which means it loads faster).

To use the object properly, follow the steps necessary to see it in D3Edit, I recommend method number 133- [Customs](#), [again](#); This is where you need the object just in<descent3dir>\data\modelsput it down, done.

Then load the before opening your level.gam.



Opens the level, opens the object bar (5), pulls your object into theCustom-List, select a face where it should go and click**Insert**:

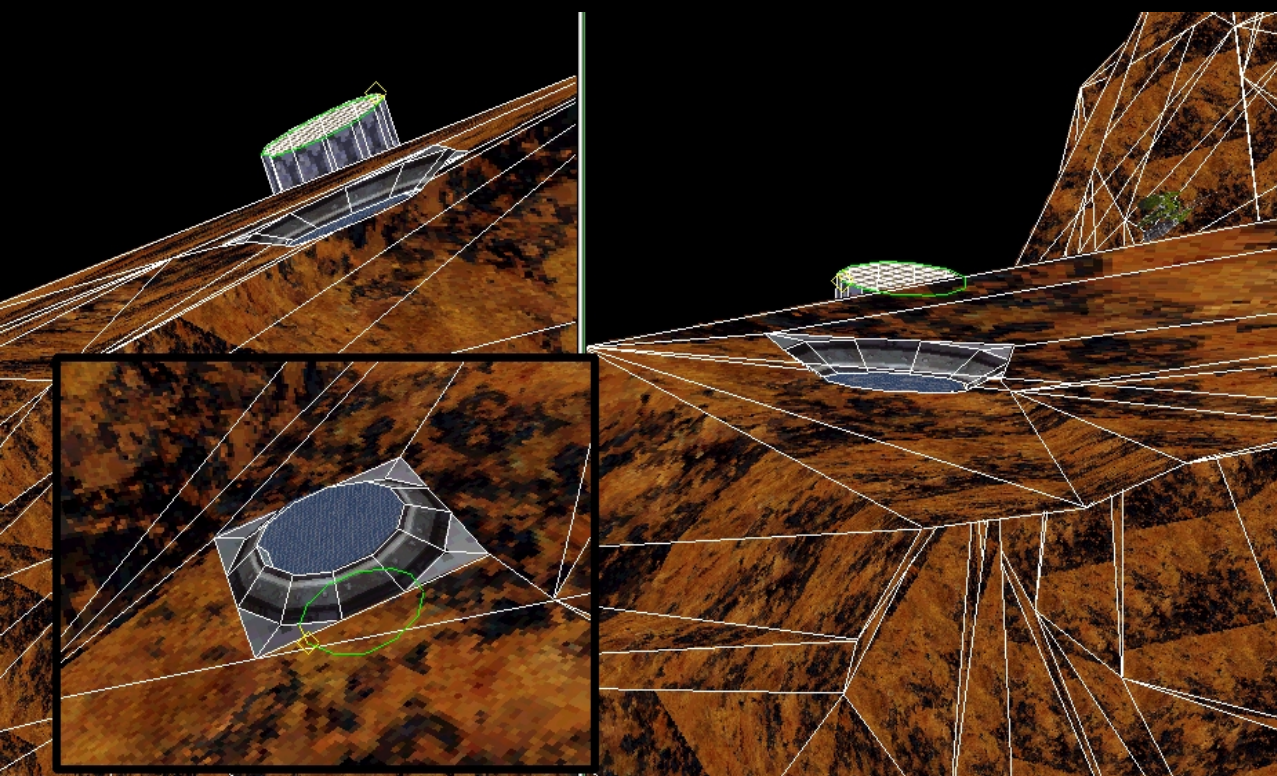
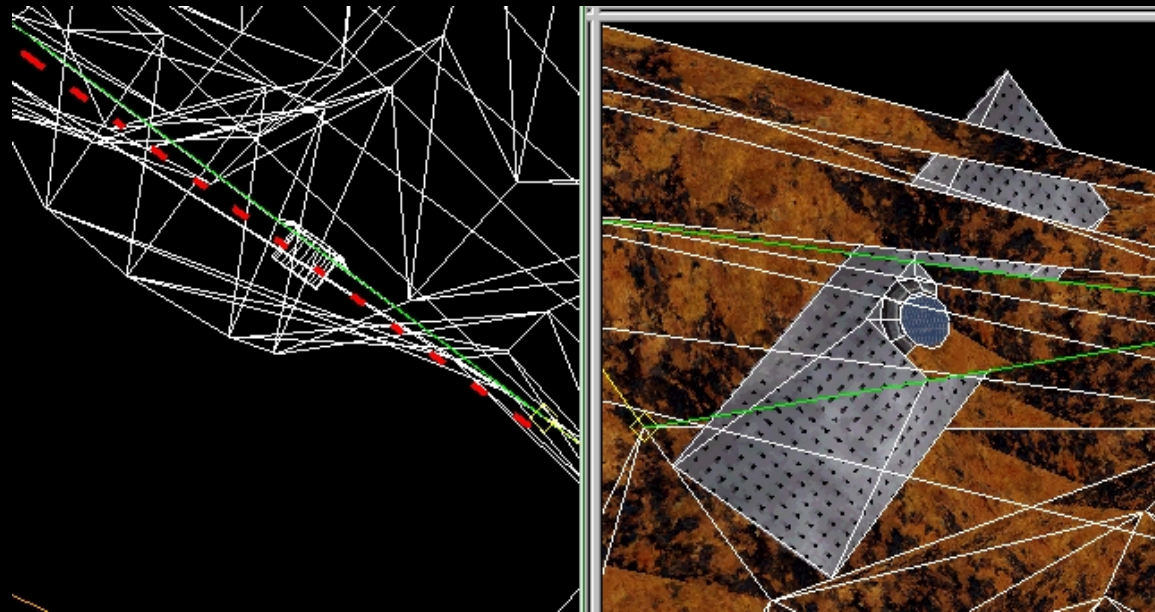


The object is inserted with the plane of the face, which is now a ground point, plane-parallel to the current face.

Then go to World View and click Room->Merge Object into Room, and that Object is transformed into a face construct that you can then integrate into your shell (below).

The red line is intended to underline that the object is plane-parallel.

Turn a little, and or move, then nor the faces fit...



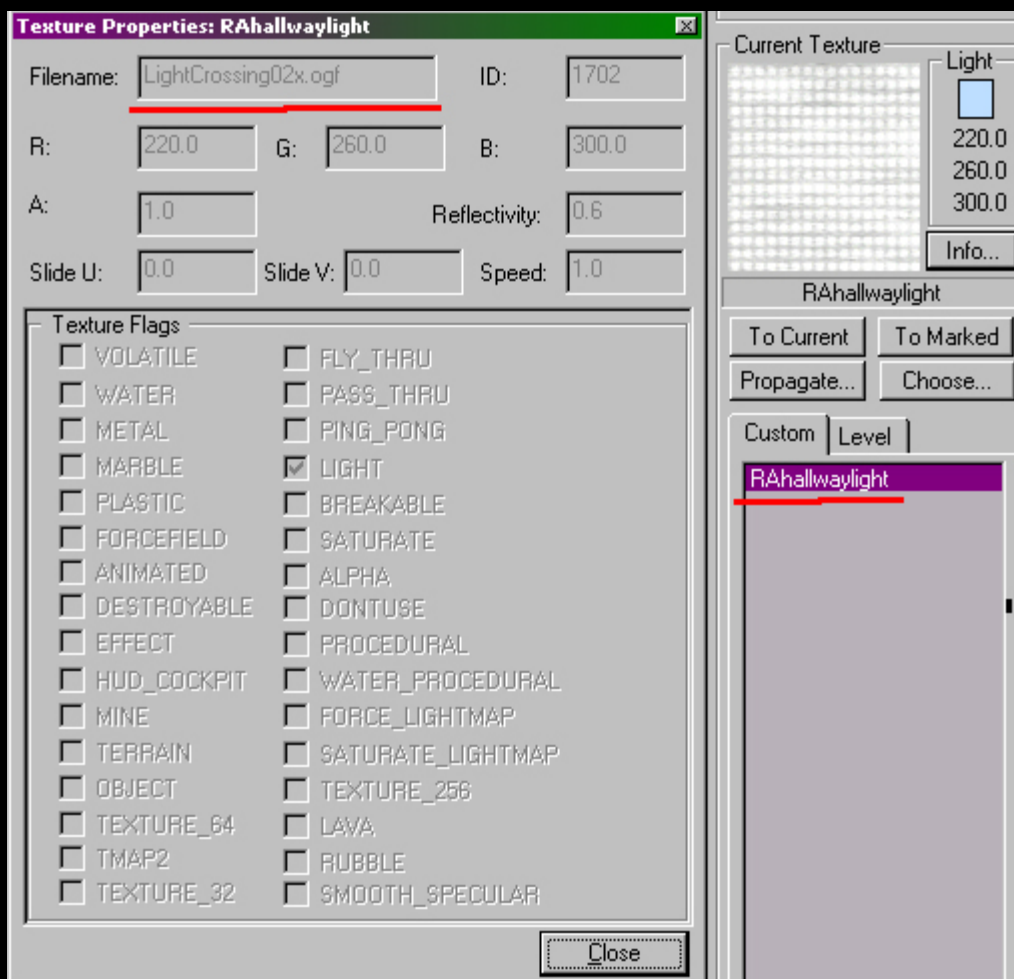
... and that Whole sits perfect!

Verify will be here no Mistake report.

One thing is though:

When using light textures, make sure that you have the right one in the merged object; it often becomes one and the same .oif-File used for multiple lights. Do you have anything like that?

Object the 'RAhallwaylight' as a light texture, it becomes the after merging underlying texture changed, in this case the 'LightCrossing02x'; and there is no light at all.

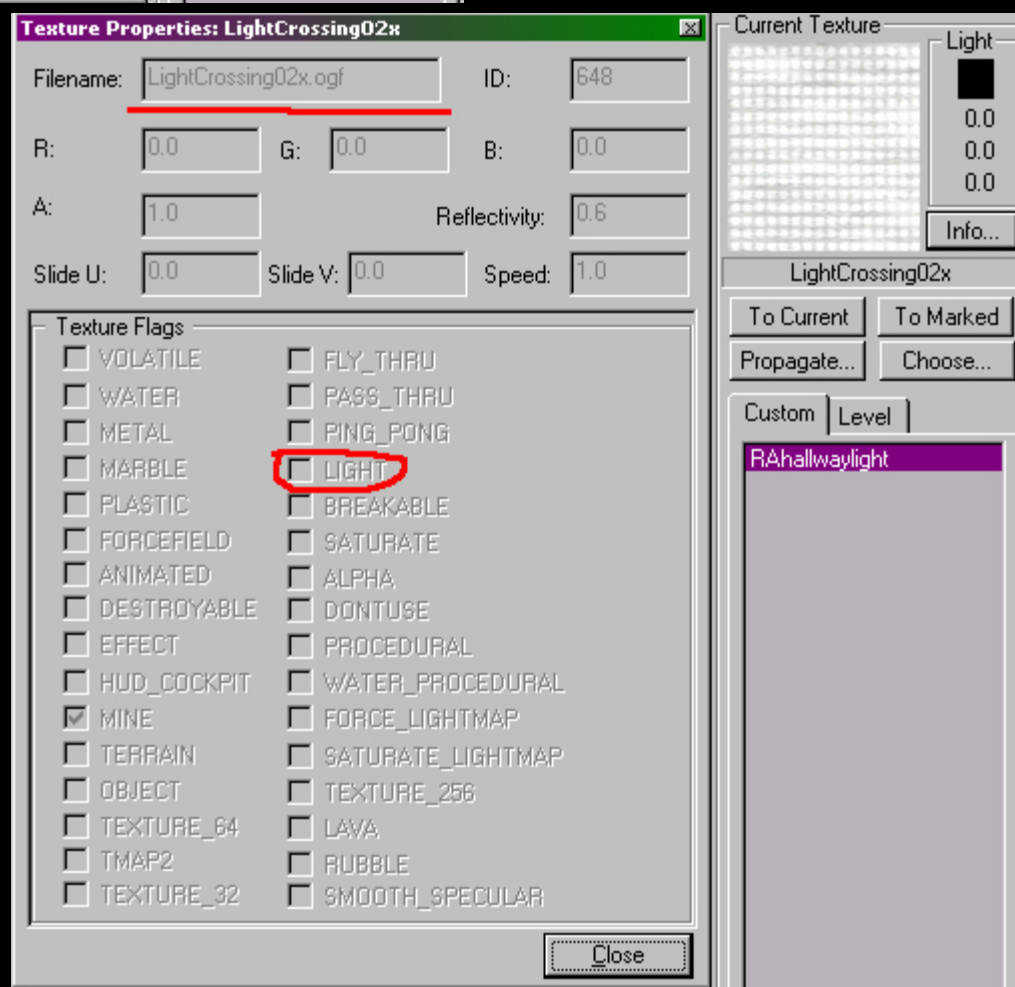


Left: the RAhallwaylight properties, pay attention to the filename and the light values.

Right: The underlying one .oif does not have the light flag

Thanks again to Atan for this Schmee!

Back to the overview of section L



135 - Use Mercenary bots

Ragil Ral

It is possible to bring the Mercenary bots into the level as objects, but then they either do nothing or little. This is because a lot has been programmed for the bots in the addon pack, which is why they are used with the standard `aiido` and do not carry out any actions.

To use Merc-Bots, you have to add everything relevant to your own level. This is in detail:

1. The bot (na nona),
2. all objects that belong to it,
3. the textures that belong to these objects,
4. of course the GAM pages for the objects and textures,
5. the corresponding one.dll for the behavior of the bot.

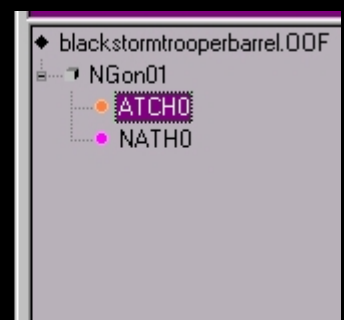
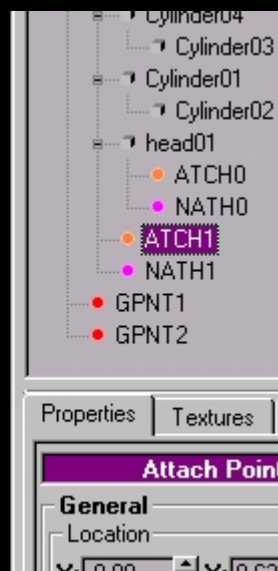
I've always liked the Black Stormtrooper from Mercenaries level 7, the story with the red laser sight is just awesome. This tut will use it. So first extract them with the quick edit `merc.hog` and `thmerc.mn3`, the files we need are split between these two hog files.

The bot itself is installed quickly; treats it like a normal custom object. So you take that.oof of the bot and its GAM entry and incorporate both into your level. If you look at it in-game, you'll notice that the bot doesn't even turn - you have to fly directly into its line of fire before it fires. So not quite the truth.

Another feature of this bot is the explosion when you kill it, this is realized via another object that is attached to the bot.

Unfortunately, finding out which object that is is a Sisyphean task; First you have to install the bot

Look at the functional original, then open all the objects one after the other in OOFEdit until you found the right thing. In this example it was (fortunately) relatively easy because the naming was convenient. The object files are `blackstormtrooper.OOF` for the bot and `blackstormtrooperbarrel.OOF` for the 'backpack'.

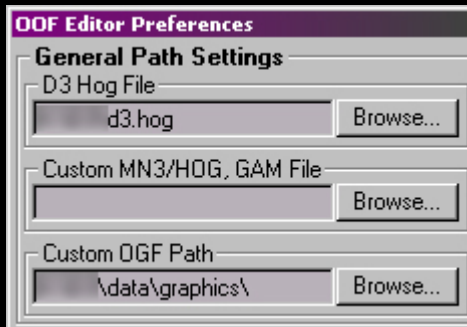
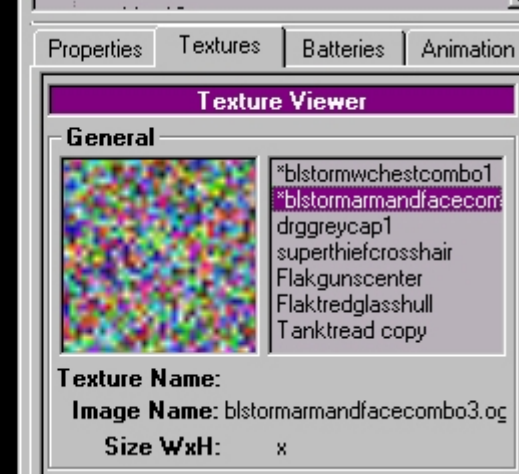


... and his 'backpack'

the Black Stormtrooper...

If you find 'color rush' textures on the object in OOFEdit, you still have to locate the associated textures. That's easy, go to the registerTextures; those that are missing are marked with an asterisk. Find these together and recreate them

<D3dir>/data/graphics and shares it
OOFEdit with where these are: Opens the
OOFEdit settings dialog
(File->Preferences...),



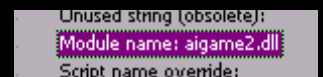
and under Custom OGF Path Simply point to the relevant directory. It is most convenient if you do it as described in 'Error: Reference not found'. Of course you can use any directory you want here, but the textures should also be there.

As soon as OOFEdit has this information, it will suggest that you reload the textures and then display them straight away.

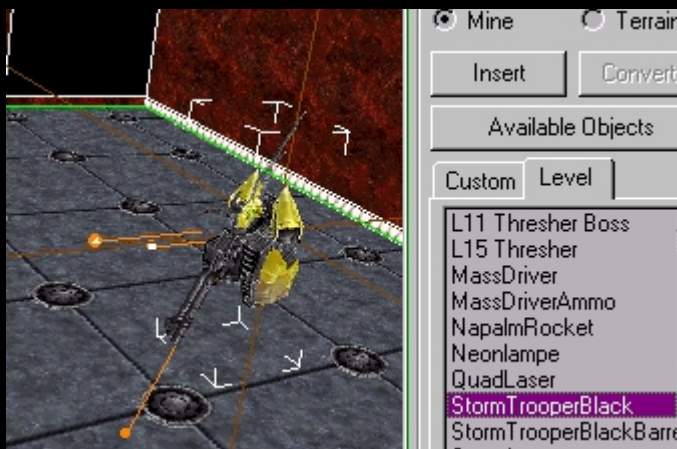


You get the GAM pages from thetable.gam, which in the merc.hogis filed; Don't forget the GAM entries for the required textures.

If you open the GAM entry for the bot, you will see that below Module name something is entered - this is thedll, which 'enlivens' the bot. You must find these; it is located in themerc.mn3. The GAM for the 'backpack' also points to this file.

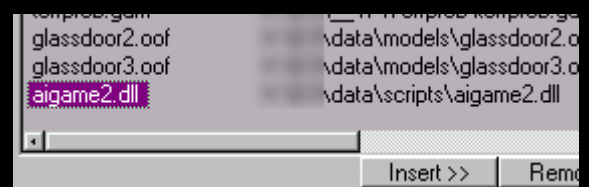


Put these in<D3dir>/data/scriptsaway.



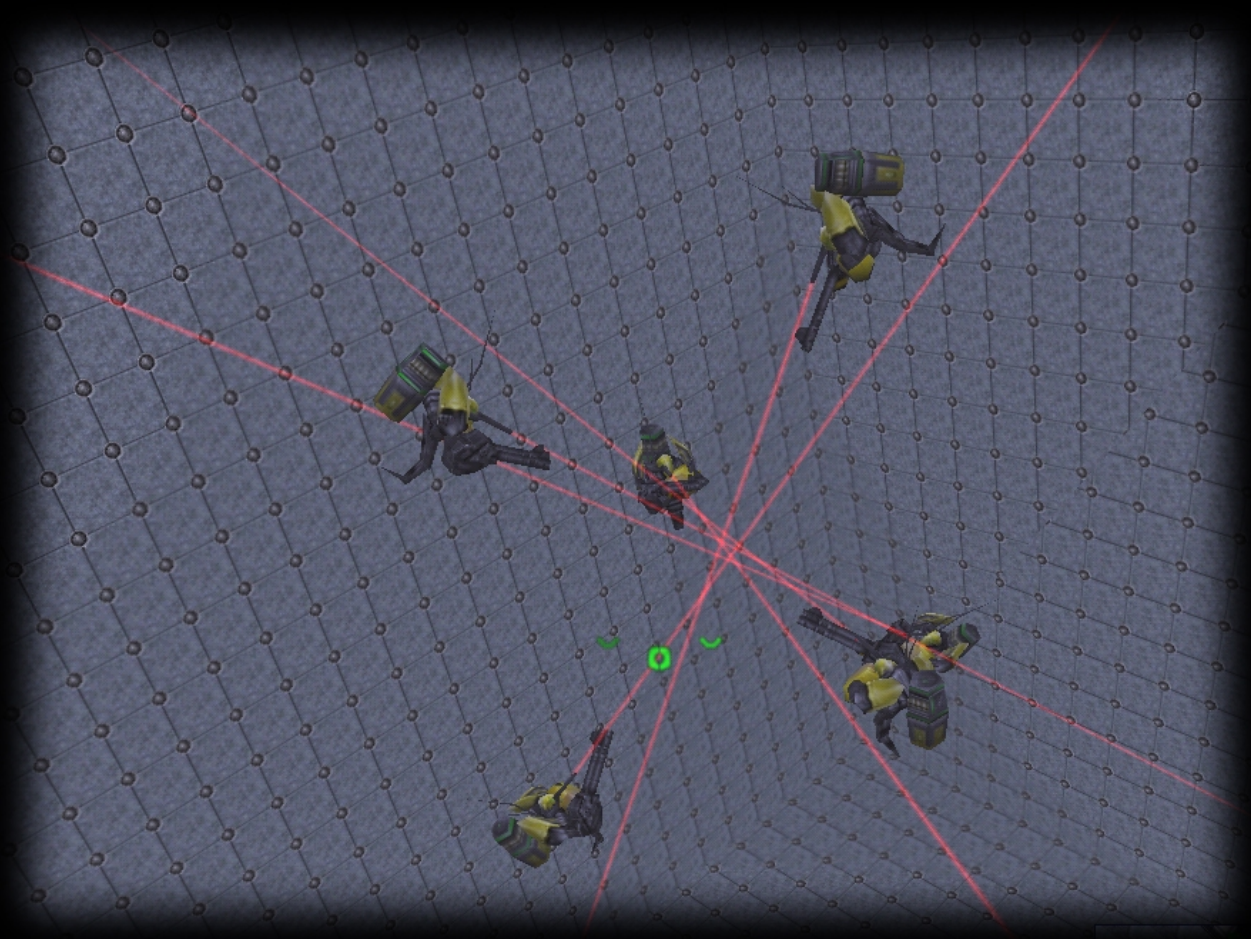
Now build the bot into the level.

In order for it to do what you expect, you still have to add it into the level, you add it in the quick edit a:



Then pack up the level and get in your pyro.

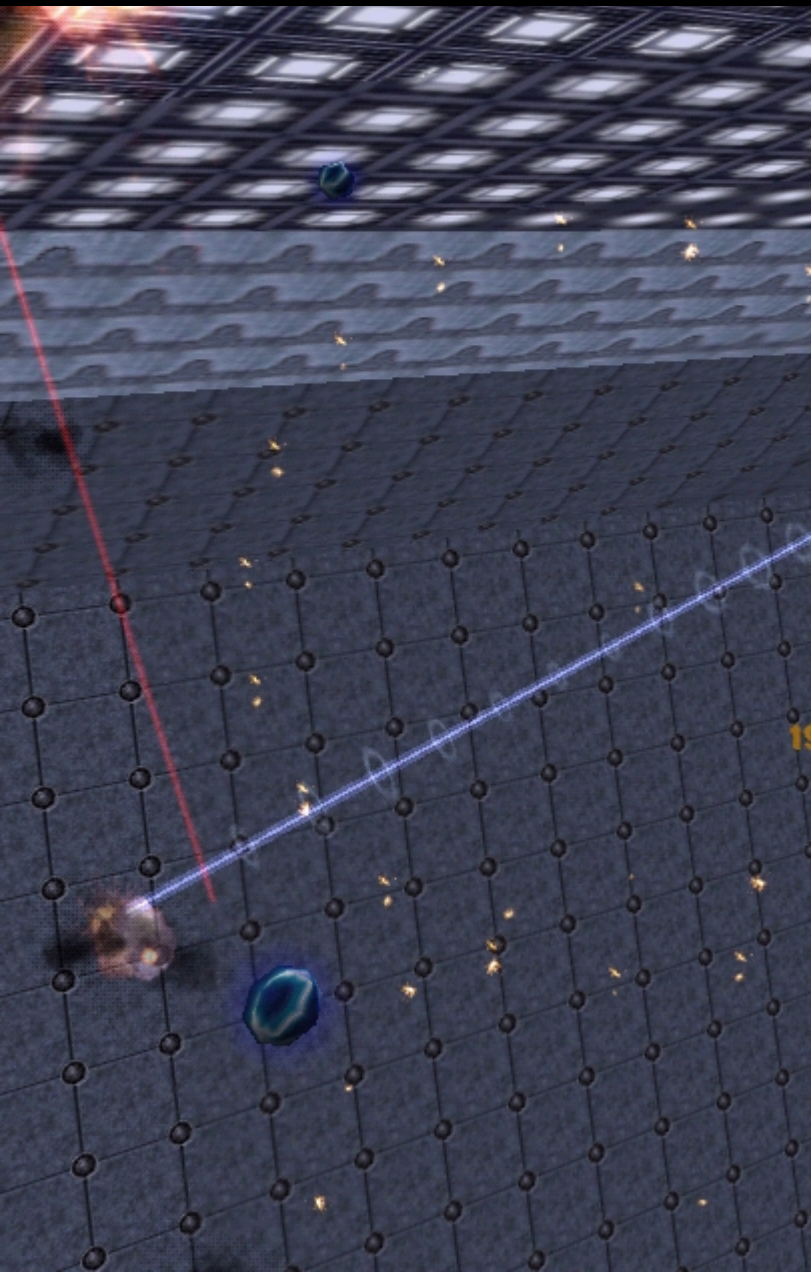
The Robot
is now
sharp!



As you can see above, the bots now have
their 'backpacks.'

On the left you can see the shrapnel coming
from the exploding bots... Better keep your
distance! 🙄

Back to the overview of section L



136 - Scripting & Matcen <>

Ragil Ral

For good single player action, but perhaps also for a special multiplayer mod, it will be interesting to combine materialization centers (Matcen, MC for short) with scripting. There was a thread on the DescentBB board that was a little more specific:

<http://www.descentbb.net/viewtopic.php?t=14392>; The point here was that an MC spits out bots, and when the player destroys them all a door is supposed to open, we'll cover that here because it's not that trivial. Therefore, you should be reasonably fluent in DALLAS.

The whole thing is a bit more complex, which is why there is a level for this chapter that is fully illuminated and contains some objects and MC's so that you can get started right away.

The two MC are called **MCleft** and **MCright**, they are in the rooms

LeftDoorRoom respectively **RightDoorRoom**.

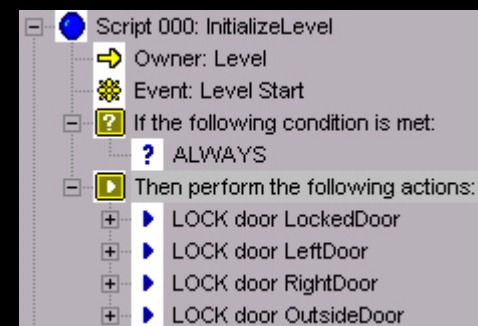
The object list from the **yo** on the right.

There is also a trigger called

playerEnteredSwArea, on the portal in Room 0. The doors are all still open,

so that you can test fly the level briefly. But the doors should all be locked, so either sit in from D3Edit **Locked**- Flag all doors or lock them using a script at the start of the level:

ID	Type	Type Name	Object Name...	Room Name	Room
0	PLAYER		0		0
1	DOOR	SEAN'S ENERGY SECR...	RightDoor	RightDoorRoom	2
2	DOOR	SEAN'S ENERGY SECR...	LeftDoor	LeftDoorRoom	3
3	DOOR	PTMC Covert H	LockedDoor	LockedDoorRoom	4
5	CLUTTER	Forcefieldswitch	Switch1	SwitchesRoom	6
6	CLUTTER	Forcefieldswitch	Switch3	SwitchesRoom	6
7	CLUTTER	Forcefieldswitch	Switch2	SwitchesRoom	6
8	DOOR	SEAN'S HEATSINK DO...	OutsideDoor		11



Task

The player should open a door by pressing switches in the correct order. If the order is not followed, the MC's should start spitting out bots. If the player all

Once the bot has finished, the switches should be reset so that he can try again. This is where the trigger comes into play, it is supposed to lock the player in to prevent freedom of movement

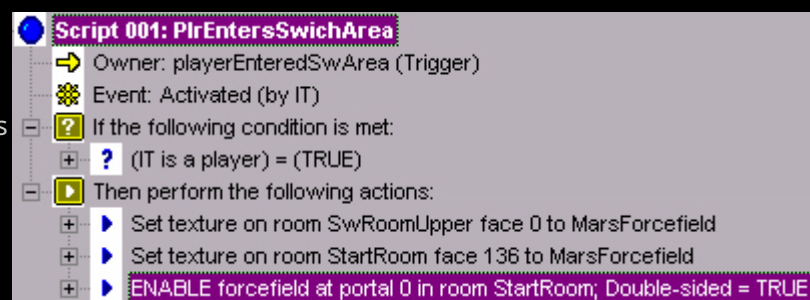
to narrow down.

Now, how do you go about something like that?

First of all, the player should be locked up, so let's first turn on a force field after the trigger has flown through (right):

To do this you have to give room 0 a name (here **StartRoom**). This action does those textures visible on the portal

are applied, so put a Forcefield texture on both sides of the portal otherwise the Palmleaf1 comes to light and can be flown through.



The logic

Now to 'programming'. The player should press switches in a specific order, and the switches' presses should be evaluated; Depending on the result, the action is more or less pleasant for the player.

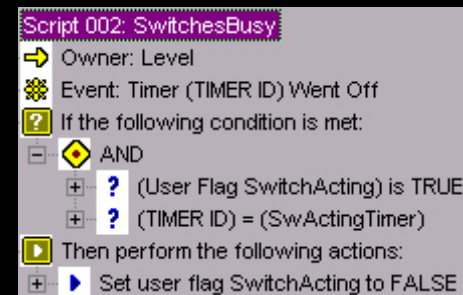
This means that we have to remember which switches were activated. You could do this with flags; Pressing a switch sets an associated flag that is queried and thereby decides whether the player has done the 'right' or 'wrong' thing. With three switches that would mean you need three flags; But we also need a flag that prevents the switches from being triggered too quickly, see also [094 - Switch scripts II](#) by WillyP. That would mean we have four flags to check for; not particularly beautiful or efficient.

There is another way. As soon as the player doesn't press the switches in the correct order, the MC's should be turned on. If the order is correct, nothing should happen - except that we still have to remember whether the order is correct or not. This can also be done with a variable by creating a User Type of typeUserVar, which we use for counting - every time the correct switch is pressed we simply increase this variable by 1, and so we only have to check for this one value and the switch brake.

Switch brake

As mentioned, this has already been covered, so here's just the script:

The necessary variables are called here **Switch Acting** (the flag) and **SwActingTimer** (the timer). In the scripts for the switches you then trigger this script with a level timer, and for the switch you ask for the status of the **Switch Acting**-Flags.



The switches

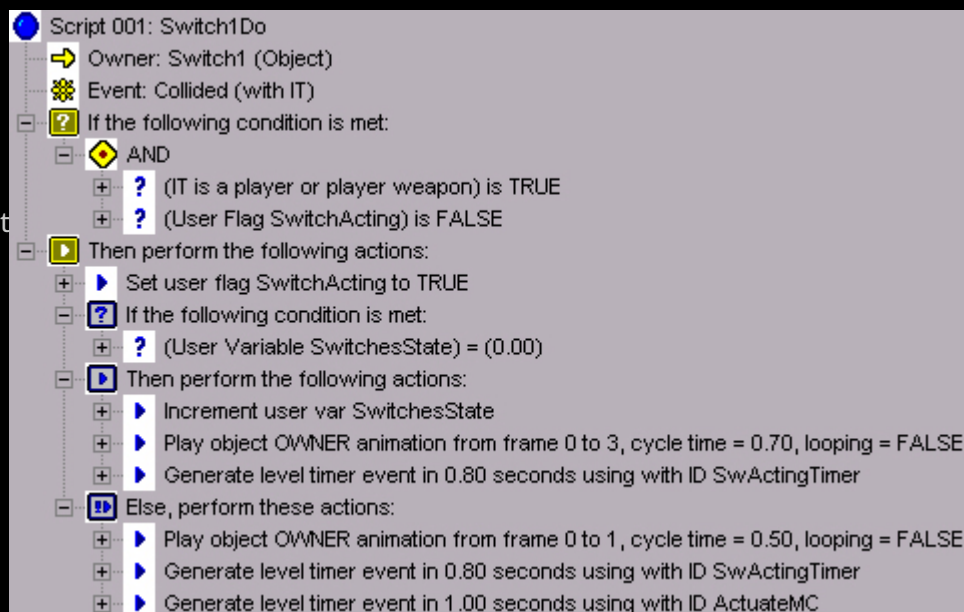
So we know so far that we have to query for a flag and a variable. The flag is intended to prevent the switches from being activated too quickly one after the other; So it decides whether actions belonging to the switch should be executed at all. Only when this is the case should the UserVar be queried, here it is called **SwitchesState**. In order to avoid eventualities, this is the case

Level start to 0 (it is generally a good idea to preset variables at the start of the level, this ensures that they have a precisely defined state).

The owner of the script is the respective switch, the event is the interaction of the player with the switch object, the query for the **SwitchActing** flag must be included in the conditions, and in the actions we first need an action for the switch brake and then an IF...THEN...ELSE construction, because of the query for **SwitchesState**. If the evaluation is true, the incrementation should take place, the switch should be animated and a timer should be triggered, which triggers the switch brake script after the switch's animation duration has expired. If the evaluation is not true, the same thing should always happen - the MC's begin their deadly cargo to release 😡

So that you don't have to include these actions three times in the respective switch scripts, it is recommended to create a script for the MC's and call it up at the switches. Unfortunately, this doesn't work directly, so we have to use a timer (here **ActuateMC**).

