PVRGeoPOD

User Manual

Contents

List of Figures

1. Introduction

PVRGeoPOD is a file-exporter plug-in for 3D Studio MAX and Maya. Some notable supported features are as follows:

- Parented nodes
- Mesh instancing
- Capable of exporting as many texture-coordinate sets as MAX and Maya supports
- Boned meshes
- Bone batching (can split a mesh into sub-meshes when the mesh has more bones than the bone palette can fit)
- Complete choice of data formats (e.g. floats, bytes, …)
- Choice of interleaved vertex data or separate channels (e.g. position, normal, texture coordinates).
- Tangent space generation
- Polygon/vertex sorting
- Polygon stripping

2. Installation

2.1.   3DS MAX

2.1.1.   3DS MAX Data Exchange Interface (3DXI) - previously known as IGame

PVRGeoPOD for 3DS MAX uses Autodesk's 3DXI interface, which relies on a file called "igame.dll" that resides in the 3DS MAX installation directory. This file is frequently updated by Service Packs for 3DS MAX, as and when Autodesk releases them, and by specific updates to 3DXI.

Unfortunately, the various versions of "igame.dll" are often incompatible, and only one may be installed at any point in time. It is therefore very difficult to use two different 3DS MAX plug-ins that both use 3DXI, but are compiled to use different versions of it.

PVRGeoPOD attempts to work around this issue by having each build of the plug-in simultaneously support as many versions of 3DS MAX and 3DXI as is possible and practical.

2.1.2.   Supported configurations

Plug-in	3DS MAX release	"igame.dll" version
`3dsmax6\PVRGeoPOD.dle`	3DS MAX 6	6.0.0.56
	3DS MAX 6 + SP1	6.0.1.62
	3DS MAX 7	7.0.0.65
	3DS MAX 7 + SP1	7.0.1.76
	3DS MAX 7 + 3DXI 2.0	7.0.1.78
	3DS MAX 8 + 3DXI 2.0	8.0.0.40
	3DS MAX 8 (contains winding order bug?)	8.0.0.92
	3DS MAX 8 + SP1	8.0.1.11
	3DS MAX 8 + SP2	8.0.1.18
	3DS MAX 8 + SP3	8.0.1.24
`3dsmax9\PVRGeoPOD.dle`	3DS MAX 9	9.0.0.100
	3DS MAX 2008	10.0.0.86
	3DS MAX 2009	11.0.0.57
	3DS MAX 2010	12.0.0.106

2.1.3.   Installation

3DS MAX 6

Copy "`3dsmax6\PVRGeoPOD.dle`" to "`C:\3dsmax6\plugins\`", or the equivalent location where your copy of 3DS MAX is installed.

3DS MAX 7

Copy "`3dsmax6\PVRGeoPOD.dle`" to "`C:\3dsmax7\plugins\`", or the equivalent location where your copy of 3DS MAX is installed.

3DS MAX 8

Copy "`3dsmax6\PVRGeoPOD.dle`" to "`C:\Program Files\Autodesk\3dsMax8\plugins\`", or the equivalent location where your copy of 3DS MAX is installed.

3DS MAX 9

Copy "`3dsmax9\PVRGeoPOD.dle`" to "`C:\Program Files\Autodesk\3DS MAX 9\plugins\`", or the equivalent location where your copy of 3DS MAX is installed.

3DS MAX 2008

Copy "`3dsmax9\PVRGeoPOD.dle`" to "`C:\Program Files\Autodesk\3DS MAX 2008\plugins\`", or the equivalent location where your copy of 3DS MAX is installed.

3DS MAX 2009

Copy "`3dsmax9\PVRGeoPOD.dle`" to "`C:\Program Files\Autodesk\3DS MAX 2009\plugins\`", or the equivalent location where your copy of 3DS MAX is installed.

3DS MAX 2010

Copy "`3dsmax9\PVRGeoPOD.dle`" to "`C:\Program Files\Autodesk\3DS MAX 2010\plugins\`", or the equivalent location where your copy of 3DS MAX is installed.

2.2. Maya

2.2.1. Installation

Maya 7

Windows

Copy "`PVRGeoPOD_v7.mll`" to "`C:\Program Files\Alias\Maya7.0\bin\plug-ins\`", or the equivalent location where your copy of Maya is installed.

Linux

Copy "`PVRGeoPOD_v7.so`" to "`/usr/aw/maya7.0/bin/plug-ins/`", or the equivalent location where your copy of Maya is installed.

