

PVRShaman

User Manual

Copyright © 2009, Imagination Technologies Ltd. All Rights Reserved.

This document is confidential. Neither the whole nor any part of the information contained in, nor the product described in, this document may be adapted or reproduced in any material form except with the written permission of Imagination Technologies Ltd. This document can only be distributed to Imagination Technologies employees, and employees of companies that have signed a Non-Disclosure Agreement with Imagination Technologies Ltd.

Filename : PVRShaman.User Manual.1.10f.External.doc
Version : 1.10f External Issue (Package: POWERVR SDK 2.05.25.0737)
Issue Date : 31 Jul 2009
Author : POWERVR

Contents

1.	Introduction	4
1.1.	Requirements	4
1.1.1.	Graphics Card	4
1.1.2.	Operating System	4
1.2.	3D Graphics APIs Supported	4
1.2.1.	OpenGL	4
1.2.2.	DirectX	5
2.	PVRShaman Interface.....	6
2.1.	Main Interface	6
2.1.1.	Scene Browser	7
2.1.2.	Scene Container	7
2.1.3.	Visualisation	8
2.1.4.	Editor	8
2.1.5.	Debug Output	8
2.2.	Menubar	9
2.2.1.	File Menu	9
2.2.2.	Edit Scene Menu	9
2.2.3.	Edit Shader Menu	10
2.2.4.	Build Menu	10
2.2.5.	Render Menu	10
2.2.6.	View Menu	10
2.2.7.	Tools Menu	11
2.2.8.	Help Menu	11
2.3.	Toolbars	12
2.3.1.	Main Toolbar	12
2.3.2.	Scene Toolbar	12
2.3.3.	Editor Toolbar	13
2.4.	Dialog Boxes	14
2.4.1.	Material Properties Dialog	14
2.4.2.	Add Material Dialog	15
2.4.3.	Light Properties Dialog	15
2.4.4.	Camera Properties Dialog	16
2.4.5.	Preferences Dialog	16
2.4.6.	POD Information Dialog	17
2.4.7.	Object Data Viewer	17
2.5.	Window Configuration Modes	18
2.5.1.	Tab Book - Scene	18
2.5.2.	Tab Book - Editor	18
2.5.3.	Floating Scene	19
2.5.4.	Floating Windows	19
2.5.5.	Split Window	19
2.6.	Render Modes	20
2.6.1.	Effects	20
2.6.2.	Wireframe	20
2.6.3.	Wireframe No Effects	20
2.6.4.	No Effects	21
2.6.5.	Depth Complexity	21
2.7.	Navigation Modes	22
2.7.1.	Select	22
2.7.2.	Pan	22
2.7.3.	Rotate	22
2.7.4.	Zoom	22
2.7.5.	FPS Navigation	22
2.8.	Object Viewer	23
3.	How use PVRShaman	24

3.1.	Adding an effect from the library to a 3D file (OpenGLS2)	24
3.2.	Building a new shader for a 3D file (OpenGLS2).....	24
3.3.	Launching PVRShaman from the command line	25
4.	PFX Overview	26
4.1.	Keywords & Blocks.....	26
4.2.	PFX Example.....	27
4.3.	OpenGL2 Interface.....	27
5.	PVRShamans' Semantics	28
5.1.1.	Attributes	28
5.1.2.	Uniforms	28

1. Introduction

PVRShaman is a development environment which allows shaders for OpenGL ES 2.0, OpenGL, DirectX9 and DirectX10, to be created and edited, with the results displayed in a 3D scene. Effect files are used to link the shader code to objects in the scene. An effect file is a complete definition of a material, or a set of materials. It contains textures and shaders. These files can be edited and compiled on the fly, with the graphic results applied to the scene.

The 3D scene must be in POWERVR Geometry format (POD files). These files can be generated with the geometry exporters in the POWERVR SDK from models created with Autodesk's 3dsMax, or Maya. Shaders are applied to the objects in the scene via effect files.

PVRShaman also functions as a viewer for POD files, including those which don't contain shaders.

1.1. Requirements

1.1.1. Graphics Card

You must have a recent shader capable graphics card in your computer to utilise PVRShaman's full functionality. However, PVRShaman can function as a viewer for POD files on older hardware.

1.1.2. Operating System

PVRShaman is available for Windows, Linux and Mac OS X.

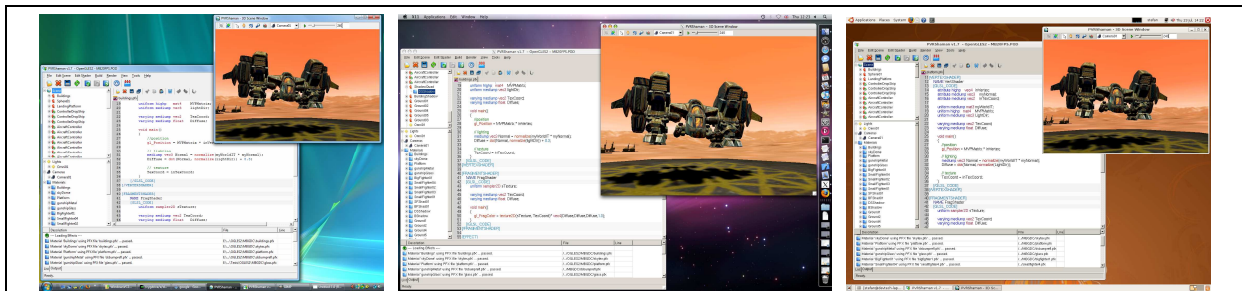


Figure 1 PVRShaman running on Windows (Vista), Mac OS X, Linux (Ubuntu)

1.2. 3D Graphics APIs Supported

1.2.1. OpenGL

For the OpenGL ES2.0 and OpenGL APIs the POWERVR FX (PFX) file format is used. Source code supplied with the POWERVR SDK will allow applications to load and display both PFX and POD files. OpenGL ES1 does not use shaders or effect files.

PVRShaman does full OpenGL ES2.0 validation of shaders each time an effect is applied in the MS Windows version, this feature is not yet available for Mac, or Linux.

1.2.2. DirectX

PVRShaman uses Microsoft Effect (FX) files for DirectX 9 and DirectX 10 modes. The DirectX modes are only available in the MS Windows version.

To use DirectX10 mode, you must have a DirectX10 capable graphics card and you must have the DirectX10 runtime (directx_nov2007_redist.exe) which can be downloaded from Microsoft's website installed.

2. PVRShaman Interface

2.1. Main Interface

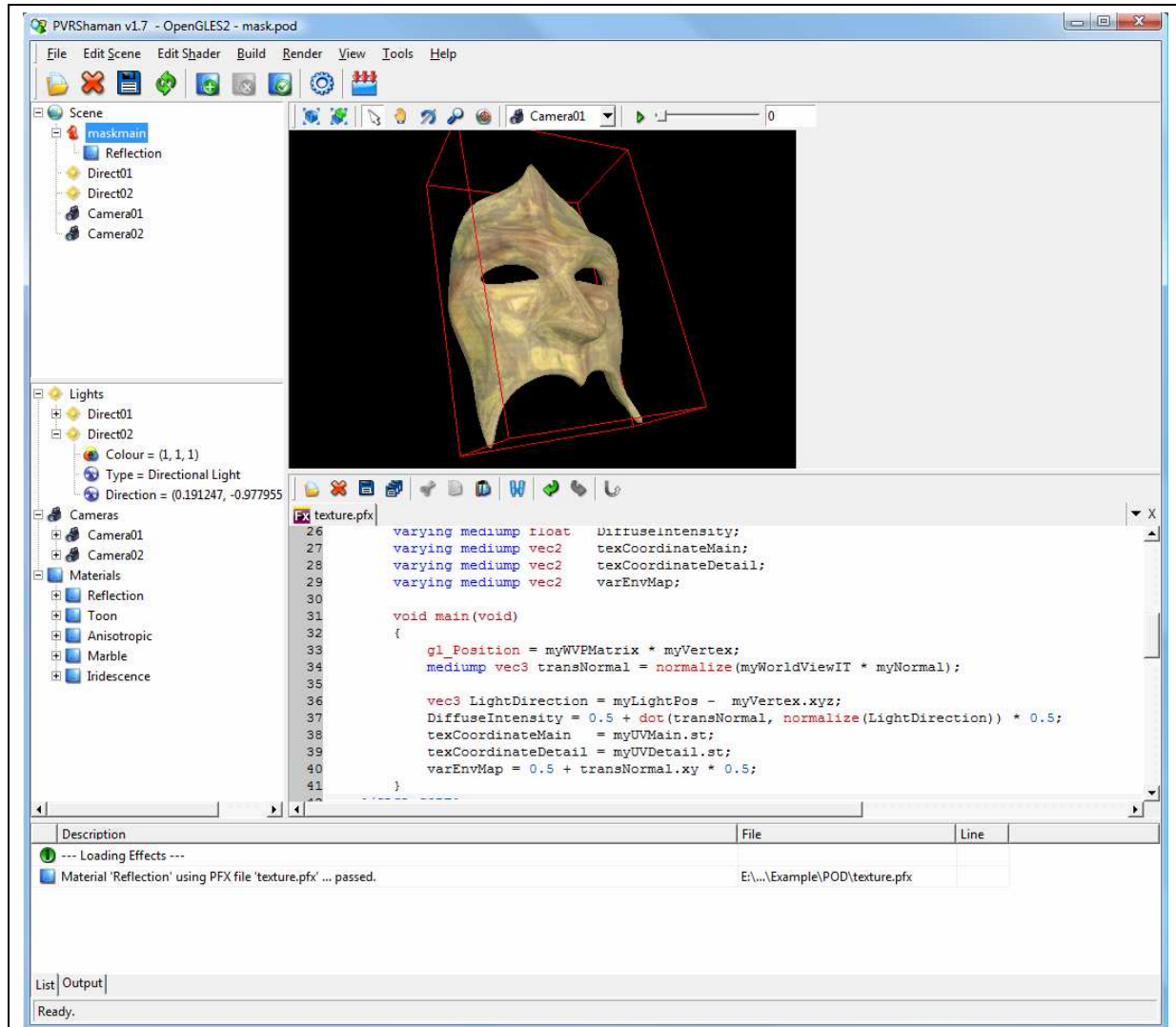
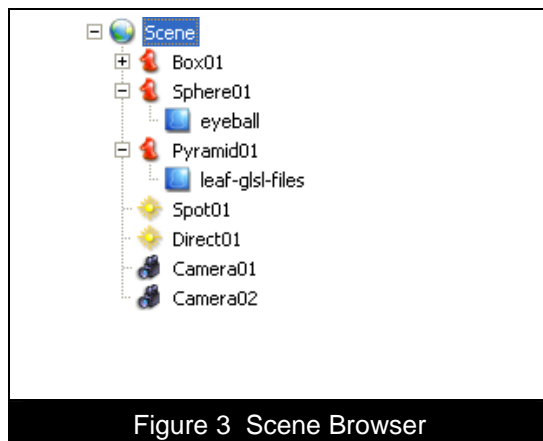