With the scripts for the other two switches you now only have to evaluate the **SwitchesState** and the **Owner**. You can therefore copy the script for the first switch, which saves work.



In the script for the last switch, the action differs for the correct evaluation because the player has then solved the task; then a door should open. So replace the action for the increment with an action that unlocks the door (in this example the door named **LockedDoor**). To spice things up, you can play a sound and/or display a message on the HUD.

The Matzen script

If the player has made a mistake with the switches in the order, a timer triggers this script. So what is supposed to happen?

- 1- the MC's should start production
- 2- the doors to them must be opened beforehand.
- 3- after the player has destroyed all bots, the switches should be reset and the doors closed again.

further,

- a- if the player dies, he respawns in the starting room, so the force field has to be switched off again so that he can get to the switches again.
- b- maybe a powerup for the crushed player wouldn't be bad, maybe a shield orb?

For Matcenters there is under...Actions->Mixthe following options:

WithActivate/Deactivate

you switch an MC on/off,
withEnable/Disablecertainly

Your whether an MC can be activated or not. With the third action you can tell an MC more precisely what to do.

[Activate/Deactivate] matcen [Matcen]

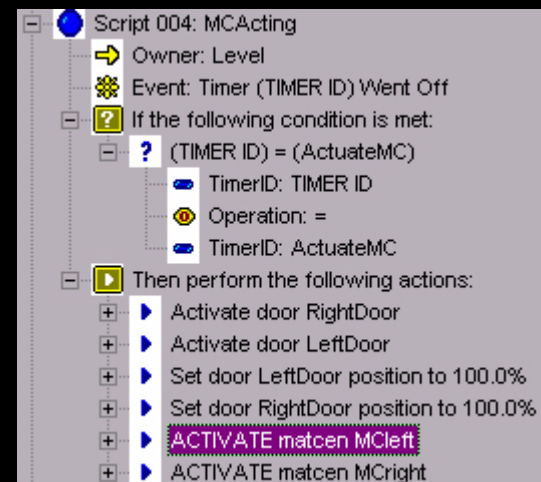
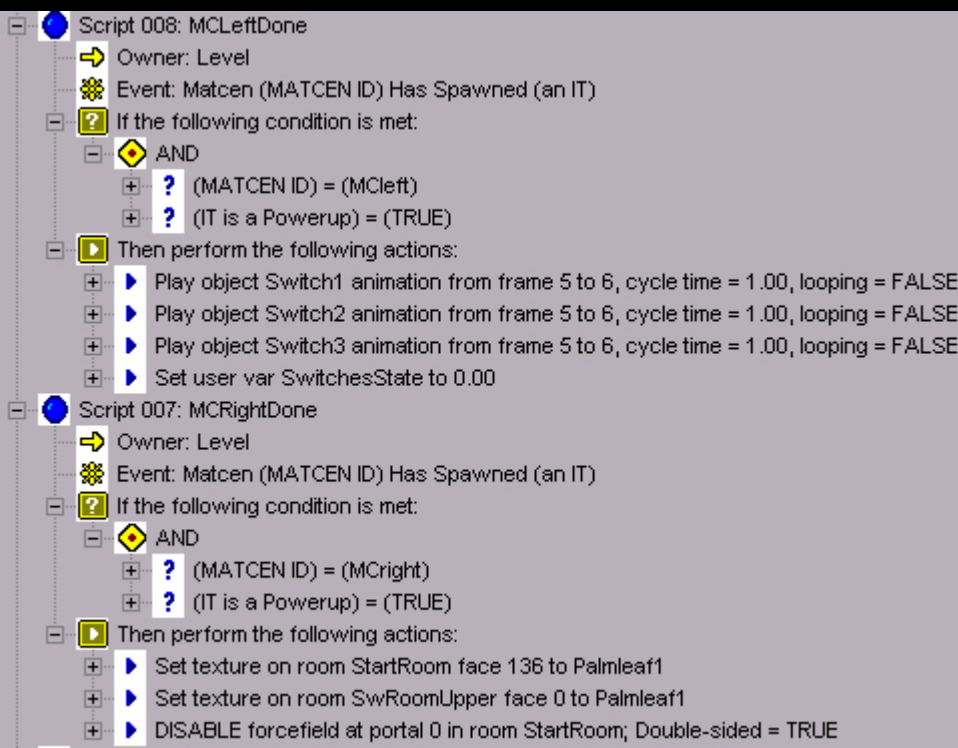
[Enable/Disable] matcen [Matcen]

Set matcen [Matcen] max produced = [MaxProduced], production rate multiplier = [Multiplier], max alive children = [MaxAlive]

Robot
Player
Weapon
Viewer
Powerup
Debris
Shockwave
Clutter
Building
Door

There is only one event for the MC, namelyMatcen (MATCEN ID) has Spawned (an IT), so you can query which object type the object produced by the MC is. In addition sets in the conditions...Query > Objects > [Object] is a [ObjectType], where your then on the type of the object. You can see which are possible in the screenshot on the left.

So we can evaluate or react to what an MC has spit out. Unfortunately I was unable to complete the queryObjectType=Weapon to get it to work, perhaps because DALLAS is under the ObjectTypeWeaponunderstands something different than a Vaussgun. This allows us to fulfill condition 3 by simply letting the MC generate a powerup and react to its appearance. If necessary, the pre or Post-production time, so that there is a certain time interval between 'powerup appears' and 'door opens'.



Of course, you could also go here and put a named bot in it, ghost it at the start of the level and deghost it at the MC event, and unlock the door when this bot is finished.

MCRightDonealso removes the force field from the entrance.

If the player has not solved the task and fails a second time, the MC's will not spit out anything. In order to intercept this you have to for the relevant MC's `MaxProduced` increase; To do this, you can either set a high value natively and then set the MC's again using a script turn it off or use `itSet matcen [Matcen] max produced = [MaxProduced], production...`-Function to increase the value.

Okee, that might have been a bit much all at once and besides, I only included screenshots of the most important scripts with the text. But if you think it through, I'm convinced you'll get the whole thing done 😊

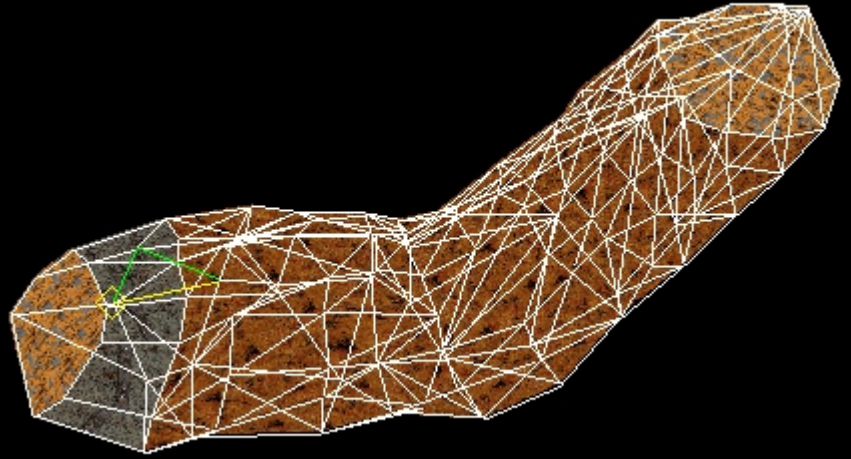
[Back to the overview of section L](#)

137 - Partially scale rooms

Ragil Ral

Maybe you've already had the situation where the room was generally okay, but... certain parts of it are too big or too small. In this example, I have a cavernous passageway that I want to extend the end of:

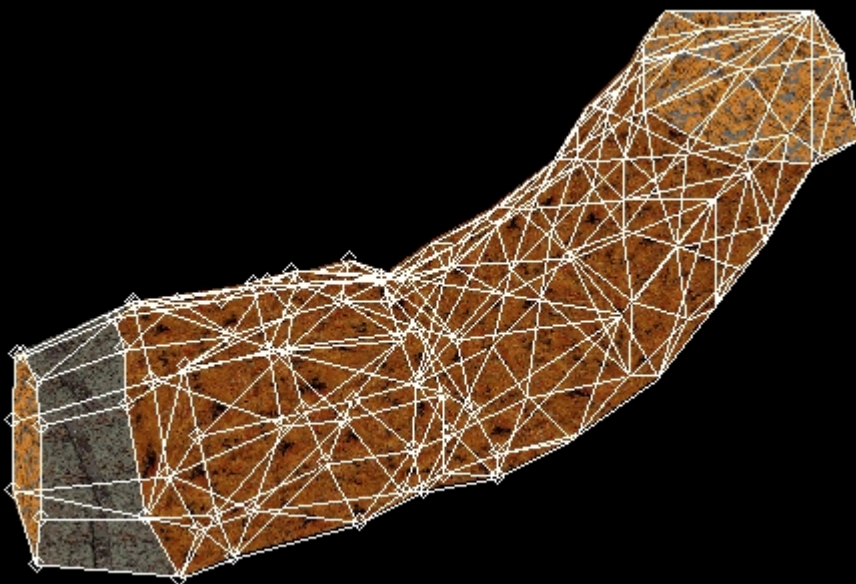
This is where the function for face expand and contract comes into play - perhaps a little surprising... 🤖



Process...

This takes advantage of the fact that Face Expand and Contract process EVERY face, regardless of whether it is valid or not. So you mark the verts of the area you want to scale, insert a face and scale it. Then adjust if necessary and align the textures.

... and the practice

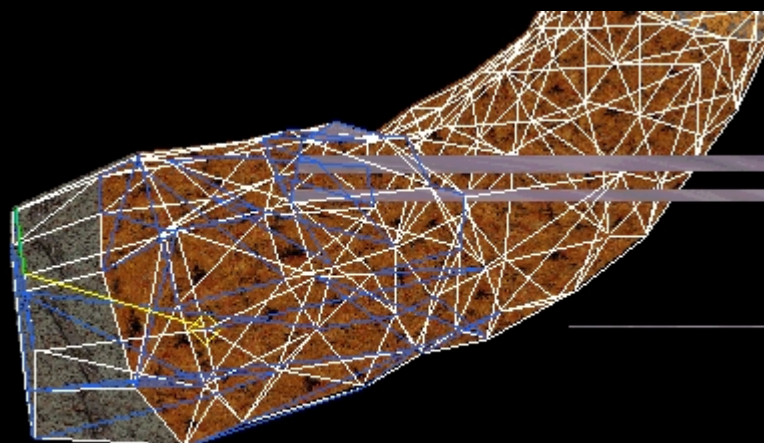


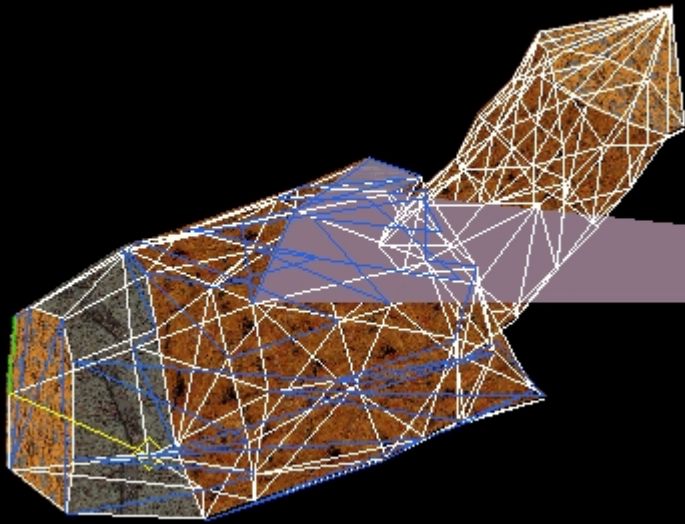
First mark the verts of the area to be scaled, since v40 you can also have these displayed in the 3d view:

Here it goes straight, there are 52 verts.

Then add a face there (**Shift Ins**, below)

As you can see from the display errors, the face is anything but valid. The inserted face is current and marked.

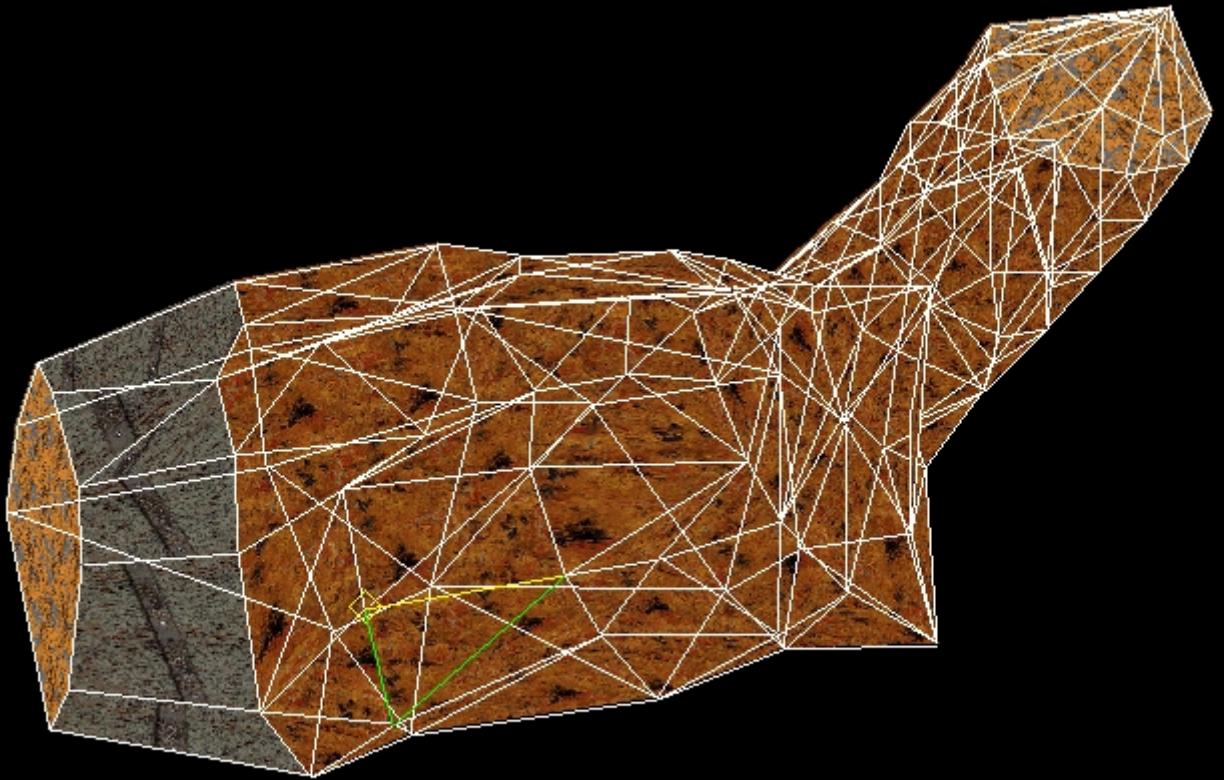





A few clicks on Face Expand later and things will look like the one on the left.

Now switch to face mode. Now you can adjust the changed area. Then simply delete the marked face.

Et voila...



Just align the textures and you're done.

The method has the advantage that the face bandage is not torn, which unfortunately happens when using the dialog-controlled scaling function  (`scaling`).

Update v40:

The scaling function had a positioning error, which is now fixed. The 'splitting' of the face is still happening, so don't forget to run Remove Extra Verts if you're using dialog-controlled scaling.

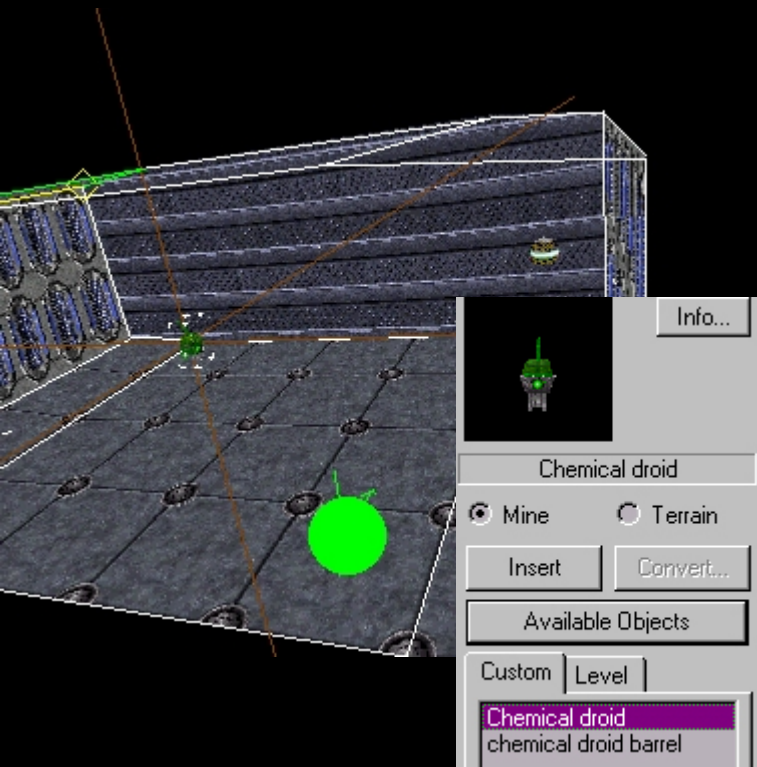
Back to the overview of section L

138 - Composite Objects <>

Ragil Ral

As you know from the section Error: Reference not found - Error: Reference not found, it is possible to attach objects to others. The prerequisite for this is that both objects have one **ATCH** as well as one **NATH**, they serve to connect and align the objects with each other.

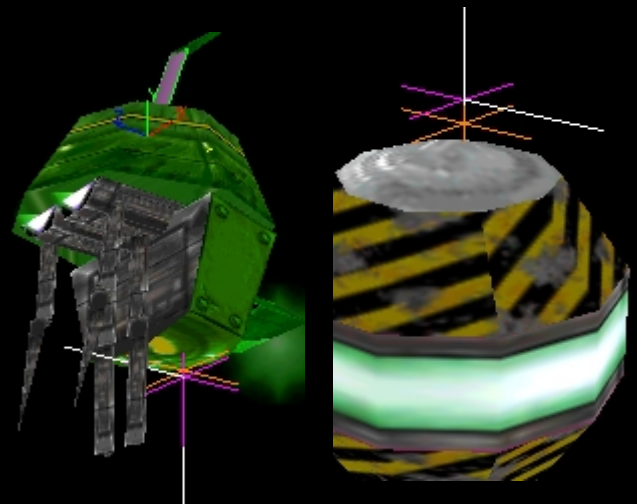
In principle, the matter is not that complicated, but using DALLAS should no longer cause any difficulties; You should also be familiar with OOFEdit.



In the first example we will use the Chemical Droid and its capsule from Retri Level 2.

So start with a level, get the two objects in and name them. Here the name is Chemical DroidCarrier, the capsulePayload.

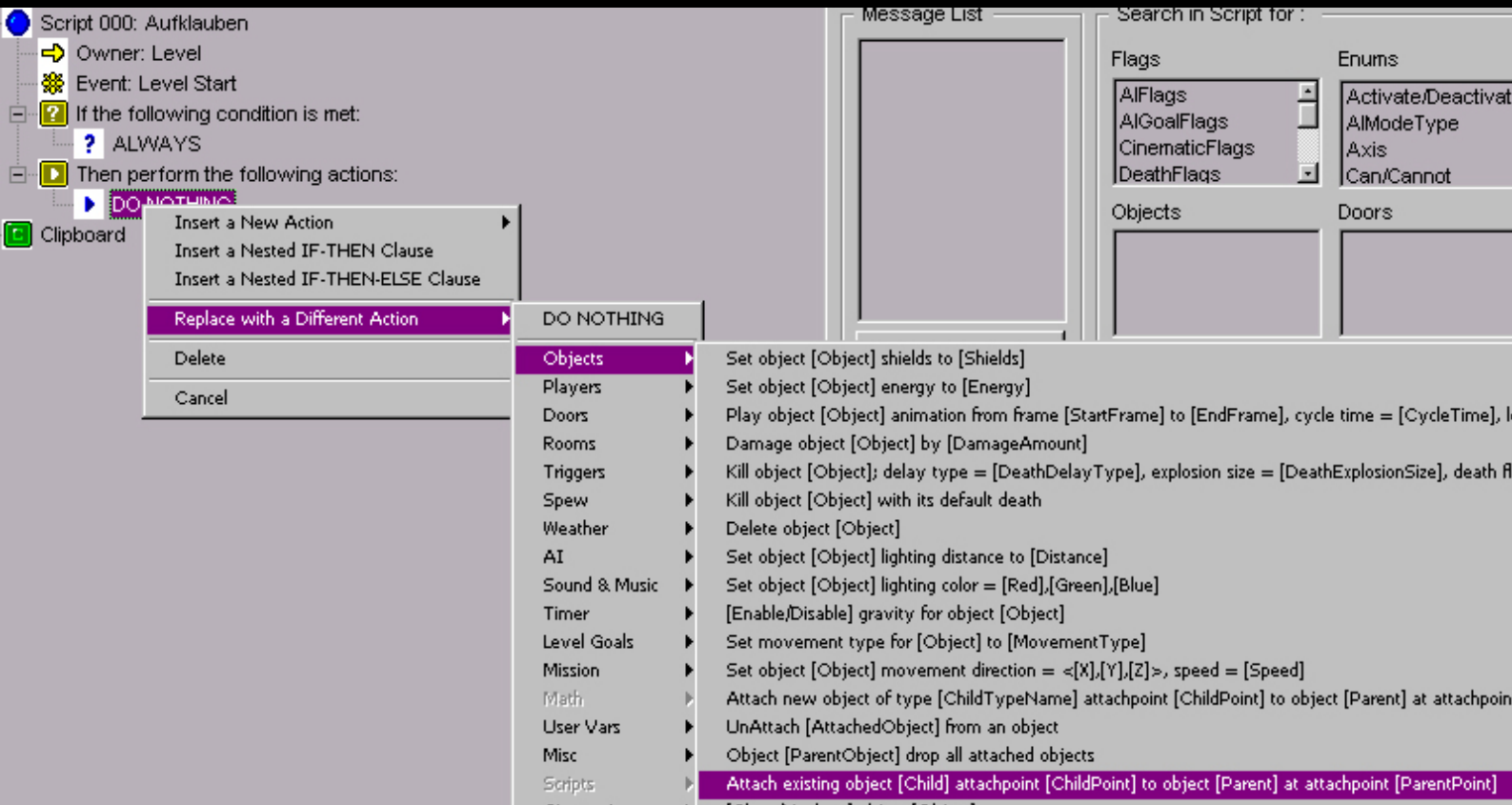
Both objects have one each **ATCH** and one **NATH**:



When you put the objects in the level and start it, they are of course separated; This is where DALLAS comes into play.

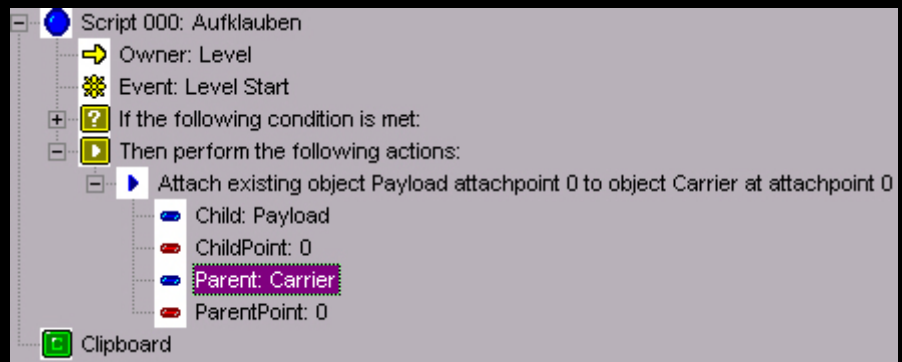


Build this script:



Parent is the carrier (here the Chemical Droid with the name Carrier) and Child is the attached object (here the capsule named Payload).

Since both objects only have one **Attach** entry it fits in this example for ChildPoint and ParentPoint.



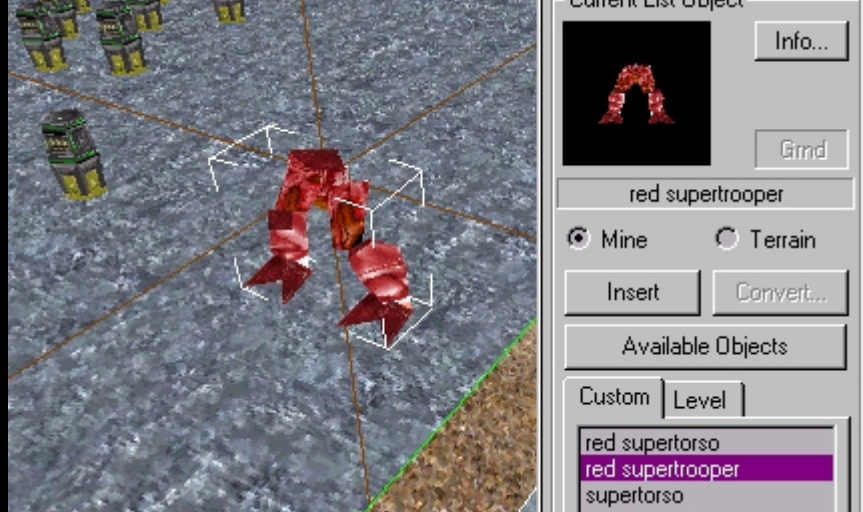
Save/compile the script and incorporate it into your level.

Lo and behold, the bot is carrying the capsule!

If you destroy the bot and have gravity in the level, the capsule will fall down.

This is the 'Hello World variant'. But there is more...

If you think about the supertroopers running around, they also consist of two objects - the legs and the top. The game takes care of the assembly here, as the scripts are already included there. The same goes for the Juggernaut, the PLAGUE bot, the Dragon Boss bot, and the other composite things from the original game. With Mercenary bots you have to do this a game2.dll incorporate.

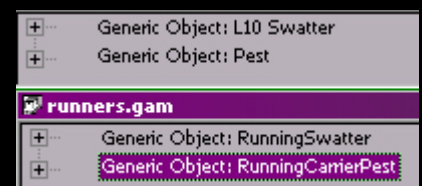


But how about walking gun turrets?

Preparatory work

To do this we first need two suitable objects, a bot that moves on the 'ground' as a carrier and a turret, let's say the L10Swatter. For the 'carrier' take the PEST-Bot, it is relatively mobile.

First, copy their GAM entries into the level GAM and rename them so that they are treated as separate objects (right). Additionally I have the damage to the turret (Hit Points) increased to 1500 so that things don't break so quickly.



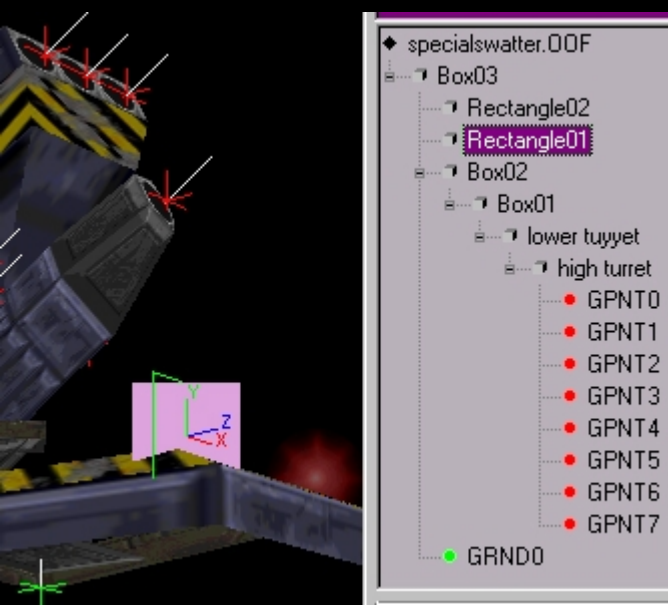
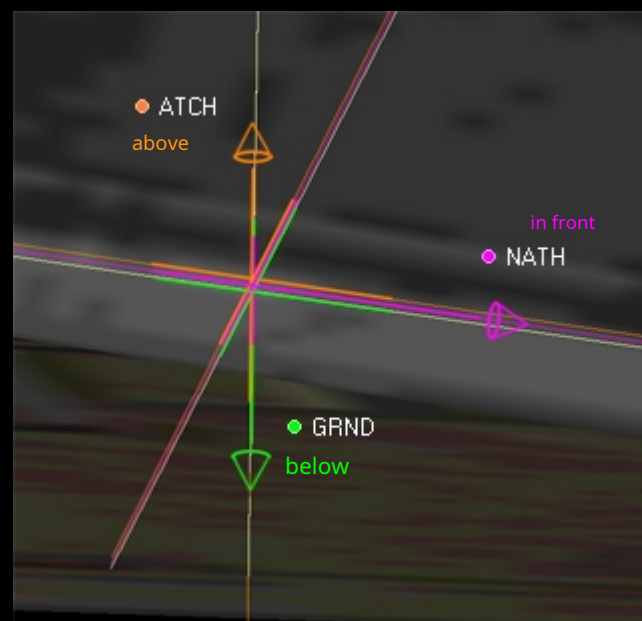
Above the original entries, below the renamed ones



We can do the PLAGUE body Take over as it is, it already has an attach point and an associated attach normal point.

To connect objects together, you need one **ATCH**(which indicates in which direction the object is attached) also one **NATH**(which indicates in which direction relative to the **ATCH**the object looks); see the graphic on the right.

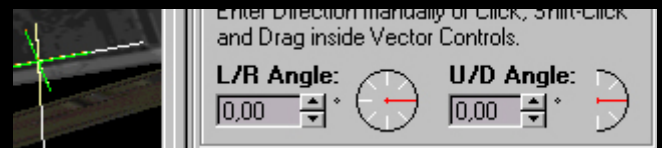
You have to rebuild the tower a little because the object is missing these two points. The **REASON** You must not take it away under any circumstances, otherwise the turret property of the object will no longer work.



The model

specialswatter.OOF but has two faces that are used as lights, and they are suitable for incorporating the missing points, they are called **Rectangle01** respectively **02** and are used for the red lights. First click that **GRND0** and write down its coordinates (should be 0.08 / -6.09 / -1.39). Then move both rectangles there by entering the coordinates of the **GRND0** in the tab **Properties** entered - Affect Pivot Only switch off if necessary. Then one of the faces converts to an attach point, the other to an attach normal.

Now align the points **ATCH0** sets as shown on the left **NATH0** according to the right picture:



Save the modified turret under its own name and change the GAM entry accordingly. (In this example **runningswatter.OOF**)

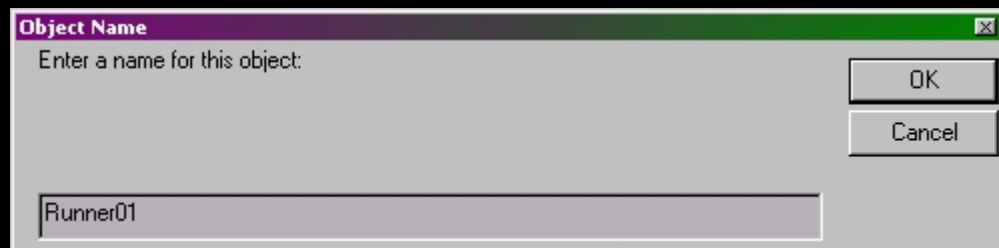
It was easy here, if you first have to add faces to the object in order to use it, you also have to recreate the entire object with hierarchy and animation.

Bulk

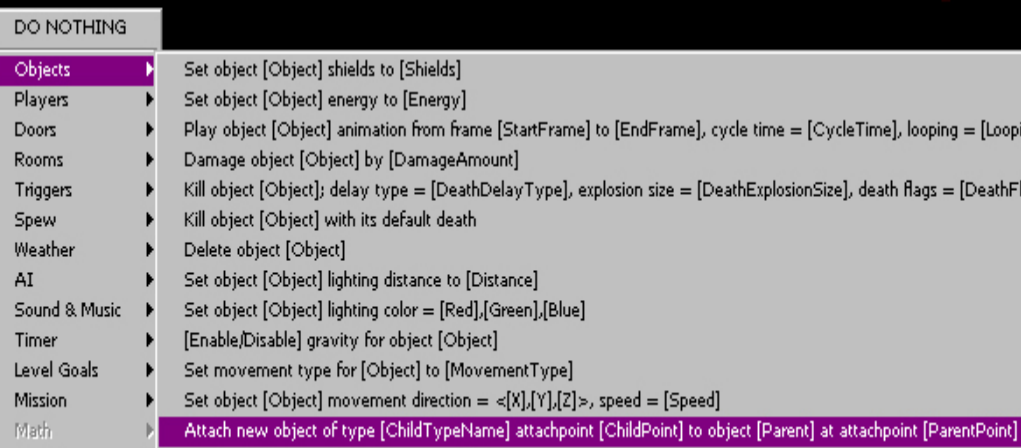
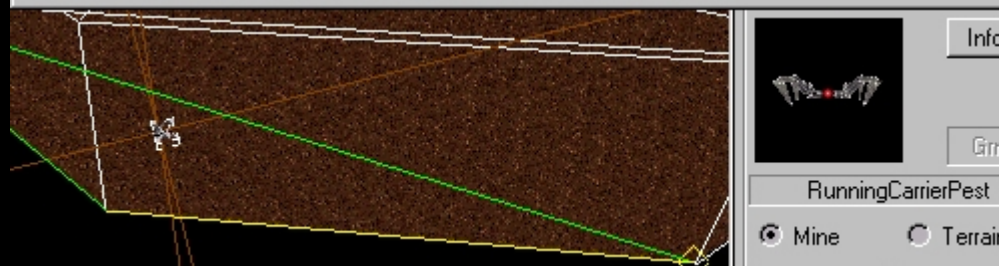
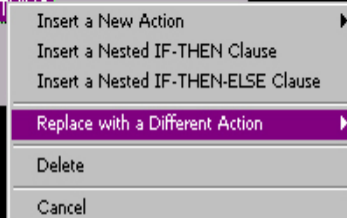
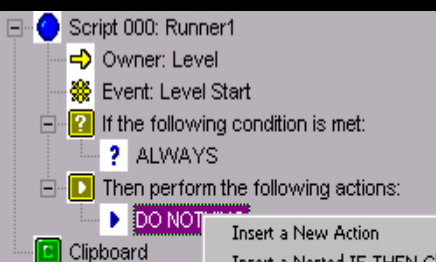
Now all the stuff has to go into the level.

To do this, first get the two objects in the list and set one or more of the **PLAGUE Bodies** in.