Once the plug-in is copied to the correct location, enable it in Maya's plug-in manager.

Maya 8

Windows

Copy "`PVRGeoPOD_v8.mll`" to "`C:\Program Files\Alias\Maya8.0\bin\plug-ins\`", or the equivalent location where your copy of Maya is installed.

Linux

Copy "`PVRGeoPOD_v8.so`" to "`/usr/aw/maya8.0/bin/plug-ins/`", or the equivalent location where your copy of Maya is installed.

Once the plug-in is copied to the correct location, enable it in Maya's plug-in manager.

Maya 8.5

Windows

Copy "`PVRGeoPOD_v8.5.mll`" to "`C:\Program Files\Autodesk\Maya8.5\bin\plug-ins\`", or the equivalent location where your copy of Maya is installed.

Linux

Copy "`PVRGeoPOD_v8.5.so`" to "`/usr/aw/maya8.5/bin/plug-ins/`", or the equivalent location where your copy of Maya is installed.

Once the plug-in is copied to the correct location, enable it in Maya's plug-in manager.

Maya 2008

Windows

Copy "`PVRGeoPOD_v2008.mll`" to "`C:\Program Files\Autodesk\Maya2008\bin\plug-ins\`", or the equivalent location where your copy of Maya is installed.

Linux

Copy "`PVRGeoPOD_v2008.so`" to "`/usr/autodesk/maya2008/bin/plug-ins/`", or the equivalent location where your copy of Maya is installed.

Once the plug-in is copied to the correct location, enable it in Maya's plug-in manager.

Maya 2009

Windows

Copy "`PVRGeoPOD_v2009.mll`" to "`C:\Program Files\Autodesk\Maya2009\bin\plug-ins\`", or the equivalent location where your copy of Maya is installed.

Once the plug-in is copied to the correct location, enable it in Maya's plug-in manager.

Maya 2010

Windows

Copy "`PVRGeoPOD_v2010.mll`" to "`C:\Program Files\Autodesk\Maya2010\bin\plug-ins\`", or the equivalent location where your copy of Maya is installed.

Once the plug-in is copied to the correct location, enable it in Maya's plug-in manager.

2.3. Blender

2.3.1. Requirements

Platforms:

Linux and Windows .

Blender:

Known to work with 2.48, 2.48a, 2.49b

Other:

Python 2.5 (or newer)

2.3.2. Installation

Linux

Copy "lib`PVRGeoPOD.so.xxx`" and "blenderPodExporter.py" to "~/.blender/scripts" (default location where export scripts are stored)

Windows XP

Copy "lib`PVRGeoPOD-xxx.dll`" and "blenderPodExporter.py" to "C:\Documents and Settings\USERNAME\Application Data\Blender Foundation\Blender\.blender\scripts" (location where scripts are stored. Alternatively: "C:\Program Files\Blender Foundation\Blender\.blender\scripts")

Windows Vista

Copy "lib`PVRGeoPOD-xxx.dll`" and "blenderPodExporter.py" to "C:\Documents and Settings\USERNAME\AppData\Roaming\Blender Foundation\Blender\.blender\scripts"

3. Supported File Formats

3.1. POD File

PVRGeoPOD exports to a binary format (file extension: "POD"). The POWERVR SDK Tools library, part of the POWERVR SDK, contains code to load and save POD files.

```
CPVRTModelPOD m_model;

// Load the model
if(!m_model.ReadFromFile("model.pod"))
        return false;

// Do stuff

// Free the memory
m_model.Destroy();
```

3.1.1. Example

See *ChameleonMan* in the POWERVR SDK.

3.2. Header and Source File

PVRGeoPOD can export the binary POD file directly to a header-file (file extension: "H") or a cpp-file (file extension: "CPP") as if it had been wrapped using the POWERVR SDK's FileWrap utility.

To load from a cpp-file, follow the instructions for loading a conventional POD binary file. To load from a header-file, use the POWERVR SDK Tools library ReadFromMemory function.

```
// Include the scene data
#include "model.h"

CPVRTModelPOD m_model;

// Load the model
if(!m_model.ReadFromMemory(_model_pod, _model_pod_size))
        return false;

// Do stuff

// Free the memory
m_model.Destroy();
```

3.3. POD Files Versus Header Files:

3.3.1. Memory

A POD file can be loaded into memory and the vertex data copied into HW friendly buffers, (e.g. a vertex buffer object (OpenGL [ES]) or a vertex buffer (Direct3D [Mobile])). As the graphics API now has its own copy of the data, the POD file can be released from memory (or just the POD file's vertex data if the rest of the POD data is still of use to the application).
This saves memory, compared to headers, on operating systems which do not support page files: the memory used by the data in a header file will be around for the lifetime of the application, and therefore two copies of mesh data are in memory. This is not a problem for operating systems which support a page-file, since the unused memory can be swapped out.
On OpenGL [ES] this is not an advantage if glVertexPointer() and similar functions are used, as no copy operation is required.