Figure 2 PVRShaman v1.7 (Split Window View)

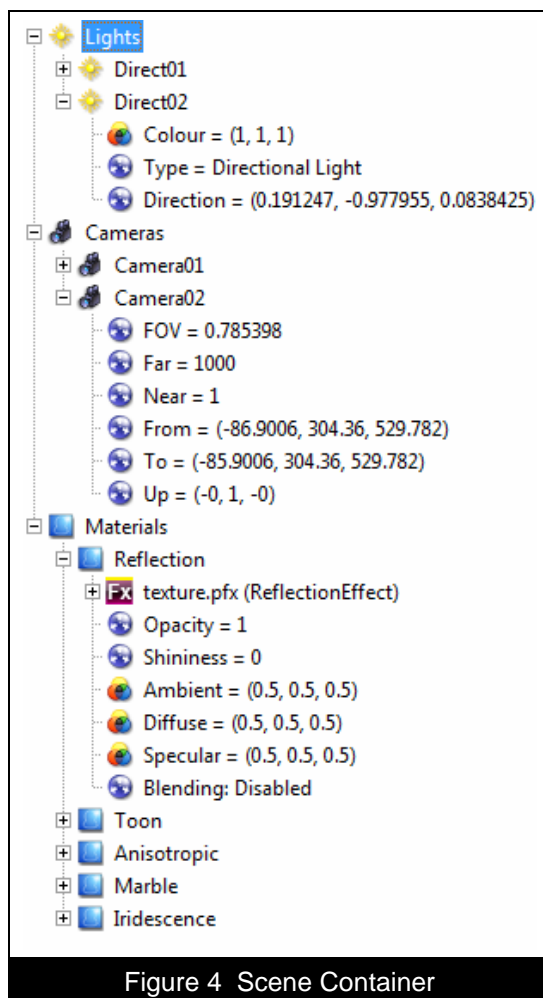
PVRShaman Interface is composed by several panels which hold the information about the current scene, shader editing and the debug output.

2.1.1. Scene Browser



At the top left is the 'Scene Browser'. This shows all the objects in the scene and any materials applied to the meshes. If a mesh has a parent, this will be reflected in the tree structure. A click with the left mouse button will select an object; the object will also be highlighted in the visualisation window if the object is a mesh. Right clicking a mesh will bring up a menu to select the material applied to that object. Double clicking a material, light, or camera, will expand the corresponding item in the 'Scene Container', allowing its properties to be viewed.

2.1.2. Scene Container



Below the 'Scene Browser' is the 'Scene Container' where the components of the scene are 'stored': lights, cameras and materials. Expanding a component will display its properties and double clicking on a property will bring up the properties dialog box to edit these values.

There are two types of materials: A single textured one, which can be exported with a scene from 3ds Max or Maya, and a 'shader' material. The 'shader' material is composed of a single PFX file.

Double clicking on the PFX item will bring up the PFX file in the integrated editor (right from the 'Scene Container' panel). Expanding the PFX item will show any external GLSL files, or any texture files used by the PFX file. Double clicking on the GLSL files will open them in the integrated editor. Double clicking on the texture files will open the texture using PVRTexTool.

Pressing Delete when a material is selected deletes the material. Cameras and lights can not be deleted.

When blending is enabled the blending options are also displayed in the tree.

2.1.3. Visualisation

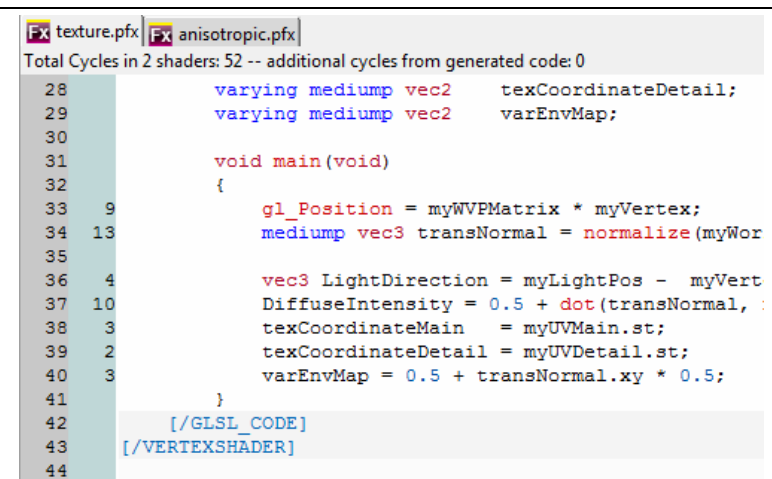


Figure 5 Visualisation

At the top right is the 'Visualisation Panel' where the scene is displayed. This displays and meshes in the scene with the corresponding materials applied. Clicking an object in the scene selects it, when an object is selected right clicking brings up a menu for selecting its' material, similarly to the 'scene browser'. Lights are represented in the scene by yellow squares and cameras by green triangles.

The mouse is used to navigate the scene in a number of different modes (See section 2.7). The mouse wheel moves the scene in and out.

2.1.4. Editor



```

28  varying mediump vec2 texCoordinateDetail;
29  varying mediump vec2 texCoordinateMain;
30
31  void main(void)
32  {
33      9      gl_Position = myWVPMatrix * myVertex;
34      13      mediump vec3 transNormal = normalize(myWor
35
36      4      vec3 LightDirection = myLightPos - myVert
37      10      DiffuseIntensity = 0.5 + dot(transNormal, :
38      3      texCoordinateMain = myUVMain.st;
39      2      texCoordinateDetail = myUVDetail.st;
40      3      varEnvMap = 0.5 + transNormal.xy * 0.5;
41  }
42  [/GLSL_CODE]
43  [/VERTEXSHADER]
44
  
```

Figure 6 Editor

The integrated shader editor works in the same way as PVRUniSCoEditor. It can be used to edit GLSL code, POWERVR FX files, Microsoft effect files, and plain text files. Open files are arranged as tabs, so multiple files can be open at any time. Right clicking in the text brings up the context menu which allows easy insertion of attributes or uniforms to a PFX file. Syntax highlighting is applied to PFX and GLSL files to improve readability. Cycle counters can be enabled for GLSL.