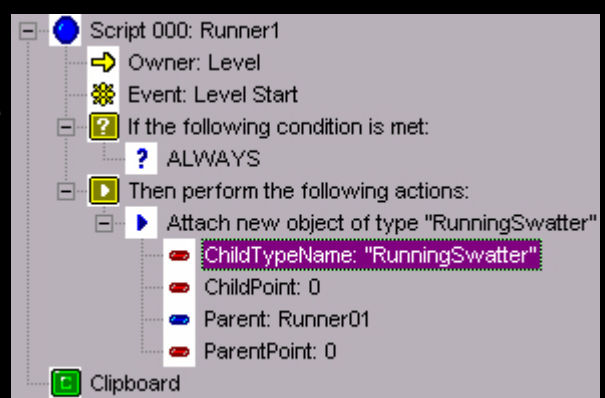
Names the bot(s) so that they can be addressed via script.

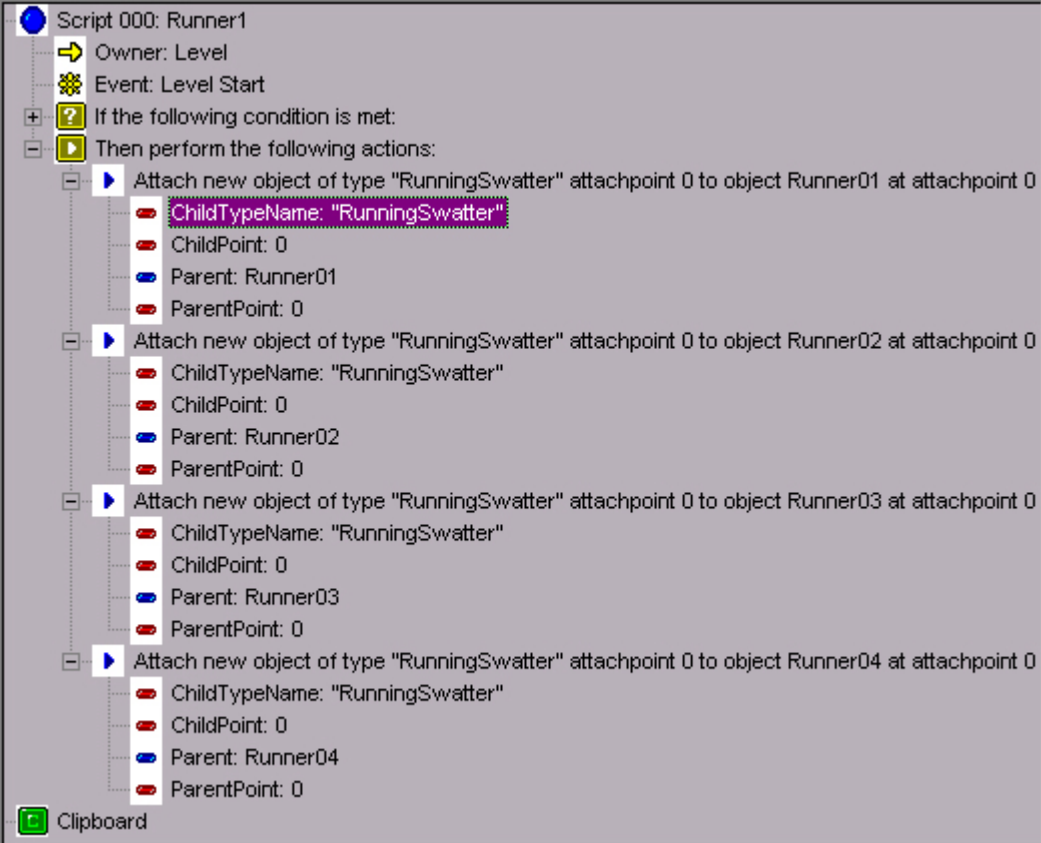


Now insert the following script:



The way it works here is that the 'carried' object (the turret) is created at the start of the level and is placed on the 'carrier' (the PEST body). To do this, you assume Parentone of the named bots, and under ChildTypeName write the name of your modified turret:





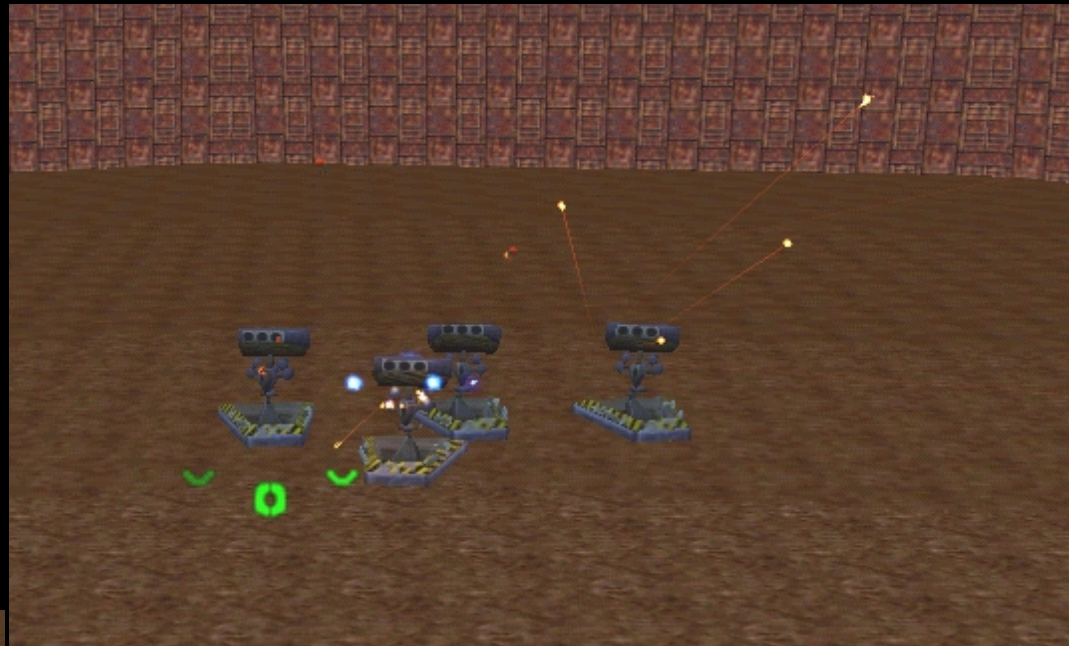
For each bot that is supposed to carry a tower you also have to insert a corresponding action.

Saves the script and builds it along with the other files (.gam and object(s) into the level file.

Ingame it looks like this:

The front of the Turret base always points towards the player, as the PEST always runs towards them (left).

If the 'carrier' is destroyed, the turrets remain stationary and behave like normal turrets. (front of the Turret base no longer points to the player, below)



In this example, the towers are positioned about one unit too high, which is why you can see underneath them. To fix that you would have to **ATCH** reposition accordingly.

Furthermore, according to Destruction of the tower does not show the destroyed base, but you would have to do that again Install appropriate script.

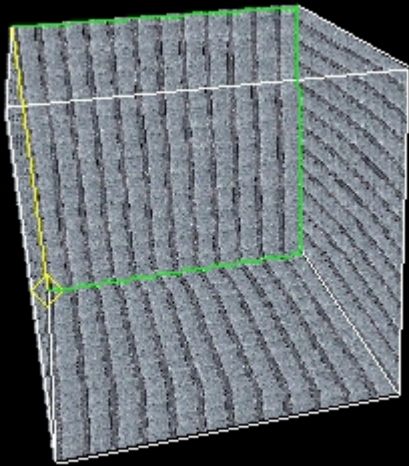
Back to the overview of section L

139 - Recycling

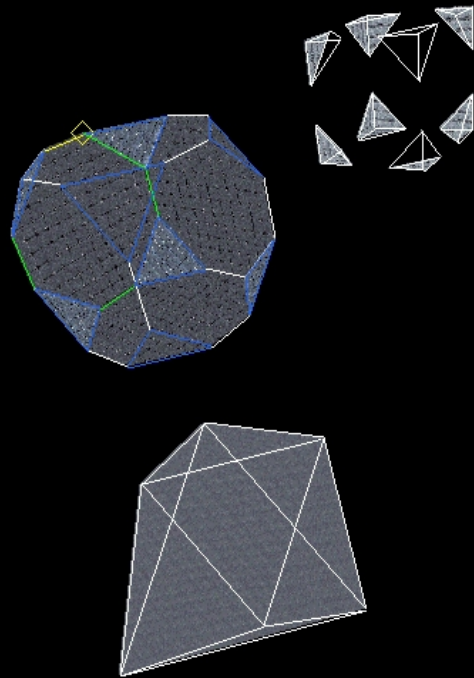
Ragil Ral

If you are designing rooms, there will be a lot of 'waste' when installing the interior; So remaining faces. In most cases, auxiliary constructions are used when creating a room or its shell

Waste is created when forming the shell from a primitive.



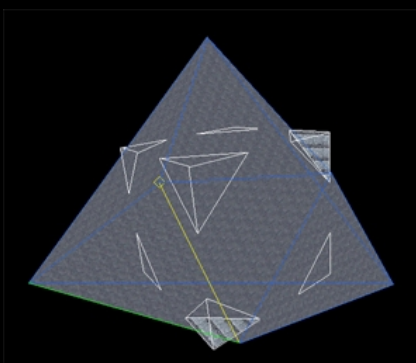
Dice...



... cropped, offcuts and cutting figure

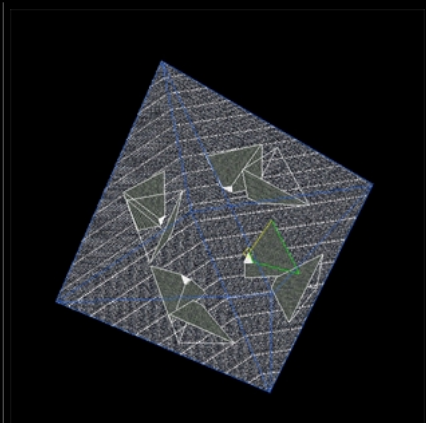
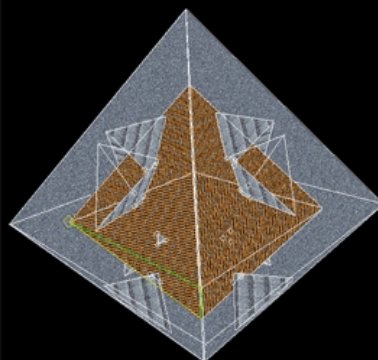
As a rule, you will delete the scrap, but wait...! There are a few things that speak against simply throwing it away.

- The angles and lengths in the offcuts are, so to speak, the 'negative' of your room.
- Verts and faces are aligned relative to space
- Leftover polygons often form quite interesting constellations.
- A room needs a few regions anyway, so why not use what's there



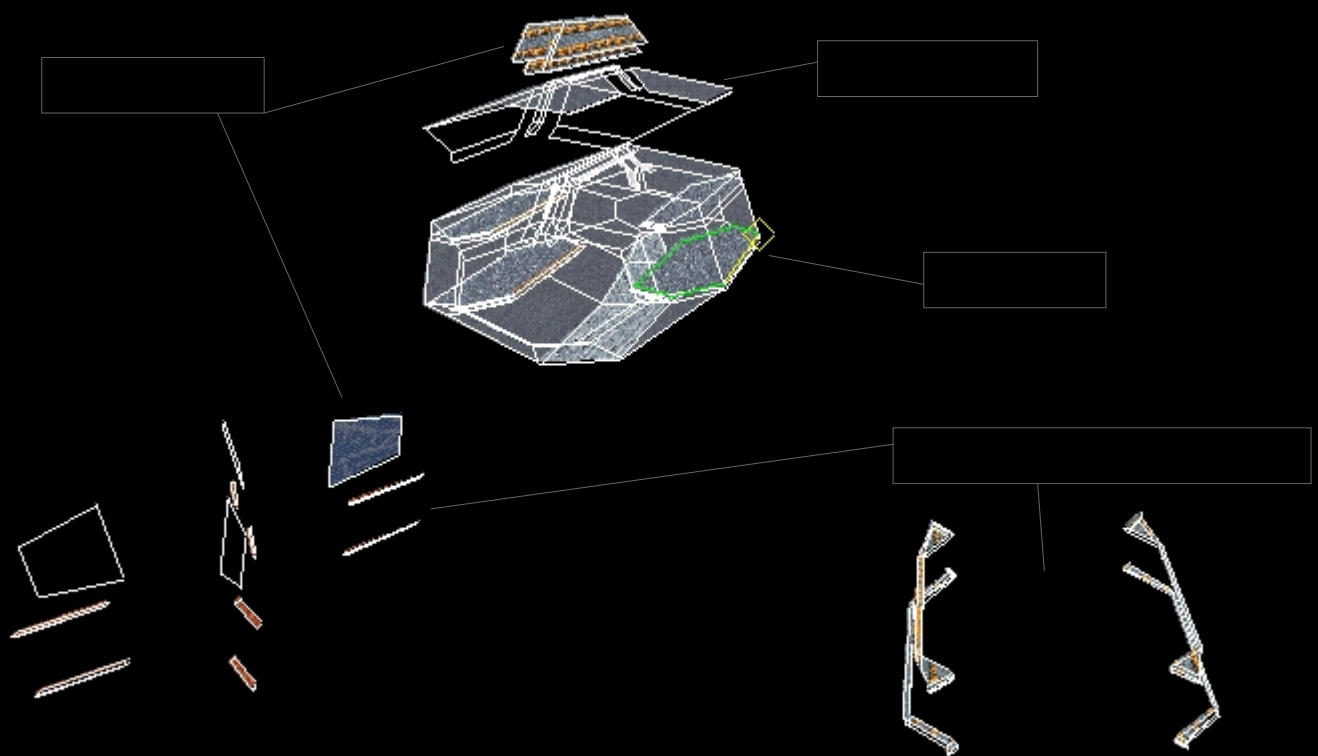
Offcut and offcut figure fit...

about intermediate steps...

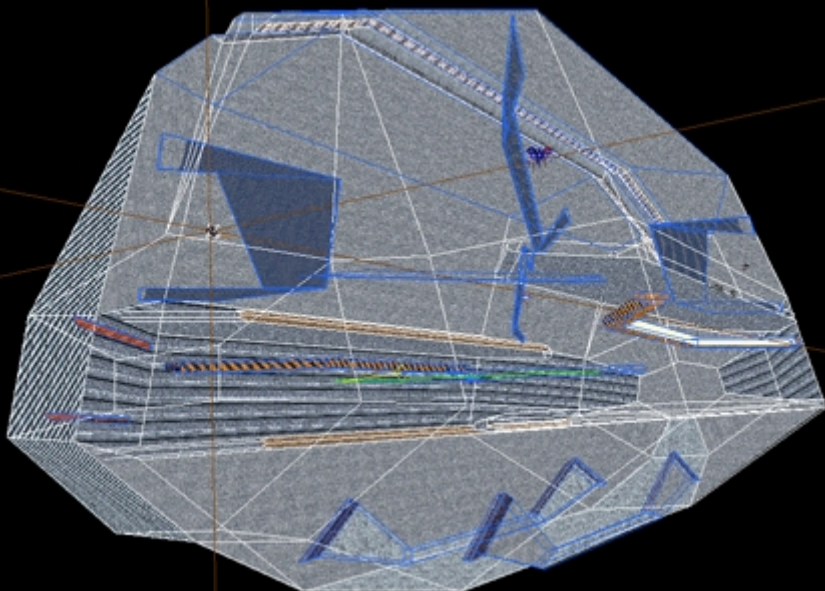
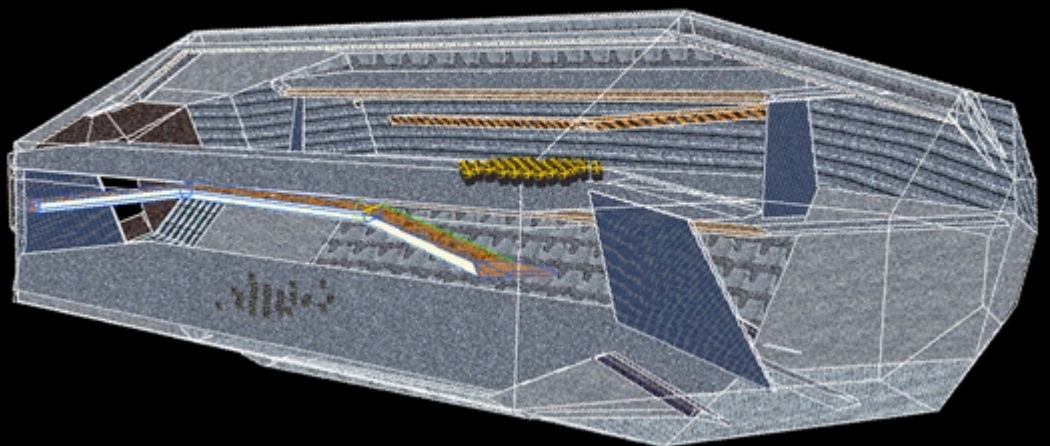


A second room is created.

Here is a storage room whose construction involved a lot of polygon waste:



The arched one
Offcuts left over from the
storage areas were
turned into lights
convert, from the
trapezoid
Cut surfaces were
'Protective discs':



In the left screenie there is another
storage room, all constructs created
from leftovers are marked.

The 'glass panes' were created
from offcuts from cut faces.



The
smaller
Space
ingame...

... and the
Bigger.



Naturally is not it always tried to improvise something from scraps and leftovers. But sometimes it's a good idea, and if you take advantage of it you can save yourself a lot of work and time.

Back to the overview of section L

140 - Open Mercenary levels

Ragil Ral

Attention: You have to stick to the given (C)!

It may happen that you would like to find out how the guys solved certain things in the Mercenary Pack. If you have already extracted all or parts of the Mercenary Pack, you may also have tried opening one of the levels - which is the following produced an error message should:

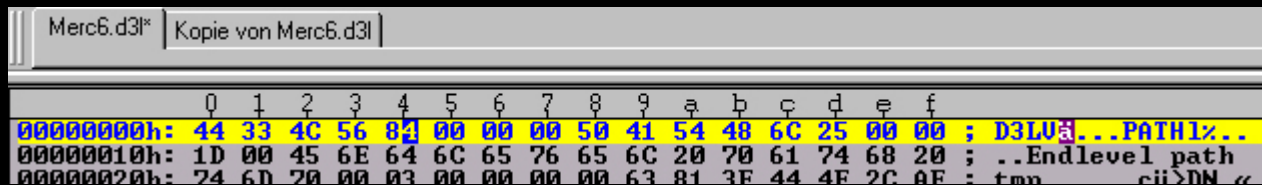


There is a solution, but you have to do something.

First, extract the whole thing merc.hog, all the custom stuff is needed when the levels are to be loaded.

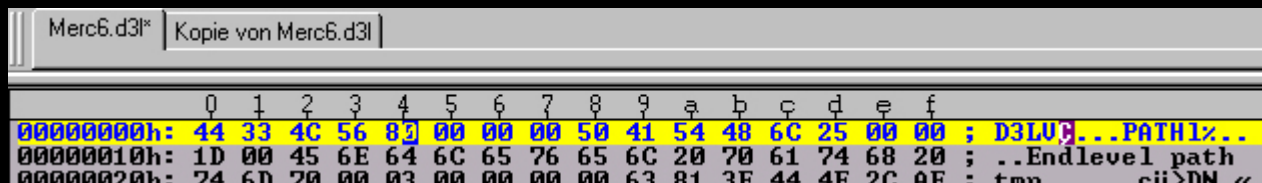
Then you need a hex editor; This is - if you don't already know - a program that can open and edit files as byte code (which is then represented as hexadecimal numbers).

Now open one of the Merc levels in the hex editor, it will look something like this:



It's about the spot4in the line 0h, there it says '84';


This corresponds to the decimal value 132. This is the version number that D3Edit criticized as being too new - and we are now changing it. So write instead of '84' so '80' (corresponds to decimal 128) into:



Save that File, and off she leaves immediately in Open D3Edit.

To open a Merc level, you should definitelytable.gamload that you extracted from the Merc Pack; Otherwise you will be overwhelmed with error messages because D3Edit cannot find valid references for a truck full of objects.

Furthermore, it is convenient to put the models, textures and the rest of the customs in the developer folder structure (seeNo.117 - Customs again) to distribute.

The levels are very large and contain some geometrical impurities (no pun intended, so ), the opening them takes a lot of time and editing is also slower because of this.

Back to the overview of section L

Attachment-Useful information

	List of all D3 tools	=Legion=	480
House of Mirrors	Images from the Abyss		485
In-game help for Testing	Simple means of increasing the level 'in action' check	Ragil Ral	486
D3 Edit Limits	Limits when building	(LL)Atan	488
Application of D3 Level info	Using Atan's special tool	Atan	489
Documentation of the Doors	All doors from D3	Ragil Ral	492
D3Edit Key combinations	Hotkeys for the D3Edit - all on one A4 page		541

[Toc](#)

List of all D3 tools

=Legion=

I have tried to list as many existing tools as possible here; if several versions are available, I have used the newer one. I have left out older tools if there is a (better) replacement for them. The whole thing is grouped thematically. Unfortunately I can't guarantee completeness, but I hope I didn't miss anything 🙏

The list is a compilation of the tools available on the following sites:

<http://archives.descent.cx/d3/d3tools.html> <http://descent3fischlein.de/tools/tools.php4> <http://www.planetdescent.com/site/files/d3/utilities/development.asp>

The status of the list is March 2008.

Table key...

Purpose of use	path\to\tool\	
Surname	Description	author
filename		Program version
Art		Size in kB

where:

Art	Description
packed	Simply unpack it somewhere and can usually be used straight away; read readme's!
installer	requires installation to function properly.
neither nor	. . . , be sure to follow the instructions in the readme's here

First of all, of course, you need it:

D3 Edit	The real, true, only D3 editor!	Outrage / (LL)Atan
D3EditAV40.zip		1.1 [AV40]
packed		1465 kB

You can find the source in the introduction.

Script/code related	tools\coding\	
Ming C compiler	This is used to compile the DALLAS scripts.	
egcs-1.1.2-mingw32.zip		
packed, see tutorials		6200.0 kB
SDK TOOL v0.01 beta	The tool makes it possible to translate game modes under LINUX. It installs the SDK, a compatible compiler and is included in the Next versions will be improved, with more functions	D. Cent
sdk-tool_v0.01_beta.zip		0.01
packed, see tutorials		30.7 kB
SDK 1.4 (Linux)	You need SDK to incorporate scripts into your levels.	Jeff Slutter
d3-sdk_v14.tar.gz		1.4
packed, see tutorials		573.9 kB
SDK 1.4 (Windows)	You need SDK to incorporate scripts into your levels.	Jeff Slutter
d3-sdk14.zip		1.4
packed, see tutorials		634.1 kB
HOG OGF	A text file containing the data of all .ogf files.	?
hog-ogf.zip		?
packed		18.5 kB

converter	tools\converter\	
2 ORF	Converts .stl, .dxf and .ase to .orf	Magus
2orf_v091.zip		0.91
packed		119.8 kB
3DSGen	Converts OOF files to 3DS format. Basically everything that is stored in an .oof is converted (textures, texture mapping, hierarchy, special points, keyframe data...) Also includes the MAXOOFUtility. Cmd line tool	SuperSheep
3dsgen.zip		1.0
packed		88.6 kB
DXF2ORF	DXF2ORF is a utility for converting various Drawing eXchange Files to Outrage Room Files used by the Descent3 Level Editor, D3Edit.	Cougars
dx2orf_vb.zip		beta
packed		11.4 kB
MS3D2ORF	Import rooms created in Milkshape to D3Edit's ORF format. Very handy for those of you who don't like D3Edit's interFace. Cmd line tool	Descenterace
ORFTOOL.zip		0.5 beta
packed		57.3 kB
OOF to 3DS	With this program you can convert the geometry information stored in OOF files (Outrage Object Files) to 3DS files (3D Studio). Cmd line tool, requires 3dmax installed	Tycoon
oof23ds.zip		1.0
packed		44.7 kB
OOF2ORF	Converts OOF's (Outrage Object Format) to ORF's (Outrage Room Format). The nice thing about it is that you can also unpack the textures with. You also get all the sub-objects, UV mappings and the rest from the .oof	Robert Piontek
oof2orf_vb123.zip		beta 1.23
installer		1607.8 kB
PD3 to OOFGen Converter for 3DS Max	3dMax plugins to convert to .oof. requires 3dmax installed. Plugins for newer and older versions	Outrage
p3d-oof-utilities-MaxAllVer.zip		3
plugins		534.7 kB
X2orf	Converts x exports from blender 1.42 to .orf files. Cmd line tool	Ouch
x2orf.zip		0.9.0a :)
packed		27.2 kB

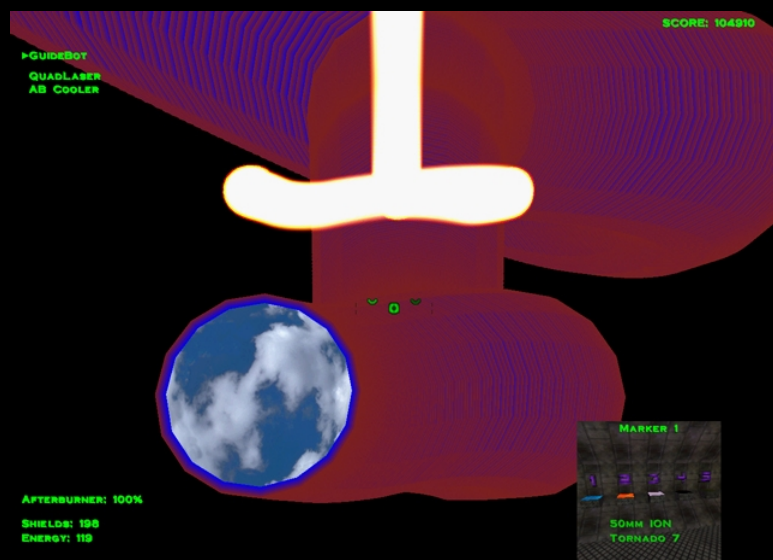
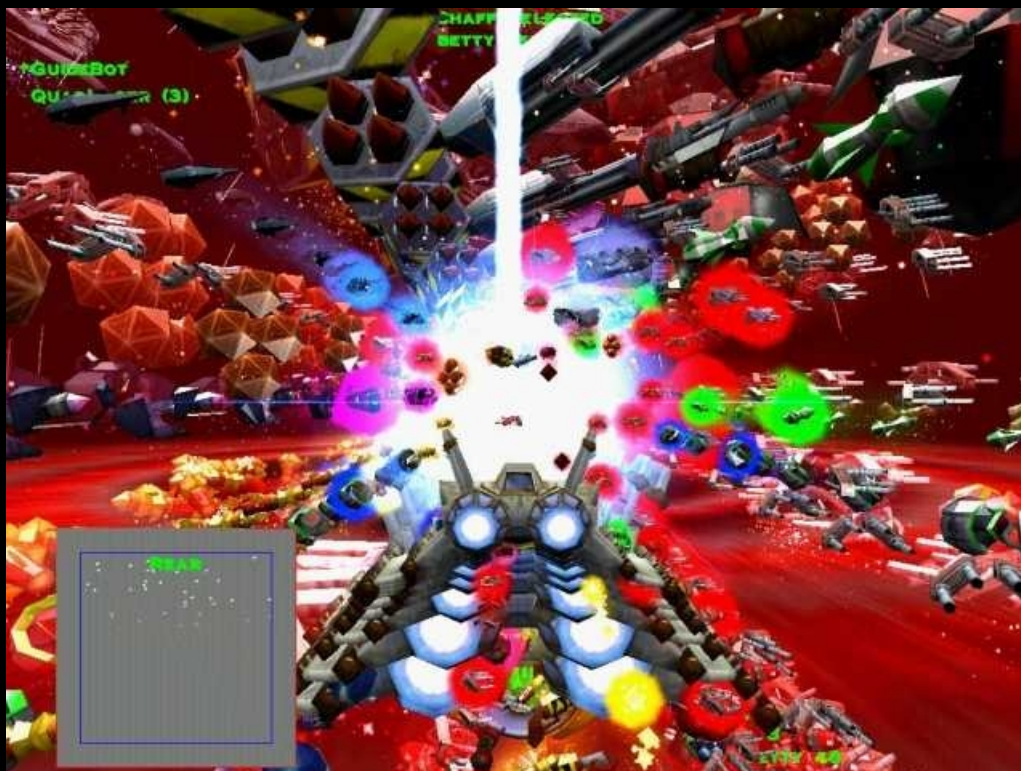
GAM file	tools\gamfile\	
Descent 3 Control Panel	View and edit GAM files	Dark Knight
d3controlpanel_v102.zip		1.02
packed		350.3 kB
Gamtool	Edits .gam files. The tool is used to change or create properties of weapons, ships and your own textures. (and even more)	Aldel
gamtool0611.zip		0.611
packed		163.3 kB
Hog Explorer	View and edit HOG2 files	Gwar
hogexplorer_v05.zip		0.5
packed		28.0 kB
Hog View 32	Extracts files from .MN3s and .HOGs	Mike Menefee
hogview32_v3b.zip		1.0 Beta 3 build10
installer		488.7 kB
HOG2 Workshop	A very good program for editing and extracting files from D3 HOG2 files.	Dark Knight
hog2workshop_v10.zip		1.0
packed		165.3 kB
Lex Hogedit	Extracts and creates Descent3 HOG files	Lex
dh2e11.zip		0.0.0.0
packed		1807.9 kB
Packafied 0.5 (PR's)	Packafied is a package file editor. Examples are Doom WAD files, Quake PAK files and Descent HOG files.	Peter Runge
packafied_v05.zip		0.5
packed java		295.8 kB
Table Editor 0.4 (PR's)	With this program you edit GAM files; Weapons, ships, objects, etc...	Peter Runge
tableedit_v04.zip		0.4
packed java		147.4 kB
Tycoons Hog 2 Viewer	Displays the contents of D3's HOG2 files. No editing!	Tycoon
hogedit.zip		1.1
installer		210.5 kB

Objects	tools\objects\	
alOOF 0.8	alOOF is an editor for Descent 3's Outrage Object Format (OOF) files.	Aldel
aloof08.zip		0.8
packed		110.8 kB
D3View	Displays Descent 3 poly models.	Garry Knudson
d3viewer.zip		0.1
packed		40.2 kB
Descent 3 tool	This program can display the objects used with the Descent3. You can also use it to create your own objects and manipulate them. This tool has import/export functions for OOF	Garry Knudson
d3tool_vb14.zip		beta 0.14
packed		73.9 kB
Model view 32	View and edit all Descent 3 polymodels.	Heiko Herrmann
modv32b5.exe		1.0 Beta 05 Build 28
installer		616.6 kB
OOF view	View of Descent3 3d .oof models.	Gwar
oofview_v040.zip		0.4
packed		44.6 kB
Oofedit1,518	A great tool for creating and animating your own objects! If you get any ActiveX error message, you have to do the following, go to the program directory, open a Dos box there and enter: regsvr32 SSubTmr.dll	SuperSheep
oofedit1518.zip		1,518
installer		2304.1 kB
tExtractor 3.0	Maya plugin that allows you to unwrap the UV coordinates of a selected polygon and output the results as a graphic file. This can then be used as a perfectly fitting texture	Outrage
textractor30.zip		3.0
maya plugin		18.3 kB

sound	tools\sound\	
Descent3 Audio Taunt Manager	Converts Wave files to Descent3 for immediate use.	Lex
tauntmgr.zip		1
packed		260.6 kB
OSFPlay	Tons of OSF Taunts in the "\custom\sounds" directory? You can listen to this here. "MusicTester" is required and is included with D3Edit.	Lex
osfplay.zip		1
packed		210.7 kB
Sound Composer + Plugins	A new sound tool. The composer, if he proves himself, will replace the Taunt Manager. With a new plugin concept for extensions. Plugins included: Sound Effects 1.1, Sound Recorder 1.1	Lex
d3sc+Plugins.zip		Alpha 1.0 Build 38
packed		647.6 kB

Textures	tools\textures\	
Animake	A tool with which you can create animated textures.	DCrazy
animake_vb0201.zip		0.201
installer		89.2 kB
D3 Image Tool	Converts TGA files to OGF files. However, it is easier to handle than the OGFTOOL.	SuperSheep
d3it103.zip		1.03
packed		122.7 kB
GIMP OGF/OAF plugin	This plugin allows you to read/write .ogf and .oaf files directly, so you don't have to work with a converter tool. Simply unpack the contents of the file into the Gimp plugin directory	?
Gimpp-ogfoafPlugin.zip		0.11
gimp plugin		9.7 kB
OAF Editor (PR's)	OAF Editor is a Descent 3 OAF (Outrage Animated File) editor.	Peter Runge
oafedit.zip		1
packed java		151.6 kB
Pic View 32	Converts D1 & D2 textures to Descent 3 textures.	Chris Becker
picview_vb01.zip		1.0 beta 01 build 03
packed		459.9 kB
Tycoons Logo Editor	Create and edit multiplayer ship logos.	Tycoon
tyclogoed_v20-p.zip		2.0
installer		2073.3 kB

Other tools	tools\the_rest\	
AutoEdit	A small tool that makes it easier to create mn3 files. It's useful when testing levels and having to repack them again and again.	[DCG]Roadrunner
autoedit.zip		1.0.4
installer		1563.4 kB
LETTER32	This allows you to create briefings. With tutorial!	Topher
briefs32.zip		1.0 Beta 01(Build05)
packed		856.3 kB
D3 Quick Test 2.1	Quicktest as a standalone app. Practical if you're not at the level but in.gam's makes changes, you don't have to start D3Edit every time	Anthony Galica
D3Quicktest21.zip		2.1
packed		40.6 kB
D3Cleaner	With this tool you can start D3 faster. You can configure a pilot, its sounds and graphics, and you can also set command line switches	Road runners
d3cleanerfull.zip		0.0.78
packed		97.9 kB
D3MInstaller 1.0	An all-in-one setup program that supports and makes installing D3 mods easier. Simply specify a custom data file, eula file and the main mod file (.d3m) and you have an installer!	DCrazy
d3minst.zip		1.0
packed		13.2 kB
Descent3 Mission Browser	Shows information such as author, URL, etc. of the mission files. Is also very useful for server OP's, because the program also has an extract function with which you can unpack the OOF's.	Lex
d3msninfo.zip		1.0
packed		215.2 kB
Unstrung	This is used to edit the 'string' files; HUD messages or other.	Josh Allen
UnStrung03.zip		0.3
installer		1718.0 kB



... back

In-game help for testing

Ragil Ral

There are 'cheats', at least that's how they are used, which make developing & testing a little easier. Simply enter it in-game like a cheat, here is a list:

framelength || frametime || frame timer

Shows you the current refresh rate. If there is a problem with the display in the game, you can easily find the places that are still need to be optimized.



