

# PVRTrace

## User Manual

Copyright © Imagination Technologies Ltd. All Rights Reserved.

This publication contains proprietary information which is subject to change without notice and is supplied 'as is' without warranty of any kind. Imagination Technologies and the Imagination Technologies logo are trademarks or registered trademarks of Imagination Technologies Limited. All other logos, products, trademarks and registered trademarks are the property of their respective owners.

Filename : PVRTrace.User Manual.1.23.2.3f.External.doc  
Version : 1.23.2.3f External Issue (Package: POWERVR SDK 2.09.29.0646)  
Issue Date : 11 Oct 2011  
Author : Imagination Technologies Ltd

# Contents

- 1. Introduction .....4**
  - 1.1. Software Overview.....4
    - 1.1.1. Recording Libraries .....4
    - 1.1.2. PVRTraceGUI .....4
  - 1.2. Document Overview .....4
  - 1.3. File Types .....4
- 2. Recording Libraries .....5**
  - 2.1. Overview .....5
  - 2.2. Compatibility .....5
    - 2.2.1. API.....5
    - 2.2.2. Supported Extensions .....5
  - 2.3. Installation.....6
    - 2.3.1. Package Installation .....6
    - 2.3.2. Windows.....6
    - 2.3.3. Linux .....6
    - 2.3.4. Android .....7
  - 2.4. Configuration .....8
- 3. PVRTrace GUI.....9**
  - 3.1. Overview.....9
  - 3.2. Installation.....9
    - 3.2.1. From Installer .....9
    - 3.2.2. From GZIP.....9
  - 3.3. Compatibility .....9
  - 3.4. Main Interface .....10
    - 3.4.1. Frame Summary.....10
    - 3.4.2. Frame Function Counts.....11
    - 3.4.3. Data Viewer.....12
    - 3.4.4. Render State Tab.....12
    - 3.4.5. Textures Tab .....13
    - 3.4.6. Shaders Tab.....13
    - 3.4.7. Function Call List.....14
  - 3.5. Toolbars.....15
    - 3.5.1. Data Viewer Toolbar.....15
    - 3.5.2. Function Call List Toolbar.....16
  - 3.6. Dialogs.....17
    - 3.6.1. Find.....17
    - 3.6.2. Filter Functions.....17
  - 3.7. Menus .....18
    - 3.7.1. File.....18
    - 3.7.2. Tools.....18
    - 3.7.3. Help .....18
- 4. Related Materials .....20**
- 5. Contact Details.....21**
- Appendix A. Regular Expression Syntax .....22**

# List of Figures

- Figure 2-1 Recording library overview ..... 5
- Figure 3-1 PVRTrace Main Window ..... 10
- Figure 3-2 Frame Summary ..... 10
- Figure 3-3 Frame Function Counts ..... 11

---

Figure 3-4 Data Viewer .....	12
Figure 3-5 Render State Tab .....	12
Figure 3-6 Textures Tab.....	13
Figure 3-7 Shaders Tab .....	13
Figure 3-8 Function Call List .....	14
Figure 3-9 Zoom In.....	15
Figure 3-10 Zoom Out.....	15
Figure 3-11 Scale To Fit.....	15
Figure 3-12 Actual Size.....	15
Figure 3-13 Vertical Flip.....	15
Figure 3-14 Zoom Level.....	15
Figure 3-15 Show Draw Calls .....	16
Figure 3-16 Toggle Filters.....	16
Figure 3-17 Show Filter Dialog.....	16
Figure 3-18 Find.....	16
Figure 3-19 Find Dialog.....	17
Figure 3-20 Filter Functions .....	17
Figure 3-21 File Menu .....	18
Figure 3-22 Tools Menu .....	18
Figure 3-23 Help Menu .....	18

# 1. Introduction

## 1.1. Software Overview

PVRTrace is a scene recording and analysis utility; it captures all the API calls made by an OpenGL ES application as it is running and records the data for analysis at a later date. It consists of two main components, a recording tool and an analysis tool.

