CeX3D Converter 0.4.9 - User's Manual

Ánoq of the Sun, Hardcore Processing *

July 17, 2003

1 Introduction

CeX3D Converter can do:

- `LightWave 3D` to `Unreal Editor` conversion.
- `LightWave 3D` to `RenderMan` conversion.
- `LightWave 3D` to `LightWave 3D` conversion.
- `Unreal Editor` to `RenderMan` conversion.
- `Unreal Editor` to `LightWave 3D` conversion.
- `Unreal Editor` to `Unreal Editor` conversion.

It is available for:

- Windows (x86/win32)
- Linux (x86/Linux)

See *Appendix A: Features* for details about what is supported.

2 This document in other formats

This document is a available in the following different file formats:

- `http://www.CeX3D.net/converter/documentation/manual.html`
- `http://www.CeX3D.net/converter/documentation/manual.pdf`
- `http://www.CeX3D.net/converter/documentation/manual.ps`

3 Disclaimers and copyrights

- CeX3D Converter is copyright © 2000-2001 by Hardcore Processing.
- NO ONE MAY REDISTRIBUTE CeX3D Converter or any parts of it.
- The customer of CeX3D Converter will indemnify, hold harmless, and defend Hardcore Processing against lawsuits, claims, costs associated with defense or accusations that results from the use of CeX3D Converter.
- Hardcore Processing is not responsible for any damages whatsoever, including but not limited to loss of data, interruption of business, personal injury and/or any consequential damage without limitation, incurred before, during or after the use of CeX3D Converter. Hardcore Processing's entire liability, without exception, is limited to the customer's reimbursement of the purchase price of the software (maximum being the suggested retail price listed by Hardcore Processing) in exchange for a signed paper contract of assurance that the customer have deleted all versions of CeX3D Converter that the customer has ever had anything to do with.

4 Installing

To install CeX3D Converter on your computer you should follow the instructions given in the one of the subsections below which correspond to your system.

4.1 Installing on Windows

The Windows version with a graphical user interface is not released yet. It will be very soon though.

4.2 Installing on Linux

1. Make sure that you have installed SDL for Linux. It is available on the `download page` for CeX3D Converter.

2. Unpack the file `CeX3DConverter*_x86-linux.tar.gz`. This is can usually be done with these commands:

   ```
   gzip -d CeX3DConverter*_x86-linux.tar.gz
   tar -xvf CeX3DConverter*_x86-linux.tar
   ```

You can now start the converter under X-Windows by running the program file CeX3DConverter.

5 Using CeX3D Converter

The user interface for CeX3D Converter consists of two main parts:

- A filebrowser on the left side
- Conversion settings on the right side

The way to convert files is basically to select the files that require conversion by using the filebrowser. Then set the conversion settings as desired and press the *Convert* button. This will now be described in detail.

5.1 Using the filebrowser

The filebrowser on the left side of the screen is used for selecting any number of files in a directory to be converted.

5.1.1 Current directory

At the top left of the screen there is a field called *Current directory*. This is a text field for entering the path to the directory where the files to be converted are located. On Windows a path could for example be `C:\LightWaveObjects\`. On Linux an example path might be */home/anoq/LightWaveObjects/*.

5.1.2 Files

A list of files should appear in the listbox called *Files*. These are the files found in the directory entered in the field *Current directory*. It is possible to scroll through the files by using the scrollbar on the right side of the listbox.

The files to be converted should be selected in the *Files* listbox. Selecting a single file is done by pressing the left mousebutton on the name of a file in the list.

Selecting multiple files can be done in one of two ways:

- Hold down the *Ctrl* key while selecting files. When pressing the mousebutton on the name of a file the selection of that file is toggled between selected and not selected.
- Hold down the *Shift* key. Press the mousebutton on one of the filenames and move the mouse up or down while keeping both the *Shift* key and the mousebutton pressed.

You can use both ways of multiselecting in combination as needed.

5.2 Using the basic conversion settings

Before converting files you have to specify how you want the files converted. In this section we will go through the most important settings.

5.2.1 Output format

The most important thing to set is probably the output format. This is done in the listbox at the bottom right of the screen. You select which format you want to convert to by clicking on the name of the desired format in the listbox using the left mousebutton. Currently the following formats are supported:

- *RenderMan RIB* - this converts to RenderMan with shaders
- *Unreal Brush T3D* - converts to a brush for the Unreal editor
- *LightWave 6 and above* - converts to a LightWave 6 object

Notice that you don't have to set the format of the files you convert from. The input formats of the files are automatically detected. This detection is not done based on the names of the files (as many programs do), but is instead based on the actual content of the file. This should be a very robust way of detecting input formats. See *Appendix A: Features* for a list of supported input formats.