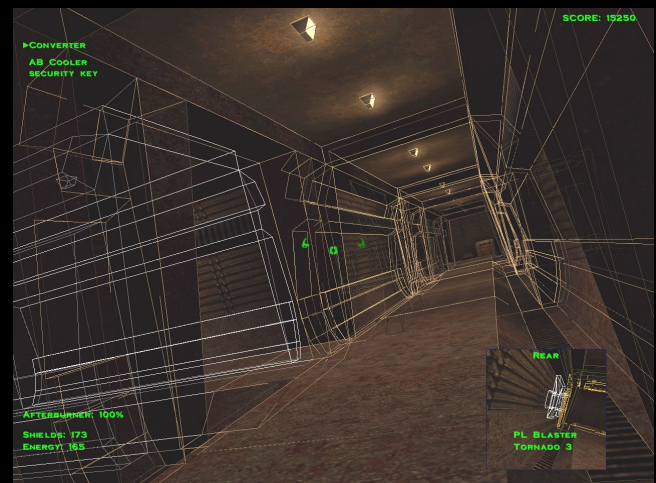
renderstat

Shows you the vertices and polygons currently displayed on the screen. However, the ones that the graphics card calculates and not those from D3Edit!



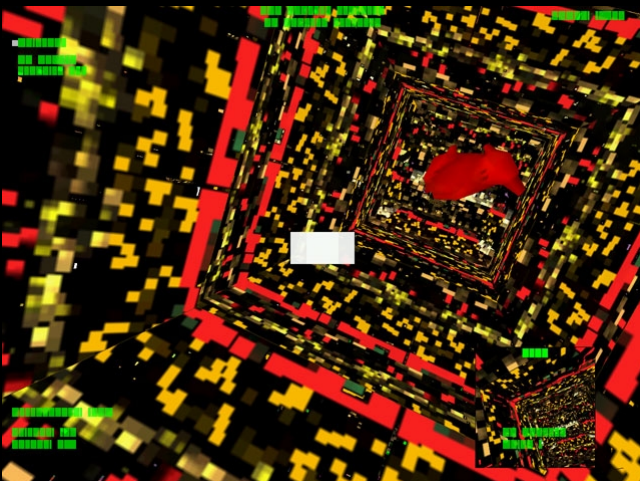
outlinem

When you test your level, you'll see it in the textured wireframe, about three rooms across. However, terrain and objects are not outlined

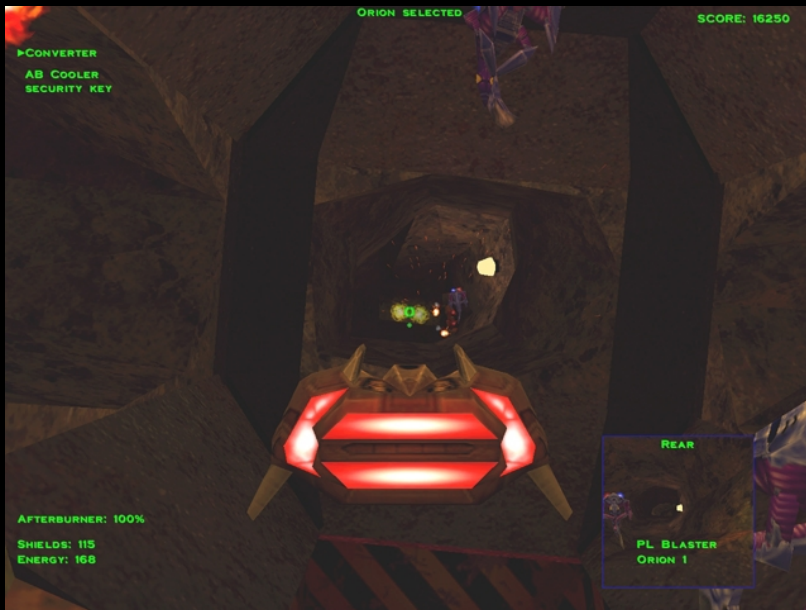


weirdtexture || shananigans

Mix up the textures... maybe an Easter egg from the developers?



Press and hold the key (circumflex accent) and shoot a flare at a face: you get the room and the number; helpful with
Aligning the textures.



byebyemonkey

Chase View - you see yourself from a position behind and slightly above your pyro.

These can also be helpful when developing single player, although they are 'real' cheats:

moreclang
jump to a level; then
just the number
input.



Then there are still

treesquid Complete map
deadofnight removes all bots from the Cloak
testicus level

. . . There is another one for invulnerability, but I won't reveal it now :]

D3-Edit Limits

(LL)Atan

v1.0

To ensure that everything always goes well, Atan has published this list. You should take these limits seriously and not exceed them, as I know from my own experience I just HAD to try it out

Mine:

MAX_LEVELS_PER_MISSION	30	
MAX_ROOMS	400	// 0-399 (Mine rooms)
MAX_PALETTE_ROOMS	50	// max number of loaded rooms (400-449) //
MAX_OBJECTS	1500	total number of objects in world
MAX_POLY_MODELS	1000	// total number of POLY_MODELS in world //
MAX_ACTIVE_DOORWAYS	30	specific doors inside mine -still unclear
MAX_DOORS	60	// predefined Doors usable by Editor
MAX_TRIGGERS	100	
MAX_PATHS	100	
MAX_PATH_PORTALS	40	
MAX_GAME_PATHS	300	
MAX_NODES_PER_PATH	100	
MAX_GOAL_ITEMS	12	
MAX_LEVEL_GOALS	32	
MAX_GOAL_LISTS	4	
MAX_WAYPOINTS	25	
MAX_SOUNDS	1000	

Rooms:

MAX_REGIONS_PER_ROOM	200	// Facestructures, not connected to room shell
MAX_Faces_PER_ROOM	3000	
MAX_VERTS_PER_ROOM	10000	
MAX_BNODES_PER_ROOM	127	

Faces:

MAX_VERTS_PER_Face	64
--------------------	----

Textures:

MAX_BITMAPS	5000	// Mine
MAX_TEXTURES	2600	//Room
MAX_LIGHTMAP_TEXTURES	60	// Mine
MAX_VCLIPS	200	// OAF
VCLIP_MAX_FRAMES	50	// oaf frames
MAX_BUMPMAPS	500	//
MAX_FORCE_FIELD_BOUNCE_TEXTURES	3	// TF_FORCEFIELD
MAX_SPECIAL_Faces	13000	// TF_METAL + TF_MARBLE TF_PLASTIC

Objects:

MAX_MODEL_TEXTURES	35
MAX_POLYGON_VECS	2500
MAX_DETAIL_LEVELS	3
MAX_PROP_LEN	256
MAX_NAME_LEN	32
MAX_GROUND_PLANES_PER_MODEL	10
MAX_GUNS_PER_MODEL	64
MAX_SUBOBJECTS	30
MAX_POINTS_PER_SUBOBJECT	300
MAX_WB_GUNPOINTS	8th
MAX_WB_FIRING_MASKS	8th
MAX_WB_TURRETS	8th

Application of D3 Levelinfo

Atan

Version BETA 1.3

New since 1.2:

With ~~-~~ on the number pad you jump to the previous valid room. With ~~+~~ on the number pad you jump to the next valid room. Jump in terrain removed

Jump into doors removed

New since 1.0:

With 'comma' on the number pad you jump to the previous valid space.

New since 0.9:

Pressing 'Enter' on the number pad jumps to the next valid room.

New since 0.8:

Freeze bug fixed D3 would hang when flying from internal areas to external areas.

New checks on:

Nonplanar faces

Flipped Faces

Missing Faces

Unused Verts

Double Verts

built-in

and added visual support to make it easier to find errors.

Menu:

Number pad ~~/~~ activates/deactivates the extended check mode. In rooms with many faces, this check slows down the D3 engine.

What is D3 LevelInfo?

LevelInfo is intended to be a little helper with which you can examine your D3 levels, or rather, with which you can get information while flying through the level. It checks the known limits, shows the room names and what kind of room you are currently flying through, as well as which flags are set. This can be helpful for Entropy Level.

That's why it's more powerful than Renderstat.

This little mod has to be activated in multiplayer mode, for this you have to prepare a few little things:

A)

this step must be carried out once:
the file `check.d3m` to the `Descent3/netgames` folder.

B)

For each level to be checked with D3 LevelInfo:

Since versions > 0.5 it should also work without Step B! Try it. If it doesn't work, please continue reading here:

Open the level in D3Edit, start Quicktest, and search for the Keywords section. Mark anarchy, even in single player levels that should be checked!

Add the keyword:сheckand save the level again.

C)

* * !!! No reason to go online if you want to use LevelInfo!!! **

start D3,
choose multiplayer,
choose DIRECT TCP/IP,
select Start New Game.

On the right is: GAME TYPE,
select CHECK,
If CHECK is not visible there, you may have forgotten step A or you have to scroll through the mod types to see the mod.

Now select the version to be tested on the left.

D)

Start the game

If the level is not compatible with the mod, Descent3 will issue a message. maybe you have the keywordсhecknot registered, or the filecheck.d3mnot copied to the Descent3/netgames folder.

But if everything went well, then you should now be in the level, ignore 'waiting for players' and start the game.

On the top right of the screen you should see the 'Key menu'.

Insert: Shows room information **Pos1:**Shows region information **ImageUp:** Shows special room information **Remove:** Toggle font size **End:** Shows version number **ImageDown:** Show/Hide the 'key-menu'

Press '**Insert**'

Information about the current room should now be visible on the left side of the screen.

Fly through the level,
the information changes as soon as you fly into a room.
The object information changes when you pick up an object, for example.

If a problem is detected, a message is issued.

Because the region check works very slowly, you can use it**Pos1**To switch on and off.

The font size can be set with 'Remove' can be switched.

'End' shows the version number of the mod

If you want to see information about an Entropy level, press 'ImageUp' and the information is then displayed at the bottom right.

With 'PageDown' the 'Key Menu' is switched on and off.

With '/' on the numeric keypad turns the advanced checks and visual effects on/off

With 'Enter' on the number pad jumps to the next valid room with ',' on the number pad jumps to the previous valid space

This little mod was created during my experiments, use it at your own risk. No guarantee of correct results.

I do not provide support, but I may expand the tool from time to time

Note, this mod does not replace careful work in D3Edit. Every error can be found there in advance.

Documentation of the doors in Descent 3

Ragil Ral

Door name:

PTMC Industrial 2

Size about.:

20x18 units

Verts of the front face:

Thickness approx.:6 units

Has region built-ins:

No

Bugs:

7 duplicate verts, 6 unused verts, 4

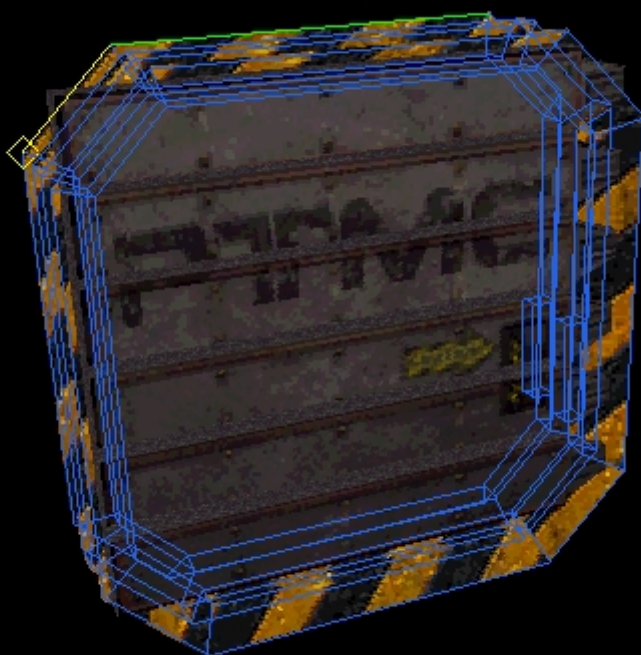
T-joints

Miscellaneous:

The T-joints are located on the faces, which become portals when the door is installed.

Repair=Must!

Many faces of the door are triangulated, uniting with **Ctrl-Shift-LClick** is recommended.



Door name:

PTMC Industrial 4

Size about.:

20x20 units

Verts of the front face:

Thickness approx.:6 units

Has region built-ins:

No

Bugs:

8 Duplicate Verts

Miscellaneous:

-

Door name:

SEANS STEAMVENT DOOR

Size about.:

50x35 units, front face 30x30

Verts of the front face:8th

Thickness approx.:23 units **Has region built-ins:**

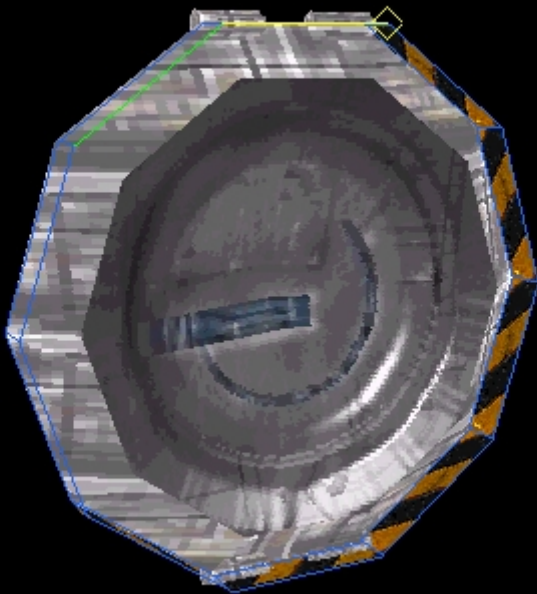
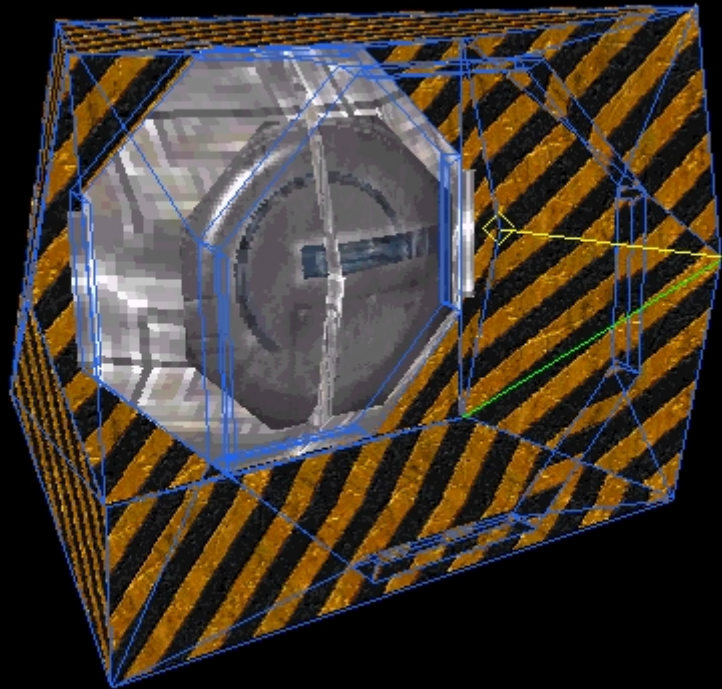
No

Bugs:

8 duplicate verts, 8 unused verts

Miscellaneous:

Once the door has been flown through, a sound is played. The door is a good connection to the terrain.



Door name:

SEAN'S HEATSINK DOOR

Size about.:

Diameter 35 units

Verts of the front face:10 **Thickness approx.:**2.5 units, 5 with the Door object

Has region built-ins:

No

Bugs:

10 Duplicate/Unused Verts

Miscellaneous:

Is the SEANS STEAMVENT DOOR from the other side.

Door name:
DAN'S VAULT DOOR

Size about.:
200x170 units

Verts of the front face:

12, Diameter 140

Thickness approx.:71 units

Has region built-ins:

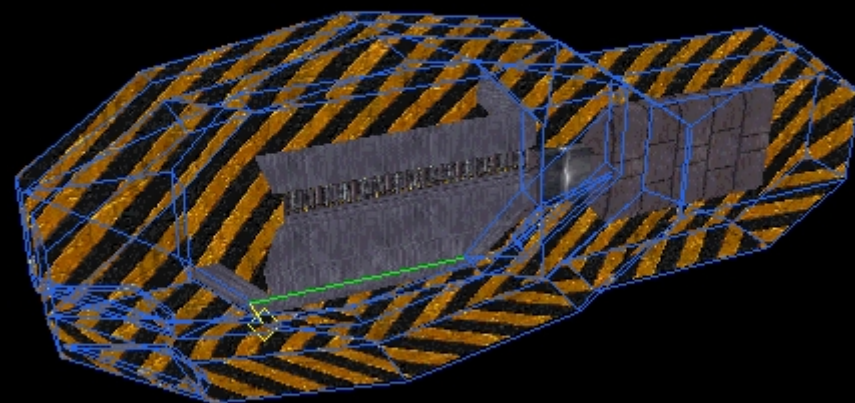
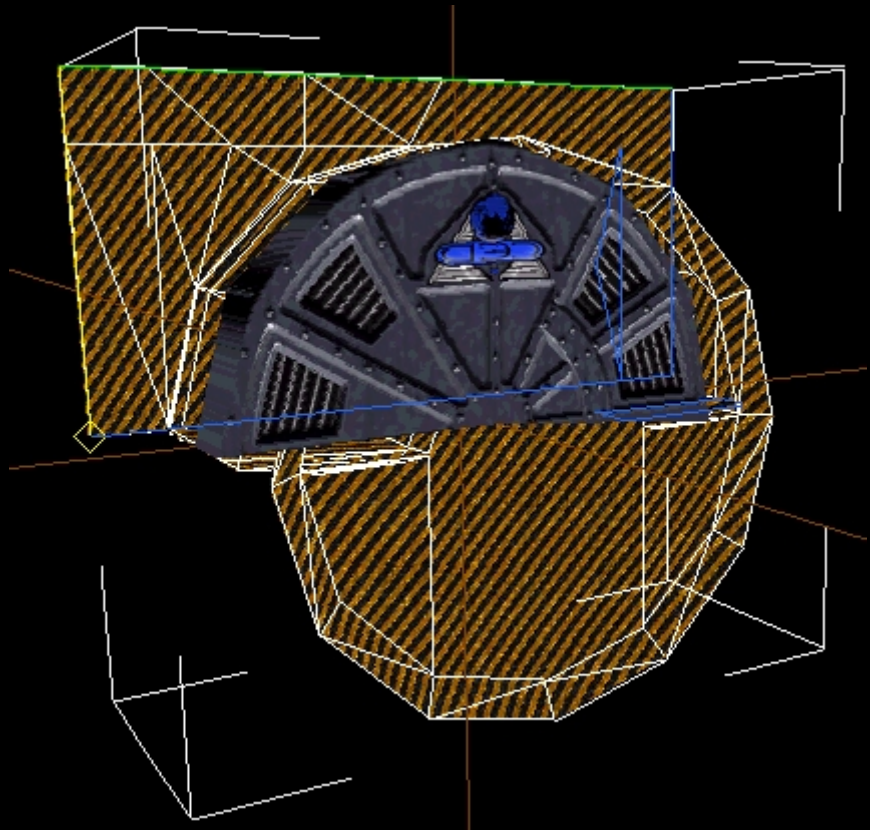
Yes

Bugs:

13 duplicate verts, 12 unused
verts, 4 T-joints

Miscellaneous:

The T-joints are located on the faces,
which become portals when the door
is installed. Repair=Must! Many faces
of the door are
triangulated, a uniting with **Ctrl-
Shift-LClick** is recommended. The
door is a little rough and the repair
and optimization effort is a little
higher.



Door name:

IBD 1

Size about.:

80x30 units, FF 60x30

Verts of the front face:10

Thickness approx.:14 units

Has region built-ins:Yes

Bugs:

10 duplicate/unused verts, 2 T-joints

Miscellaneous:

-

Door name:
SEAN'S TRAPDOOR

Size about.:
28x26 units

Verts of the front face:4

Thickness approx.:16 units

Has region built-ins:

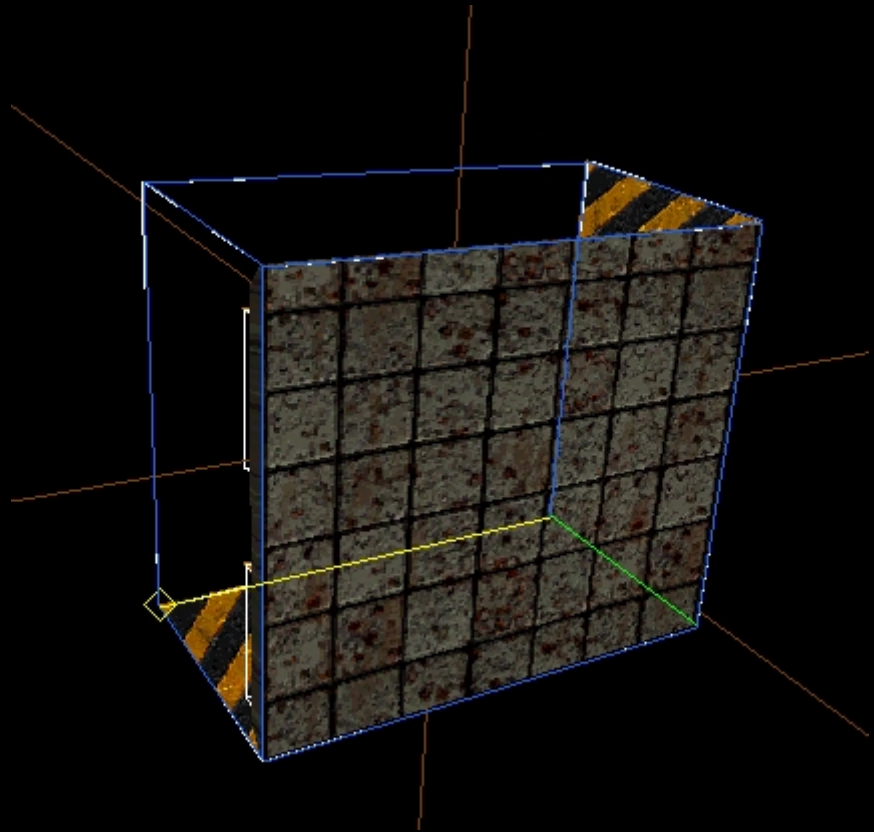
Yes

Bugs:

4 duplicate / unused verts

Miscellaneous:

The door object has 4 concave faces. Repair recommended.

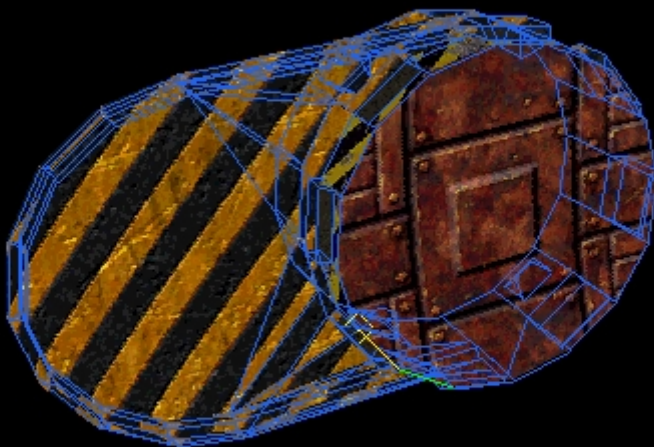


Door name:
LUKE'S SECRET DOOR

Size about.:
35x20 units, FF diameter 19
Verts of the front face:16
Thickness approx.:9 units
Has region built-ins:Yes
Bugs:16 duplicate / unused verts, 3 T-joints

Miscellaneous:

The door should be optimized.



Door name:
PTMC Industrial 5

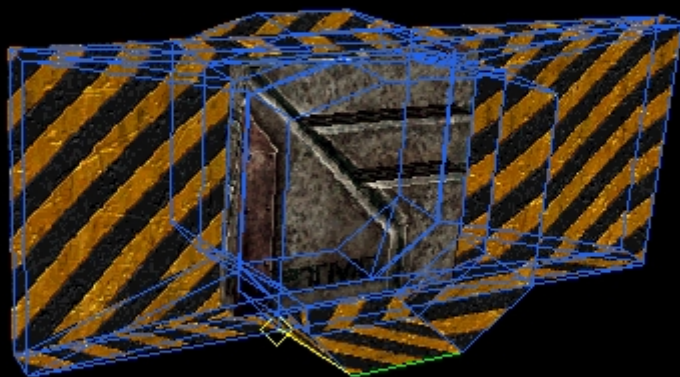
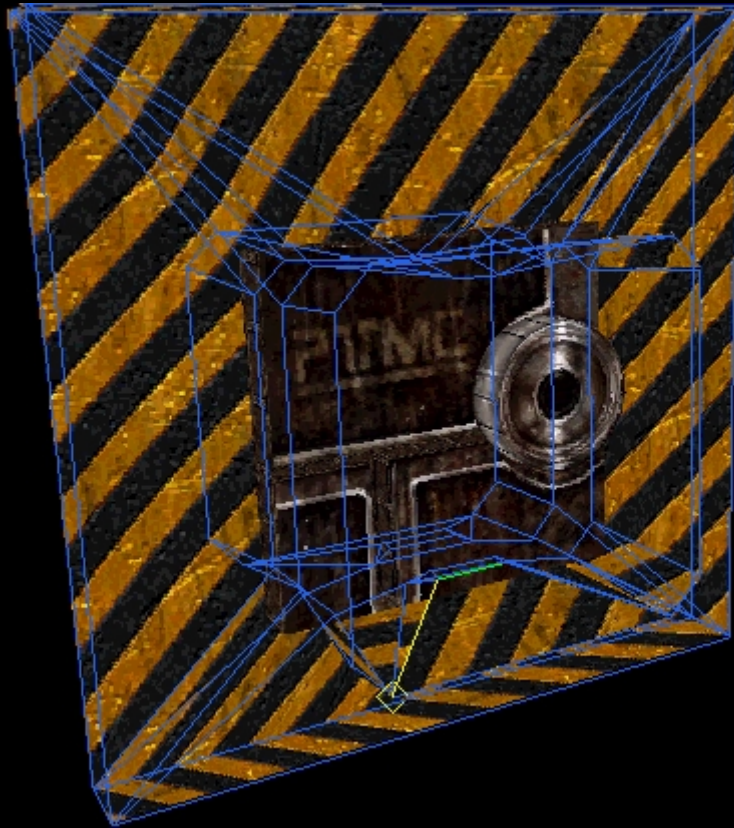
Size about.:
38x38 units, FF 20x20 Verts
of the front face:8th

Thickness approx.:20 units

Has region built-ins:
No

Bugs:
8 Unused Verts

Miscellaneous:
-



Door name:
PTMC Industrial 6

Size about.:
55x22 units, FF 20x20 Verts
of the front face:8th

Thickness approx.:20 units

Has region built-ins:No Bugs:8
Duplicate / Unused Verts

Miscellaneous:
A few faces need to be optimized.

Door name:

PTMC Industrial 7

Size about.:

21x63 units, FF 20x20 Verts

of the front face:10

Thickness approx.:20 units

Has region built-ins:

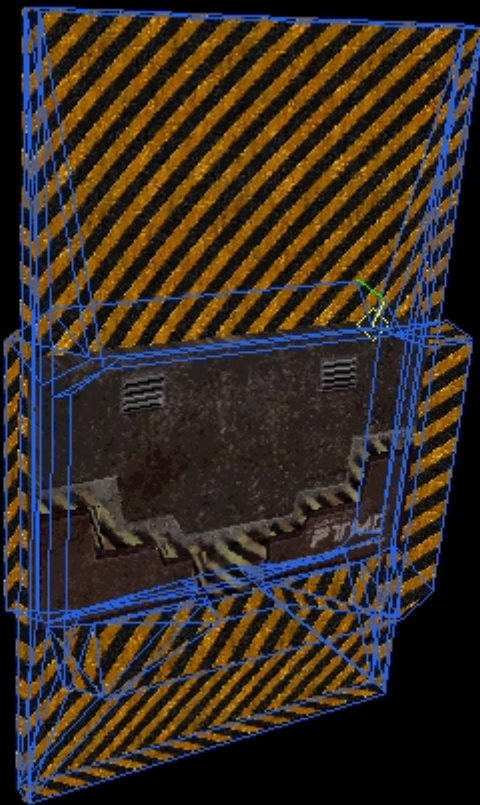
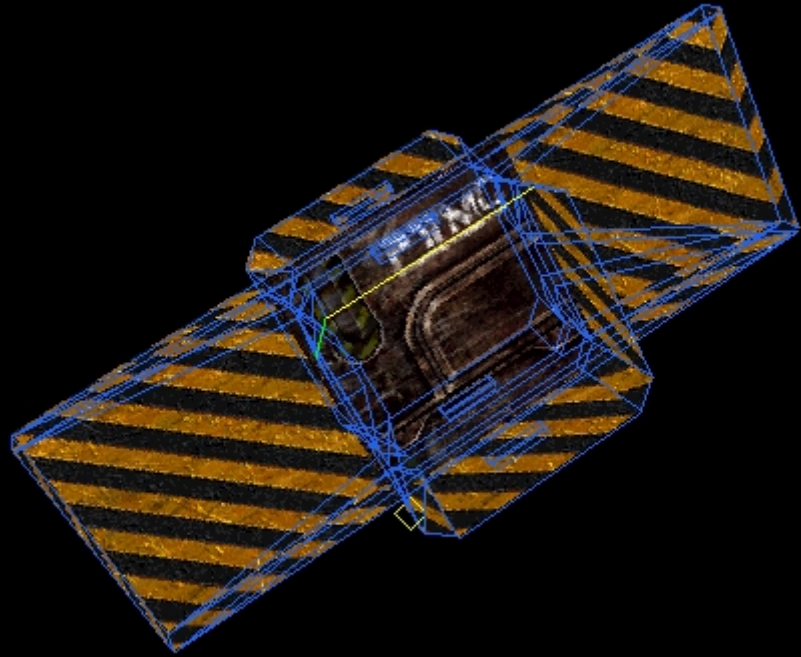
Yes

Bugs:

10 duplicate / unused verts, 1 T-joint

Miscellaneous:

The faces that become portals have recommended extra verts on the edges -> optimization.



Door name:

PTMC Industrial 3

Size about.:

62x105 units, FF 60x46 Verts

of the front face:8th

Thickness approx.:20 units

Has region built-ins:No Bugs:8

Duplicate / Unused Verts

Miscellaneous:

The door would be complex, but could still be optimized.

Door name:
SEAN'S ENERGY SECRET

Size about.:
26x53 units, FF 20x25 Verts
of the front face:4

Thickness approx.:26 units

Has region built-ins:

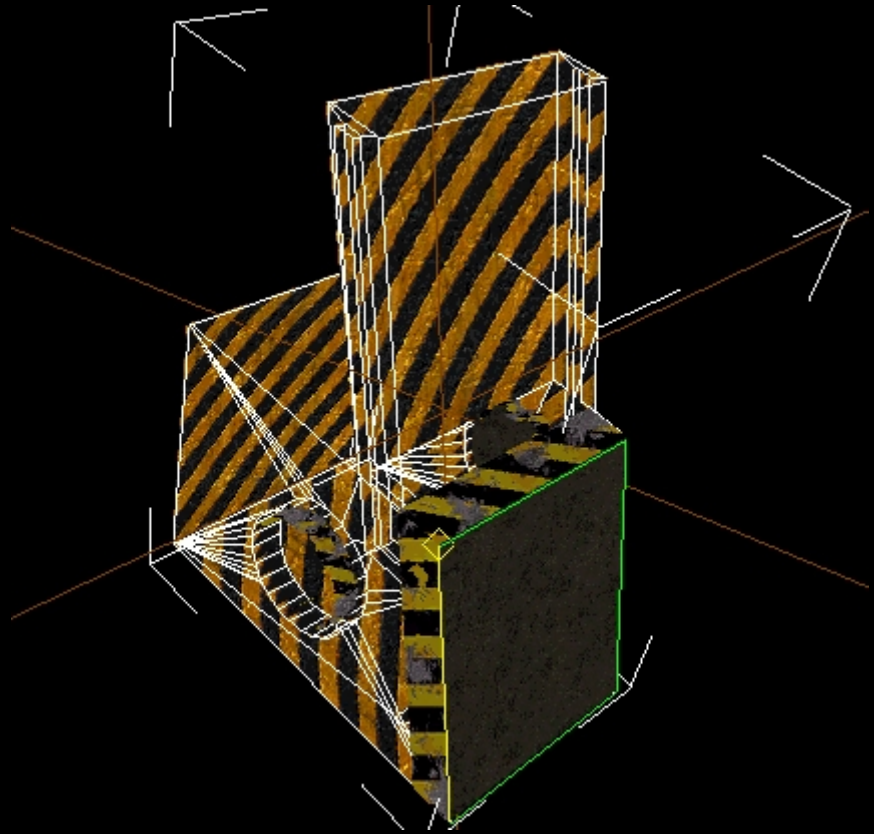
No

Bugs:

4 duplicate / unused verts

Miscellaneous:

Portal faces not plane-parallel.