2.1.5. Debug Output

Description	File	Line
--- Apply Effects ---		
error: 'a' unexpected on line 9	E:\...\AnisotropicLighting.pfx	9
Material 'Bumpmap' using PFX file 'Bumpmap.pfx' ... passed.	E:\...\PFX_Library\Bumpmap.pfx	
warning: Variable not used by GLSL code: a WORLDVIEWPROJECTIONIT	E:\...\PFX_Library\CellShading.pfx	
Material 'CellShading' using PFX file 'CellShading.pfx' ... passed.	E:\...\PFX_Library\CellShading.pfx	
Material 'ComplexLighting' using PFX file 'ComplexLighting.pfx' ... passed.	E:\...\ComplexLighting.pfx	

Figure 7 Debug Output

The panel at the bottom is for the output of debug information. In particular if an effect file or shader fails to load, it will report error messages here and the mesh the effect is applied to will appear as a grey un-textured object in the scene.

2.2. Menubar

2.2.1. File Menu

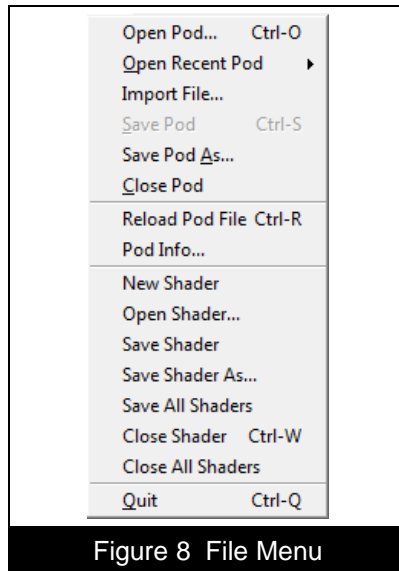


Figure 8 File Menu

This menu is composed of the file input and output options for POD files and shader files. Collada (dae) files can be imported automatically (using Collada2POD) and saved to a POD file using the 'Import File' option. 'Reload Pod File' can be used to reload the POD from its' file, useful if the file has been edited by an external program. The 'Pod Info' option brings up a dialog window showing information about the data contained in the POD file (see section 2.4.6).

2.2.2. Edit Scene Menu

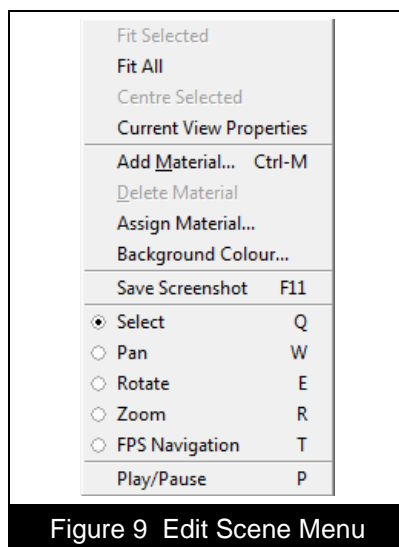


Figure 9 Edit Scene Menu

This menu contains all the options to edit the scene and control animation. 'Fit Selected' and 'Fit All' are used to re-centre and zoom the view so either the selected object or whole scene, respectfully, fill the visualisation window. 'Current View Properties' brings up the camera properties dialog for the current view and allows it to be changed, These changes are for the current view only and will not be saved back to the POD file.

'Add Material' brings up the dialog window, which is used to add new materials to the scene (see section 2.4.2). 'Delete Material' deletes the currently selected material in the 'Scene Container' and 'Assign Material' can be used to assign a material to a mesh, similarly to the pop-up menu when right clicking a mesh. 'Background Colour' brings up the colour dialog window to change the background colour that is stored in the POD file.

'Save Screenshot' saves a screenshot of the current scene as a bitmap file in the same directory as the POD file.

See section 2.7 for a description of the navigation modes. 'Play/Pause' is used to start and stop the animation, if the POD file contains any animation.

2.2.3. Edit Shader Menu

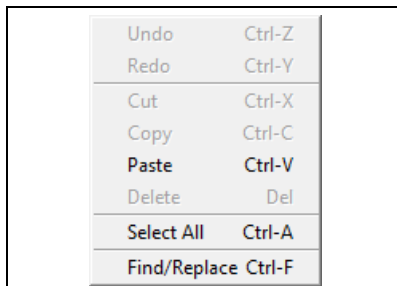


Figure 10 Edit Shader Menu

This menu contains advanced edition options for editing shader code. The menu options work as those expected from any standard editor.

2.2.4. Build Menu

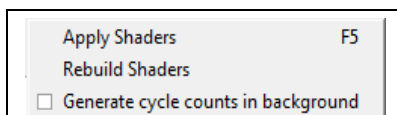


Figure 11 Build Menu

'Apply Shaders' is used to compile (and apply) changes to the shaders after they have been edited. Only files, or images, that have been changed will be re-applied. 'Rebuild Shaders' forces all the shaders to be rebuilt. 'Generate cycle counts' enables the calculation and display of the cycle counts in the editor.

2.2.5. Render Menu

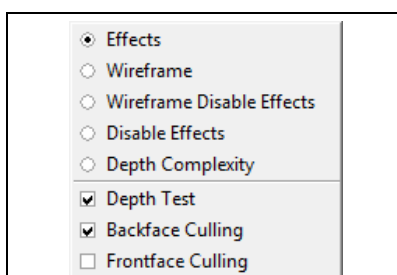


Figure 12 View Menu

This menu allows selection of the render modes (see section 2.56) and enables, or disables, depth testing and culling modes.

2.2.6. View Menu

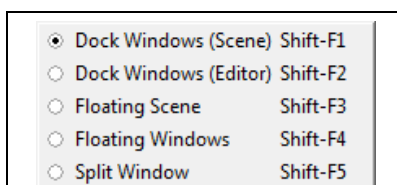
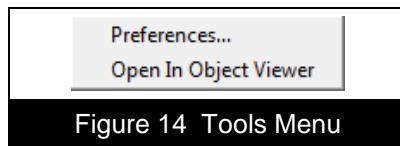


Figure 13 View Menu

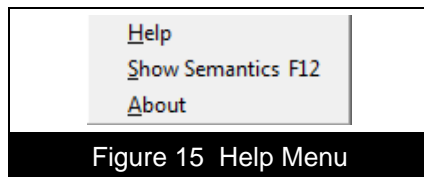
This menu allows selection of the window configuration modes. (See section 2.5.)

2.2.7. Tools Menu



This menu allows setting the preferences (see section 2.4.5). If an object in the scene is currently selected, it can be opened in the Object Viewer (see section 2.8).

2.2.8. Help Menu



The 'Help' option opens this user manual in the default PDF reader. The 'About' option displays information about PVRShaman. The 'Show Semantics' option brings up a dialog window with a list of the semantics (used in the effect file) to tell PVRShaman to assign values to these uniforms.

2.3. Toolbars

2.3.1. Main Toolbar



Figure 16 Main Toolbar

The icons from the left to the right are:

Open POD:

Open a new POD file.

Close POD:

Close the current POD file.

Save POD:

Save the current POD file.

Reload POD:

Refresh the loaded file. This is useful if shader or POD files have been edited outside PVRShaman.

Add Material:

Bring up the dialog box for adding a new material.

Delete Material:

Deletes the material currently selected in the Scene Container.

Assign Material:

Bring up dialog box to assign materials to meshes.

Preferences:

Bring up the preferences dialog.

Apply Shader:

Will validate and apply the current shader to the scene.

2.3.2. Scene Toolbar

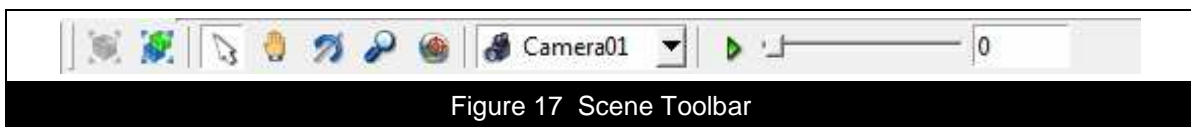


Figure 17 Scene Toolbar

Fit selected:

The scene is scaled so the selected object is fitted to the screen.

Fit All:

The whole scene is scaled to fit the screen.

Select:

Enable selection mode in the scene. Clicking on an object will select it.

Rotate:

Enable rotation mode.

Pan:

Enable Pan mode. Use this mode to displace the scene within the display.

Zoom:

Enable Zoom mode. Use this mode to move the scene in and out.

FPS Navigation:

Enable FPS Navigation mode. Use this mode uses the mouse and keys W, A, S & D to navigate the scene like a FPS game.

Camera selection:

Pull down menu to select the current camera for the viewport. 'No Camera' means that they current view is not from a camera defined in the scene. 'Follow Selected' follows the currently selected object, useful for observing one object through an animated scene.

Play/Pause:

Play or pause the animation, if there is any.

Current Frame:

Used to select the frame to display in the visualisation.

2.3.3. Editor Toolbar



Figure 18 Editor Toolbar

Open File:

Open an effect file or shader file.

Close File:

Closes the current file in the editor.

Save File:

Saves the current file in the editor.

Save All:

Save all files open in the editor.

Cut:

Cut the selected text to the clipboard.

