Sniper's Paradise

Who Loves You Baby!!!

Sniper's Paradise!

Introduction

ASEtoT3D is an application for converting from Discreet's Ascii Scene Exporter file format to Epic's Unreal Editor T3D format. It is designed in particular for the importation of textured brushes into the editor. This is version 1.0.2 and any suggestions for changes or bugfixes are welcomed.

About

The application was written over the period of six weeks by Daire Stockdale in response to a request by Daniel Patton for a convertor to T3D which preserved the texture co-ordinates. The Unreal Editor supports several file formats, including the autocad DXF format, however only the T3D format contained texture descriptions. Daniel identified the ASE format as being a suitable format for conversion, as it was exportable from '3ds max', contained the necessary data, and was a human readable Ascii format. He contacted Daire, who had previous experience with the ASE format, and the project was begun. The convertor is not perfect, and there may yet be hidden bugs or problems waiting to be found. If you find any of these, or have any suggestions as to how the convertor might be improved, please contact the author at solosnake@hotmail.com, as he is committed to incorporating all useful ideas and bugfixes.

Instructions

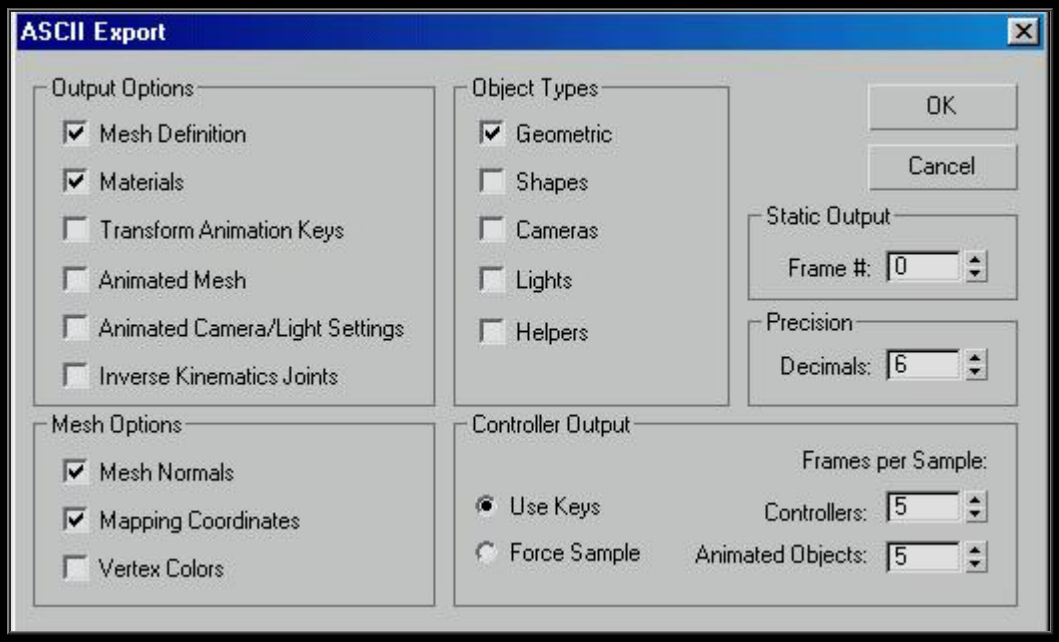
Using

The convertor is simple to use. To begin with, choose the model you have created in 3ds max which you wish to import into UnrealEd as a brush. For best results it is recommended that you texture the object in 3ds max. This is what the convertor is designed for. As the Unreal Editor only supports single textured brushes, the only texture that the convertor uses in the 3ds scene is the DIFFUSE texture map. All other textures will be ignored, or may even confuse the convertor.

Note: As yet the convertor cannot convert correctly textures which are both rotated and translated in 3ds max. This means that you cannot combine both the U or V offset with any rotations.

Note: In 3ds max there are three possible axis of rotation allowed for a texture map : U, V and W. These correspond to the texture's X, Y and Z axis. However the ASE file format only supports rotation in the W/Z axis, so it is advised you do not use the U and V rotations, as the results will not be the same in UnrealEd This is due to the limitations of the ASE file format, not the convertor.

Once you have your model ready, export it to an ASE (ASCII Scene Export) file, ensuring to export all of the following at the minimum: the Mesh Definition, the Materials, the Normals, the Texture Co-ordinates, as shown below:



Remember that once converted the model will be a single brush. The units of your scene will become one unit within the editor.