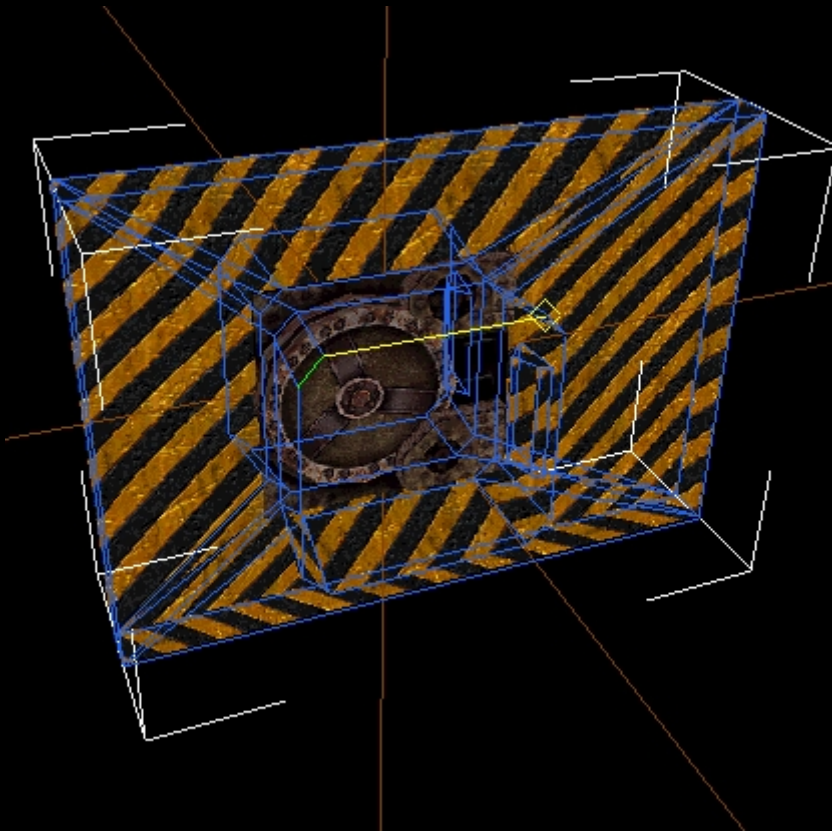


Door name:
PTMC Industrial 1

Size about.:
52x39 units, FF 20x20 Verts of the
front face:8th Thickness approx.:20
units Has region built-ins:Yes Bugs:8
Duplicate / Unused Verts

Miscellaneous:

-



Door name:

PTMC Covert 1

Size about.:

39x41 units, FF 20x20 Verts

of the front face:8th

Thickness approx.:20 units

Has region built-ins:

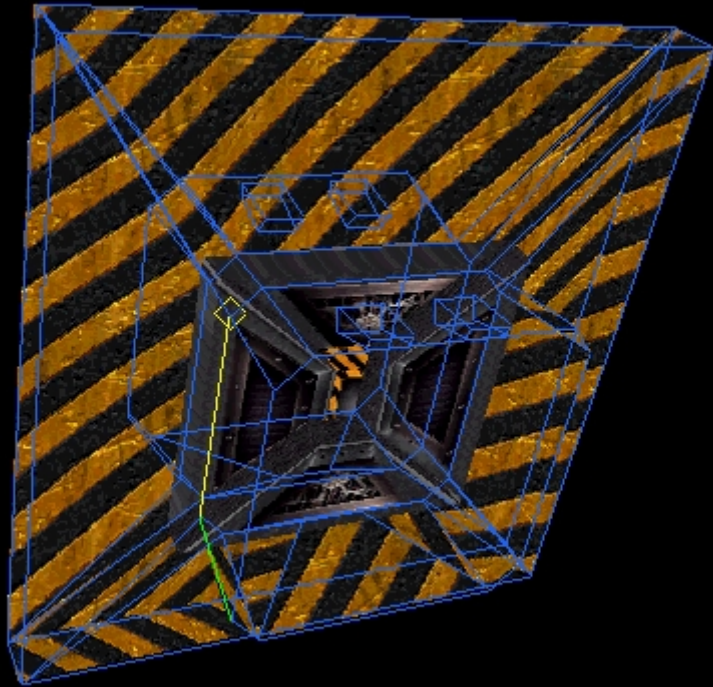
Yes

Bugs:

8 duplicate / unused verts

Miscellaneous:

-



Door name:

ced_acc_door

Size about.:

38x31 units, FF 17x28 Verts **of the front**

face:8th **Thickness approx.:**21 units **Has**

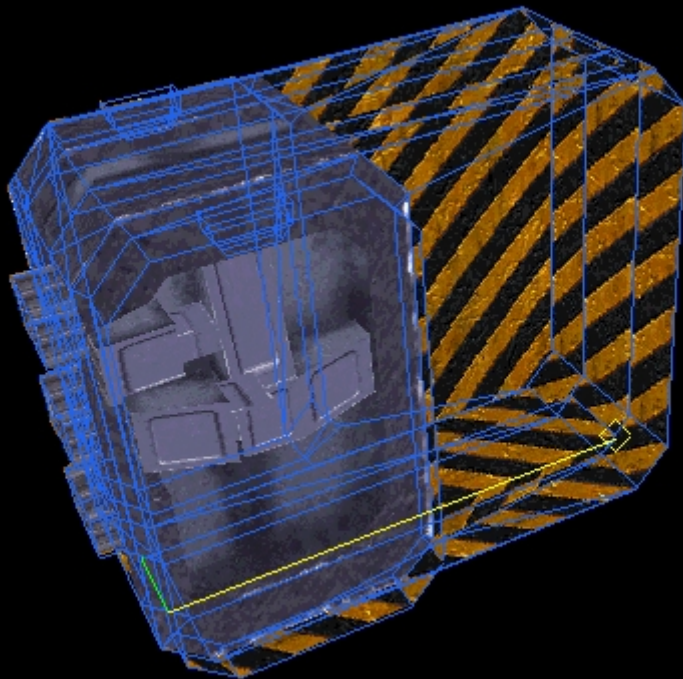
region built-ins:Yes **Bugs:**8 duplicate /

unused verts, 8 T-joints

Miscellaneous:

The page where you would continue building is split -> combine faces. Gaps open when errors are ironed out.

The door should be optimized.



Door name:
ced_wide_door

Size about.:
56x36 units, FF 50x17 Verts
of the front face:4

Thickness approx.:19 units

Has region built-ins:

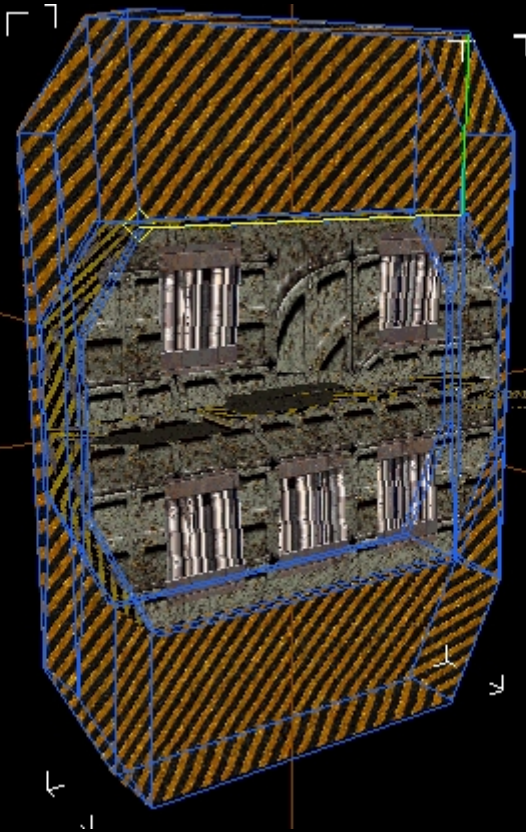
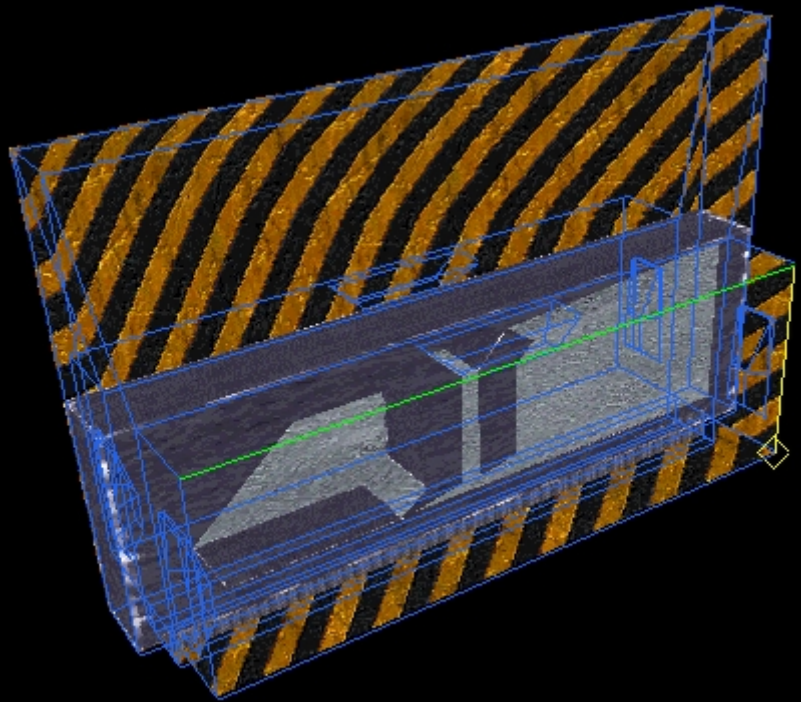
Yes

Bugs:
4 duplicate / unused verts, 16
gaps

Miscellaneous:

The substructure must be
optimized.

The door is not without its
problems; It happened to me
twice that she corrupted a level.



Door name:
PTMC Solid Door

Size about.:
106x150 units, FF 102x76 Verts
of the front face:8th Thickness
approx.:20 units

Has region built-ins:No **Bugs:**8
duplicate / unused verts, 8 gaps

Miscellaneous:

The door object has concave faces,
the shell is not closed.

Should be repaired.

Optimization recommended.

Door name:

exp_cargo_door

Size about.:

106x150 units, FF 102x76

Verts of the front face:8th

Thickness approx.:20 units

Has region built-ins:

No

Bugs:

8 duplicate / unused verts, 8 gaps

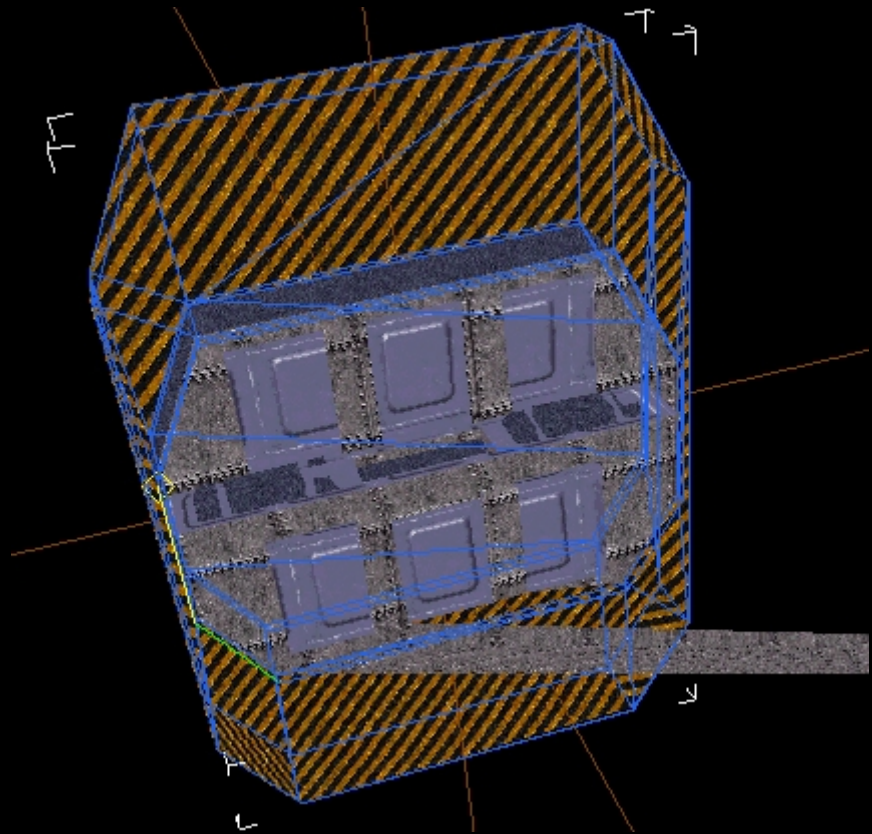
Miscellaneous:

The door object has concave faces, the shell is not closed.

Repair=Must!

The side where you would continue building is split - a face union with **Ctrl-Shift-LClick** is to be carried out.

Optimization recommended.



Door name:

PTMC Corporate A

Size about.:

49x24 units, FF 23x22 Verts

of the front face:8th

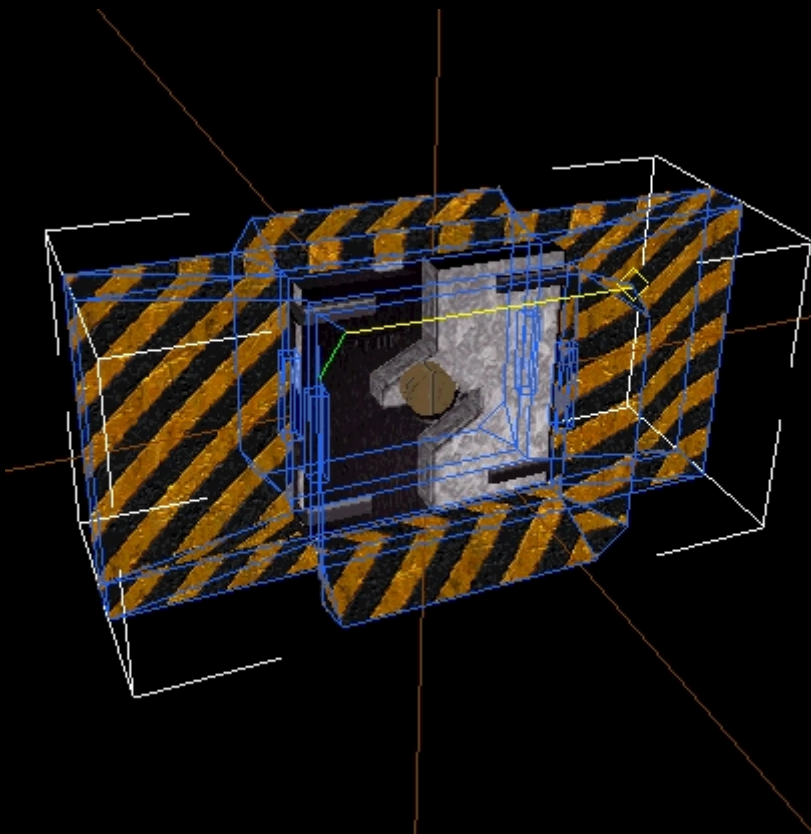
Thickness approx.:20 units

Has region built-ins:No Bugs:8

Duplicate / Unused Verts

Miscellaneous:

-



Door name:

PTMC Corporate B

Size about.:

34x25 units, FF 23x22 Verts

of the front face:8th

Thickness approx.:20 units

Has region built-ins:

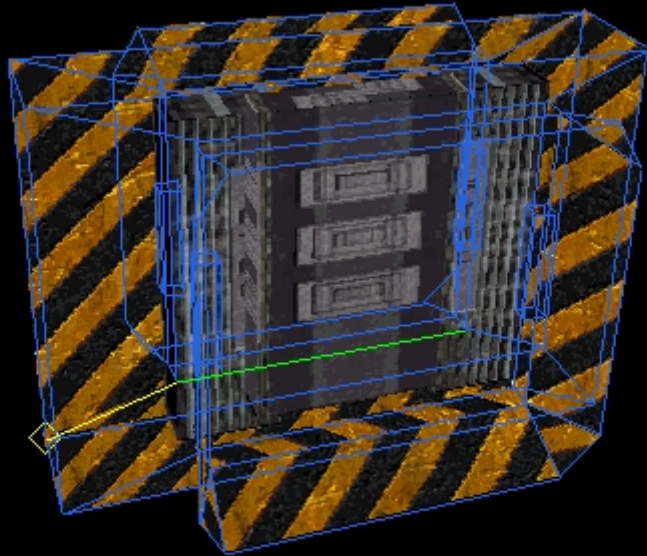
Yes

Bugs:

8 duplicate / unused verts

Miscellaneous:

Very compact door.



Door name:

PTMC Corporate E

Size about.:

49x31 units, FF 19x22 Verts

of the front face:8th

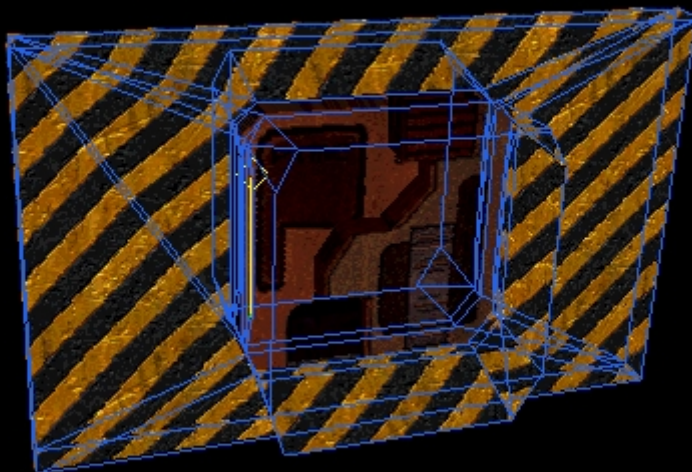
Thickness approx.:20 units

Has region built-ins:No Bugs:8

Duplicate /Unused Verts

Miscellaneous:

Door can be optimized.



Door name:

PTMC Covert A

Size about.:

31x48 units, FF 21x21 Verts

of the front face:4

Thickness approx.:20 units

Has region built-ins:

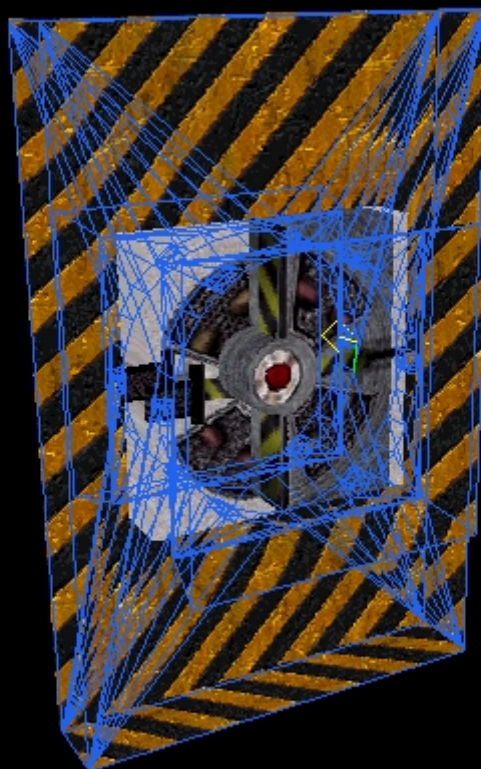
Yes

Bugs:

4 duplicate / unused verts

Miscellaneous:

-



Door name:

PTMC Industrial A

Size about.:

54x42 units, FF 19x22 Verts of the

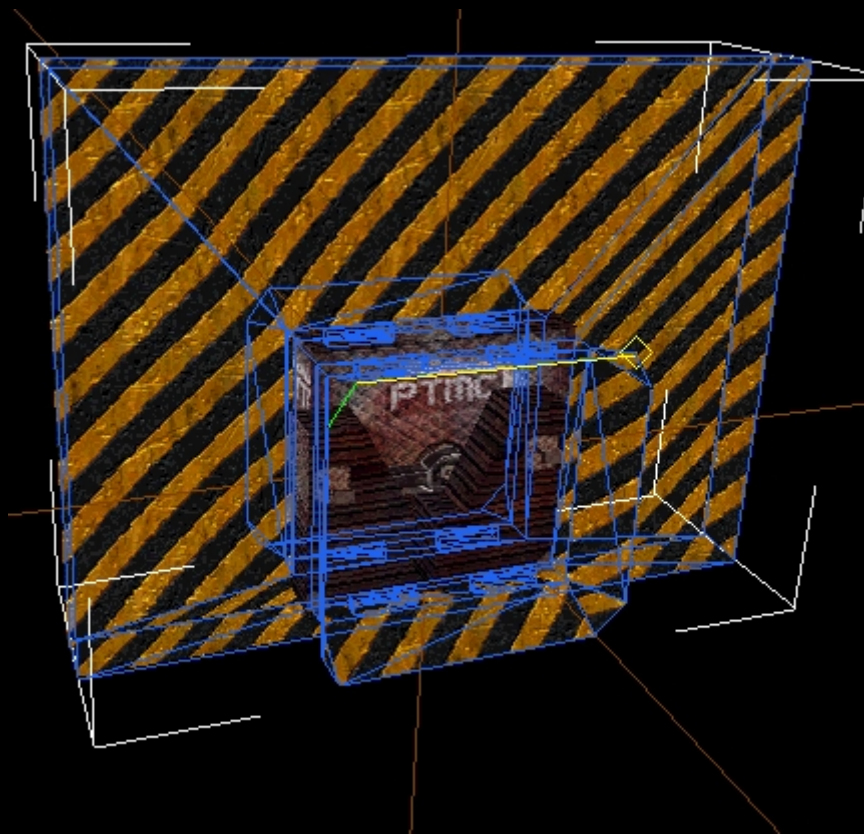
front face:8th Thickness approx.:20

units Has region built-ins:Yes Bugs:8

Duplicate / Unused Verts

Miscellaneous:

Shell optimization recommended.



Door name:

Nomad 1

Size about.:

21x46 units, FF 21x21 Verts

of the front face:8th

Thickness approx.:20 units

Has region built-ins:

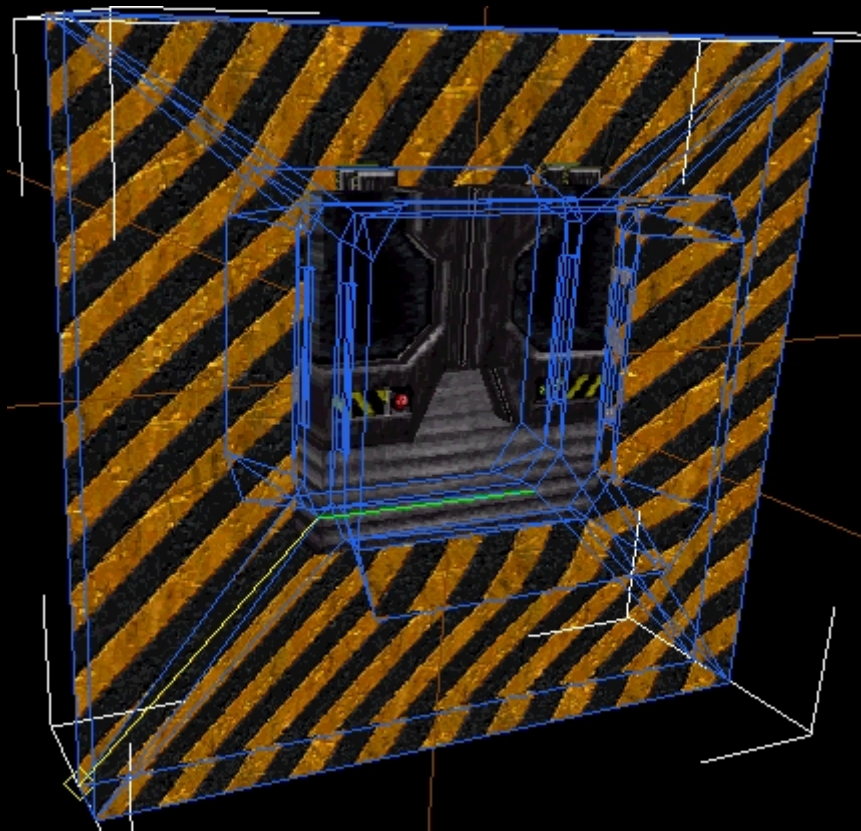
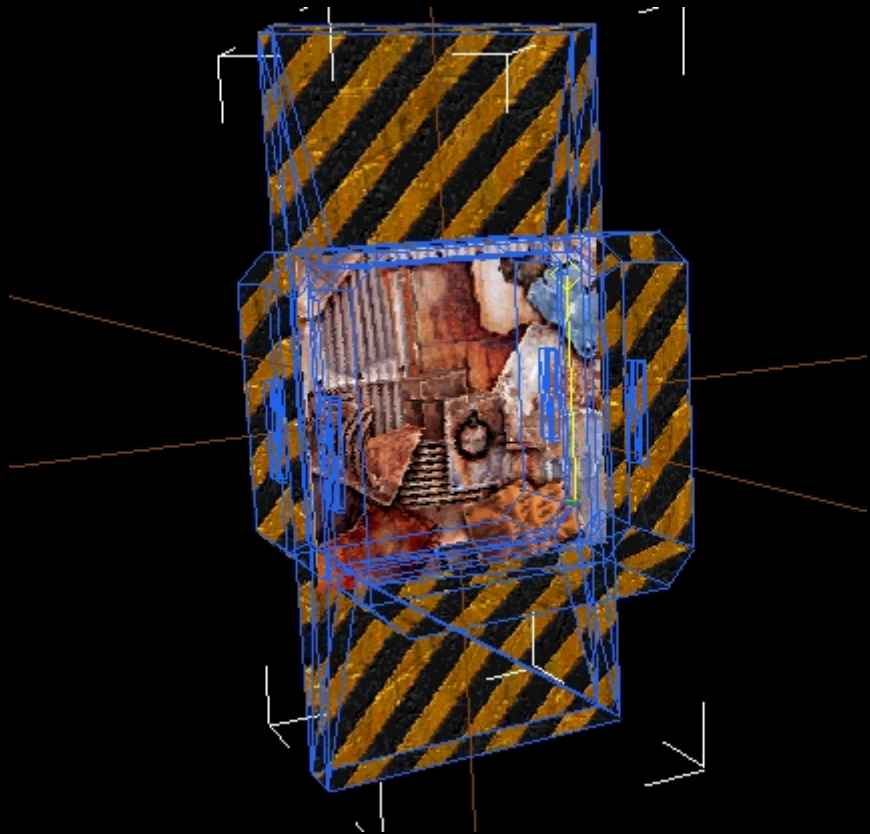
Yes

Bugs:

8 duplicate / unused verts

Miscellaneous:

-



Door name:

PTMC Covert B

Size about.:

45x45 units, FF 23x22 Verts

of the front face:8th

Thickness approx.:20 units

Has region built-ins:No Bugs:8

Duplicate /Unused Verts

Miscellaneous:

Shell could use some optimization.

Door name:

Red Acropolis A

Size about.:

51x53 units, FF 21x21 Verts

of the front face:12

Thickness approx.:20 units

Has region built-ins:

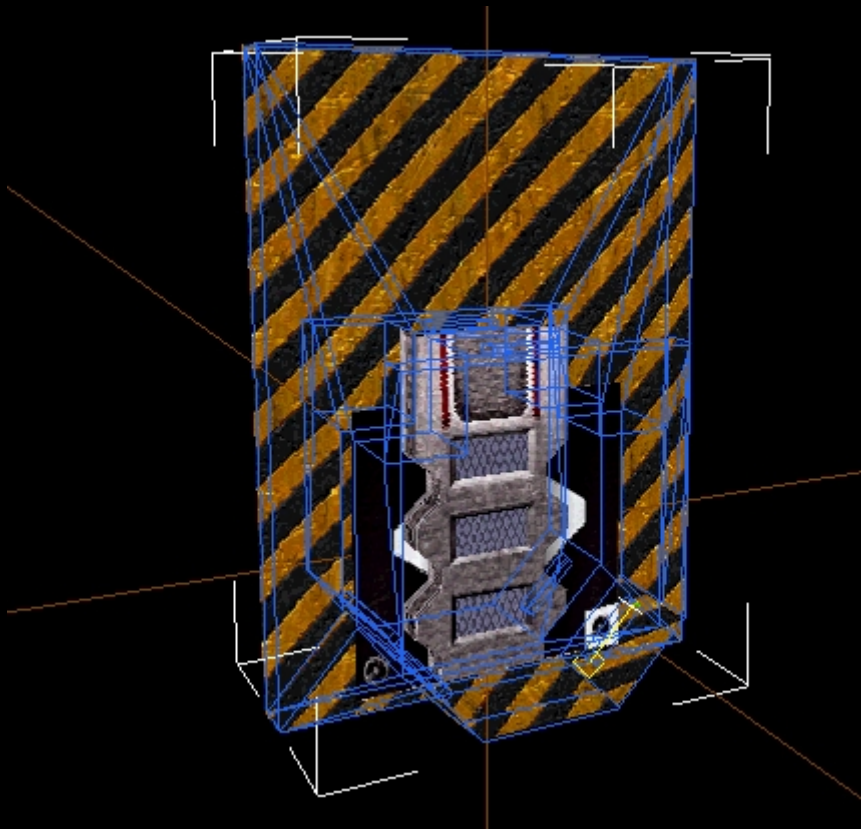
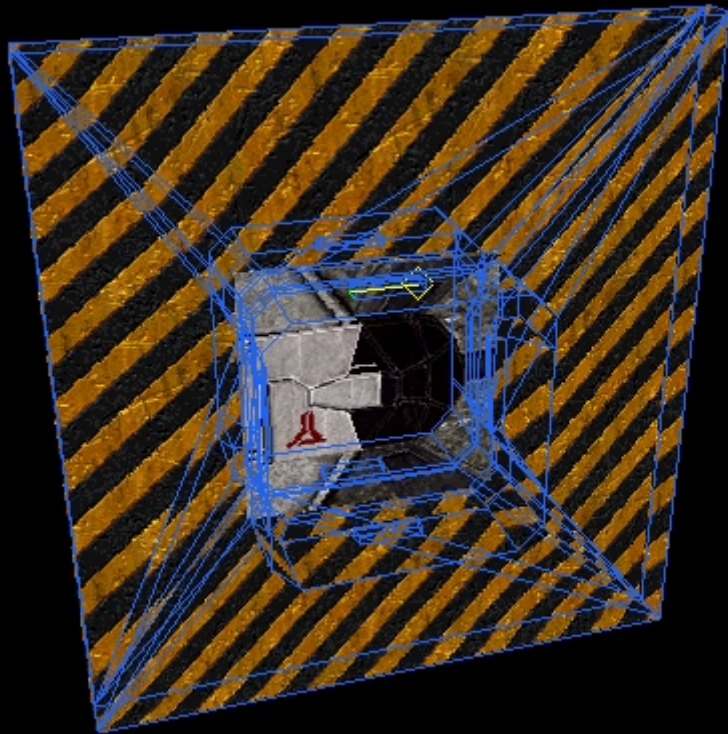
Yes

Bugs:

12 duplicate / unused verts

Miscellaneous:

-



Door name:

IBD station

Size about.:

31x43 units, FF 20x25 Verts of the

front face:6 Thickness approx.:

20 units Has region built-ins:Yes Bugs: 6

Duplicate / Unused Verts

Miscellaneous:

Has the see-through flag

Door name:

Nomad 2

Size about.:

41x37 units, FF 21x21 Verts

of the front face:8th

Thickness approx.:20 units

Has region built-ins:

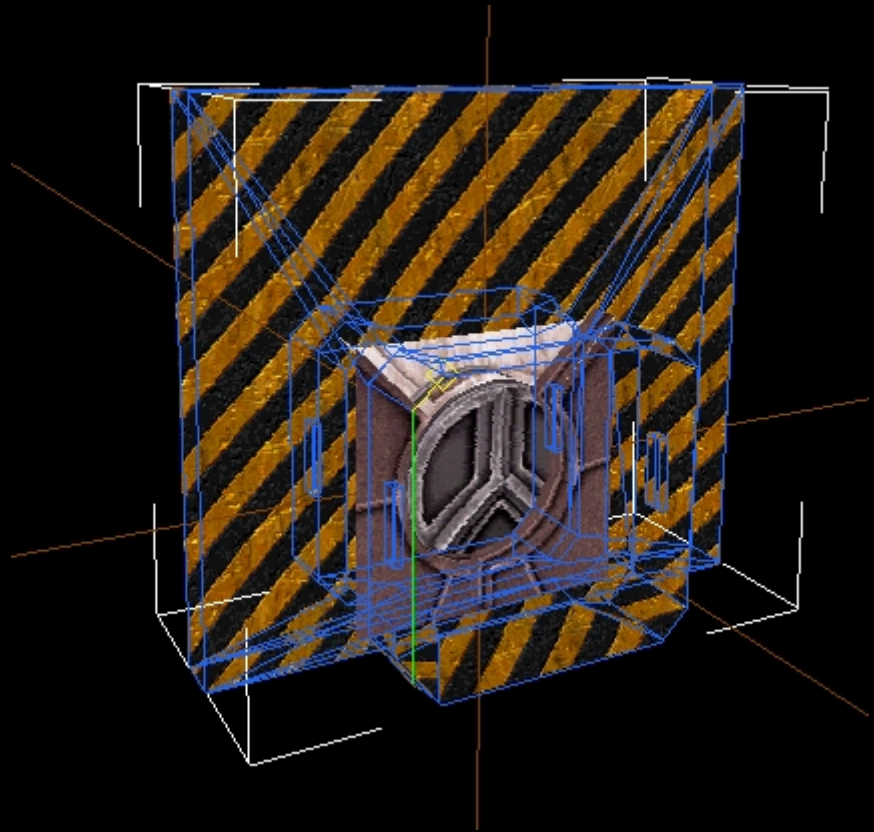
Yes

Bugs:

8 duplicate / unused verts

Miscellaneous:

-



Door name:

Novak A

Size about.:

23x42 units, FF about 22 diameter Verts of

the front face:6 **Thickness approx.:**20

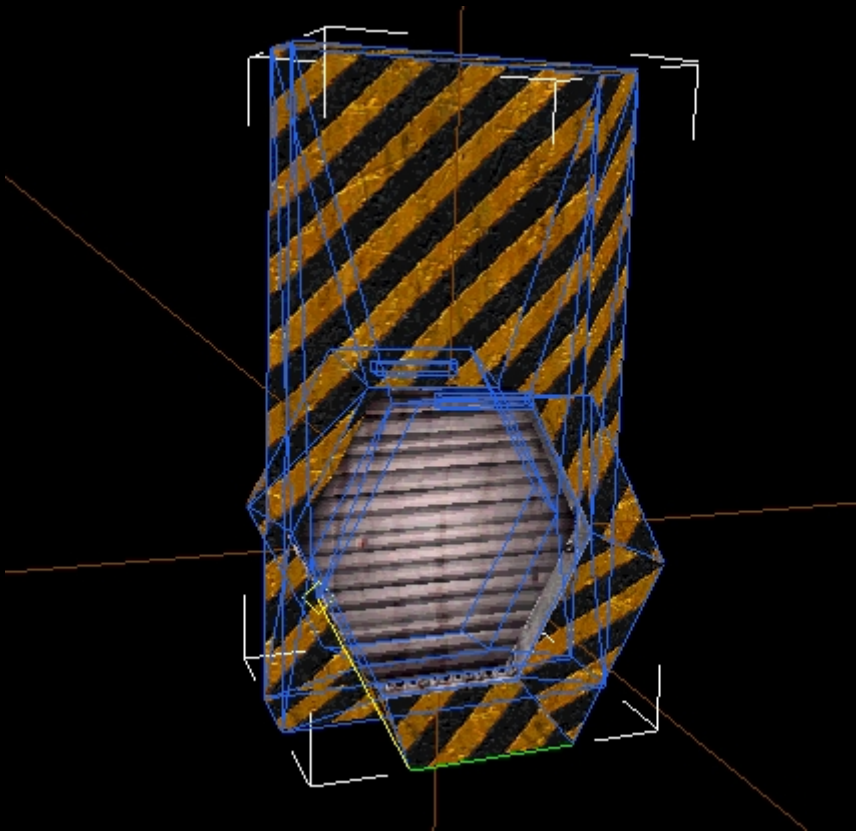
units

Has region built-ins:No **Bugs:**6

Duplicate / unused Verts

Miscellaneous:

You can get stuck in the door with the Magnum



Door name:

Novak B

Size about.:

41x40 units, FF 19x19 Verts

of the front face:8th

Thickness approx.:20 units

Has region built-ins:

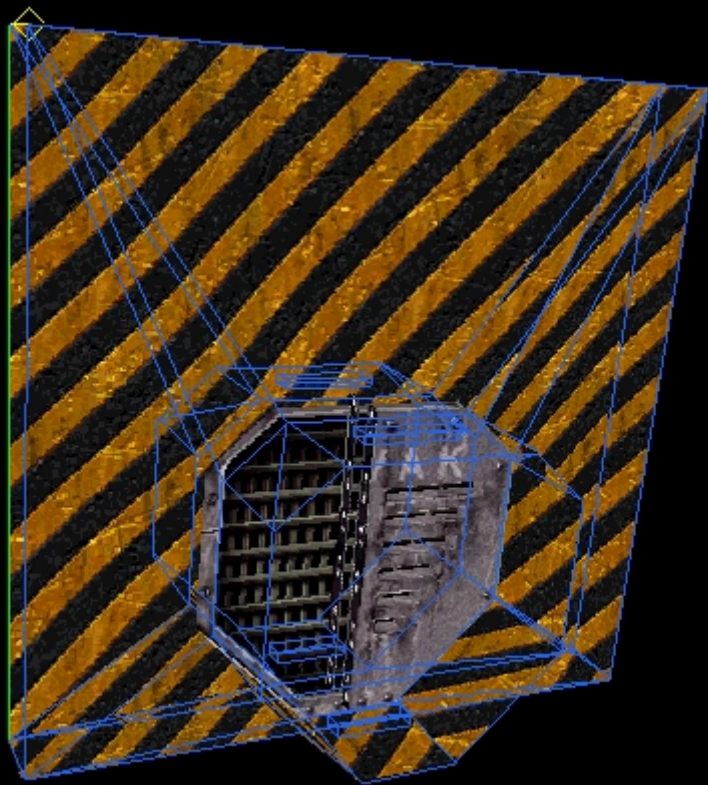
Yes

Bugs:

8 duplicate / unused verts

Miscellaneous:

The Door object has concave faces.