Copy:

Copy the selected text to the clipboard.

Paste:

Paste text currently in the clipboard.

Find:

Search for text in the current file.

Undo:

Undo the last action.

Redo:

Redo an action that has been undone.

Generate Cycle Counts:

Enables calculation and display of shader cycle counts.

2.4. Dialog Boxes

2.4.1. Material Properties Dialog

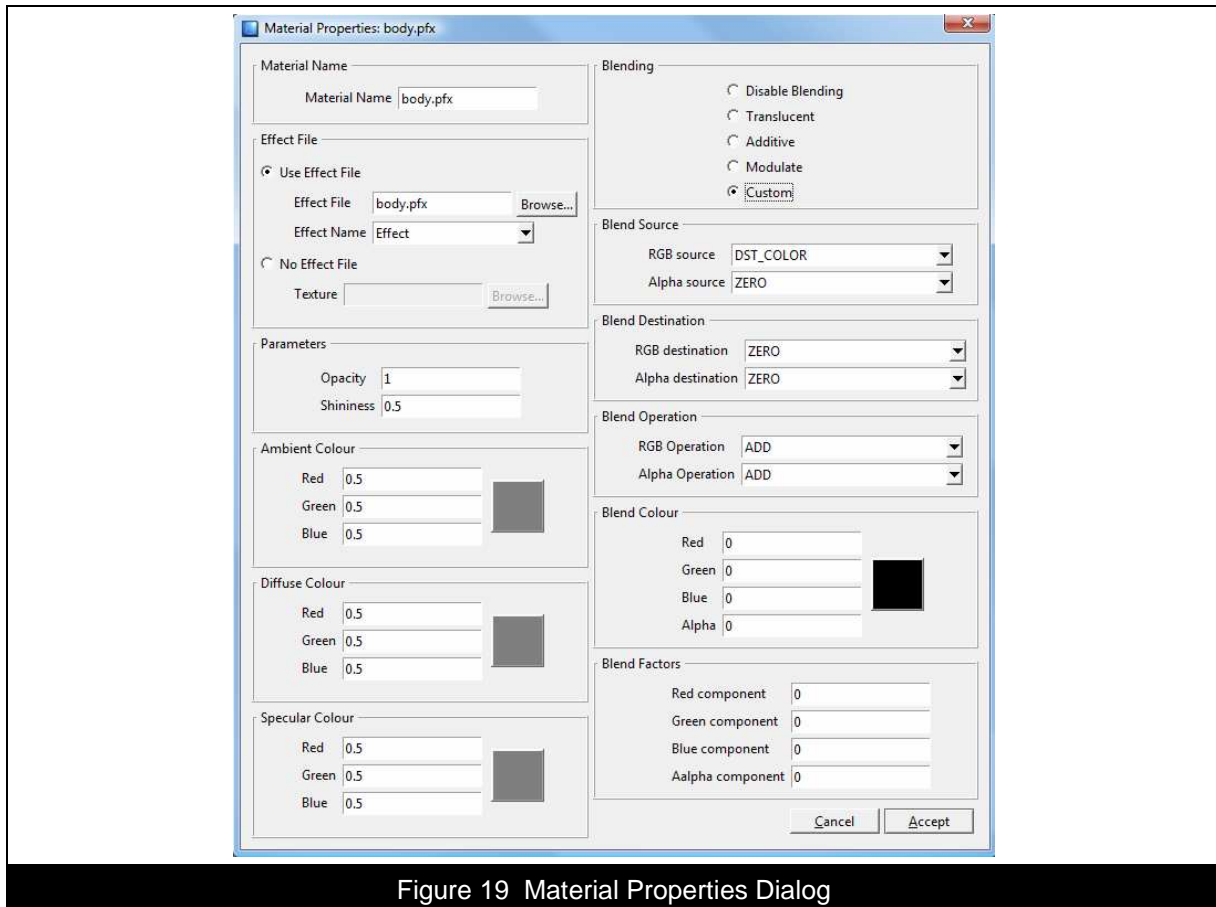


Figure 19 Material Properties Dialog

A material may use an effect file, if no effect is used a texture file can be optionally specified. If a mesh does not have any material assigned to it, it will show up in the default grey. A material must be assigned to a mesh to be displayed; each material can be assigned to multiple meshes.

The material properties dialog box displays a material's properties and allows them to be changed. The type of material can be changed between effect files, or not, and an effect name must be specified if the material is using effect files. Values for opacity, shininess, ambient colour, diffuse colour and specular colour must all be in the range 0 to 1. Clicking a colour panel brings up the colour selector dialog, which is an alternative way to select the colour.

Blending is enabled if the currently set blend mode requires it or if Opacity is set to < 1.0 . Source, destination functions and blend operation are set from the drop down list for both RGB and Alpha. Blend colour and blend factor are used as appropriate for the current blending mode.

The changes are applied to the scene as the values are changed, and can be saved back to the POD file. The path to effect files and texture files is not stored in POD. It is advisable to store PFX and PVR files in the same directory as the POD file to avoid specifying the path each time the POD is loaded.

Please refer to the 'PVRGeoPOD' document, or the IntroducingPOD training course for further information about the POD format.

2.4.2. Add Material Dialog

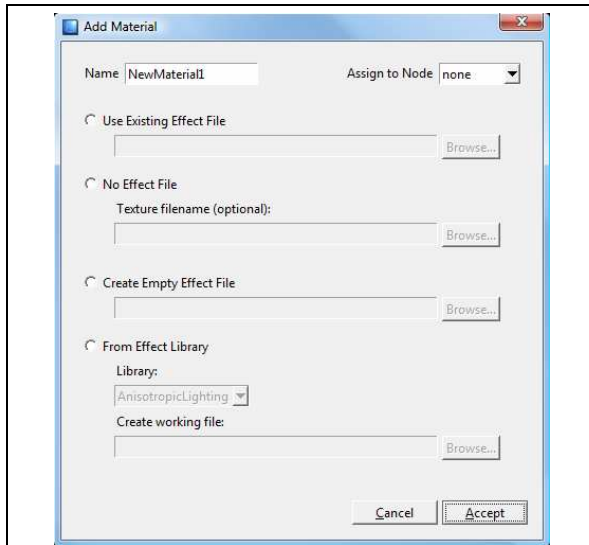


Figure 20 Add Material Dialog

When adding a new material it must be given a name and has the option of being assigned to a mesh at the same time, which can be removed, or changed, at any time.

There are four options when adding materials: use an existing effect file, create a material that doesn't use an effect file (it may use one texture), use an existing texture file, create a new empty effect file, or create a new effect file using one of the library examples.

The new material will be added to the list of materials in the scene container. The POD file can now be saved to store the details of the new material.

2.4.3. Light Properties Dialog

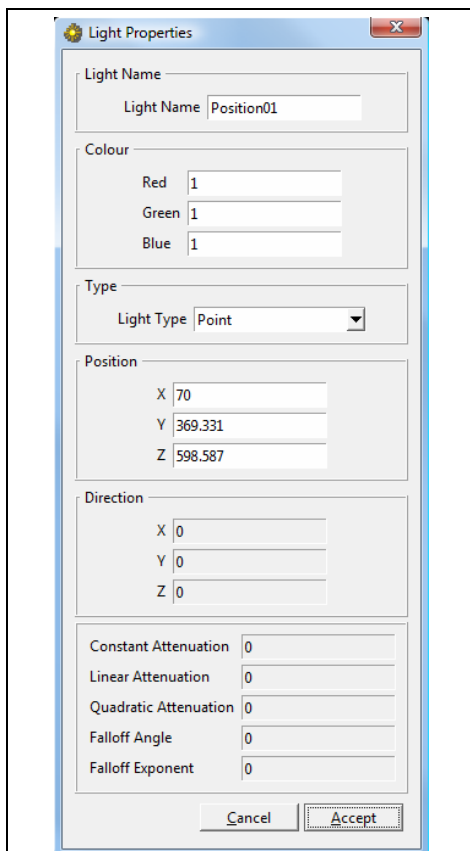


Figure 21 Light Properties Dialog

The light properties dialog allows changing the properties of the light. Colour values must be in the range 0 to 1. The type of light can be either point, directional, or spot light.

Position is the coordinates of the light source (disabled for directional lights). Direction is the direction of the light (disabled for point lights). Values for Falloff and Attenuation are for spot lights only.

Changes will be displayed as they are made and saving the POD file will store the changes.