3.3.2. Flexibility

It allows POD files to be changed without rebuilding the application.

3.4. Storage

A POD file is smaller on disk (although there should be little size difference between two compiled executables, one using a header file and one with a POD file in resources).
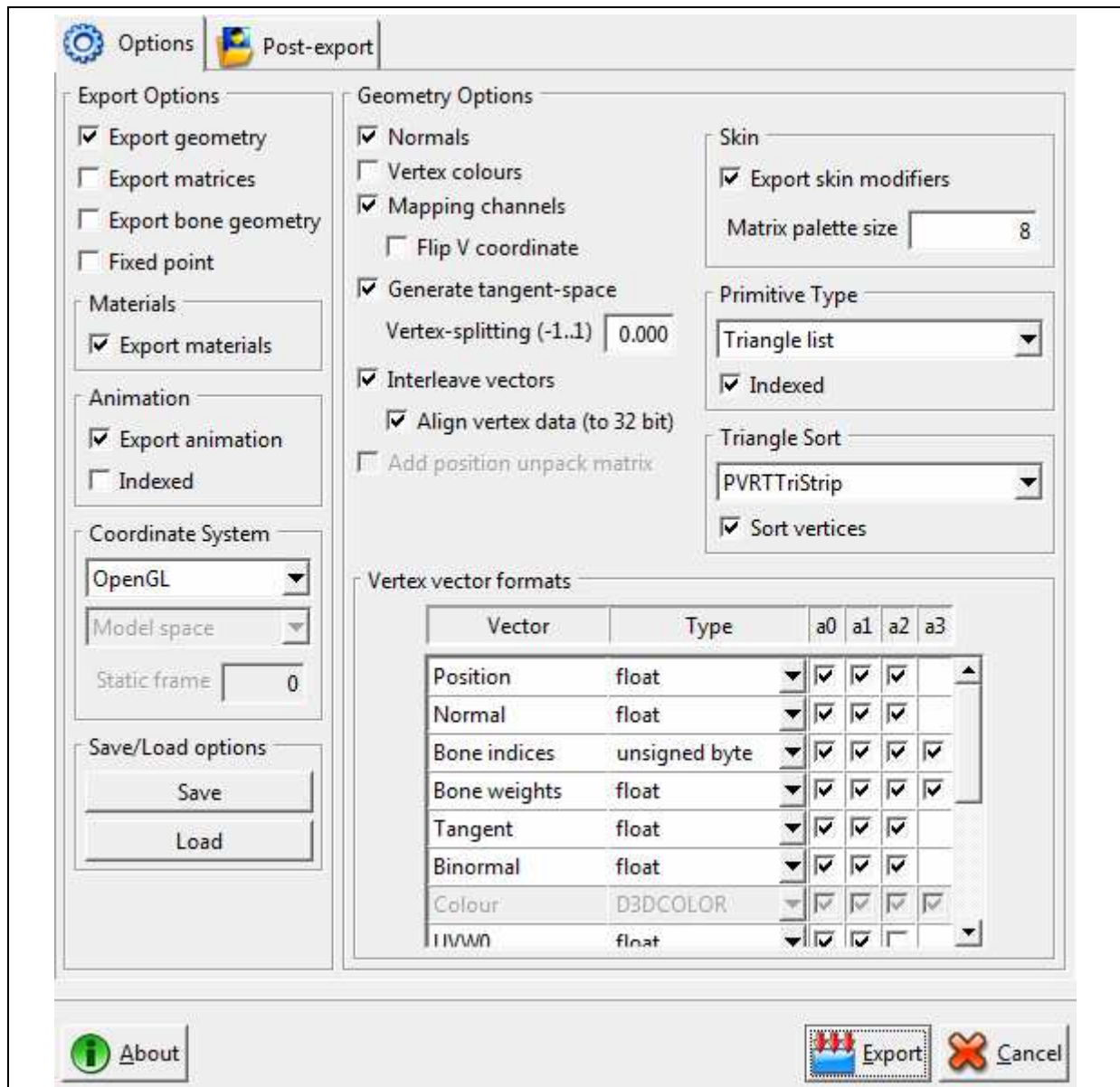
4. Using the Exporter