### 1.1.1. Recording Libraries

The recording libraries are shim libraries that are installed on a platform and capture all calls to that platform's native graphics libraries. These calls are captured and written into a .pvrt file for reading back by the analysis GUI.

### 1.1.2. PVRTraceGUI

PVRTraceGUI serves as the analysis interface of PVRTrace allowing 'human-readable' access to the contents of pre-recorded .pvrt files.

## 1.2. Document Overview

The purpose of this document is to serve as a complete user manual for the PVRTrace analysis tool and its associated libraries. It includes compatibility information, installation instructions, a guide to the functionality of the application and a complete listing of all interface options and preferences, broken down by application.

## 1.3. File Types

### PVRT (.pvrt)

A trace of a given application output by PVRTrace recording libraries at runtime.

## 2. Recording Libraries

### 2.1. Overview

Recording of a trace is performed by multiple libraries; shim libraries for each graphics API, and one core recording library. Once installed on a platform, the shim libraries intercept graphics API calls and send them to the recording library to write them to a file. The calls are then passed on to the host libraries, as stated in the 'pvrttrace.cfg' file. Calls are streamed to the .pvt file during runtime so that even if an application crashes, data will still be recorded.

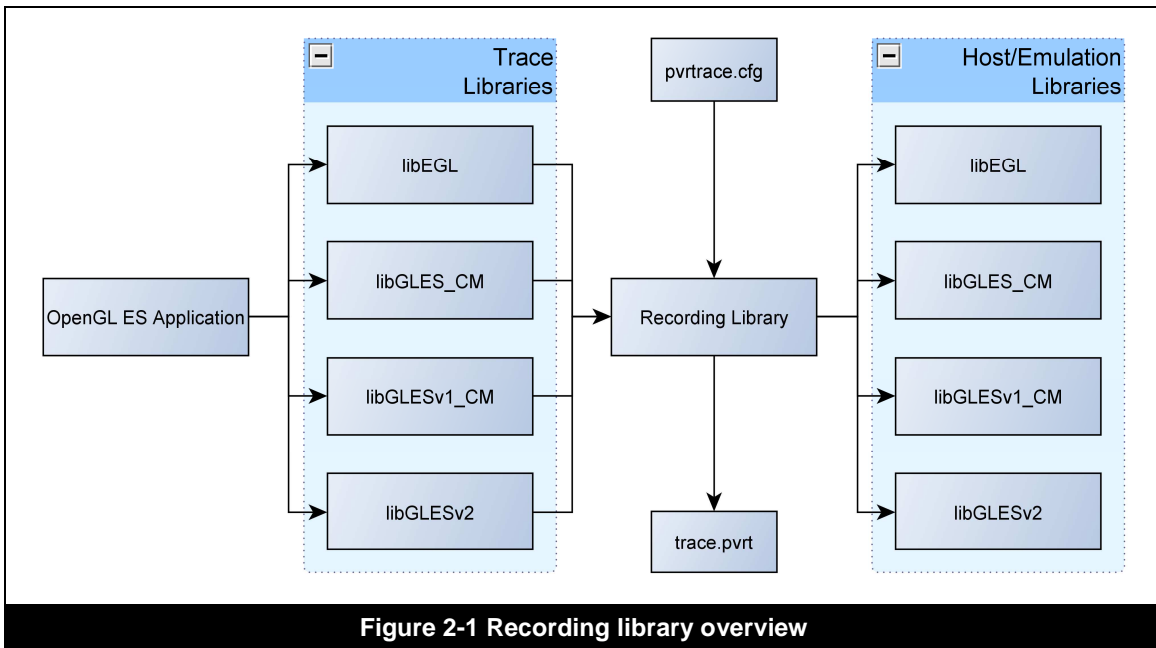


Figure 2-1 Recording library overview

### 2.2. Compatibility

#### 2.2.1. API

The PVRTrace recording libraries are compatible with EGL, OpenGL ES 1.1 and OpenGL ES 2.0.