2.4.4. Camera Properties Dialog

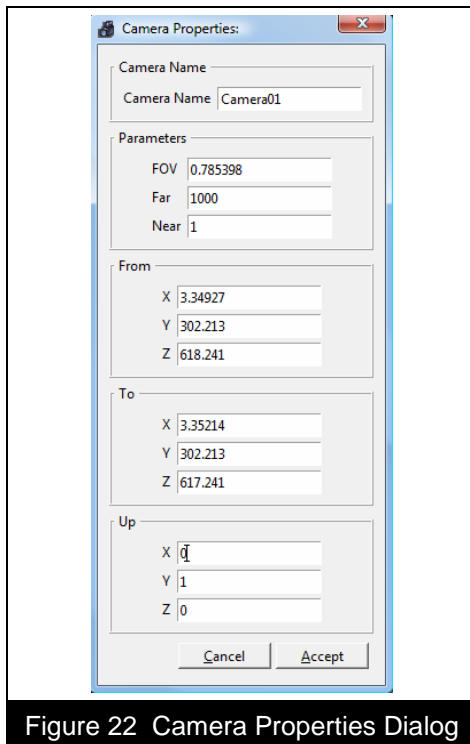


Figure 22 Camera Properties Dialog

The camera properties dialog allows changing the properties of the camera. Changes will be displayed as they are made and saving the POD file will store the changes. The same dialog can also be used to edit the current view.

In the Parameters section FOV is the field of view, and near and far are the positions of the clip plane. 'From' is the position of the camera. 'To' is a point the camera is pointing towards. 'Up' is the up direction of the scene.

2.4.5. Preferences Dialog

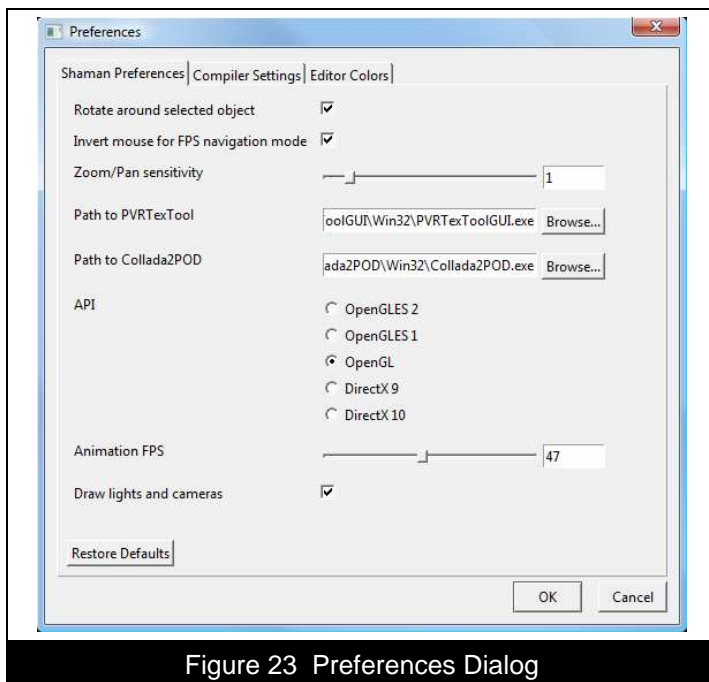


Figure 23 Preferences Dialog

The Preferences dialog is used to change settings for PVRShaman. The editor's syntax highlighting and compiler settings are shared with PVRUniSCoEditor.

The 'Zoom/Pan sensitivity' is used to select the mouse sensitivity for the zoom and pan modes. 'Animation Interval' sets the animation speed.

Location of PVRTexToolGUI and Collada2POD executables to use. An alternative image viewer that accepts filenames as arguments can be used in instead of PVRTexTool, if required.

The API that PVRShaman uses can be changed, PVRShaman will need to be restarted for this change to take effect. OpenGL ES2 and OpenGL modes

expect the effect files to be PFX format and the geometry in the POD file to be left handed. DirectX modes expects effect files to be Microsoft effect files and the geometry in the POD to be right handed.

2.4.6. POD Information Dialog

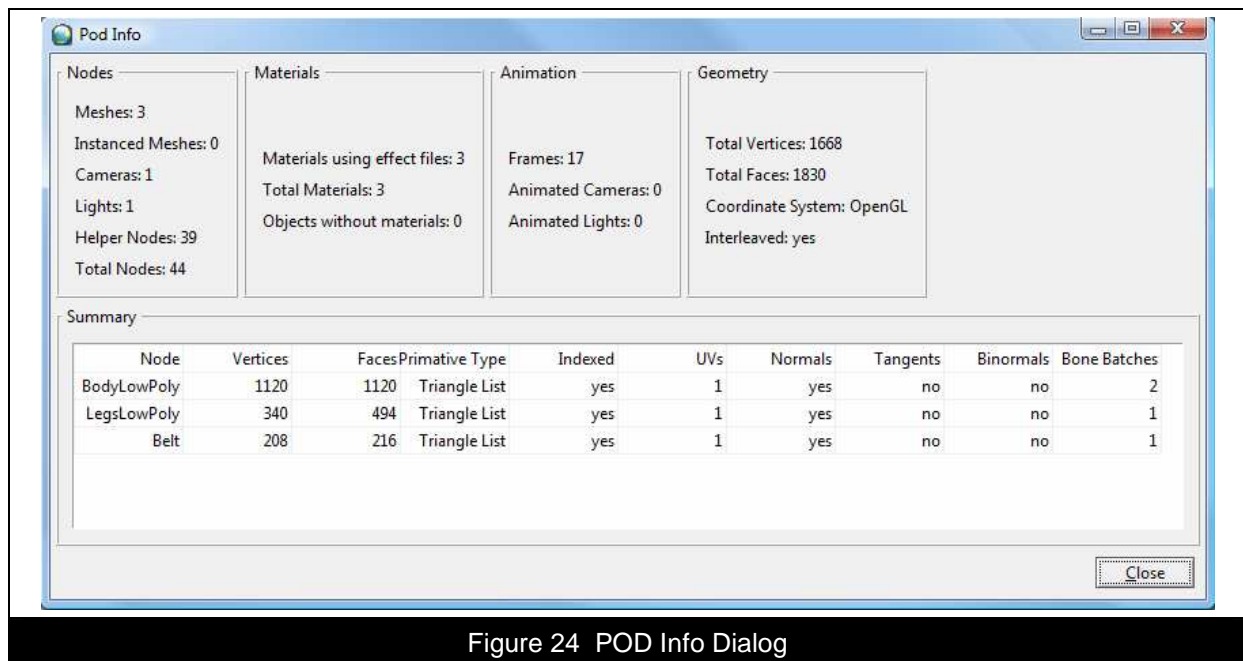


Figure 24 POD Info Dialog

This dialog window displays statistics from the data in the POD file. Totals are at the top, and below is a list of details about each mesh.

2.4.7. Object Data Viewer

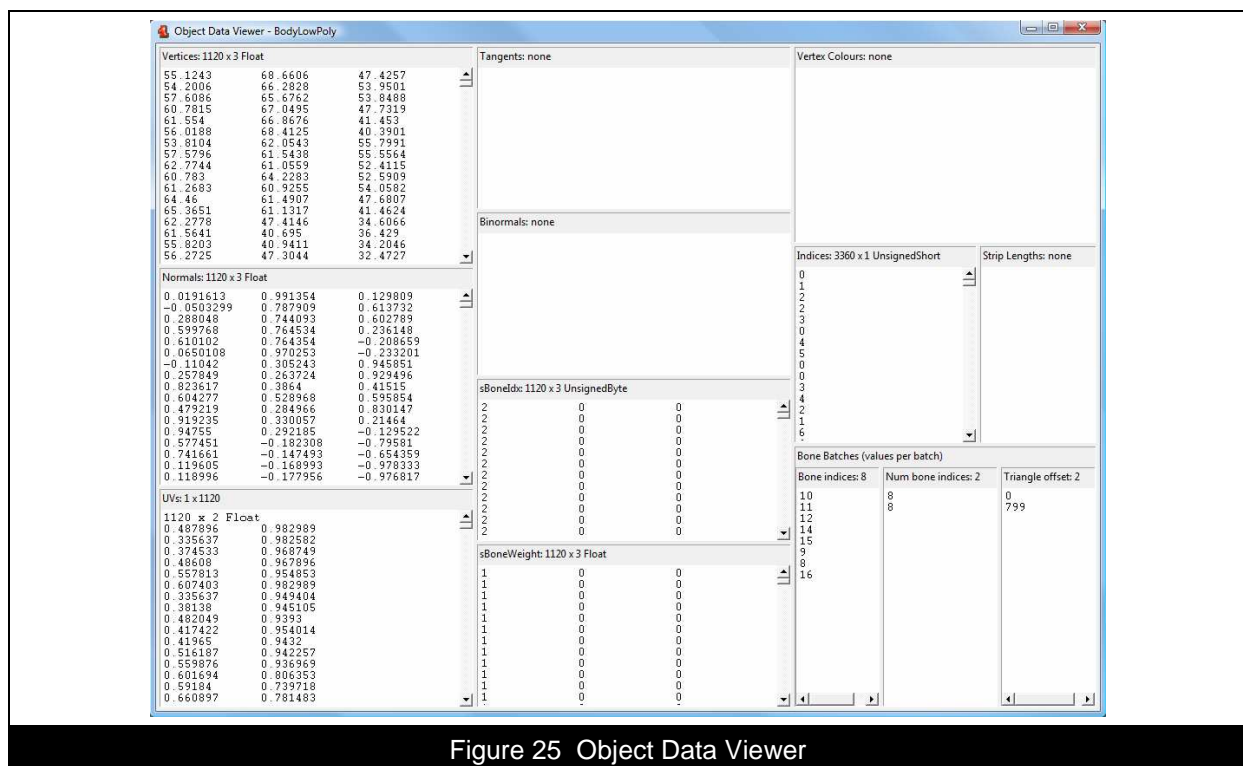


Figure 25 Object Data Viewer

This dialog window displays the data for the object (e.g. vertices, normals, indices, etc).