Now run the convertor, and click on the square button near the ASE edit box window. You will be prompted for the ASE file to convert.

Once you have chosen or nominated the ASE file to convert, you may then choose the T3D filename to Save it as, or an existing T3D file to overwrite.

You may wish to change some of the 'Advanced' flags prior to conversion.

Clicking on convert will convert your ASE file to the T3D file. A summary window will be displayed:

Brush Name
This will indicate the name of the brush within the T3D file. This name is usually the name of the ASE file that was converted.

Polygon Count
A polygon count is also shown. This is equivalent to a triangle count or a face count, as all objects are stored as triangles by this convertor. Thus three times this is the total vertex count.

Texture Count
More correctly this is the number of materials used in the ASE file. As only the diffuse map parameter in the ASE format appears in the T3D format, the actual number of textures used in the T3D file may be less then this.

Warnings

Warnings are generated when a nonfatal error occurred during the translation. These usually result from the ASE file containing 'null' geometry objects, ones that contain no actual data. This might result from creating and then deleting objects within the 3ds max scene, or perhaps from geometry subtraction operations that remove entire objects. In general you can ignore them, however if your scene does not look entirely right, or perhaps an entire part of the scene is missing, then you might need to go back and ensure that you have correctly applied materials and textures to it, and exported it correctly.

Errors

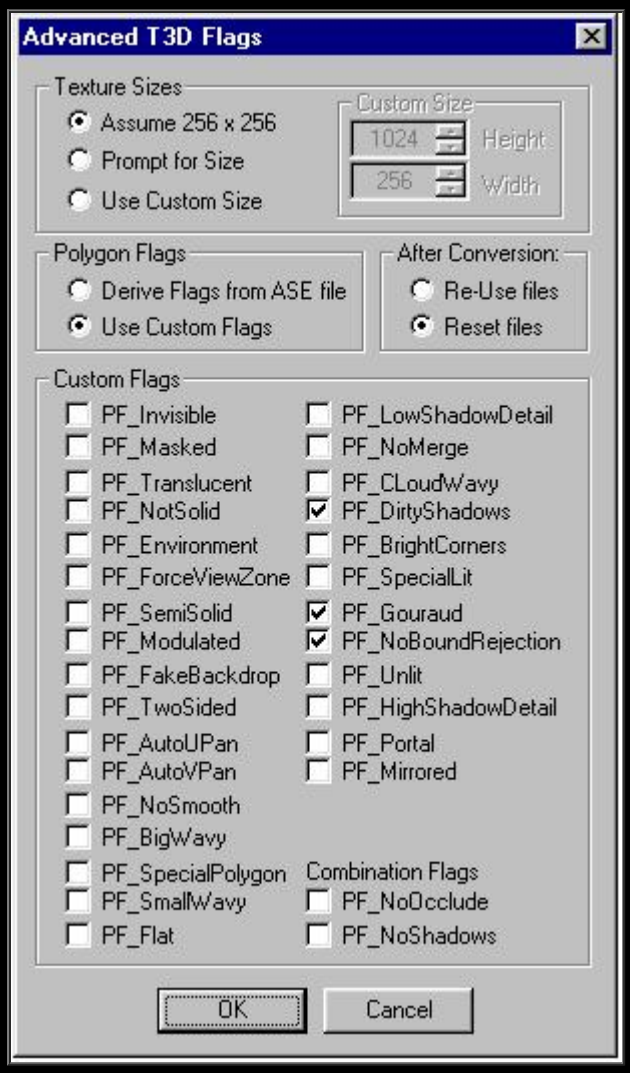
Errors are generated when the convertor cannot convert a file for some reason. The author has attempted to make the errors 'user friendly' and self explanatory. Possible errors are due to:

An object did not have the correct texture faces listed.
An object did not have its mesh normals listed.
An object had mesh normals but they are not correctly listed.
A material was not in the correct format for the convertor.
The file is not a valid ASE file.
Anonymous error. This is a 'catchall' error, meaning something unusual happened.
To any programmers: these errors are the results of try / catch exception blocks in the code.

It is possible (hopeful) that some of these errors may never occur.

Advanced

The 'Advanced Options' window allows you to change the T3D file output to some degree. Shown here is the advanced options window, and some possible settings.