#### 2.2.2. Supported Extensions

OpenGL ES 1.1	OpenGL ES 2.0
GL_OES_vertex_array_object	GL_OES_vertex_array_object
GL_EXT_multi_draw_arrays	GL_EXT_multi_draw_arrays
GL_OES_mapbuffer	GL_OES_mapbuffer
GL_EXT_discard_framebuffer	GL_EXT_discard_framebuffer
GL_OES_framebuffer_object	
GL_OES_fixed_point	
GL_OES_EGL_image_external	
GL_OES_matrix_palette (GL_OES_extended_matrix_palette)	

## 2.3. Installation

### 2.3.1. Package Installation

#### From Installer

Download either the PowerVR Insider SDK or the individual PVRTrace package and follow the on screen instructions. Once the package has successfully installed the recording libraries can be found within the PVRTrace folder in the install directory as follows:

```
<SDK_ROOT>\Utilities\PVRTrace\Recorder\<API>\<PLATFORM>\
```

#### From GZIP

Download either the PowerVR Insider SDK or the individual PVRTrace package. Unzip the .tar.gz file, and then untar the .tar file. From the ensuing folder browse to:

```
<SDK_ROOT>\Utilities\PVRTrace\Recorder\<API>\<PLATFORM>\
```

This folder will contain the recording libraries.

### 2.3.2. Windows

1. Copy: `\libEGL.dll`, `\libGLESv2.dll` (OpenGL ES 2.0) or `\libGLES_CM.dll` and `\libGLESv1_CM.dll` (OpenGL ES 1.1), and `\PVRTrace.dll`  
To: Executable folder
2. If `\pvrtrace.cfg` does not exist, generate one by running the application to be traced or make one manually as in Section 2.4 Configuration.
3. Place `\pvrtrace.cfg` into the executable folder.
4. Update `\pvrtrace.cfg` so that `EgLibraryPath`, `Es1LibraryPath` and `Es2LibraryPath` are set to the location of the systems graphics libraries, see Section 2.4 Configuration.

### 2.3.3. Linux

1. Copy: `\libEGL.so`, `\libGLESv2.so` (OpenGL ES 2.0) or `\libGLES_CM.so` and `\libGLESv1_CM.so` (OpenGL ES 1.1), and `\libPVRTrace.so`  
To: Any folder
2. Set the `LD_LIBRARY_PATH` to the folder containing the new libraries.
  - a. This can be set globally using `\set LD_LIBRARY_PATH=...` or `\export LD_LIBRARY_PATH=...`, and reverted once the trace is complete.
  - b. This can also be set by using `\LD_LIBRARY_PATH=... ./<APPLICATION_NAME>` as part of running the trace.
3. If `\pvrtrace.cfg` does not exist, generate one by running the application to be traced.
4. Place `\pvrtrace.cfg` into the executable folder.
5. Update `\pvrtrace.cfg` so that `EgLibraryPath`, `Es1LibraryPath` and `Es2LibraryPath` are set to the location of the systems graphics libraries, see Section 2.4 Configuration.

### 2.3.4. Android

1. Copy: `'libPVRTrace.so'`  
To: `/system/lib`
2. Copy: `'libEGL_PVRTRACE.so'`, `'libGLv2_PVRTRACE.so'` (OpenGL ES 2.0) or `'libGLv1_CM_PVRTRACE.so'` (OpenGL ES 1.1).  
To: `/system/lib/egl` or `/system/vendor/lib/egl` (gingerbread)  
Essentially these need to be where the OpenGL ES driver libraries are located.
3. If `'pvrtrace.cfg'` does not exist, generate one by running the application to be traced.
4. Place `'pvrtrace.cfg'` into the root of the file system (`/`) or the SD card (`/mnt/sdcard`). It should be noted that the root of the file system (`/`) will be checked first and used in preference over the root of the SD Card.
5. Update `'pvrtrace.cfg'` so that `EglLibraryPath`, `Es1LibraryPath` and `Es2LibraryPath` are set to the location of the systems graphics libraries, see Section 2.4 Configuration
6. File permissions are tightly regulated on Android, typically you can only write to a location either on an SD Card, or the process folder for your app (`/data/data/<PROCESS_NAME>/`). So the "TraceFile" variable must be set accurately with an absolute path. If it isn't correctly set, no file will be output. It is suggested that you check write permissions for your app before setting this. As all subsequent processes will be traced after installation, it is recommended that you use an output folder that allows write access for all processes so that PVRTrace output doesn't fail and potentially threaten the stability of your device.
7. Edit `'egl.cfg'` in `/system/lib/egl/` to use PVRTrace instead of the current driver. For example if the original file stated:

```
0 0 android
0 1 POWERVR_SGX_540_120
```