2.5. Window Configuration Modes

These are the window configuration modes that PVRShaman can use. Shortcut keys Shift+F1 to Shift+F5 can be used to quickly switch between views.

2.5.1. Tab Book - Scene

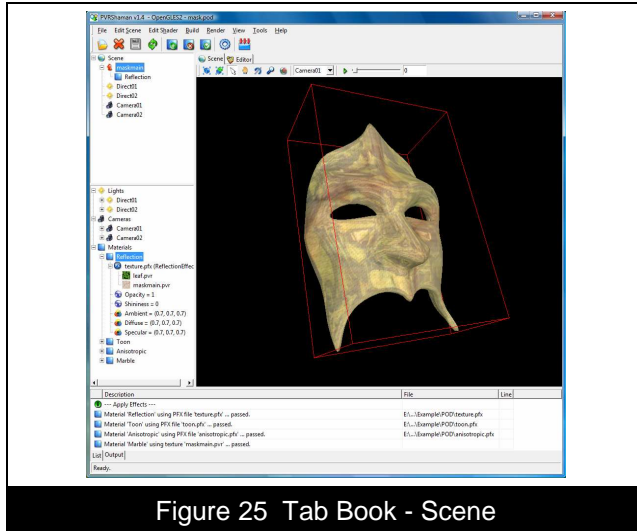


Figure 25 Tab Book - Scene

In this view the Scene and Editor are both docked as tabs in the main window. The Scene tab is displayed.

2.5.2. Tab Book - Editor

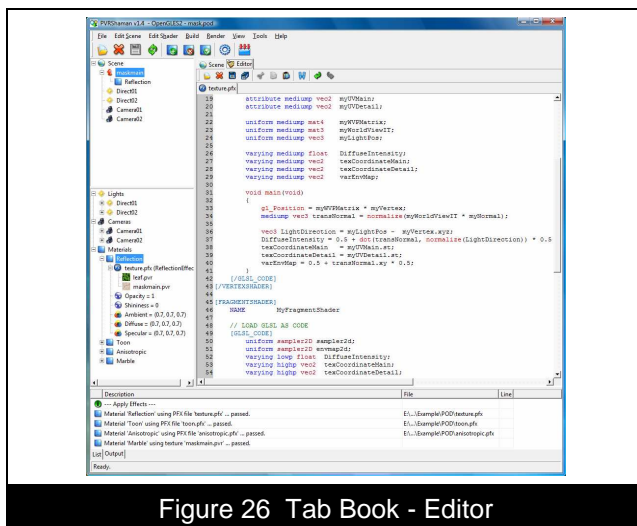


Figure 26 Tab Book - Editor

In this view the Scene and Editor are both docked as tabs in the main window. The Editor tab is displayed.

2.5.3. Floating Scene

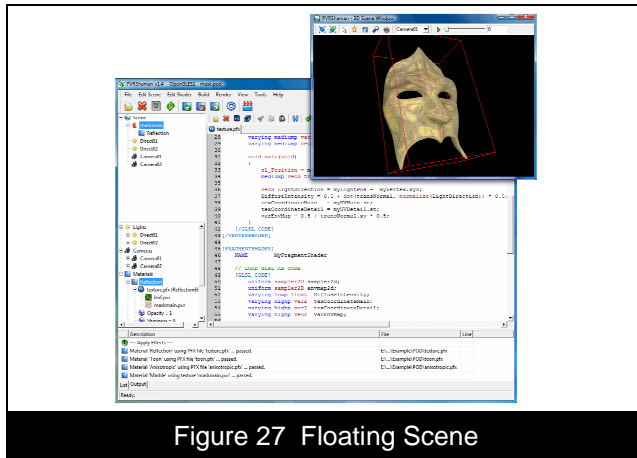


Figure 27 Floating Scene

In this view the scene is in a floating window and the editor is docked into the main window. This mode can be useful when using multiple monitors.

2.5.4. Floating Windows

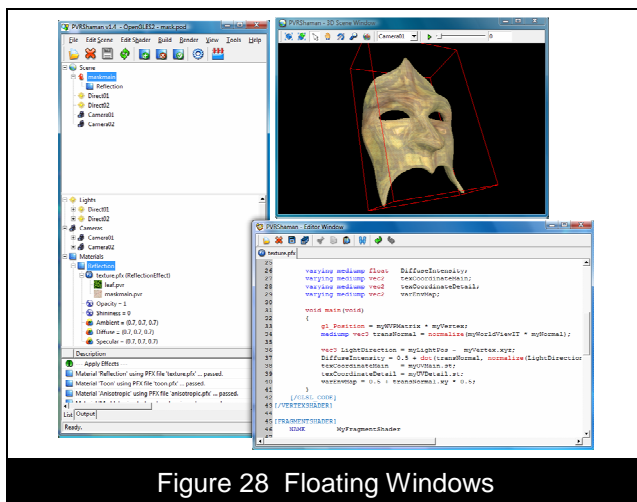


Figure 28 Floating Windows

In this view both the scene and editor are in floating windows and the main window is reduced in size.

2.5.5. Split Window

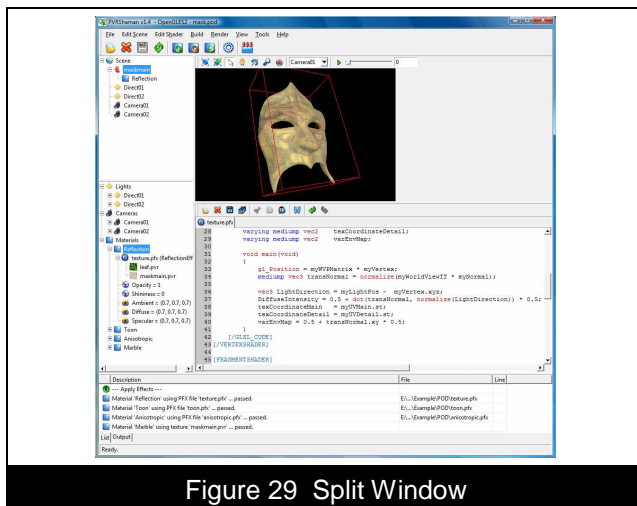


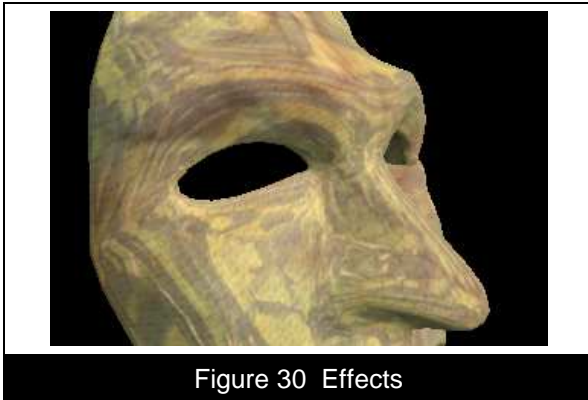
Figure 29 Split Window

In this view the main window is split, with the scene in the top half and the editor below.

2.6. Render Modes

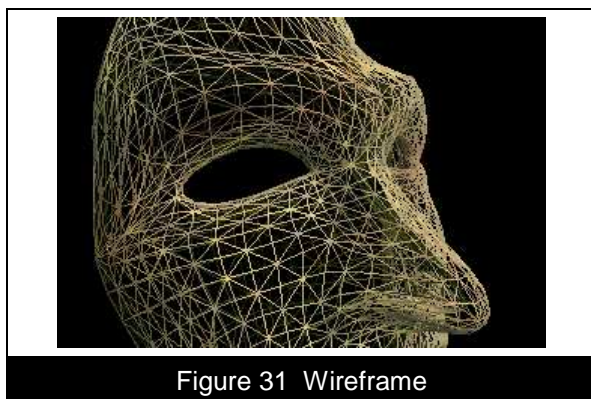
These modes change the ways the mesh is rendered and effects are applied.