The Advanced window can be broken down and described as its three subunits:

Texture Sizes

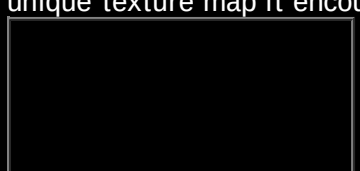
The Unreal Editor / Engine is unusual in that its texture co-ordinates are dependant on the size of the texture map being used. This is not normally the case, and it is possible and arguably more convenient to use dimensionless texture co-ordinates, such as are employed by OpenGL and DirectX. It is the opinion of the author that the Unreal texture co-ordinate system is perhaps a legacy of a software rendering system. As a result texture co-ordinate conversion is not as straightforward as might be hoped. The system requires knowledge of the dimensions of the texture map in order to correctly calculate its texture co-ordinates.

Detailsof a textures dimensions can be passed to the editor in four ways:

Assume 256 by 256
As most, if not all, of the standard set of textures in Unreal are of size 256 x 256 pixels, the convertor provides a means of using this as a default. When this is selected, all texture coords will be calculated based upon a texture of these dimensions.

Use Custom Size
This option should be used when you know that all of the texture maps you are using in your scene are of the same dimension, which is perhaps not 256 x 256. When this option is chosen the Custom Size box becomes active for you to change the size.

Prompt for Size
When this option is chosen, the convertor will display the window shown below for each unique texture map it encounters whilst converting the scene.



The window title indicates the name of the texture it is expecting the dimensions of. Note that this might not be a PCX file: it will be the name of the texture map as specified in the ASE file. You can manually enter the size of the texture here, which is the third method of entering the data, or, if the texture map exists as a PCX format, you can use the 'Browse' button. Select the PCX texture you wish to use, and the convertor will extract the dimensions from the file itself.

Polygon Flags

Derive flags from ASE
The convertor can in a very limited way 'guess' what flags you might want to use, by looking at the data in the ASE file. However as there is not a great degree of overlap in the ASE object description and the possible flag descriptors in Unreal, power users are advised to use the Custom Flags option. Flags that the convertor may set, and why, are as follows:

PF_LowShadowDetail : If the object is set to receive shadows in the 3ds scene, then this flag will be set. The choice of low over high was made by the author out of a desire to minimise possible performance costs associated with higher details.

PF_Translucent : If the material used upon an object contains any degree of translucency, then this flag will be set. To ensure it is not set, the material must have a Transparency value of zero.

PF_Gouraud : Although 3ds does not have a specific 'Gouraud' shading option, this flag will be set when the shading mode is set to 'Blinn'. However the author is uncertain as to whether the PF_Gouraud flag is supported itself in Unreal, so this flag is of limited usefulness.

Custom Flags
Each checkbox simply corresponds to its namesake flag as used in the Unreal Editor. As these flags are Unreal Editor details, there exact uses are not detailed here. The flags are just bit switches, and you may note that several are repeated. This is what will cause the setting of one flag to set another, apparently unrelated flag. Presumably these flags that share the same value are intended to be used in a mutually exclusive manner. When used, each polygon in the created brush will have these flags.

After Conversion

This option is provided because you may want to keep the convertor open and reconvert an ASE file often, perhaps whilst tweaking it in 3ds max and viewing it as a brush in UnrealEd. When 'Re-Use Files' is chosen, the last ASE and T3D file remain visible in the selection windows, and clicking on 'Convert' will convert them, again and again. If the 'Prompt' option in the texture size options box is chosen, you will still be asked to input the texture sizes again, as the convertor has no way of being sure that the textures have not been changed since the last conversion, or that perhaps it is the texture size that you wish to vary. Choosing the 'Reset' option clears the windows and waits for a new selection.

Limitations

In creating the convertor the author has attempted to allow the Unreal Editor user to import almost any geometric description from 3ds max as possible. Perhaps the single greatest limitation is that the author has been unable to find a way to correctly incorporate both the texture W rotation and the Texture U and V offset. Models which use both abilities are unlikely to display correctly textured as a T3D brush. It is the sincere hope of the author that this limitation does not detract too greatly from the usefulness of this tool to the Unreal community, and it is possible that this problem will be solved for later releases.

The convertor can also convert untextured / material-less objects, however these will have unusual texture formations which will need to be corrected.

There is no immediate upper polygon limit, although objects of a count greater then 1,431,655,765 faces will most likely result in integer overflow. It is the opinion of the author that if you are creating objects with this many faces then you may need to rethink your model design.

Note also that the Unreal Editor prefers 'solid' closed geometric shapes. Shapes that are

impossible in teh real world, such as polylines and stand alone planes are not recommended to be used.