Figure 1 The PVRGeoPOD dialog box

The function of each dialog-box element is individually described in a tool-tip that will appear when the mouse is hovered over them, as shown in Figure 2.
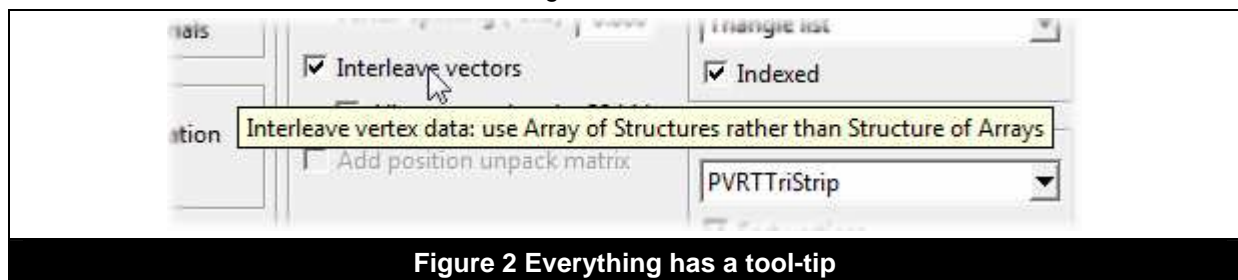

Figure 2 Everything has a tool-tip

The interface is very similar to that of Collada2POD, also available in the SDK and many of these elements are explained in more detail in its documentation.

4.1. 3D Studio Max

PVRGeoPOD can be used in 3DS MAX by performing the following:

1. From the menu, select "`File/Export…`" or "`File/Export Selected…`".
2. In the "`Save as type`" drop-down box, select "`POWERVR Exporter (*.pod,*.h,*.cpp)`".
3. Browse to the location you wish to save the file to.
4. Double-click an old file to overwrite, or type in a filename; use the extension .h for a header-file, and .pod for a binary file. If the filename does not specify an extension, .pod will be assumed.

4.2. Maya

PVRGeoPOD can be used in Maya by performing the following:

1. From the menu, select "`File/Export All…`" or "`File/Export Selection…`"
2. In the "`Files of type:`" drop-down box, select "`POWERVR Exporter (*.pod,*.h and *.cpp)`".
3. Browse to the location you wish to save the file to.
4. Double-click an old file to overwrite, or type in a filename; use the extension .h for a header-file, and .pod for a binary file. If the filename does not specify an extension, .pod will be assumed.

4.3. Blender

PVRGeoPOD can be used in Blender by performing the following:

1. From the menu, select "`File->Export->POD`"
2. Browse to the location you wish to save the file to.
3. Type the filename you wish your POD file to have.
4. Click OK.

5. Using POD Data in your Application

The POWERVR Tools library, part of the POWERVR SDK, contains code to load and save POD files; it also contains many utility functions to help use the resultant data structures, which can be used both when the data is loaded from a POD file and when compiled in from a header file.

The POD data structures are relatively straightforward and easy to use. For example, in OpenGL ES, the vertex (position) data pointer can be used directly as an input to `glVertexPointer()`; the same goes for the pointer to the normals, and all similar data.

For tasks such as acquiring World matrices, bone matrices, etc, you can use the code in the `PVRTModelPOD` module from the POWERVR Tools library.

Appendix A.   POD File Format

This information is irrelevant if the code supplied in "PVRTModelPOD.cpp" - to load binary POD files - is used, as we recommend.

A.1.   Binary File ("POD")

A.1.1.   Binary File Format

POD files are stored in a tagged, nested structure. A marker consists of two 32-bit values:

DWORD	Bit	Symbol	Description
0	31	End	End of data block bit 0 This marker is the beginning of a block 1 This marker is at the end of a block
	30:0	Name	Marker name, identifying the data which follows
1	31:0	Length	Amount of data which follows: i.e. distance to next marker.

- After each marker, Length bytes of data follow.
- The list of possible Name values is in the enumerated type EPODFileName in the file "PVRTModelPOD.cpp".

A.1.2.   Overview of File Reading

```
10 Read marker
20 If recognised marker, read the following data, otherwise skip it
30 GOTO 10
```