2.6.1. Effects



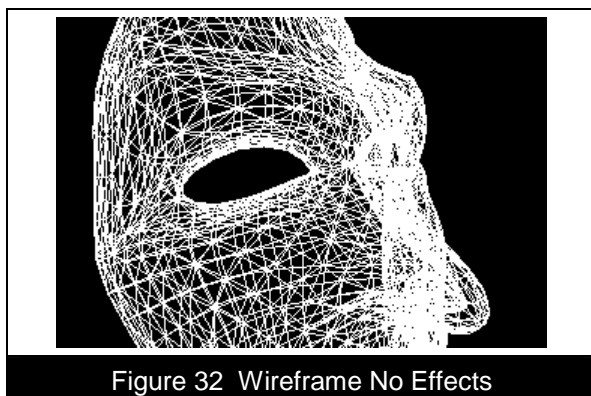
This is the default mode, displaying any PFX and texture effects which are applied to the meshes.

2.6.2. Wireframe



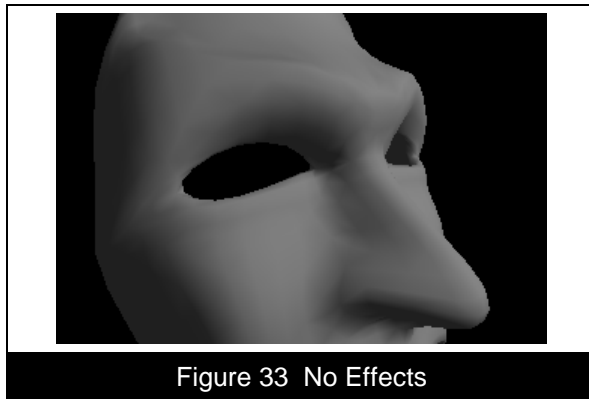
This mode shows the effects applied to a wireframe version of the mesh.

2.6.3. Wireframe No Effects



This mode shows the wireframe mesh without any effects applied.

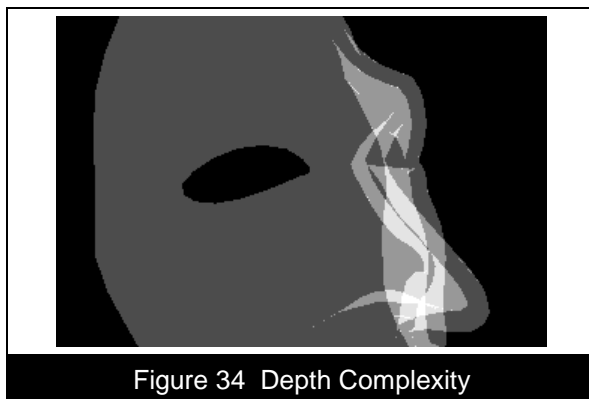
2.6.4. No Effects



This mode shows the meshes without the effects applied.

If no materials are applied to a mesh, or the material fails to load, the mesh will be displayed in this way.

2.6.5. Depth Complexity



This mode shows the depth complexity of the scene. The brighter white means greater complexity.

2.7. Navigation Modes

These modes are used by the mouse to navigate the scene in the visualisation window.

2.7.1. Select



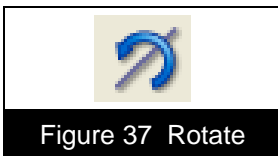
This mode allows selection of the mesh under the cursor by clicking the left mouse button.

2.7.2. Pan



This mode allows moving the scene up, down, left and right.

2.7.3. Rotate



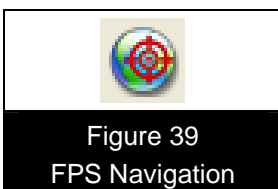
This mode allows rotation around the centre of the scene, or the centre of the current object, if one is selected.

2.7.4. Zoom



'Zoom' moves the scene in/out as the mouse is moved forward/backward.

2.7.5. FPS Navigation



This mode allows navigation of the scene similarly to that of a first person shooter game. The mouse cursor is grabbed by the visualisation window and can no longer be used until the mode is quit, this can be achieved by left clicking the mouse, or by pressing Escape. In this mode the mouse is used to change the view direction and the keyboard is used to move. The keys W/S are used to move forward/backwards, and keys A/D are used to strafe left/right.

2.8. Object Viewer

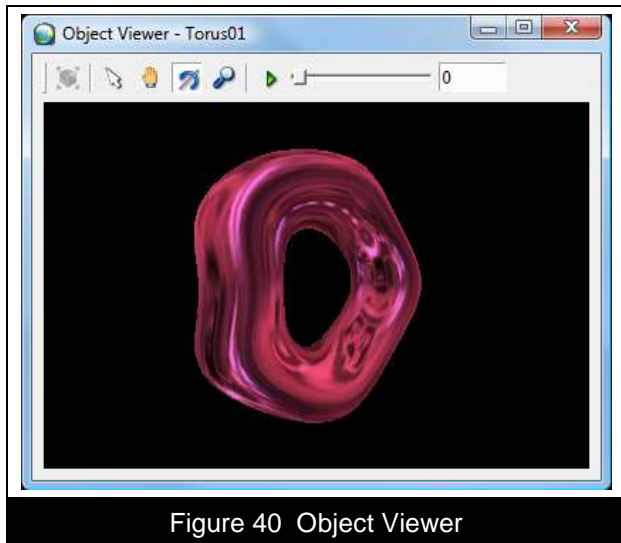


Figure 40 Object Viewer

The Object Viewer is used to view a single object from the scene in a separate window. The pan, rotate and zoom tools can be used to navigate around the object, without affecting the view in the main window.

The Object Viewer is useful if you have a complex scene and only want to view a single object in isolation, or if you want to see an alternative view without effecting the main window.

3. How use PVRShaman

3.1. Adding an effect from the library to a 3D file (OpenGL ES2)

1. Create a 3D file in 3DStudioMAX or Maya and export it as POD format using the OpenGL coordinate system.
2. Open PVRShaman in OpenGL ES2 mode (see section 2.4.5) and open the POD file just created (File -> Open Pod).
3. From the menu select 'Edit Scene' -> 'Add Material...'.- 4. In the Add Material dialog set the material name to 'Cell Shader' (or a name of your choice).
- 5. Select the 'From Effect Library' radio button and select the option 'CellShading' from the Library drop down box.
- 6. Enter the name 'mycellshader.pfx' in the file name box and click 'Accept'.
- 7. The editor will open the newly created file mycellshader.pfx and your material will appear in the 'Scene Container' on the left, under the 'Materials' branch. If the editor is not open you can double click on the PFX file in the 'Scene Container' to open it.
- 8. Right click on a mesh in the 'Scene Browser' and select 'Cell Shader' from the list to apply the material.
- 9. The object will now be displayed with the cell shading effect.
- 10. The debug output panel will show the result of the compilation. If there is an error the object will be displayed un-textured.
- 11. Save the POD file which will now store this information.

3.2. Building a new shader for a 3D file (OpenGL ES2)

1. Create a 3D file in 3DStudioMAX or Maya and export it as POD format.
2. Convert all textures to PVR format and keep them in the same place as the POD file, so PVRShaman can find them.
3. Open PVRShaman and open the POD file just created (File -> Open Pod). This file will be displayed as it was displayed in the application that created it.
4. From the menu select 'Edit Scene' -> 'Add Material...' and select 'Create empty effect file'. Give the new file a name (e.g. mynewshader.pfx) and click 'Accept'.
5. The editor will open the newly created empty PFX file and your material will appear in the 'Scene Container'.
6. Right click on an object in the 'Scene Browser' and apply the material just added. As the PFX file is incomplete, it will not compile and the object will be displayed un-textured. Error messages will be displayed in the output window.
7. Add names to your vertex shader, fragment shader and effect by specifying the name after the NAME keyword in each of the 3 blocks.
8. Link the shaders to the effect by specifying the names of the shader to use after the keywords FRAGMENTSHADER and VERTEXSHADER in the effect block.
9. Add your vertex shader code and fragment shader code to the PFX file between the GLSL_CODE tags.
10. Add any textures required to the [TEXTURES] block in the format 'FILE texname filename.pvr', where texname is the name you want to refer to it as and filename is the name of the pvr file.
11. Define these textures to have a number in the [EFFECT] block using 'TEXTURE 0 texname' where texname is the name you specified in the [TEXTURES] block and 0 is the number.

12. The texture must now be passed to the shader with the line 'UNIFORM myTex TEXTURE0' also in the [EFFECT] block, where myTex is the name of the sampler in the shader code and 0 is the texture number you defined in the previous step.
13. Add any attributes and uniforms required by the shaders to the EFFECT block in the format 'ATTRIBUTE myVertex POSITION', where myVertex is the name used in the shaders. To simplify this step you can right click in the editor and select the attribute or uniform you want to add from the list. The new attribute or uniform will be added on the current line and you must change '_name_' to be the name used in the shader.
14. Click 'Apply Shader' (Edit Scene -> Apply Shader, or hit F5). This will automatically save the shader to disk and update the display. Any error messages will be displayed in the output window.
15. Save the POD file.