Door name:

PTMC Corporate C

Size about.:

34x20 units, FF 20x19 Verts of the

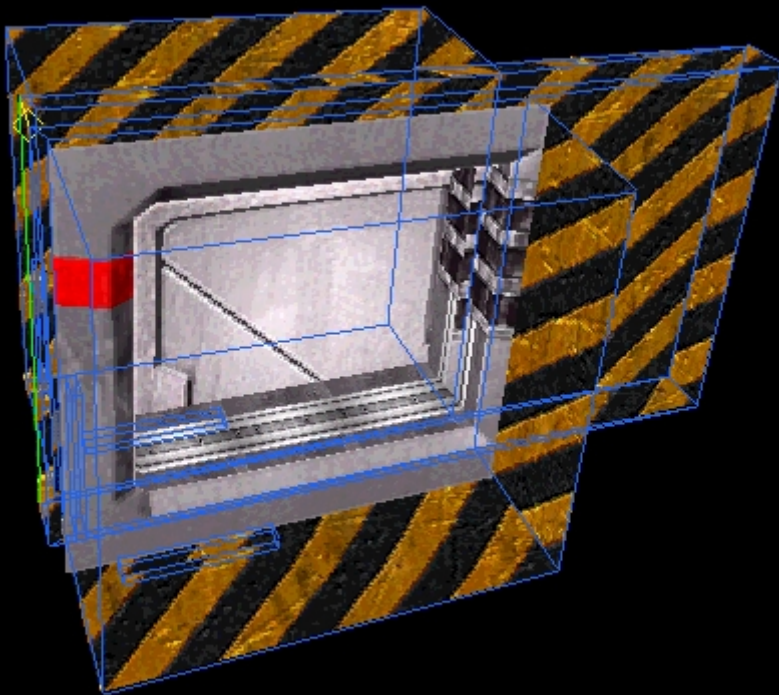
front face:4 Thickness approx.:

20 units Has region built-ins:Yes Bugs:4

Duplicate / Unused Verts

Miscellaneous:

You tend to get stuck in that door



Door name:

PTMC Covert D

Size about.:

40x23 units, FF 21x21 Verts

of the front face:4

Thickness approx.:20 units

Has region built-ins:

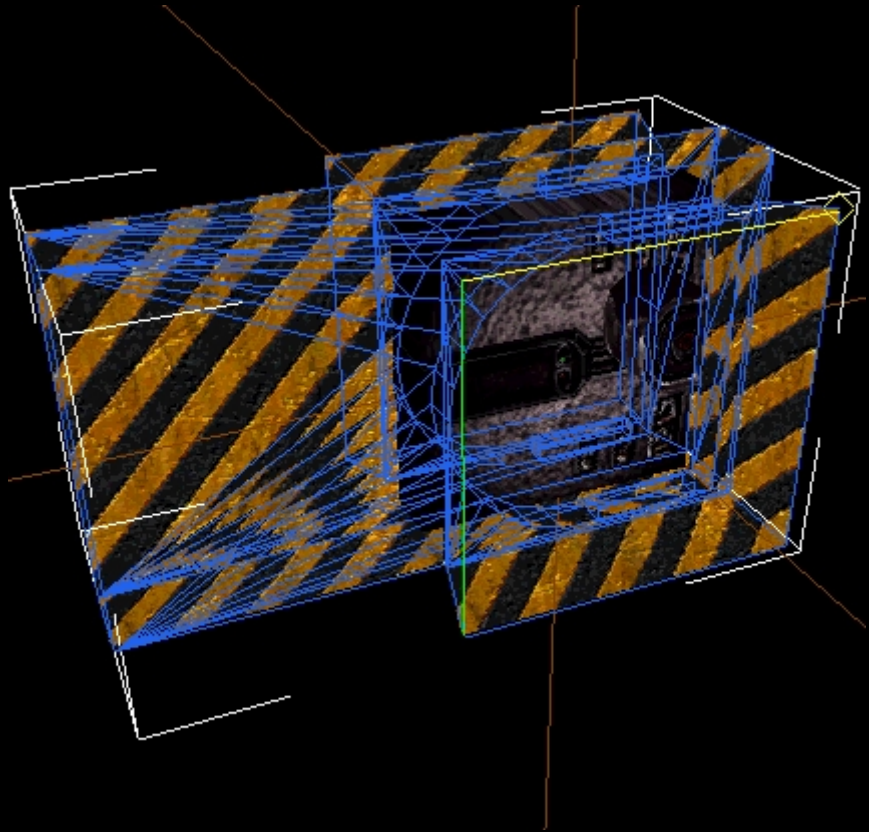
Yes

Bugs:

4 duplicate / unused verts

Miscellaneous:

-



Door name:

PTMC Covert E

Size about.:

60x24 units, FF 21x21 Verts

of the front face:8th

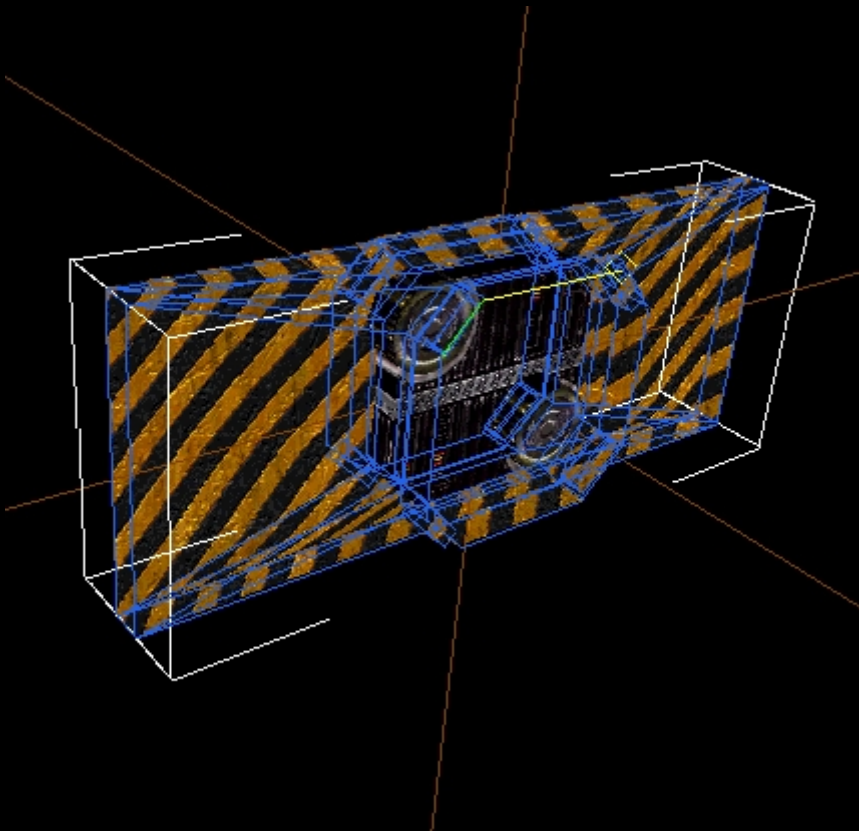
Thickness approx.:13 units

Has region built-ins:No Bugs: 8

Duplicate / Unused Verts

Miscellaneous:

-



Door name:

Red Acropolis B

Size about.:

36x43 units, FF 21x21 Verts

of the front face:8th

Thickness approx.:13 units

Has region built-ins:

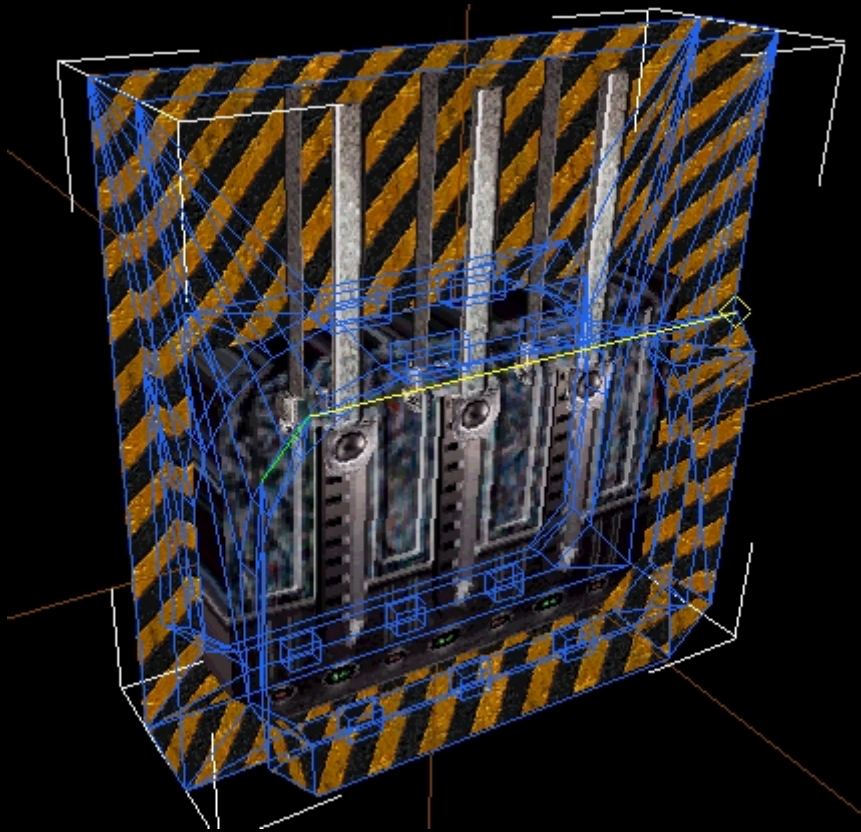
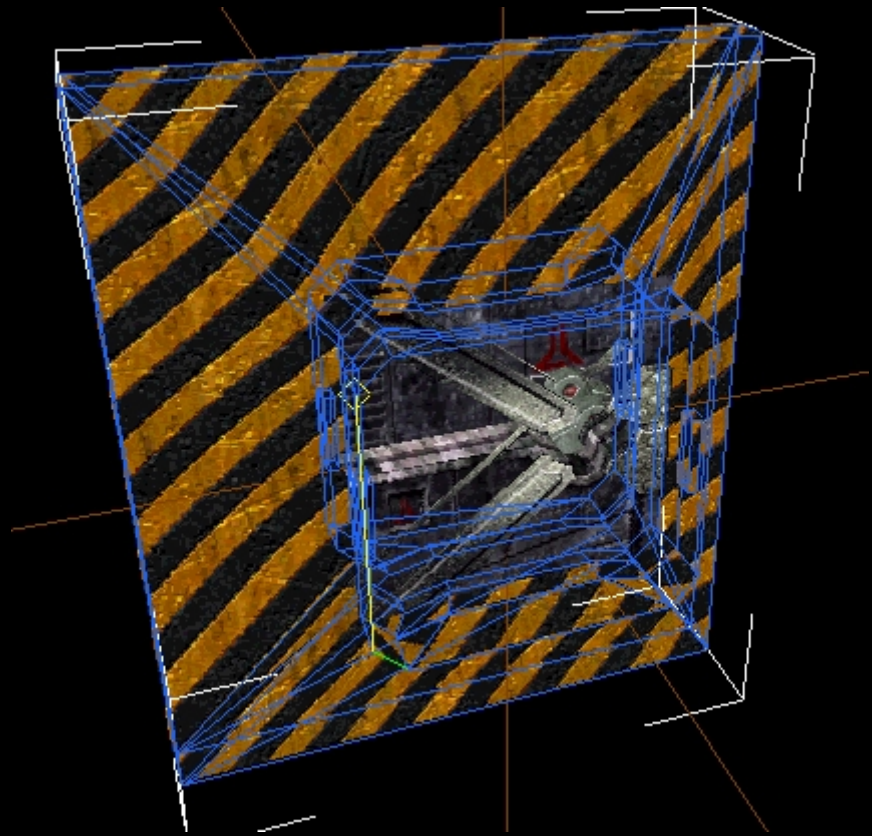
Yes

Bugs:

8 duplicate / unused verts

Miscellaneous:

Parts of the door object do not form a closed polygon network.



Door name:

PTMC Covert F

Size about.:

50x51 units, FF 42x31 Verts of the

front face:8th **Thickness approx.:**20

units **Has region built-ins:**Yes **Bugs:**8

Duplicate / unused Verts

Miscellaneous:

Door name:

PTMC Covert C

Size about.:

36x40 units, FF 21x21 Verts

of the front face:8th

Thickness approx.:20 units

Has region built-ins:

Yes

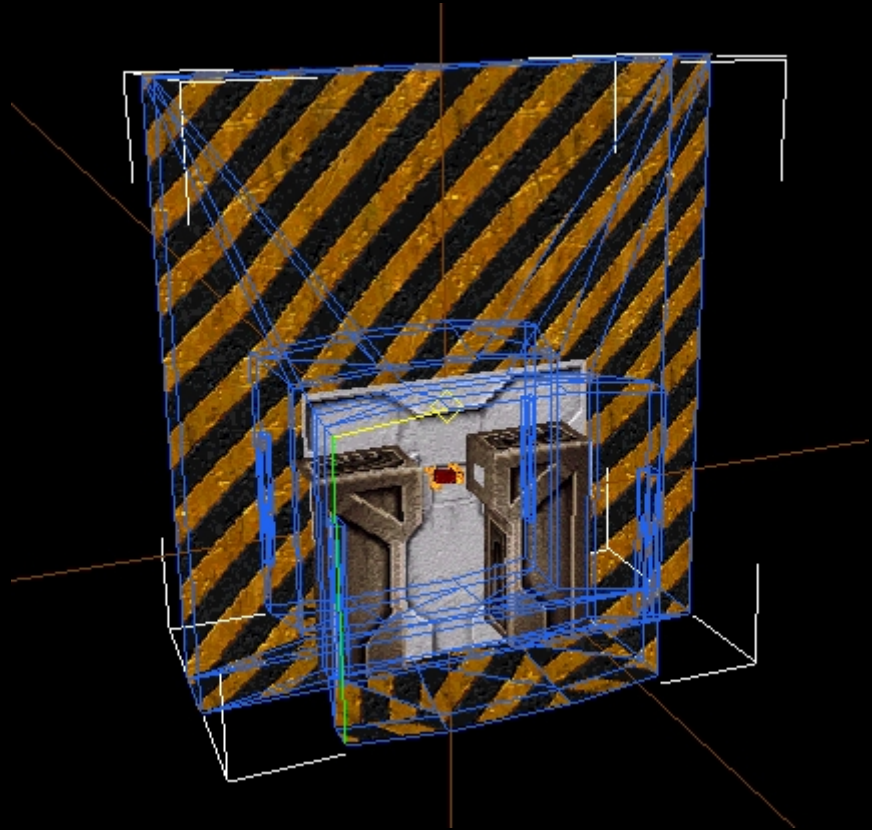
Bugs:

8 duplicate / unused verts

Miscellaneous:

Shell optimization recommended.

Has the see through flag.



Door name:

PTMC Covert H

Size about.:

84x83 units, FF is 37 in diameter Verts of

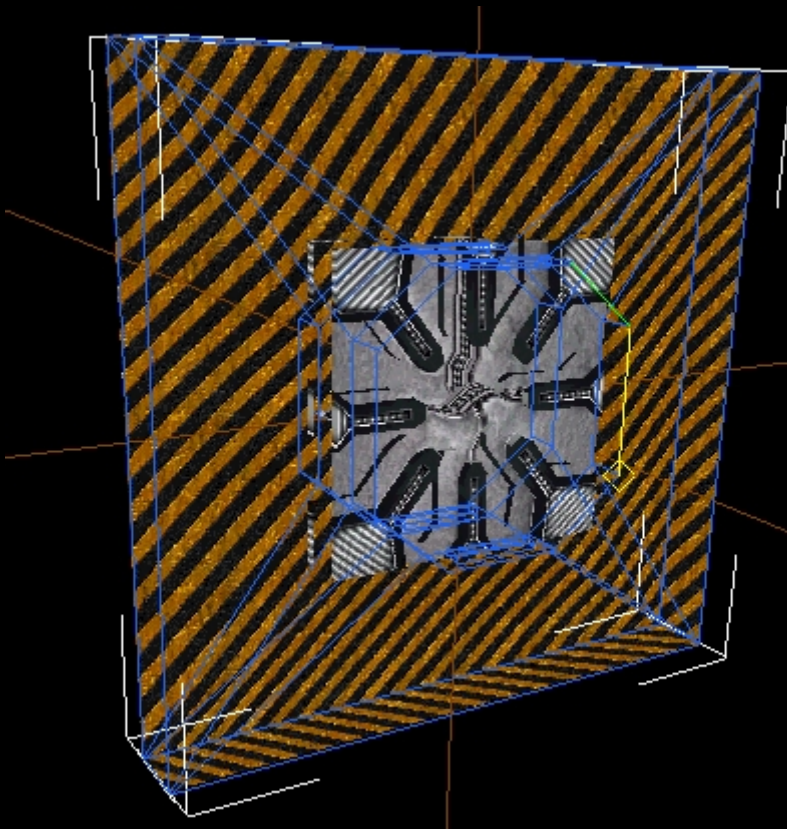
the front face:8th Thickness approx.:20

units Has region built-ins:Yes Bugs:8

Duplicate / unused Verts Miscellaneous:

The shell is large enough for smaller bots to fly in, which can be very annoying.

The components of the Door object do not form a closed polygon network.



Door name:
PTMC Covert G

Size about.:
23x43, FF 20x19 units **Verts of**
the front face:6 Thickness
approx.:20 units Has region
built-ins:Yes Bugs:

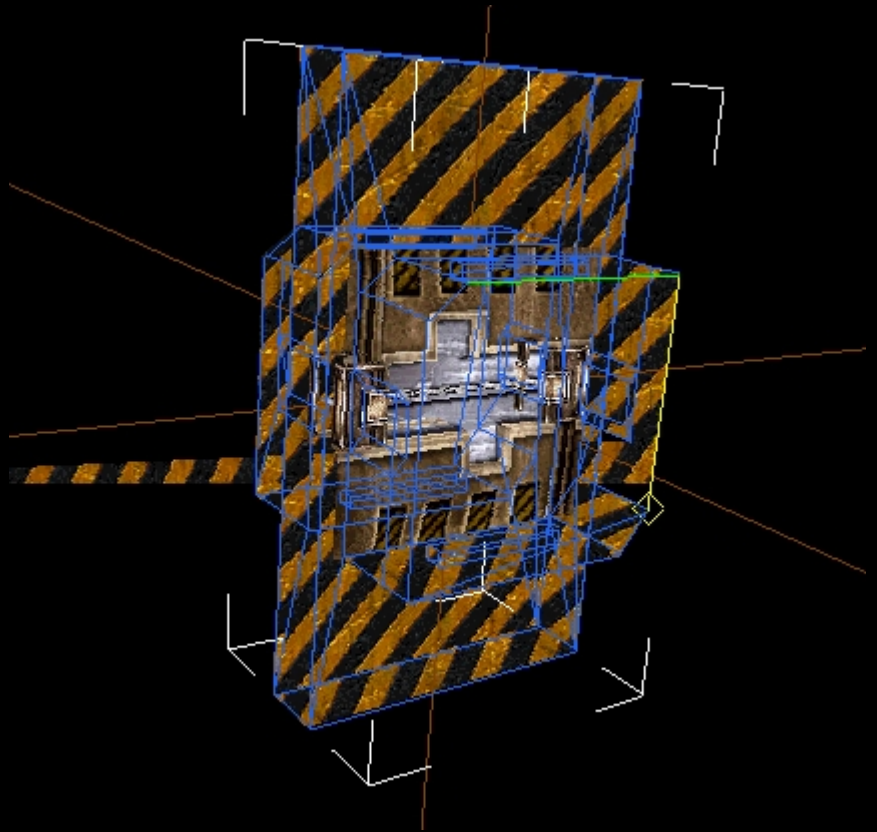
2 concave faces, 1 duplicate face, 6
duplicate / unused verts, 2 T-joints

Miscellaneous:

The duplicate face lies above the
cultivation portal. The FF is split and
creates gaps when the door is
installed -> repair portals on both
sides.

The side where you continue
building is also split and must
first be united.

Repair=Must!
Shell optimization is highly
recommended.



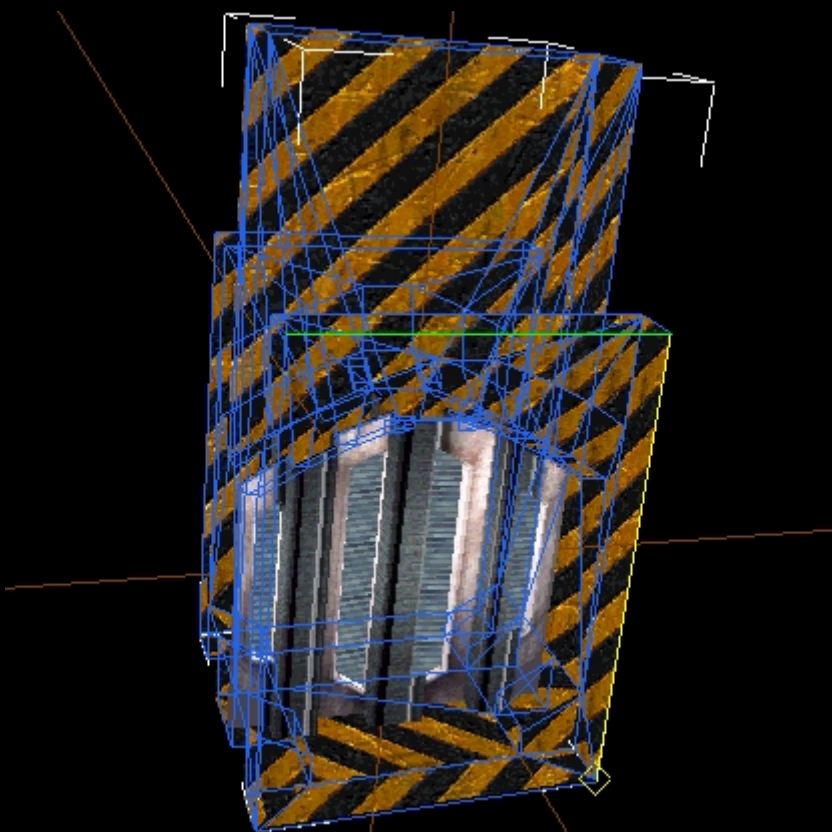
Door name:
PTMC Corporate D

Size about.:
20x40 units, FF 20x27 **Verts of**
the front face:4 Thickness
approx.:20 units Has region
built-ins:Yes

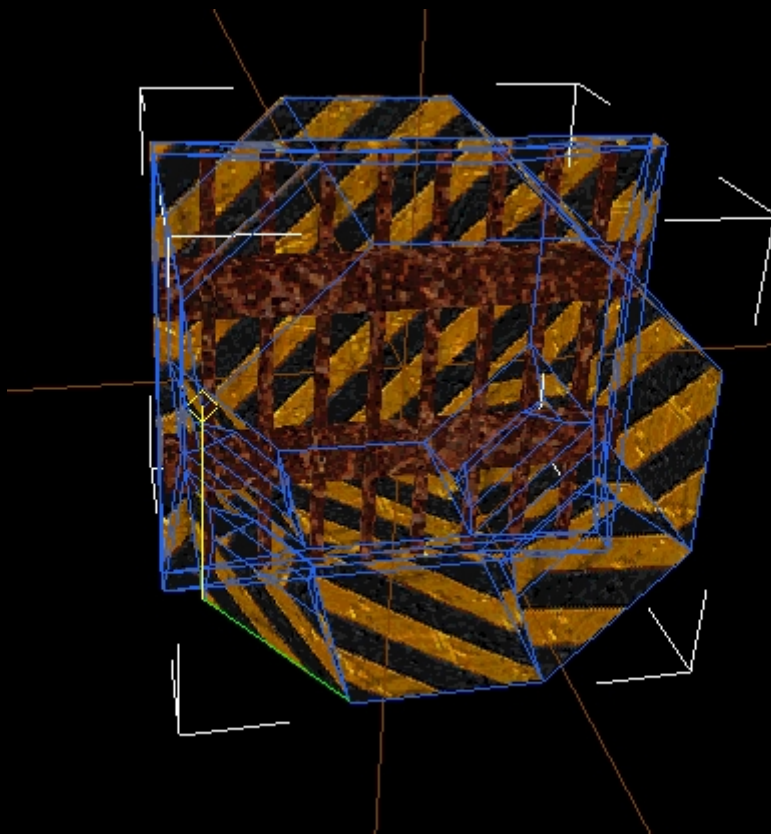
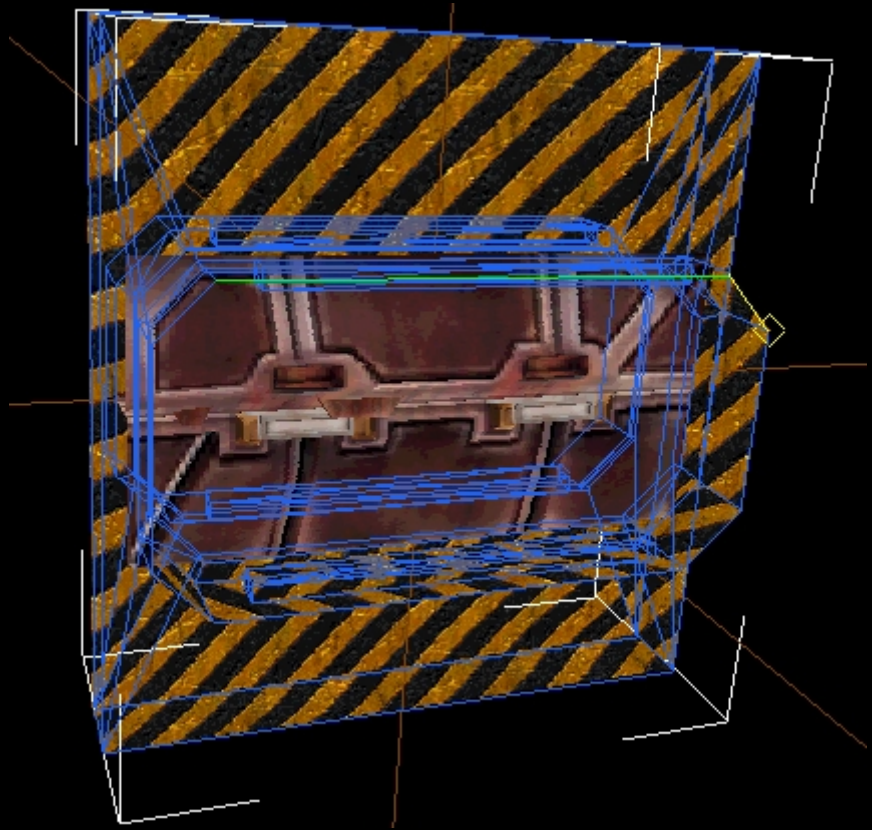
Bugs:4 duplicate / unused verts, 2 gaps

Miscellaneous:

There are two faces in the side area that
overlap each other, that's where the gaps are.
The Door object does not form a closed
polygon mesh.

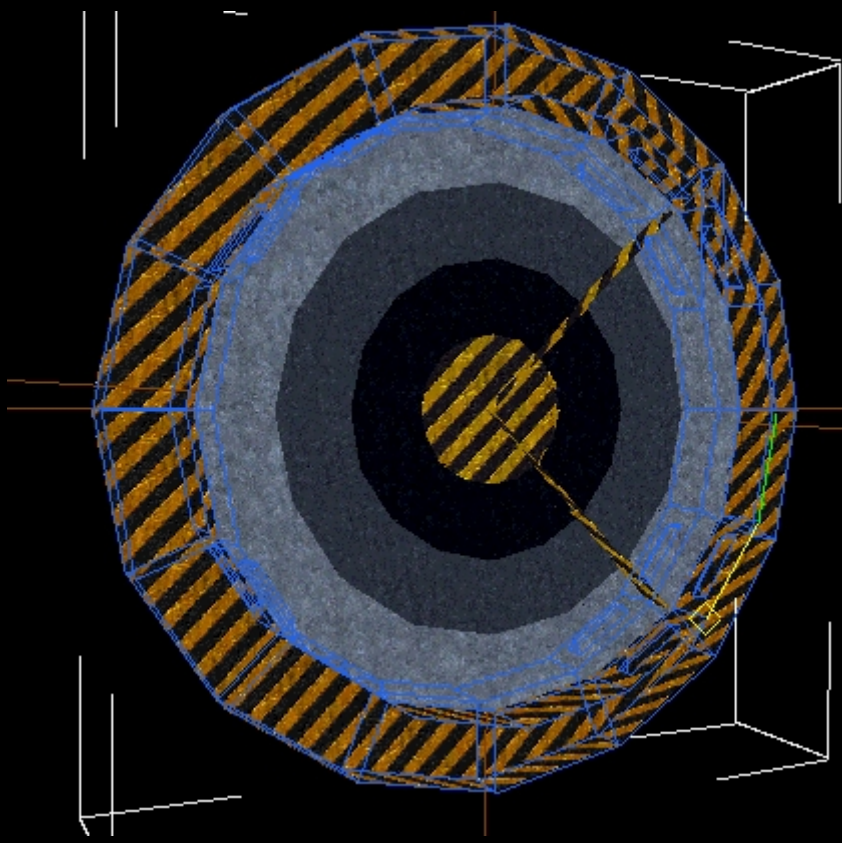
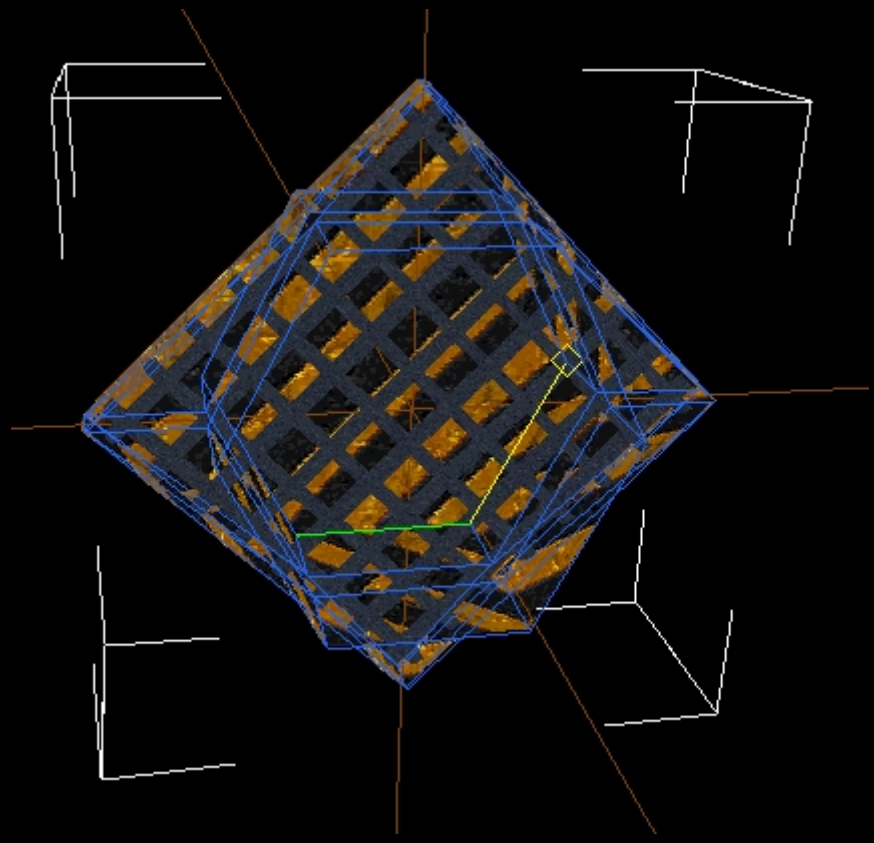


Door name:
Red Acropolis Big Door
Size about.:
44x46 units, FF 40x21 Verts
of the front face:8th
Thickness approx.:20 units
Has region built-ins:
Yes
Bugs:
8 duplicate / unused verts
Miscellaneous:
-



Door name:
PTMC Industrial 4
Size about.:
22x21 units, FF 20x20 Verts
of the front face:8th
Thickness approx.:20 units
Has region built-ins:No Bugs:8
Duplicate / unused Verts
Miscellaneous:
Blastable Door

Door name:
 NovakBlastServiceGrate
Size about.:
 28 units on the diagonal, FF
 18x17.
Verts of the front face:6
Thickness approx.:10.5
 units **Has region built-ins:**
 No
Bugs:
 6 duplicate / unused verts
Miscellaneous:
 Blastable



Door name:
 Transfer Door
Size about.:
 96 units diameter, FF 86 in
 diameter.
Verts of the front face:16 **Thickness**
approx.:20 units **Has region built-ins:**
 Yes **Bugs:**16 Duplicate / unused Verts
Miscellaneous:

Door object components are not a closed polygon mesh.