Edit it to:

```
0 0 android
0 1 PVRTRACE
```

This change can be done at any time and will only affect new processes that are launched after the change. It is recommended that two copies of `'egl.cfg'` be used, one referencing the PVRTrace libraries, one referencing the default libraries; each file should be named separately and copied over the original `'egl.cfg'` as required.

8. Once a trace is complete the changes to `'egl.cfg'` must be reverted to avoid any undesired behaviour. This is particularly important before rebooting your device as the OpenGL ES libraries are required to draw the UI, and if these are traced it can cause issues at boot time.

**WARNING:** If the `'egl.cfg'` file is inaccurate and points at libraries that do not exist or are in the wrong location and the device is rebooted it will be unable to boot correctly. If this happens the device may need to be reflashed unless external access to the file system is available.

## 2.4. Configuration

The PVRTrace recording libraries require the configuration file 'pvrtrace.cfg' to be present. If this file is not present then one will be created the first time the application being traced is run. An example of a configuration file can be found below:

```
[host]
EglLibraryPath = %SYSTEMDIR%\libEGL.dll
Es1LibraryPath = %SYSTEMDIR%\libGLES_CM.dll
Es2LibraryPath = %SYSTEMDIR%\libGLESv2.dll

[record]
TraceFile = trace-%pid.pvrt
RecordData = 0
StartFrame = 0
EndFrame = 1
```

### EglLibraryPath

'EglLibraryPath' refers to the location of the original (non-PVRTrace) EGL library on the host system (eg. /system/vendor/lib/egl on Android Gingerbread). On Windows, environment variables can be used (such as %SYSTEMDIR%). This functionality is not supported on Linux or Mac OS.

### Es1LibraryPath

'Es1LibraryPath' refers to the location of the original (non-PVRTrace) OpenGL ES 1.1 library on the host system (eg. /system/vendor/lib/egl on Android Gingerbread). On Windows, environment variables can be used (such as %SYSTEMDIR%). This functionality is not supported on Linux or Mac OS.

### Es2LibraryPath

'Es2LibraryPath' refers to the location of the original (non-PVRTrace) OpenGL ES 2.0 library on the host system (eg. /system/vendor/lib/egl on Android Gingerbread). On Windows, environment variables can be used (such as %SYSTEMDIR%). This functionality is not supported on Linux or Mac OS.

### TraceFile

'TraceFile' sets the name of the trace file that will be output by the recording libraries. The process ID of an application can be used as part of its trace file name by using %pid in this field.

### RecordData

This option states whether the trace libraries should record the data associated with each call or not. With 'RecordData' set to '1', texture information and buffer data etc. is recorded along with the graphics api calls. With 'RecordData' set to '0' only the graphics api calls are recorded. In general, 'RecordData' should be set to '1' unless file size is a problem.

### StartFrame

'StartFrame' states the frame at which recording should begin.

### EndFrame

'EndFrame' states the frame at which recording should stop.

## 3. PVRTrace GUI

### 3.1. Overview

PVRTrace GUI is the graphical interface to the PVRTrace analysis tool. It opens .pvr files created with the PVRTrace recording libraries and displays them in a human readable and easy to understand format. It provides a wealth of information on a trace, broken down frame by frame; everything from the number of times a given call occurs in a trace to the exact values of a specific matrix.