5.2.2 To OS type

Another important thing is to set the operating system type that the converted files are intended to be used on. This has an effect on how file paths are converted - for instance the paths to texture files. The operating system type is selected by using the left mousebutton to click on the name of the desired operating system type. Currently the supported operating system types are:

- Windows - for file paths of the type `C:\mydir\`
- Unix - for file paths of the type */mydir/*

It is even possible to rename texture file paths during conversion. This is described in more detail in the section *Using advanced conversion settings.*

5.2.3 Other important settings

When converting to *Unreal Brush T3D* files it is also very important to set the texture resolution settings correctly - otherwise the textures will not work correctly. Setting the scale field correctly is also important when converting to *Unreal Brush T3D* files and you might also want to look into setting the coordinate transformations to get the axis alignment correct.

However we will not go through these settings here. Please turn to the section *Using advanced conversion settings* for a description of this.

5.3 Doing the conversion

When converting files the newly converted files will appear in the same directory as the files they were converted from. Be careful not to overwrite any files by accident! CeX3D Converter will NOT warn you if you overwrite any existing files. If you have converted a file from LightWave to Unreal for instance, and that you later decide to convert the Unreal file back to LightWave, you will most likely be overwriting your original LightWave file! And in this example

CeX3D Converter would produce a LightWave file without textures (among other things).

To convert files press the *Convert* button. This will convert all files selected in the filebrowser on the left of the screen according to the conversion settings entered on the right of the screen.

The next section will go through some more advanced conversion settings.

5.4 Using advanced conversion settings

5.4.1 Default texture width and height

If you are converting to Unreal Brush T3D format then you will most likely want to specify the resolution of the textures you are using. This can be done by filling out the fields *Default texture width* and *Default texture height*. For example if you enter *256* in both fields, the textures used in the object must have the resolution 256x256 pixels in the Unreal Editor.

These settings only affect the Unreal Brush T3D format.

5.4.2 Autodetect texture resolutions

The *Autodetect texture resolutions* lets CeX3D Converter try to determine the resolution of a texture file during conversion. You can click on this checkbox to set or remove a checkmark.

When *Autodetect texture resolutions* is enabled, CeX3D Converter will try to look at all texture files encountered during conversion, to determine their resolution. CeX3D Converter can currently detect the resolution of *Targa* and *BMP* files. If the resolution of a texture cannot be detected, then the resolution entered in the fields *Default texture width* and *Default texture height* are assumed. If you are changing texture filenames during conversion (as described later) CeX3D Converter will try to detect the resolution of the texture after the name has changed, because it is expected that you want to use the texture with the new name in the Unreal editor.

If *Autodetect texture resolutions* is enabled, CeX3D Converter will assume the resolutions entered in the fields *Default texture width* and *Default texture height* for all texture files. This may be handy if you know that you are going to use a different texture resolution in the Unreal editor.

This setting only affects the Unreal Brush T3D format.

5.4.3 Scale

If you wish to scale objects during conversion it can be done by entering a scale factor in the field *Scale*.

For instance if you wish to scale an object by a factor of 2.5 it is done by entering *2.5* in the *Scale* field.

The default scaling factor when is set to *1*. This should be OK when converting to RenderMan. When converting to Unreal, a scaling factor of about *40* or *50* might be more appropriate.

The reason for setting the Unreal scale factor to *40* or *50* is that one unit in *LightWave* is 1 meter. One unit in *Unreal* on the other hand, is about one inch.

5.4.4 From texture root and to texture root

To change the root directory where your texture files are placed during the conversion you can use the text fields *From texture root* and *To texture root*.

The field *From texture root* specifies the prefix of the path to be removed from the texture filenames. The field *To texture root* specifies the new texture root directory.

As an example of this you could change the texture directory from `C:\Textures\` to `T:\` by entering this in the *From texture root* field:

```
C:\Textures\
```

And entering this in the *To texture root* field:

```
T:\
```

5.4.5 Replace texture extension

It is possible to replace the filename extension of all texture filenames. To do this, click on the checkbox *Replace texture extension* so that it gets a check mark. A text field called *New texture extension* will appear. In this field you can enter the new texture extension that you need. Then all texture filenames will have their old file extension removed and the specified extension will be used instead.

For example you can enter the following in the *New texture extension* field:

```
TIFF
```

In this case a texture called *Texture.JPEG* would be renamed to *Texture.TIFF*.

5.4.6 Coordinate transformations (left handed)

When converting objects it is possible to rotate them. This can be done by using the field *Coordinate transformations (left handed)*.

The value to be entered in the field is a comma separated list of transformation operations. Each transformation operation can be one of:

- rx<angle> - for rotating <angle> degrees around the X axis.
- ry<angle> - for rotating <angle> degrees around the Y axis.
- rz<angle> - for rotating <angle> degrees around the Z axis.

For example if you want first to rotate 12.5 degrees around the X axis and then 45 degrees around the Y axis you can write:

```
rx12.5,ry45
```

When converting from LightWave to the Unreal Editor you will probably want to use:

```
rz90,ry90
```

This is because in LightWave (and in RenderMan) the coordinate system looks like this:

- Positive direction of X axis points to the right.
- Positive direction of Y axis points up.
- Positive direction of Z axis points away from the camera.