Door name:

Lvl5BossDoor

Size about.:

140x312 units, FF 140x140 Verts

of the front face:4 Thickness

approx.:32 units Has region

built-ins:

No

Bugs:

4 duplicate / unused verts, 1

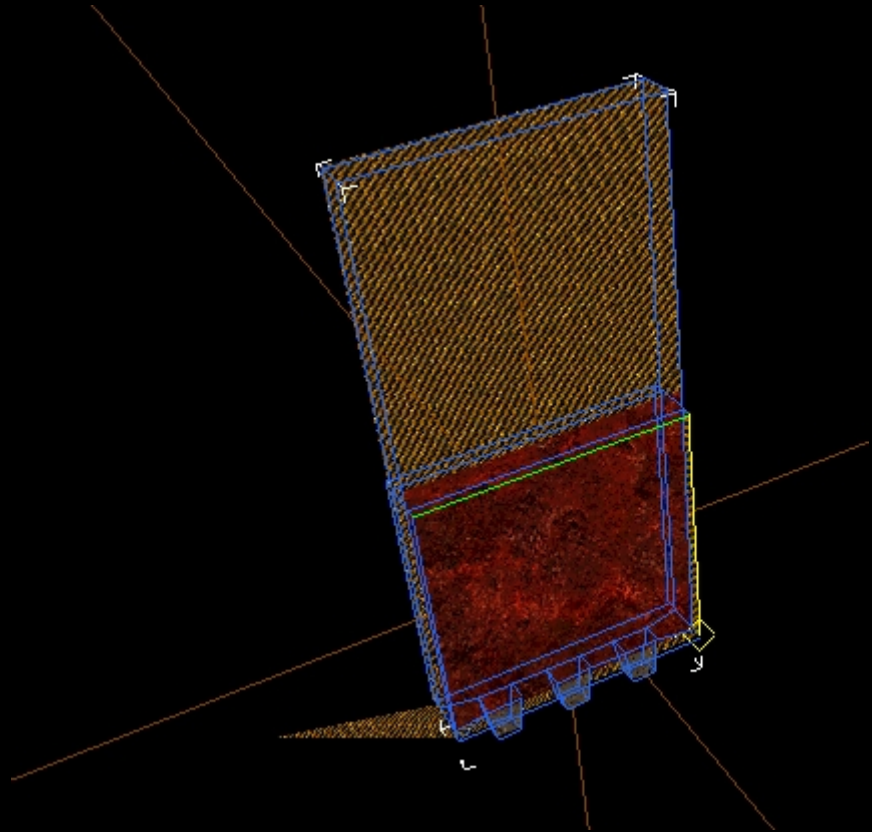
concave face

Miscellaneous:

Wow!

The concave at the foot of the door just needs to be split.

Optimization possible.



Door name:

SS_airlok_door

Size about.:

31x15 units, FF diameter 15 Verts of

the front face:10 Thickness approx.:

20 units

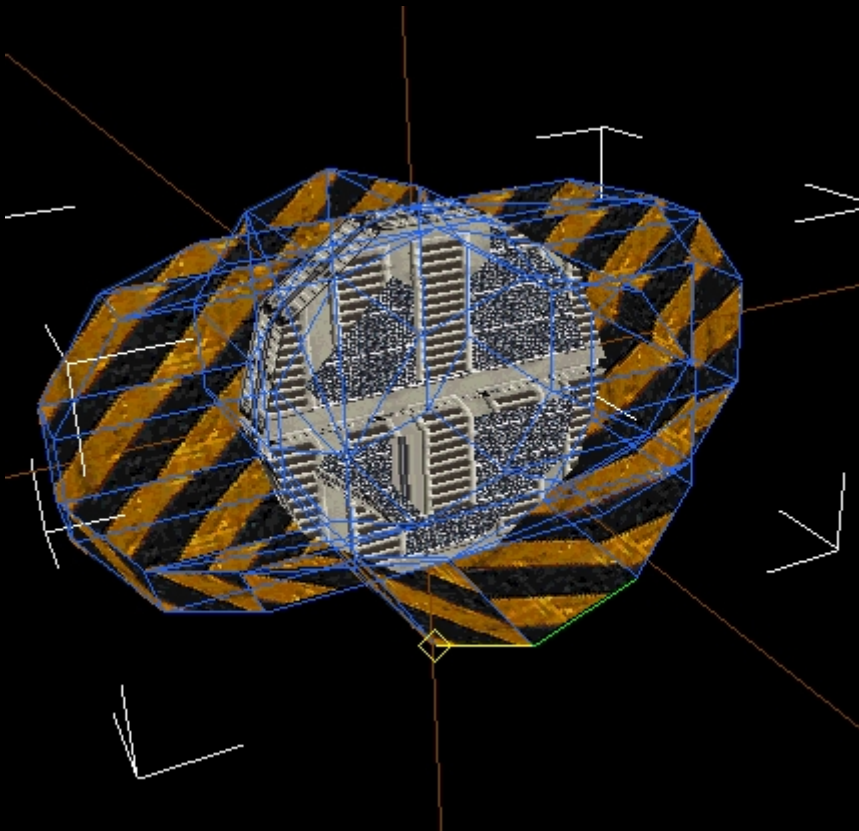
Has region built-ins:No Bugs:10

Duplicate / unused Verts

Miscellaneous:

The side where you continue building must first be unified.

Optimization possible.



Door name:

piccusecretdoor

Size about.:

35x20 units, FF diameter 19

Verts of the front face: 16

Thickness approx.: 9 units

Has region built-ins:

Yes

Bugs:

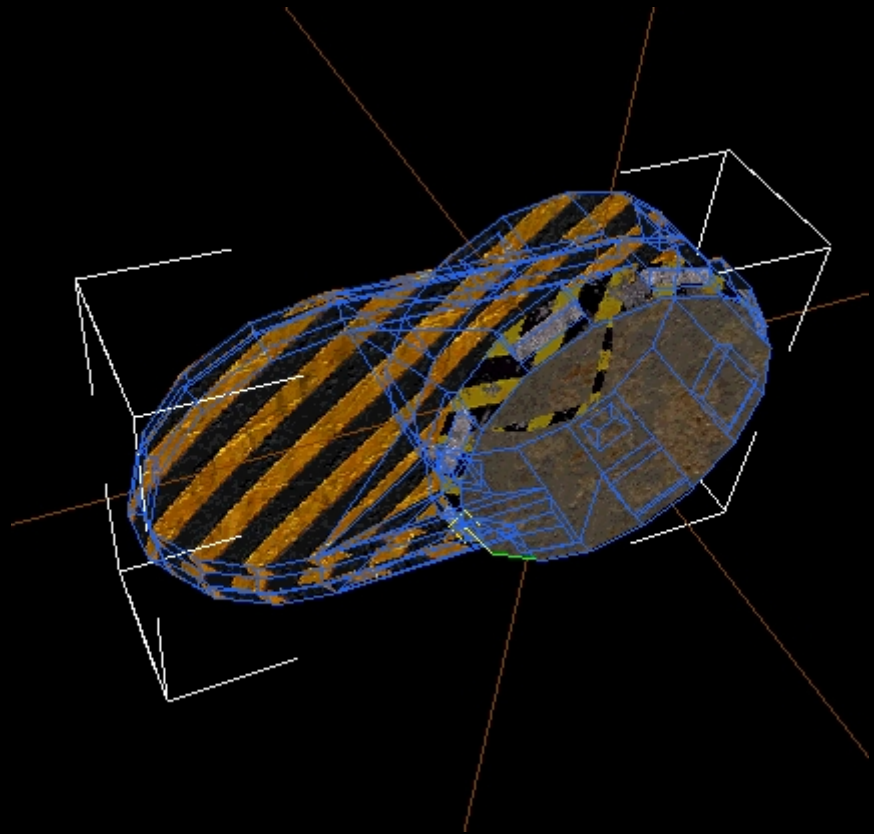
16 duplicate / unused verts, 3 T-joints

Miscellaneous:

Repair=Must!

Optimization recommended.

It's basically the same door as the 'LUKE'S SECRET DOOR'.



Door name:

deimstrapdoor

Size about.:

28x26 units

Verts of the front face: 4

Thickness approx.: 16 units

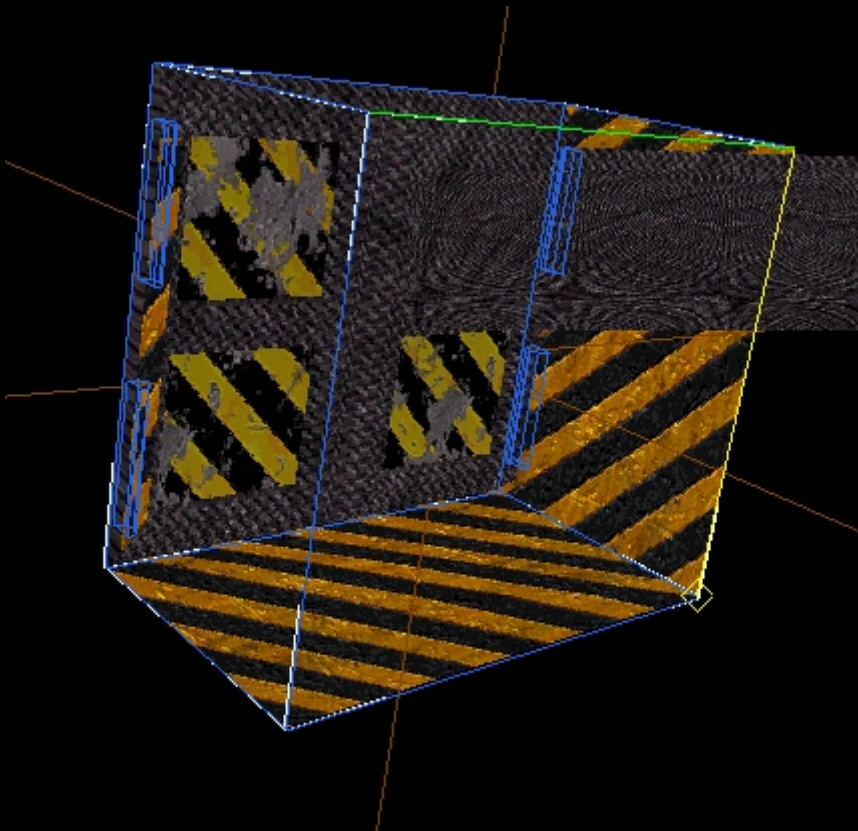
Has region built-ins:

Yes Bugs: 4 Duplicate / Unused Verts

Miscellaneous:

the door object contains 4 concave faces.

is basically the same door as 'SEAN'S TRAPDOOR'.



Door name:

seoulsecretdoorround 2

Size about.:

35x20 units, FF diameter 19 **Verts of the front face:** 16 **Thickness**

approx.:9 units **Has region built-ins:**

Yes

Bugs:

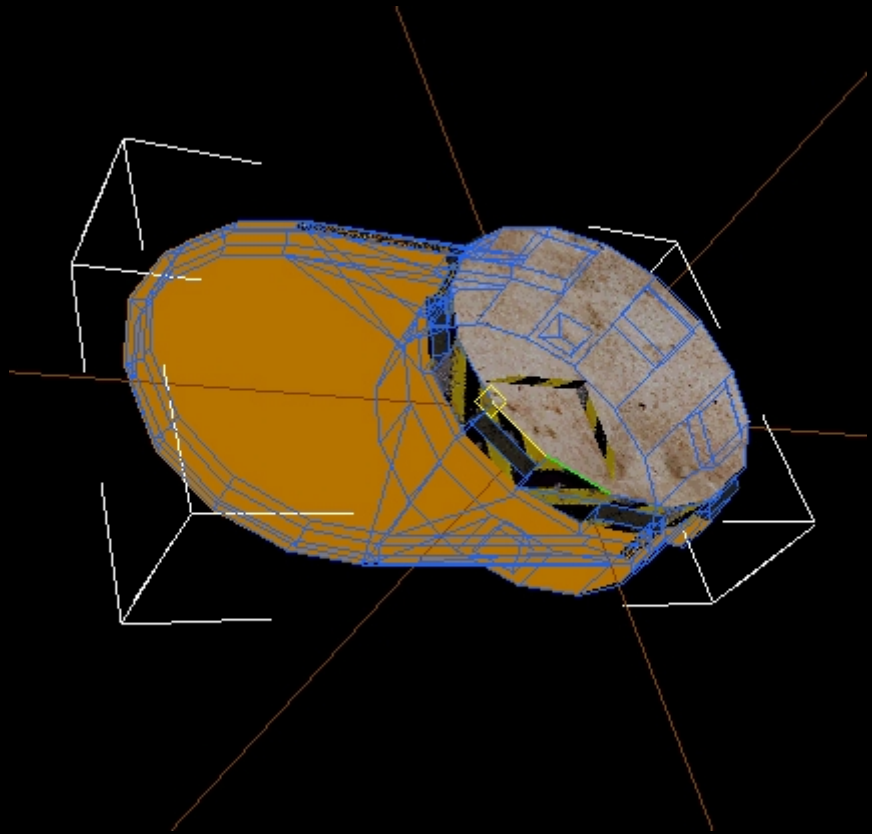
16 duplicate / unused verts, 3 T-joints

Miscellaneous:

Repair=Must!

Optimization recommended.

It's basically the same door as the 'LUKE'S SECRET DOOR'.



Have custom objects visible in D3Edit

Papacat

Note: this is for completeness, since this work can actually be saved since Atan's discovery - see error: reference not found.

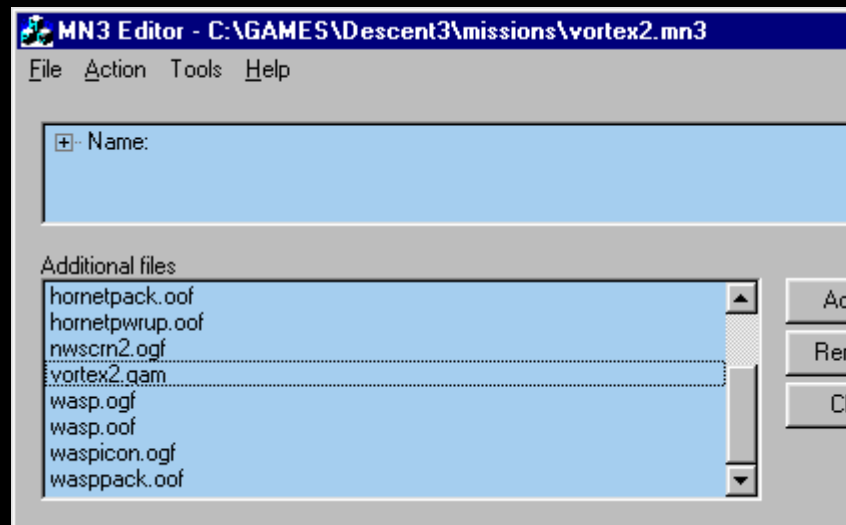
Custom textures and objects are easy to use in D3Edit, but they are not always visible during editing; This naturally makes alignment and positioning somewhat difficult. The process described here has already been mentioned several times before, but here is the whole thing again in all the details.

If you start creating the .oof's that you want to use are ready and a custom.gam you have to create one.hog-make file. You do this by using a.mn3-file created; because this IS a HOG file.

Simply start the MN3 packager and add all your custom.oof's,.oif's,.oaf's and those

Custom.gam via the button **Add file(s)** added. You are NOT allowed to add levels or any mission information. Save it with any name you like. I'm using mine here.mn3 from Vortex2 as an example, but you can Custom.mn3, Temp.mn3...whatever you call it. Just remember that this is not your FINALE .mn3 is, but only a temporary one.

Start D3Edit. Before you start or open anything, you must explain to D3Edit that it is next to the d3.hog yours too.mn3 (=hog) loads to objects and Make textures available.



Go to Data->Configure A Mission Hog like here on the left. This opens the mission

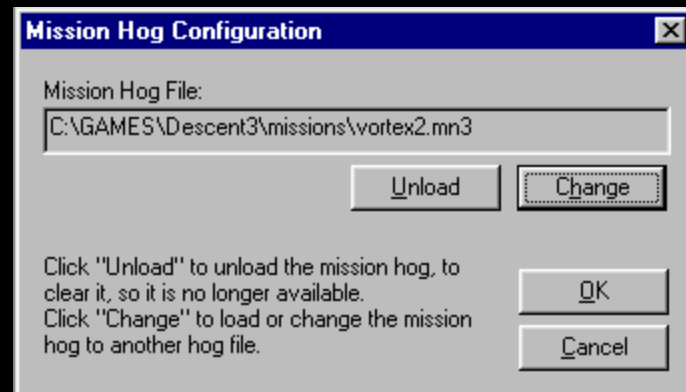
Hog configuration-Dialog. The Textbox should initially be empty. Click **Change** and navigate to your .mn3-File. choose them,

click **Open** and its path will be entered in the text box. Then click OK.

Your hog file (=the.mn3) is now loaded and available for D3Edit. Now you have to .gam-File from your custom.mn3 choose.

Then go to Data -> Tablefile Manager. In the Mission table file-Line click **HOG Browse**, not **File Browse**. The Hog File Browser now opens and lists those of loaded.hog files available.gam files. Select yours from the list and close the Tablefile Manager **OK**.

That's it! Your custom objects and textures are now not only available, but also VISIBLE in both the selection palettes and the textured views.



Here you can see the advantage of the Extra 5 Minutes around your .mn3 close. The object is visible, which helps with positioning.



You can have one Custom.gam also without creating one.mn3 use, but then you see Your objects not.



The Gam Format: The Specification File

Outrage

The Descent 3 Tablefile Format (.GAM) File Spec. (Text file version: 0.501)

(extended)

History:

<i>version</i>	<i>Description</i>
0.505	Info for some texture flags added - Ragil Ral
0.504	Update for Sound Flag - Dark
0.503	Info for some Parameters added - Ragil Ral
0.502	Fixed error in Weapon page, terrain damage was reported as a float, but should be a byte.
0.501	(added stuff from Jeff's email - aldel) Fixed error in Texture page, related to procedural textures. The loop reading the palette went 256 iterations, when it should only go 255.
0.50	Initial Creation. Very rough.

A Tablefile is a collection of pages that define the properties of textures, weapons, doors, ships, sounds and objects in the Descent 3 universe. A page is a chunk of data that are the properties for one such building block. Some pages contain data primitives along with property data. A data primitive is the raw data needed to define the building block. For instance, one type of primitive for objects is the .OOF (Outrage Object Format) file for the polymodel to be associated with the object. Byte order is little endian (INTEL format) unless stated otherwise.

Each page contains its own version number. There is no versioning information for the table file in general. Each page is responsible for its own version control.

The following is a list of page types that you will find in a Tablefile, along with the value associated with them:

<i>PageType</i>	<i>Value</i>	<i>Description</i>
PAGETYPE_UNKNOWN	0	An empty page. Usually used to create an empty table file. Also the first page is usually found in a table file.
PAGETYPE_TEXTURE	1	Contains data and properties for a texture.
PAGETYPE_WEAPON	2	Contains data and properties for a weapon (player or robot).
PAGETYPE_DOOR	5	Contains data and properties for a door
PAGETYPE_SHIP	6	Contains data and properties for a player ship.
PAGETYPE_SOUND	7	Contains data and properties for a sound.
PAGETYPE_GAMEFILE	9	Contains data and properties for a game file. Game files are any kind of file that may be used by Descent 3 in some way. Any extra files (that are not a data primitive) in the D3.HOG file are enumerated in these pages.
PAGETYPE_GENERIC	10	Contains data and properties for a generic object. A generic object is an object of type robot, powerup, clutter or building.

The following page types are outdated and no longer supported. They are listed here because it is possible they may be in a table file. Should you come across one of these page types, it should just be skipped over:

PageType	Value
PAGETYPE_ROBOT	3
PAGETYPE_POWERUP	4
PAGETYPE_MEGACELL	8th

TABLEFILE STRUCTURE

A Tablefile is a collection of the above listed pages formatted in the following way:

'Offset' is the offset from the start of the file (for Page 0) or the offset from the end of the previous page.

<i>offset</i>	<i>Size</i>	<i>Description</i>
0	1 byte	Page type (see above)
1	4 bytes int	Page length in bytes (which includes the 4-byte length)
5	(x-4) bytes	Page data

This pattern is continued until the EOF (End of File). Pages are listed in no specific order, usually a PAGETYPE_UNKNOWN page will be page 0. For optimal performance (in Descent 3) it is suggested that pages that are dependent on another page come after those pages it is dependent on.

The easiest way to handle this is to sort the pages by pagetype in the following way (from beginning of the file to the end):

```
PAGETYPE_TEXTURE
PAGETYPE_SOUND
PAGETYPE_WEAPON
PAGETYPE_GENERIC
PAGETYPE_DOOR
PAGETYPE_SHIP
PAGETYPE_GAMEFILE
```

PSUEDOCODE FOR TABLEFILE READING

```
FILE *file;
```

```
// open the table file file = fopen(
filename,"rb");
```

```
while(!feof(file)) {
```

```
    unsigned char ptype=FILE_READBYTE(file); int
    length=FILE_READINT(file);
```

```
    switch(ptype) {
```

```
        case PAGETYPE_TEXTURE:
            tablefile_ReadTexturePage(file);break; case
            PAGETYPE_DOOR:
            tablefile_ReadDoorPage(file);break; case
            PAGETYPE_SOUND:
            tablefile_ReadSoundPage(file);break; case
            PAGETYPE_GENERIC:
            tablefile_ReadGenericPage(file);break; case
            PAGETYPE_GAMEFILE:
            tablefile_ReadGamefilePage(file);break; case
            PAGETYPE_SHIP:
            tablefile_ReadShipPage(file);break; case
            PAGETYPE_WEAPON:
            tablefile_ReadWeaponPage(file);break; /* old page
```

```
types, no longer valid
and not supported */ case
            PAGETYPE_ROBOT:
```

```
        case PAGETYPE_POWERUP:
        case PAGETYPE_MEGACELL:
        case PAGETYPE_UNKNOWN:
            fseek(file,length*sizeof(int),SEEK_CUR);break;
```

```
    }
```

```
}
```

```
fclose(file);
```

Bake

THIS DOCUMENT IS NOT AS COMPLETE AS IT POSSIBLY COULD BE. IT IS RECOMMENDED THAT IF YOU ARE WRITING A TABLEFILE EDITOR/CREATOR OF SOME SORT TO ANALYZE THE VALUES OF THE DATA OF THE OBJECTS, TEXTURES, SOUNDS, ETC. IN THE D3.HOG IN YOUR DESCENT 3 DIRECTORY. DOING THIS, YOU WILL BE ABLE TO GET A GOOD IDEA OF THE RANGES OF THE VARIABLES AND WHAT FLAGS/SETTINGS DO WHAT.

Page Type Description

GRADE: For the remainder of the document:

the symbol "SURNAME"	represents a value of
"PAGENAME_LEN"	35
"MAX_MODULENAME_LEN"	32
"MAX_PLAYER_WEAPONS"	21
"MAX_WB_GUNPOINTS"	8th
"MAX_WB_FIRING_MASKS"	8th
"MAX_PROC_ELEMENTS"	8000
"MAX_WEAPON_SOUNDS"	7
"MAX_DSPEW_TYPES"	2
"NUM_MOVEMENT_CLASSES"	5
"NUM_ANIMS_PER_CLASS"	24
"MAX_WBS_PER_OBJ"	21
"MAX_OBJ_SOUNDS"	2
"MAX_AI_SOUNDS"	5
"MAX_DEATH_TYPES"	4

Data Size's and Description

Type	Size
byte	1 byte
short	2 bytes
int	4 bytes
float.float	4 bytes
vector	3 floats (x,y,z)
matrix	3 vectors (right, up, forward)
string	\0 (NULL) terminated string, of max length specified (max length includes the NULL terminator).
physics_chunk	physics information chunk. See "Physics Chunks".
lighting_chunk	lighting information chunk. Lake "Lighting Chunks".
weapon_battery_chunk	weapon battery chunk. See Weapon Battery Chunks.

PAGETYPE_DOOR

Current version: 3

Data

Type	Surname	Description
short	version	Version of the door page. Used for version control.
string	Door name	Used by Descent 3 to refer to this door by name. max length = PAGENAME_LEN
string	Model name	Filename of the .OOF file used for the polymodel of the door. max length = PAGENAME_LEN
float.float	Opening time	Time in seconds it takes for the door to open
float.float	Close time	Time in seconds it takes for the door to close
float.float	Open time	Time in seconds the door stays open
byte	Flags	See "Door Flags" section
short	Hit Points	How many hitpoints this door has (for destroyable doors)
string	Open sound	D3 Sound Page name for the sound to play when this door opens. max length = PAGENAME_LEN
string	Close sound	D3 Sound Page name for the sound to play when this door closes. max length = PAGENAME_LEN
string	Module name	Filename of the OSIRIS script for this door. This filename doesn't need an extension since it is dependent on the platform that Descent 3 is running for (.DLL for Win32, .so for Linux, etc.). If this is not set then the door will not open! By default this can be set to "Generic", which contains a default generic script for any door. max length = MAX_MODULENAME_LEN

(END OF PAGE)

Door Flags

Surname	Value	Description
DF_BLAstable	1	this door can be destroyed
DF_SEETHROUGH	2	this door can be seen through even when closed

Bake

PAGETYPE_GAMEFILE

Current version: 2

Data

Type	Surname	Description
short	version	Version of the game file page
string	Filename	Filename of the game file. max length =PAGENAME_LEN
string	Directory	Data directory where file should be located. This is used for Descent 3 development only. max length =PAGENAME_LEN

(END OF PAGE)

GRADE: Third party developers usually don't have to be concerned with Gamefile pages, as they were used for Descent 3 development. The information is given for completeness purposes.

Bake

PAGETYPE_GENERIC

Current version: 27

Data

Type	Surname	Description
short	version	Used for version control of the page
byte	Object Type	Type of generic object:
		OBJ_ROBOT 2
		OBJ_POWERUP 7
		OBJ_CLUTTER 11
string	Surname	OBJ_BUILDING 16
		Table Name of the object. max length =PAGENAME_LEN
		High resolution .OOF filename for the object polymodel. max length =PAGENAME_LEN
		Medium resolution .OOF filename for the object polymodel. Max length =PAGENAME_LEN
string	Med. Image Name	Low resolution .OOF filename for the object polymodel. max length =PAGENAME_LEN
float.float	Impact size	
float.float	Impact Time	
float.float	Damage	
short	Score	Score gained by the player by destroying this object
- - The following section is for objects of typeOBJ_POWERUPonly --		
short	Ammo Count	How much ammo this powerup contains by default
- - End of section --		
string	NO LONGER USED	Just ignore this string. max length =PAGENAME_LEN
string	Module name	The filename of the OSIRIS module that contains the default script for this object. No file extension needed as it depends on the platform Descent 3 is running on, as to what the real file extension is (Win32 = DLL, Linux = so, etc.). If this is not set, the object will have no default behavior. Usually this is set to "Generic". max length = MAX_MODULENAME_LEN
string	Script Name Override	Usually when the OSIRIS module (see Module Name) is queried for a script for this object, the Name is passed in for it to use for comparison. However, sometimes, you may want multiple objects to have the same behavior. By creating a behavior, giving it a name and putting the script in the OSIRIS module 'Module Name', you can set the Script Name Override to override the Name passed in to the module. max length =PAGENAME_LEN.
byte	Has description	If this is 0, then the object has no inventory description. Otherwise it does.
- - The following section of code is only if the object has a description --		
string	Description	Description for the object when it is in the inventory. Not used. max length = 1024 bytes.
- - End of Section --		
string	Icon name	Not Used (make it a 0 length string). max length =PAGENAME_LEN
float.float	Medium LOD Distance	Distance at which the medium resolution model is used
float.float	Low LOD distance	Distance at which the low resolution model is used
physics_chunk	Physics information	
float.float	Size	Object size
lighting_chunk	Lighting Information	

int	Hit Points	The amount of shields this object has by default
int	Flags	See "Object Flags".
int	AI flags	See "Object AI Flags".
byte	AI class	AI Class for objects. See "Object AI Classes".
byte	AIType	AI Type for object. See "Object AI Types".
byte	AI Movement Type	AI Movement Type for object. See "Object AI Movement Types".
byte	AI Movement SubTypes	AI Movement subtypes for object. See "Object AI Movement Types".
float.float	AI FOV	cos() of the AI Field of View for object.
float.float	AI Max Velocity	Maximum velocity
float.float	AI Max Delta Velocity	Maximum delta velocity
float.float	AI Max Turn Rate	Maximum Turn Rate
int	AI Notify Flags	Flags for AI notification. See "Object AI Notification Flags".
float.float	AI Max Delta Turn Rate	Maximum Delta Turn Rate
float.float	AI Circle Distance	Distance the object should circle it's targets at
float.float	AI Attack Vel Percentage	
float.float	AI Dodge Percentage	
float.float	AI Dodge Vel Percentage	
float.float	AI Flee Vel Percentage	
float.float	AI Melee Damage 1	
float.float	AI Melee Damage 2	
float.float	AI Melee Latency 1	
float.float	AI Melee Latency 2	
float.float	AI Curiosity	
float.float	AI Night Vision	
float.float	AI Fog Vision	
float.float	AI Lead Accuracy	
float.float	AI Lead Variance	
float.float	AI Fire Spread	
float.float	AI Fight Team	
float.float	AI Fight Same	
float.float	AI aggression	
float.float	AI Hearing	
float.float	AI frustration	
float.float	AI roaming	
float.float	AI Life Preservation	
float.float	AI Avoid Friends Distance	
float.float	AI Biased Flight Importance	
float.float	AI Biased Flight Min	
float.float	AI Biased Flight Max	

-- The following section is repeatedMAX_DSPEW_TYPEStimes --

byte	Destroy Spew Flags	Lake Destroy Spew Flags
float.float	Destroy Spew Percent	Percent Chance that this object would spew
short	Destroy Spew Number	How many of this object to spew if we are to spew it.
string	Destroy Spew Object	Table name of the object to spew (for this slot). max length = PAGENAME_LEN.

-- End of Section --
-- The following section is repeatedNUM_MOVEMENT_CLASSEStimes --

-- The following section is repeatedNUM_ANIMS_PER_CLASStimes --

short	From Frame	Starting frame of animation
short	To Frame	End frame of animation
float.float	Seconds Per Class	How long the animation should last

-- End of Section --

-- End of Section --
-- The following section is repeatedMAX_WBS_PER_OBJtimes --

weapon_battery_chunk	Weapon Battery Information	
----------------------	----------------------------	--

-- End of Section --
-- The following section is repeatedMAX_WBS_PER_OBJtimes --

-- The following section is repeatedMAX_WB_GUNPOINTStimes --

string	Weapon Name	Table name of the weapon associated with this gunpoint, on this weapon battery. max length = PAGENAME_LEN.
short	To Frame	End frame of animation
float.float	Seconds Per Class	How long the animation should last

-- End of Section --

-- End of Section --
-- The following section is repeatedMAX_OBJ_SOUNDStimes --

string	Sound name	Table Sound Name. max length =PAGENAME_LEN
--------	------------	--

-- End of Section --
-- The following section is repeatedMAX_AI_SOUNDStimes --

string	AI sound name	Table Sound Name. max length =PAGENAME_LEN
--------	---------------	--

-- End of Section --
-- The following section is repeatedMAX_WBS_PER_OBJtimes --

-- The following section is repeatedMAX_WB_FIRING_MASKStimes --

string	Fire sound name	Table Sound Name for fire sound. max length = PAGENAME_LEN
--------	-----------------	--

-- End of Section --

-- End of Section --
-- The following section is repeatedNUM_MOVEMENT_CLASSEStimes --

-- The following section is repeatedNUM_ANIMS_PER_CLASStimes --

string	Animation sound Surname	Table Sound Name for this animation. max length = PAGENAME_LEN.
--------	----------------------------	---

-- End of Section --

-- End of Section --

float.float	Respawn Scalar	Used for powerups to scale how often they respawn in multiplayer games. Normal is 1.0.
short	Number of Death Types	Number of types of deaths possible for this object. Maximum value isMAX DEATH TYPES.

- - The following section is repeated 'Number of Death Types' times --

int	Death Flags	flags for this death type. See "Death Type Flags".
float.float	Death Delay Min.	Minimum amount of time for this death to last.
float.float	Death Delay Max.	Maximum amount of time for this death to last.
byte	Death Probability	Probability of this death happening (0 to 100, the probabilities for all the death types should add up to 100).