3.3. Launching PVRShaman from the command line

PVRShaman can be launched from the command line as follows:

```
PVRShaman [--api=API] [FILE]
```

Where FILE is the name of the POD (or COLLADA) file, and API can be OGLES2, OGLES1, OGL, DX9, or DX10. (DirectX modes are MS Windows only.)

Examples:

```
PVRShaman
PVRShaman file.pod
PVRShaman --api=OGLES1
./PVRShaman --api=OGLES2 ../path/file.pod           (Mac, Linux)
PVRShaman.exe --api=DX10 ../path/file.pod           (MS Windows)
```

4. PFX Overview

A PFX file is text file that contains several shader definitions.

Blocks are marked by an opening and closing directive between square brackets. Standard defined keywords are the first word of a line and parameters will follow in the same line. A PFX file can contain several GLSL shaders and several ways of using them (EFFECTS). Comments are as `//` with the rest of the line ignored.

A more detailed explanation of PFX can be found in the 'POWERVR Effect File' document.

4.1. Keywords & Blocks

[HEADER]

Contains information about the PFX file: VERSION, DESCRIPTION and COPYRIGHT.

[TEXTURES]

Defines texture files to be used with the shaders. Only PVR format is supported. There is the option of specifying parameters for the textures' filtering and wrapping.

[VERTEXSHADER]

GLSL code for the Vertex Shader. It can be loaded as an external file using FILE, or explicitly defined within the PFX file between [GLSL_CODE] and [/GLSL_CODE]. See example in section 4.2.

External binary blocks are supported using BINARYFILE keyword and specifying the external file to load.

[FRAGMENTSHADER]

GLSL code for the FragmentShader. Like the vertex shader, it can be loaded as an external file or explicitly defined within the PFX file. BINARYFILE is also supported.

[EFFECT]

Effect is a material which links together a vertex shader, fragment shader, textures and data from the application.

4.2. PFX Example

```
[HEADER]
    VERSION          01.00.00.00
    DESCRIPTION       Example PFX File.
    COPYRIGHT         Imagination Technologies Ltd
[/HEADER]

[TEXTURES]
    FILE Tex1 tex1.pvr          LINEAR-LINEAR-LINEAR CLAMP-CLAMP-CLAMP
    FILE Grass grass.pvr       LINEAR-LINEAR-LINEAR
[/TEXTURES]

[VERTEXSHADER]
    NAME MyVertexShader

    FILE vs.glsl               // LOAD GLSL AS FILE
[/VERTEXSHADER]

[FRAGMENTSHADER]
    NAME MyFragmentShader

    [GLSL_CODE]               // LOAD GLSL AS CODE
        uniform sampler2D uSampGrass;
        varying mediump vec2 varTex;

        main()
        {
            gl_Color = texture2D(uSampGrass, varTex);
        }
    [/GLSL_CODE]
[/FRAGMENTSHADER]

[EFFECT]
    NAME MyEffect1

    // UNIFORMS
    UNIFORM vLight             LIGHTDIREYE0
    UNIFORM myWorldViewIT      WORLDVIEWIT
    UNIFORM uSampTex           TEXTURE0
    UNIFORM uSampGrass         TEXTURE1
    UNIFORM myColor            MYCOLOR    vec3(0.7,0.3,0.2)        // Default colour value

    // ATTRIBUTES
    ATTRIBUTE myVertex         POSITION
    ATTRIBUTE myNormal         NORMAL
    ATTRIBUTE myUV             UV0

    // SHADERS
    VERTEXSHADER MyVertexShader
    FRAGMENTSHADER MyFragmentShader

    // TEXTURES
    TEXTURE 0 Tex1
    TEXTURE 1 Grass
[/EFFECT]
```

4.3. OpenGL2 Interface

Please refer to the POWERVR SDK for source code to use PFX files in OpenGL ES 2.0. The training course 'IntroducingPFX' is an example to how to use PFX in an application.

5. PVRShamans' Semantics

PFX semantics are keywords that are used in the [EFFECT] block to pass 'application' data into the shaders. PVRShaman sets values the following semantics.

5.1.1. Attributes

POSITION

vec4. Position.

NORMAL

vec3. Normal.

TANGENT

vec3. Tangent.

BINORMAL

vec3. Binormal.

UV[n]

vec2. n-th set of UVs. Example UV0.

BONEINDEX

vec4. Bone Index.

BONEWEIGHT

vec4. Bone Weight.

5.1.2. Uniforms

WORLD

mat4. World matrix.

WORLDI

mat4. World Inverse matrix.

WORLDIT

mat3. World Inverse Transpose matrix.

VIEW

mat4. View matrix.

VIEWI

mat4. View Inverse matrix.

VIEWIT

mat3. View Inverse Transpose matrix.

PROJECTION

mat4. Projection matrix.

PROJECTIONI

mat4. Projection Inverse matrix.

PROJECTIONIT

mat3. Projection Inverse Transpose matrix.

WORLDVIEW

mat4. World-View matrix.

WORLDVIEWI

mat4. World-View Inverse matrix.

WORLDVIEWIT

mat3. World-View Inverse Transpose matrix.

WORLDVIEWPROJECTION

mat4. World-View-Projection matrix.

WORLDVIEWPROJECTIONI

mat4. World-View-Projection Inverse matrix.

WORLDVIEWPROJECTIONIT

mat3. World-View-Projection Inverse Transpose matrix.

VIEWPROJECTION

mat4. View-Projection matrix.

VIEWPROJECTIONI

mat4. View-Projection Inverse matrix.

VIEWPROJECTIONIT

mat3. View-Projection Inverse Transpose matrix.

OBJECT

mat4. Object matrix, without any parent node transformations.

OBJECTI

mat4. Object Inverse matrix, without any parent node transformations.

OBJECTIT

mat3. Object Inverse Transpose matrix, without any parent node transformations.

MATERIALOPACITY

float. Opacity of material.

MATERIALSHININESS

float. Shininess of material.

MATERIALCOLORAMBIENT

vec3. Ambient color of material.

MATERIALCOLORDIFFUSE

vec3. Diffuse color of material

MATERIALCOLORSPECULAR

vec3. Secular color of material.

BONECOUNT

int. Number of bones.

BONEMATRIXARRAY

mat4[]. Array of bone transformation matrices.

LIGHTCOLOR[n]

vec3. Color of light n (RGB). Example LIGHTCOLOR5.

LIGHTPOSMODEL[n]

vec3. Position of light n in model space. Example LIGHTPOSMODEL1.

LIGHTPOSWORLD[n]

vec3. Position of light n in world space. Example LIGHTPOSWORLD1.

LIGHTPOSEYE[n]

vec3. Position of light n in view space. Example LIGHTPOSEYE1.

LIGHTDIRMODEL[n]

vec3. Direction of light n in model space. Example LIGHTDIRMODEL1.

LIGHTDIRWORLD[n]

vec3. Direction of light n in world space. Example LIGHTDIRWORLD1.

LIGHTDIREYE[n]

vec3. Direction of light n in view space. Example LIGHTDIREYE1.

LIGHTATTENUATION[n]

vec3. Attenuation for spot lights (constant, linear, quadratic).

LIGHTFALLOFF[n]

vec2. Falloff for spot lights (angle, exponent).

EYEPOSMODEL

vec3. Eye position in model space.

EYEPOSWORLD

vec3. Eye position in world space.

TEXTURE[n]

sampler2D. Sampler for texture n. Example TEXTURE2.

ANIMATION

float. Contains the objects distance through its animation. Range 0 to 1.

GEOMETRYCOUNTER

Int. Resets to 0 at the beginning of each render frame and increases by one for each submission of geometry.

VIEWPORTPIXELSIZE

vec2. Size of the viewport in pixels

VIEWPORTCLIPPING

vec4. Near distance, far distance, width angle (radians), height angle (radians)

TIME

float. The current time

LASTTIME

float. The last frame's time

ELAPSEDTIME

float. The time between adjacent frames

BOUNDINGCENTER

vec3. Bounding box center

BOUNDINGSPHERERADIUS

Float. Bounding sphere radius

BOUNDINGBOXSIZE

vec3. Bounding box size

BOUNDINGBOXMIN

vec3. Bounding minimum for x, y, z

BOUNDINGBOXMAX

vec3. Bounding maximum for x, y, z

RANDOM

float. A random value (Range 0 to 1)

MOUSEPOSITION

vec3. The mouse position on screen (x, y, time)

LEFTMOUSEDOWN

vec4. The left mouse down state, and its position at that time (x, y, isdown, timedown)

RIGHTMOUSEDOWN

vec4. The right mouse down state, and its position at that time (x, y, isdown, timedown)