In the Unreal Editor however, it looks like this:

- Positive direction of Y axis points to the right.
- Positive direction of Z axis points up.
- Positive direction of X axis points away from the camera.

So rotating first 90 degrees around the Z axis and then 90 degrees around the Y axis will transform objects so that *up* in LightWave becomes *up* in the Unreal Editor, *away from the camera* in LightWave becomes *away from the camera* in the Unreal Editor and so forth. It should also be noted that when rotating some multiple of 90 degrees the rotations will be completely accurate - this is not always the case when working in LightWave, so you are encouraged to use CeX3D Converter to handle this :)

You can also consider using the setting:

```
rx90
```

This will at least convert *up* in LightWave to *up* in the Unreal Editor.

The attentive reader will already have noticed that both of the above coordinate systems are lefthanded coordinate systems. This is the reason why the field is called *Coordinate transformations (left handed)* and not just *Coordinate transformations*. When *Coordinate transformations (left handed)* is used, the objects are simply kept in the lefthanded coordinate systems - and this is also what makes most sense, since all supported formats use lefthanded coordinatesystems. However it should be noted that mathematical standard coordinate systems are right handed coordinate systems, so this is what CeX3D Converter uses internally - but the field *Coordinate transformations (left handed)* will not disappear in the future.

5.4.7 Destination filename

When converting files, the filename of each converted file is automatically generated from the input filename. For instance if you convert an object called *MyObject.Lwo* into *Unreal Brush T3D* format, then the destination file will be called *MyObject.t3d*. If you don't want your resulting file to have these default names you can specify your own destination filename in the field *Destination filename*.

However this means that all converted files will get this filename, so it is only useful when converting only a single file.

6 Appendix A: Features

6.1 Supported input formats

- `LightWave 3D` Object files from version 6 and above (LWO2).
- `LightWave 3D` Object files before version 6 (LWOB).
- `Unreal` brush (Unreal.T3D) files for the Unreal Editor.

6.2 Supported output formats

- `RenderMan` RIB files with RenderMan shaders.
- `Unreal` brush (Unreal.T3D) files for the Unreal Editor.
- `LightWave 3D` Object files from version 6 and above (LWO2).

6.3 General features

- Converts multiple files at the push of a button.
- Input fileformats are automatically detected.
- Objects can be scaled during conversion.
- Objects can be rotated during conversion - with all multiples of 90 degrees being completely accurate.
- Texture paths can be changed during conversion.
- Support for arbitrary polygons.
- Support for planar mapped textures.
- Support for UV textures.

6.4 Features for each format

	read LWOB	read LWO2	read Unreal.T3D	write RIB	write Unreal.T3D
Triangles	Yes	Yes	Yes	Yes	Yes
Arbitrary polygons	Yes	Yes	Yes	Yes	Yes
Planar mapped textures	Yes	Yes	No	Yes	One per polygon
UV textures	No	Yes	No	Yes	One per polygon
Other surface data	Yes	Yes	No	Yes	No
Additive surfaces	No	No	No	No	No
Reflections	Yes	Yes	No	No	No
Refractions	No	No	No	No	No
Caustics	No	No	No	No	No

Notice that a feature has to be supported for both the file format you are reading and the file format you are writing before it will actually work.

6.4.1 Limitations in the formats

The Unreal.T3D file format has limited capabilities for texture support. It only supports one color texture per polygon. So if there are multiple textures on a polygon, only the first color texture will be exported in the Unreal.T3D file. However, you can create several LightWave surfaces for different polygons, each with different texture. The Unreal Editor only supports texturing of up to 3 accurate UV texture coordinates - so when working with UV textures you should probably triangulate your objects. The Unreal.T3D file format also only seems to be capable of handling polygons with up to 16 vertices each and only polygon meshes containing up to 500 polygons in total. CeX3D Converter will try to split polygons with more than 16 vertices - but this will of course generate additional polygons, so you may want to split these polygons manually.

7 Appendix B: Planned features

The following features are currently planned to be implemented in a near future:

1. Split LightWave objects based on layers or surfaces.
2. Support for importing LightWave 3D Scene files.
3. Support for exporting Unreal levels.
4. Support for the rest of the surface attributes when converting to RenderMan RIB and Shading Language files.
5. Support for LightWave's fractal noise textures when exporting to RenderMan.

8 Credits

Credits in no particular order goes to:

- The people at ION Storm, in particular Clay Hoffman, Robert Kovach and Peter Marquardt - for using CeX3D Converter on a real production, doing lots of testing, giving lots of ideas and feedback and for being very enthusiastic.
- Erik De Neve from Epic Games - for helping with all my questions.
- Larry Gritz from Pixar - for help and for tolerating my unjustified claims of bugs in BMRT ;)
- Brad Peebler from NewTek - for his help, enthusiasm and interest in CeX3D Converter.
- Virtual Effects and Fantasies - for betatesting CeX3D Converter on a real production.
- Josh Tsui - for feedback and for supplying test material.