- - End of Section --

(END OF PAGE)

Object flags

Surname	Value	Description
OIF_CONTROL_AI	0x01	//this object uses AI
OIF_USES_PHYSICS	0x02	//this object uses physics
OIF_DESTROYABLE	0x04	//this object can be destroyed
OIF_INVEN_SELECTABLE	0x08	//this object can be selected in the inventory
OIF_INVEN_NONUSEABLE	0x10	//this object cannot be used by pressing ENTER during the game
OIF_INVEN_TYPE_MISSION	0x20	//this object is for Mission objectives
OIF_INVEN_NOREMOVE	0x40	//this object should NOT be removed from the inventory when used
OIF_INVEN_VISWHENUSED	0x80	//this object will not have it control type, movement type and render types
OIF_AI_SCRIPTED_DEATH	0x100	
OIF_DO_CEILING_CHECK	0x200	
OIF_IGNORE_FORCEFIELDS_AND_GLASS	0x400	
OIF_NO_DIFF_SCALE_DAMAGE	0x800	
OIF_NO_DIFF_SCALE_MOVE	0x1000	
OIF_AMBIENT_OBJECT	0x2000	//this object is just for show, & can be removed to improve performance

Object AI flags

Surname	Value	Surname	Value
AIF_WEAPON1	0x00000001	AIF_TEAM_PTMC	0x00000000
AIF_WEAPON2	0x00000002	AIF_TEAM_REBEL	0x00010000
AIF_MELEE1	0x00000004	AIF_TEAM_HOSTILE	0x00020000
AIF_MELEE2	0x00000008	AIF_TEAM_NEUTRAL	0x00030000
AIF_STAYS_INOUT	0x00000010	AIF_ORDERED_WB_FIRING	0x00040000
AIF_GB_MIMIC_PLAYER_FIRING_HACK	0x00000010	AIF_ORIENT_TO_VEL	0x00080000
AIF_ACT_AS_NEUTRAL_UNTIL_SHOT	0x00000020	AIF_XZ_DIST	0x00100000
AIF_PERSISTANT	0x00000040	AIF_REPORT_NEW_ORIENT	0x00200000
AIF_DODGE	0x00000080	AIF_TARGET_BY_DIST	0x00400000
AIF_FIRE	0x00000100	AIF_DISABLE_FIRING	0x00800000
AIF_FLINCH	0x00000200	AIF_DISABLE_MELEE	0x01000000
AIF_DETERMINE_TARGET	0x00000400	AIF_AUTO_AVOID_FRIENDS	0x02000000
AIF_AIM	0x00000800	AIF_TRACK_CLOSEST_2_FRIENDS	0x04000000
AIF_ONLY_TAUNT_AT_DEATH	0x00001000	AIF_TRACK_CLOSEST_2_ENEMIES	0x08000000
AIF_AVOID_WALLS	0x00002000	AIF_BIASED_FLIGHT_HEIGHT	0x10000000
AIF_DISABLED	0x00004000	AIF_FORCE_AWARENESS	0x20000000
AIF_FLUCTUATE_SPEED_PROPERTIES	0x00008000	AIF_UVEC_FOV	0x40000000
AIF_TEAM_MASK	0x00030000	AIF_AIM_PNT_FOV	0x80000000

Object AI Classes

Surname	Value
AIC_STATIC	0
AIC_PURE_PATH	1
AIC_AIS_FULL	2

Object AI Types

Surname	Value
AIT_FLYLANDER	0
AIT_STALKER	1
AIT_EVADER1	2
AIT_EVADER2	3
AIT_STATIONARY_TURRET	4
AIT_AIS	5
AIT_MELEE1	6
AIT_BIRD_FLOCK1	7
AIT_HERD1	8th

Object AI Movement Types

Surname	Value	Description
// AI Movement flying types		
AIMF_NORMAL	0	
AIMF_PATH	1	
AIMF_HELICOPTER	2	
AIMF_HOVERCRAFT	3	
AIMF_JET	4	
AIMF_PLAYERLIKE	5	// For NPC ships
AIMF_BUDDYLIKE	6	// For thief/buddy bots
// AI Movement walking types		
AIMW_RESTRICTED_FLAT	0	// Specify a min/max angle of incline (so we can restrict movement -- ie we can even do ceiling only robots)
AIMW_RESTRICTED_LOW_ANGLE	1	
AIMW_RESTRICTED_HIGH_ANGLE	2	
AIMW_NON_RESTRICTED	3	
AIMW_UNDERWATERONLY	4	// Stay in water
AIMW_WATERSURFace	5	

Object AI Notification Flags

Surname	Value	Description
AIN_NEW_MOVEMENT	(1<<1)	
AIN_OBJ_KILLED	(1<<2)	
AIN_WHIT_BY_OBJ	(1<<3)	
AIN_SEE_TARGET	(1<<4)	
AIN_PLAYER_SEES_YOU	(1<<5)	
AIN_WHIT_OBJECT	(1<<6)	
AIN_TARGET_DIED	(1<<7)	// In code, it only notifies if the target is gone
AIN_OBJ_FIRED	(1<<8)	
AIN_GOAL_COMPLETE	(1<<9)	
AIN_GOAL_FAIL	(1<<10)	
AIN_GOAL_ERROR	(1<<11)	
AIN_HEAR_NOISE	(1<<12)	
AIN_NEAR_TARGET	(1<<13)	
AIN_HIT_BY_WEAPON	(1<<14)	

AIN_NEAR_WALL	(1<<15)	
AIN_USER_DEFINED	(1<<16)	// Processed in script
AIN_TARGET_INVALID	(1<<17)	// Goal is killed
AIN_GOAL_INVALID	(1<<18)	
AIN_SCRIPTED_GOAL	(1<<19)	
AIN_SCRIPTED_ENABLER	(1<<20)	
AIN_ANIM_COMPLETE	(1<<21)	
AIN_BUMPED_OBJ	(1<<22)	
AIN_MELEE_HIT	(1<<23)	
AIN_MELEE_ATTACK_FRAME	(1<<24)	
AIN_SCRIPTED_INFLUENCE	(1<<25)	
AIN_SCRIPTED_ORIENT	(1<<26)	
AIN_MOVIE_START	(1<<27)	
AIN_MOVIE_END	(1<<28)	
AI_NOTIFIES_ALWAYS_ON	(AIN_ANIM_COMPLETE AIN_NEW_MOVEMENT AIN_PLAYER_SEES_YOU AIN_GOAL_COMPLETE AIN_GOAL_FAIL AIN_GOAL_ERROR AIN_USER_DEFINED AIN_TARGET_DIED AIN_TARGET_INVALID AIN_BUMPED_OBJ AIN_MELEE_HIT AIN_MELEE_ATTACK_FRAME AIN_TARGET_INVALID)	

Destroy Spew Flags

Surname	Value
DSF_ONLY_IF_PLAYER_HAS_OBJ_1	1
DSF_ONLY_IF_NO_1	2

Death Type Flags

Surname	Value	Description
//Not a bit flag, but an override value		
DF_DEFAULT	- 1	//use the default death for this object
DF_UNUSED	0x0000001	//Unused flag
//Delay options		
DF_DELAY_FROM_ANIM	0x0000002	//delay time from death animation
DF_DELAY_SPARKS	0x0000004	//delay with sparks
DF_DELAY_LOSES_ANTIGRAV	0x0000008	//object gets gravity during delay
DF_DELAY_SMOKES	0x0000010	//delay with smoke
DF_DELAY_FLYING	0x0100000	//delay with object flying up into the air
DF_DELAY_FIREBALL	0x0200000	//delay with fireballs
DF_DELAY_FADE_AWAY	0x0400000	//fade away
DF_DELAY_NO_TUMBLE_FLY	0x2000000	//don't tumble while flying up in the air
//Options for what happens on death		
DF_FIREBALL	0x0000020	//there are fireballs when the object dies
DF_BREAKS_APART	0x0000040	//the object breaks into pieces when it dies
DF_BLAST_RING	0x0000080	//a blast ring is created when the object dies
DF_REMAINS	0x0000100	//the object does not go away when it does
DF_LOSES_ANTIGRAV	0x0000200	//object gets gravity on death
DF_FADE_AWAY	0x0800000	//fades away
//Explosion size if there is a death fireball		
DF_EXPL_SMALL	0x0000000	//use a small explosion
DF_EXPL_MEDIUM	0x0000400	//use a medium explosion
DF_EXPL_LARGE	0x0000800	//use a large explosion
DF_EXPL_SIZE_MASK	0x0000c00	
DF_EXPL_SIZE_SHIFT	10	
//What happens when this object hits something during delay		
DF_CONTACT_FIREBALL	0x0001000	//creates fireballs
DF_CONTACT_BREAKS_APART	0x0002000	//break apart
DF_CONTACT_BLAST_RING	0x0004000	//blast ring
DF_CONTACT_REMAINS	0x0008000	//stays around
//Whether the debris puts off smoke		
DF_DEBRIS_SMOKES	0x0010000	//the debris that's created smokes
//What happens to the debris when it times out or hits something		
DF_DEBRIS_FIREBALL	0x0020000	//creates fireballs
DF_DEBRIS_BLAST_RING	0x0040000	//blast ring
DF_DEBRIS_REMAINS	0x0080000	//stays around
//Sound option		
DF_DELAY_SOUND	0x1000000	//play sound at start of fade

Bake

PAGETYPE_SHIP

Current version: 6

Data

Type	Surname	Description
short	version	Version of the ship page. Used for version control.
string	Ship Name	Descent 3 name of the ship. max length =PAGENAME_LEN
string	Cockpit filename	Filename of the .OOF model for the cockpit. max length = PAGENAME_LEN
string	HUD Config Filename	. INF filename of the HUD configuration file. max length = PAGENAME_LEN
string	Model name	Filename for the .OOF model that is the model of the ship. max length =PAGENAME_LEN
string	Dying model	Filename for the .OOF model of the ship when it is dying. max length =PAGENAME_LEN
string	Med. Model Name	Filename for the .OOF model that is the medium res (less poly) version. max length =PAGENAME_LEN
string	Low model name	Filename for the .OOF model that is the low res (really low poly) version. max length =PAGENAME_LEN
float.float	Med. LOD Distance	Distance at which the medium res model version of the ship is used
float.float	Low LOD distance	Distance at which the low res model version of the ship is used
physics_chunk	Physics information	The physics information for the ship
float.float	Size	Size of the radius of the ship
float.float	Armor Scalar	Armor scalar for the ship (Pyro-GL = 1.0f)
int	Flags	Ship flags. See "Ship Flags" section.

- - The following section is repeatedMAX_PLAYER_WEAPONStimes --

byte	Fire flags	Firing flags for this weapon. See "Ship Firing Flags" section.
string	Fire sound Surname	D3 Tablefile Sound name for the sound to play when firing. Max length =PAGENAME_LEN.
string	Release Sound name	D3 Tablefile Sound name for the sound to play when the trigger is released. max length =PAGENAME_LEN.
string	Spew Powerup Surname	The generic object name of the powerup object to spew out for this weapon if the ship is destroyed and the weapon is to be spewed. Max length =PAGENAME_LEN.
int	Max Ammo	Maximum amount of ammo this ship can carry for this weapon.
weapon_battery_chunk	Weapon Battery information	This chunk of data contains the weapon battery information for this weapon.

- - The following section is repeatedMAX_WB_GUNPOINTStimes --

string	Fire sound name	The D3 Tablefile sound name for the sound to play for this weapon for when this gunpoint is firing the weapon. max length =PAGENAME_LEN.
--------	-----------------	--

- - End of Section --

- - The following section is repeatedMAX_WB_FIRING_MASKStimes --

string	Weapon Name	The D3 Tablefile weapon name for the weapon associated with this weapon battery slot. max length =PAGENAME_LEN.
--------	-------------	---

- - End of Section --

- - End of Section --

(END OF PAGE)

Ship Flags

Surname	Value	Description
SF_DEFAULT_ALLOW	1	//Allowed by default

Ship Firing Flags

Surname	Value	Description
SFF_FUSION	1	// fires like fusion
SFF_ZOOM	4	// Zooms in
SFF_TENTHS	8th	// Ammo displays in tenths

Bake

Data

Type	Surname	Description
short	version	Version number, used for version control.
string	Surname	D3 Table file name of this sound. max length =PAGENAME_LEN.
string	Filename	. WAV file for this sound. max length =PAGENAME_LEN.
int	Flags	Sound flags. See section "Sound Flags".
int	Loop start	Sample value for start of a loop (looping sounds)
int	loop end	Sample value for end of a loop (looping sounds)
float.float	Outer Cone Volume	Percentage of volume for outer cone.
int	Inner cone angle	Degrees for inner cone.
int	Outer Cone Angle	Degrees for outer cone.
float.float	Max Distance	Maximum sound clipping distance.
float.float	Minimum Distance	Minimum sound clipping distance.
float.float	Import Adjust	Percentage of original .WAV file sound volume to use when loading the sound.

(END OF PAGE)

Sound flags

Surname	Value	Description
SPF_LOOPED	1	// Sound is looped
SPF_FIXED_FREQ	2	// No doppler shift
SPF_OBJ_UPDATE	4	// Sound updates with attached object movements and reacts to scripts
SPF_FOREVER	8th	// Always plays in high-level, this flag should be ignored in low-level
SPF_PLAYS_EXCLUSIVELY	16	
SPF_PLAYS_ONCE	32	
SPF_USE_CONE	64	
SPF_LISTENER_UPDATE	128	// Sound updates with listener movements
SPF_ONCE_PER_OBJ	256	

Bake

PAGETYPE_TEXTURE

Current version: 7

Data

Type	Surname	Description
short	version	Version of this page, used for versioning control.
string	Surname	D3 Table file name for this texture. max length =PAGENAME_LEN
string	Bitmap name	Filename of the .OGF file for this texture. max length =PAGENAME_LEN.
string	Destroy name	Filename of the .OGF file to be used as a 'destroyed' version of this texture. Max length =PAGENAME_LEN.
float.float	Red	Red component of the light this texture gives off (0 to 1.0)
float.float	Green	Green component of the light this texture gives off (0 to 1.0)
float.float	Blue	Blue component of the light this texture gives off (0 to 1.0)
float.float	alpha	Alpha value of the entire texture (0 to 1.0, 0 being fully transparent).
float.float	speed	Speed of the texture (lets it slide) (0 to 1.0)
float.float	U slide	Speed of the sliding of this texture in the U direction.
float.float	V slide	Speed of the sliding of this texture in the V direction.
float.float	Reflectivity	Reflectivity value of this texture, used during radiosity lighting (0 to 1.0)
byte	Corona style	Which type of corona this texture should give off (used if the texture gives off light). A, B, C or D.
int	Damage	Amount of damage per second this texture does if an object collides with it.
int	Flags	Texture flags. See "Texture Flags" section.

-- The following section is to be read only if the texture is a procedural texture --

// Palette section

-- The following section is to be repeated 255 times --

short	16 bit color	Color entry for this palette color
-------	--------------	------------------------------------

-- End of section --

byte	Heat	'Heat' intensity
byte	Light	???
byte	Thickness	???
float.float	Evaluation time	???
float.float	Osc. Time	???
byte	Osc. Value	???
short	Num. Proc Elements	Number of procedural elements. max value = MAX_PROC_ELEMENTS

-- The following section is to be repeated 'Num. Proc Elements' times --

byte	Type	Type of procedural element. See "Procedural element types" section.
byte	Freq.	Frequency of procedural elements
byte	speed	Speed of procedural element
byte	Size	Size of procedural element
byte	x1	X1 value for procedural element
byte	y1	Y1 value for procedural element
byte	x2	X2 value for procedural element
byte	y2	Y2 value for procedural element

-- End of Section --

-- End of Section --

string	Sound name	D3 Table Name of the sound to be associated with this texture. max length = PAGENAME_LEN.
float.float	Sound volume	Volume of the sound

(END OF PAGE)

Texture flags

Surname	Value	Description
TF_VOLATILE	1	// texture is hot - costs 7 shield at contact, steams if shot at
TF_WATER	(1<<1)	// steams if shot at
TF_METAL	(1<<2)	// Shines like metal
TF_MARBLE	(1<<3)	// Shines like marble
TF_PLASTIC	(1<<4)	// Shines like plastic
TF_FORCEFIELD	(1<<5)	// makes the texture a force field
TF_ANIMATED	(1<<6)	// Set for animated textures (.OAF files)
TF_DESTROYABLE	(1<<7)	// texture gets destroyed if shot at, new texture isDestroy name
TF_EFFECT	(1<<8)	
TF_HUD_COCKPIT	(1<<9)	
TF_MINE	(1<<10)	
TF_TERRAIN	(1<<11)	
TF_OBJECT	(1<<12)	
TF_TEXTURE_64	(1<<13)	// texture is 64x64 pixels
TF_TMAP2	(1<<14)	// for textures with an alpha map
TF_TEXTURE_32	(1<<15)	// texture is 32x32 pixels
TF_FLY_THRU	(1<<16)	// texture can be flown and shot thru
TF_PASS_THRU	(1<<17)	
TF_PING_PONG	(1<<18)	
TF_LIGHT	(1<<19)	// texture casts light
TF_BREAKABLE	(1<<20)	// Breakable (as in glass)
TF_SATURATE	(1<<21)	
TF_ALPHA	(1<<22)	// defines the texture as being transparent, seealpha
TF_DONTUSE	(1<<23)	
TF_PROCEDURAL	(1<<24)	// makes the texture procedural
TF_WATER_PROCEDURAL	(1<<25)	
TF_FORCE_LIGHTMAP	(1<<26)	
TF_SATURATE_LIGHTMAP	(1<<27)	
TF_TEXTURE_256	(1<<28)	// texture is 256x256 pixels
TF_LAVA	(1<<29)	// like volatile, but ship continues to burn after contact - costs ~50 shields at all
TF RUBBLE	(1<<30)	
TF_SMOOTH_SPECULAR	(1<<31)	
TF_TEXTURE_TYPES	(TF_MINE + TF_TERRAIN + TF_OBJECT + TF_EFFECT + TF_HUD_COCKPIT + TF_LIGHT)	
TF_SPECULAR	(TF_METAL + TF_MARBLE TF_PLASTIC)	

Procedural element types

Water procedures:	
Surname	Value
PROC_WATER_NONE	0
PROC_WATER_HEIGHT_BLOB	1
PROC_WATER_SINE_BLOB	2
PROC_WATER_RAINDROPS	3
PROC_WATER_BLOBDROPS	4

Fire (non-water) procedures:

Surname	Value
PROC_NONE	0
PROC_LINE_LIGHTNING	1
PROC_SPHERE_LIGHTNING	2
PROC_STRAIGHT	3
PROC_RISING_EMBER	4
PROC_RANDOM_EMBERS	5
PROC_SPINNERS	6
PROC_ROAMERS	7
PROC_FOUNTAIN	8th
PROC_CONE	9
PROC_FALL_RIGHT	10
PROC_FALL_LEFT	11

PAGETYPE_WEAPON

Current version: 8

Data

Type	Surname	Description
short	version	Version of this page, used for versioning control.
string	Surname	D3 Table name of the weapon. max length =PAGENAME_LEN.
string	HUD image	Filename of the HUD icon for this weapon. max length = PAGENAME_LEN.
string	Fire Image	Filename of the .OOF file used for the actual weapon model. max length =PAGENAME_LEN. (May be an .OGF if theWF_IMAGE_BITMAPflag is set)
string	Particle name	Texture Name of the image to use as particles. max length = PAGENAME_LEN.
byte	Particle Count	How many particles the weapon creates
float.float	Particle Life	The lifetime of each particle
float.float	Particle size	The size of each particle
int	Weapon Flags	See "Weapon Flags" section.
string	Spawn name	Name of the weapon this weapon spawns. max length =PAGENAME_LEN.
byte	Spawn count	How many weapons to spawn.
string	Robot Spawn	Name of the robot object to spawn on weapon death. max length = PAGENAME_LEN.
string	Alt. Spawn Name	An alternative weapon to spawn. max length =PAGENAME_LEN.
byte	Alt. Spawn chance	Percentage chance (0 - 100) that the alternate weapon will be spawned.
float.float	Gravity Time	How long to create a gravity field (see BlackShark)
float.float	Gravity size	How big of a radius is the gravity field.
float.float	Homing FOV	Cos of the Field of View for homing weapons.
float.float	Custom size	Custom size of weapon (to override polymodel size)
float.float	Size	Weapon 'blob' size.
float.float	Thrust time	How long does this weapon thrust?
physics_chunk	Physics information	
float.float	Terrain Damage Size	Radius of terrain damage this weapon depends on the terrain
byte	Terrain Damage Depth	How deep is the damage on the terrain
float.float	Weapon Alpha	How alpha'd the weapon is (0 to 1.0, 0 being transparent)
string	Explosion image	D3 Texture Name of the texture to use when the weapon explodes. max length =PAGENAME_LEN.
float.float	Explode time	How long the explosion for this weapon lasts
float.float	Explode size	Size of the explosion
float.float	Player Damage	How much damage this weapon does to a player.
float.float	Object Damage	How much damage this weapon does to a generic object.
float.float	Impact size	The radius of the sphere used when the weapon impacts and explodes.
float.float	Impact Time	How long the impact lasts.
float.float	Impact Player Damage	How much damage the impact explosion does to a player.
float.float	Impact Generic Damage	How much damage the impact explosion does to a generic object.
float.float	Impact Force	Amount of force caused by the impact explosion.
float.float	Lifetime	How long the weapon lives
lighting_chunk	Lighting Information	
float.float	Recoil Force	Recoil force applied to the object (used in Force Feedback also)

- - The following section is repeated MAX_WEAPON_SOUNDS times -

string	Sound name	D3 Table Sound name of the sound associated with this sound slot. Max length = PAGENAME_LEN .
---------------	------------	---

- - End of Section

string	Smoke name	D3 Texture Name of the texture to use for the smoke trail image
string	Scorch image	D3 Texture Name of the texture to use for the scorch mark left by this weapon. max length = PAGENAME_LEN .
float.float	Scorch size	Size of the scorch mark to leave
string	Icon name	name of the icon for this weapon. max length = PAGENAME_LEN

(END OF PAGE)

Weapon Flags

Surname	Value	Description
WF_HUD_ANIMATED	(1<<0)	
WF_IMAGE_BITMAP	(1<<1)	// whether or not the firing image is a bitmap or model
WF_SMOKE	(1<<2)	// Weapon drops smoke as it moves
WF_MATTER_WEAPON	(1<<3)	// This a matter weapon, as opposed to an energy weapon
WF_ELECTRICAL	(1<<4)	// These weapons fire as an electrical storm
WF_IMAGE_VCLIP	(1<<5)	// This weapon fire image is a vclip
WF_SPRAY	(1<<6)	// This weapon is a spray, like a flamethrower
WF_STREAMER	(1<<7)	// This weapon has a streamer effect attached
WF_INVISIBLE	(1<<8)	// This weapon is invisible
WF_RING	(1<<9)	// This weapon is drawn ring style
WF_SATURATE	(1<<10)	// Saturate this bitmap weapon
WF_BLAST_RING	(1<<11)	// Creates a blast ring upon explosion
WF_PLANAR_BLAST	(1<<12)	// Blast bitmap takes on the walls plane
WF_PLANAR	(1<<13)	// This weapon doesn't always face you
WF_ENABLE_CAMERA	(1<<14)	// This weapon can be used for missile cameras
WF_SPAWNS_IMPACT	(1<<15)	// This weapon spawns others on impact
WF_SPAWNS_TIMEOUT	(1<<16)	// This weapon spawns others when it times out
WF_EXPAND	(1<<17)	// This weapon expands when exploding
WF_MUZZLE	(1<<18)	// This weapon produces a muzzle flash when fired
WF_MICROWAVE	(1<<19)	// This weapon makes a microwave effect on the victim
WF_NAPALM	(1<<20)	// This weapon does a napalm effect to objects it touches
WF_REVERSE_SMOKE	(1<<21)	// The smoke trail gets smaller as it ages
WF_GRAVITY_FIELD	(1<<22)	// This weapon has a gravity field
WF_COUNTERMEASURE	(1<<23)	// This weapon is a countermeasure
WF_SPAWNS_ROBOT	(1<<24)	// This weapon spawns a robot upon death
WF_FREEZE	(1<<25)	// This weapon slows a ship/object down
WF_TIMEOUT_WALL	(1<<26)	// This weapon times out like a wall hit
WF_PLANAR_SMOKE	(1<<27)	// This weapon has a planar smoke trail instead of a blob
WF_SILENT_HOMING	(1<<28)	// This weapon does not give a homing lock sound
WF_HOMING_SPLIT	(1<<29)	// This weapon houses when it splits
WF_NO_ROTATE	(1<<30)	// This weapon does not rotate as a bitmap
WF_CUSTOM_SIZE	(1<<31)	// This weapon uses a custom size

Physics Chunks

Data

Type	Surname	Description
float.float	Mass	
float.float	Drag	
float.float	Full Thrust	
int	Flags	See "Physics Flags".
float.float	Rotational drag	
float.float	Full rotational thrust	
int	Number of bounces	Maximum number of bounces for this item (-1 for infinite)
float.float	Initial velocity	
vector	Initial rotational velocity (3 floats: x,y,z)	
float.float	Wiggle amplitude	
float.float	Wiggles Per Second	
float.float	CoEfficient Restitution	
float.float	Hit Death Angle	(the sin of the angle between 0 and 90)
float.float	Max turn roll rate	
float.float	Turn-roll ratio	

(END OF CHUNK)

Physics flags

Surname	Value	Description
PF_TURNROLL	0x01	// Roll when turning
PF_LEVELING	0x02	// Level object with closest side
PF_BOUNCE	0x04	// Bounce (not slide) when hit will
PF_WIGGLE	0x08	// Wiggle while flying
PF_STICK	0x10	// Object sticks (stops moving) when hits wall
PF_PERSISTENT	0x20	// Object keeps going even after it hits another object (eg, fusion cannon)
PF_USES_THRUST	0x40	// This object uses its thrust
PF_GRAVITY	0x80	// Effected by gravity
PF_IGNORE_OWN_CONC_FORCES	0x100	// Ignore it's own concussion force
PF_WIND	0x200	// Effected by wind
PF_USES_PARENT_VELOCITY	0x400	
PF_FIXED_VELOCITY	0x800	
PF_FIXED_ROT_VELOCITY	0x1000	
PF_NO_COLLIDE_PARENT	0x2000	// this object cannot collide with its parent
PF_HITS_SIBLINGS	0x4000	// this object can collide with its siblings (like a bomb) // chrishack -- add flag to weapon page
PF_REVERSE_GRAVITY	0x8000	// this object flies upwards with gravity
PF_GRAVITY_MASK	(PF_REVERSE_GRAVITY PF_GRAVITY)	
PF_NO_COLLIDE	0x10000	// No collisions AND NO RELINKS -- DANGEROUS TO USE if not used correctly
PF_NO_ROBOT_COLLISIONS	0x20000	// No collisions occur with robots
PF_POINT_COLLIDE_WALLS	0x40000	// When colliding with walls, make our radius zero
PF_HOMING	0x80000	// This object (weapon) houses
PF_GUIDED	0x100000	// This object is guided
PF_IGNORE_CONCUSSIVE_FORCES	0x200000	

PF_DESTINATION_POS	0x400000	
PF_LOCK_X	0x800000	
PF_LOCK_Y	0x1000000	
PF_LOCK_Z	0x2000000	
PF_LOCK_P	0x4000000	
PF_LOCK_H	0x8000000	
PF_LOCK_B	0x10000000	
PF_LOCK_MASK	(PF_LOCK_X PF_LOCK_Y PF_LOCK_Z PF_LOCK_P PF_LOCK_H PF_LOCK_B)	
PF_NEVER_USE_BIG_SPHERE	0x20000000	
PF_NO_SAME_COLLISIONS	0x40000000	
PF_NO_DOOR_COLLISIONS	0x80000000	

Lighting Chunks

Data

Type	Surname	Description
float.float	Light Distance	how far the light gets cast, in units
float.float	Start Red component	between 0 and 1.0
float.float	Start Green component	between 0 and 1.0
float.float	Start Blue component	between 0 and 1.0
float.float	Time Interval	
float.float	Flicker distance	
float.float	Directional Field of View	Part of Hemisphere, between 0 and 1.0 - this area will NOT be lit
float.float	End Red component	between 0 and 1.0
float.float	End Green component	between 0 and 1.0
float.float	End Blue component	between 0 and 1.0
int	Flags - see "Lighting Flags"	
int	Time bits - ??	
byte	Angle	NOT USED??? - not working
byte	Lighting Render Type: <div><div>Static lighting0</div><div>Gouraud shading1</div><div>Lightmaps2</div></div>	how the lighting gets applied

(END OF CHUNK)

Lighting flags

Surname	Value	Description
OLF_FLICKERING	1	
OLF_TIMEBITS	2	
OLF_PULSE	4	
OLF_PULSE_TO_SECOND	8th	
OLF_FLICKER_SLIGHTLY	16	
OLF_DIRECTIONAL	32	// Directional light - casts light in a cone
OLF_NO_SPECULARITY	64	// Object does not have specular light cast on it

Weapon Battery Chunks

Data

Type	Surname	Description
float.float	Energy Usage	Amount of energy used by this weapon battery slot
float.float	Ammo Usage	Amount of ammo used by this weapon battery slot
-- The following section is repeatedMAX_WB_GUNPOINTStimes --		
short	Gunpoint Weapon Index	???
-- End of Section --		
-- The following section is repeatedMAX_WB_FIRING_MASKStimes --		
byte	Fire Masks	
float.float	Fire wait time	
float.float	Animation Time	
float.float	Animation start frame	
float.float	Animation Fire Frame	
float.float	Animation end frame	
-- End of Section --		
byte	Number Masks - ???	
short	Aiming GunPoint Index	Gunpoint to use for aiming with this battery
byte	Aiming flags	???
float.float	Aiming Dot	???
float.float	Aiming distance	???
float.float	Aiming XZ Dot	???
short	Flags	Weapon Battery Flags Lake
byte	Quad Fire Mask	Firing mask override to use when this weapon battery is set with the Quad fire flag.

Fire masks: this is a bit mask as to what gunpoints the given weapon battery should fire from when used. Like if the value is 0x32 Then the firing mask for that weapon battery is gunpoints: 0011 0010 or 1,4 and 5 will fire when this weapon battery is fired.

Weapon Battery Flags

Surname	Value
WBF_SPRAY	1
WBF_ANIM_LOCAL	2
WBF_ANIM_FULL	4
WBF_ANIM_MASKS	6
WBF_RANDOM_FIRE_ORDER	8th
WBF_GUIDED	16
WBF_USE_CUSTOM_FOV	32
WBF_ON_OFF	64
WBF_USE_CUSTOM_MAX_DIST	128
WBF_USER_TIMEOUT	256
WBF_FIRE_FVEC	512
WBF_AIM_FVEC	1024
WBF_FIRE_TARGET	2048

Epilogue

So, now about three years have passed - from the first setting of a vertex in an empty space file to the first playable levels, as well as from the first chapters of this tutorial to here. If the interested reader has actually made it this far, I hope that the information here has helped.

I have tried my best to banish the devil of errors from this document, knowing full well that this is almost impossible with a text of such length. One or two explanations are definitely missing; That's why I also depend on you, the interested readership, to eliminate the last shortcomings.

Therefore,

Please report errors, criticism, complaints, suggestions or even more positive things to:

RagilRal@Descentforum.net

Shortcut list

View Control

cycle through windows =Ctrl + Tab
pan =Shift + left mouse
zoom =Shift + Ctrl + left mouse or mouse wheel
scroll left =a
scroll right =d scrolling
=arrow keys zoom in =
w
zoom out =s
Toggle Wireframe/Textured View =t
grid smaller =Page Down grid larger =
Page Up
grid size =1,2,3,4,5,6,7 on alpha block
Move camera to current Face =Shift + c
center Room and reset View =RAlt + c
center current Face =Ctrl+Shift+c center
Mine/Room =c **Mode select**

Face: ctrl + f
Vertex: ctrl + r
object: ctrl + g
path: ctrl + h
Toggle Vert/FaceMode: tab
object or path node:LClick

Selecting next...

Vertex =v
Face =f
object =O
room =r
portal =p
path =H
node =n
select previous above items addshift+ (key)

Mark items in Vertex, Face, and object modes

mark selected =space
unmark all =u
invert markings =i
mark all =m
to mark a selected Face in worldview =m

Mark multiple items

Dragging a rectangle around Vertices and Faces will
mark all that lie entirely within the rectangle. old+
drag to unmark **Modify items**

Moving marked items (Vertices, Faces, Objects, Nodes in 2D
views)

numpad keys or Shift + arrow key:
left =4 or Shift+P right =6 or Shift+P
up =8 or Shift+N down =2 or Shift+
N twist left =1

twist right =3
objects rotate left =7
objects rotate right =9
Rotate around current Vertex =Ctrl+1/Ctrl+3
Rotate around current Face center =
Ctrl+Alt+1/Ctrl+Alt+3
move parallel to current edge =
Ctrl+NumpadPlus/NumpadMinus
move parallel to current Face's normal =
Ctrl+Alt+NumpadPlus/NumpadMinus

Vertices:

place Vertex (Vertex mode) =Into
the Copy marked verts =Ctrl + c
Paste verts =Ctrl + v
Paste verts on top (of copied item) =ctrl + shift + v
Mark all Verts in Face =Ctrl + m

Faces:

flip marked Face(s) =n
copy face =Ctrl + c
paste face =Ctrl + v
paste on top (of copied item) =Ctrl + Shift + v
place Face (Face mode) =Into the place Face
(Vertex mode) =Shift + Ins combining co-planar
faces =Ctrl+Shift+Click propagate texture
alignment =Ctrl+Click center current Face =Shift
+ c
Mark all reachable faces from current =0
Mark all marked Faces Verts:Ctrl + m

Objects:

Set camera view from Object =Ctrl+Shift+g **Ortho
Window:** move Object: Lake **Modify items Level
window:**
Set object move/rotate to X axis =Alt+X Set object move/
rotate to Y axis =Alt+Y Set object move/rotate to Z axis =
Alt+Z move object along set axis in positive dir =Alt +N
move object along set axis in negative dir =Alt +N rotate
object around set axis in positive dir =Alt +P rotate
object around set axis in negative dir =Alt +P

Rooms:

Moving placed rooms
rotate =Alt + mouse slide
=Alt + Ctrl + mouse stop
moving =Old Rotate:1/3

Move up/down:8th/2
Move left/right:4/6
Set Move Step:NumpadPlus/NumpadMinus

Terrain:

texture rotate counterclock- / clockwise=</>
replace all terrain cells matching current cell =
Ctrl+Shift+Alt+LClick
apply current texture to all selected cells =
Alt+Shift+LClick
select / deselect current cell =Alt+LClick select /
deselect next sat (Top) =b/Shift + b select /
deselect next sat (horizon)=x/Shift + x

Nodes

move camera to current node =Shift+d
select node =LClick **Game path**

Ortho Window:

Create new Path =Shift+Ins Insert
GameNode =Into the Delete
GameNode =Del Cycle GameNode
=N/Shift + N Cycle Path =H/Shift +
H
Delete Path = Delete last Node of this Game Path

Other

save room =Ctrl + s save
Level =Ctrl + alt + s

All Number keys are the ones from the numpad, unless otherwise stated;P,P,N,N =Arrow keys