Some of effects that can be applied to objects in 3ds max do not appear to be expressed in the ASE file and so regrettably from time to time the brush in UnrealEd may not appear exactly as it looked in 3ds max. This applies especially with regard to textures. The ability to rotate a texture about three axes in 3ds max, but with only one axis angle stored in the ASE file is an example of such a difference.

Command Line

To facilitate integration with other applications, ASEtoT3D maybe run from the command line in a variety of manners. The first is by calling it with a single file to convert, as shown

C:\>ASEtoT3D robot.ASE

The default behavior is to strip the 'ase' suffix and to attempt to derive the flags from the ASE file. The T3D file created will be the filename with the suffix 't3d'.

The second manner, perhaps more useful, is to call the application and specify both the input and the output files, as shown.

C:>ASEtoT3D robot.ase robotbrsuh.t3d

Again the application will attempt to derive the flags from the ASE file. If this is not what is desired, you can also specify the flags, prefixed by a '-' symbol, as shown:

C:>ASEtoT3D robot.ase robotbrush.t3d -4

In this case we are requesting that the flag 4, or PF_Translucent be used. A full list of the flags, in Base 10 follows:

PF_Invisible	1
PF_Masked	2
PF_Translucent	4
PF_NotSolid	8
PF_Environment	16
PF_ForceViewZone	16
PF_Semisolid	32
PF_Modulated	64
PF_FakeBackdrop	128
PF_TwoSided	256
PF_AutoUPan	512
PF_AutoVPan	1024
PF_NoSmooth	2048
PF_BigWavy	4096
PF_SpecialPoly	4096
PF_SmallWavy	8192
PF_Flat	16384
PF_LowShadowDetail	32768
PF_NoMerge	65536
PF_CloudWavy	131072
PF_DirtyShadows	262144
PF_BrightCorners	524288
PF_SpecialLit	1048576
PF_Gouraud	2097152
PF_NoBoundRejection	2097152
PF_Unlit	4194304
PF_HighShadowDetail	8388608
PF_Portal	67108864
PF_Mirrored	134217728

Flags may be combined by simply adding them together. The editor uses the following flags as groups:

PF_NoOcclude = PF_Masked + PF_Translucent + PF_Invisible +PF_Modulated = 71
PF_NoShadows = PF_Unlit +PF_Invisible +PF_Environment + PF_FakeBackdrop = 4194449

Uninstalling

As the convertor creates and uses several entries in the windows registry, the author feels any good program should be able to 'clean up' after itself. Calling

C:\>ASEtoT3D-UNINSTALL

removes all registry entries created by the convertor. Note that running the convertor will once again reinstate these entries.

Thanks

The author would like to thank the following people who were involved in the project:

Daniel Patton, whose idea the convertor was, and for his assistance, advice on Unreal, the T3D format and the Unreal Editor, and for alpha and beta testing convertor through its development.

Vito Miliano for help with understanding Unreals arcane texture co-ordinate system and conversion from OpenGL co-ordinates. to Unreal texture axes.

William Sheriff and Robert Schwartz, for alpha and beta testing the application during its development.

His fantastic wife Kristina, for her ability to put up with him ignoring her for days on end whilst working at his PC.

Links

Some links to homesites of individuals involved in the creation of the convertor, as well as some Unreal sites

www.solosnake.com
The author's website. Contains amongst other things details of his University course (Computer Games Technology), and pictures of his pet rats, Spock, Bones and Kirk.

3dx3
3DX3 is a unique company that focuses on constructing real time virtual environments.

Perilith
Perilith provides a single banner under which individuals and organizations can pool their knowledge and expertise to advance the fields of 3D visualization, distributed networking, pervasive computing, wearable computing, media processing, and more.

Unreal Developer Network
The site for Unreal developers: unfortunately you need privileged access to most of the best parts.

About solosnake

The convertor was programmed by Daire Stockdale, a 28 year old student of Computer Games Technology at the University of Abertay Dundee, Scotland, UK, in November 2001. Previous to this course he had been studying for a degree in Computer Science at the University of Aberdeen, however the attraction of games programming swayed him to change discipline slightly. Before attending University Daire travelled extensively, from the Arctic in Norway to the Tropics in Central America. He has been a Royal Marine Commando, an officer (briefly) in the Royal Marines, and a French Foreign Legionnaire.

His interest nowadays are mostly centred about programming, in C++. When not programming he enjoys skiing at Christmas in Sweden, reading science-fiction and science-fact, looking after his rats and going to the cinema.

You may e-mail questions, comments or queries about the convertor to
solosnake@solosnake.com

Back To Top