### 3.2. Installation

#### 3.2.1. From Installer

Download either the PowerVR Insider SDK or the individual PVRTrace package and follow the on screen instructions. Once the package has successfully installed, the application will be available in:

```
<SDK_ROOT>\Utilities\PVRTrace\Analysis\<PLATFORM>\
```

#### 3.2.2. From GZIP

Download either the PowerVR Insider SDK or the individual PVRTrace package. Unzip the .tar.gz file, and then untar the .tar file. From the ensuing folder browse to:

```
<SDK_ROOT>\Utilities\PVRTrace\Analysis\<PLATFORM>\
```

### 3.3. Compatibility

On Linux and Mac OS X11 is required.

## 3.4. Main Interface

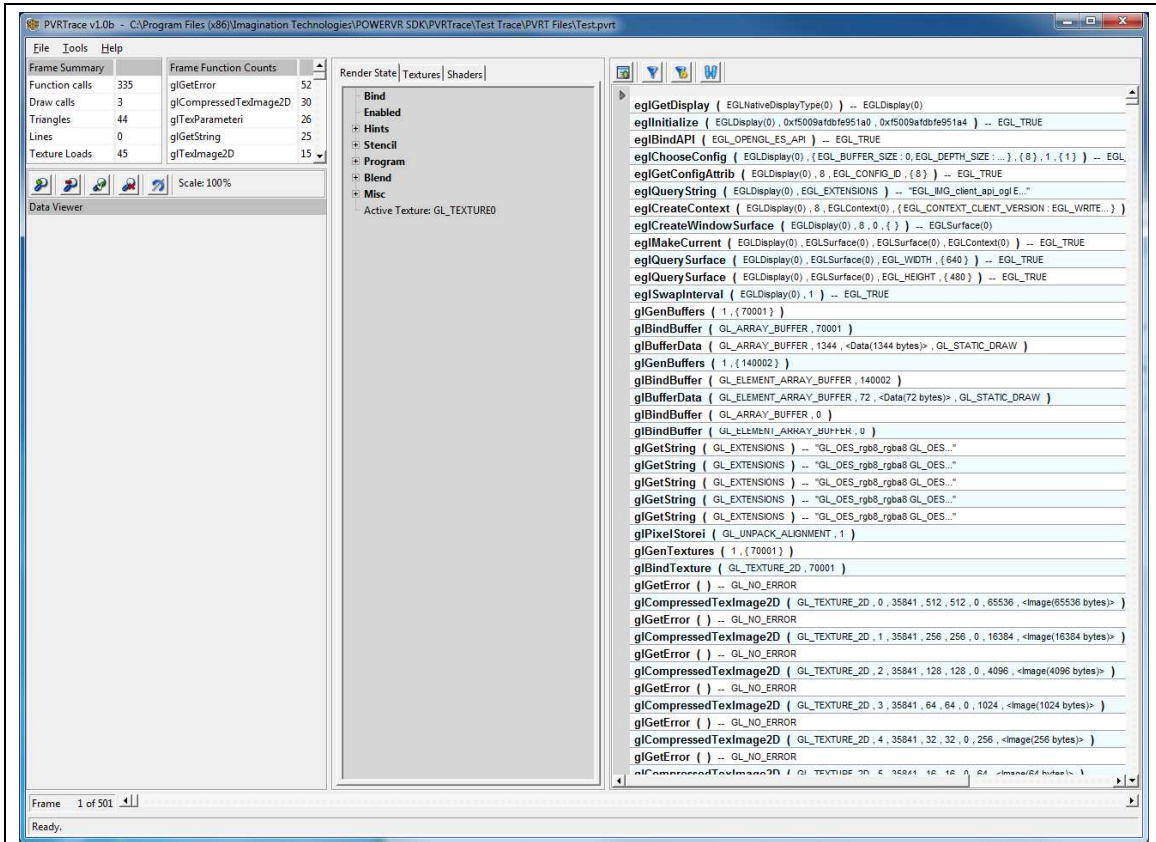


Figure 3-1 PVRTrace Main Window

### 3.4.1. Frame Summary

The Frame Summary window gives an overall summary of the currently selected frame. The following information is displayed:

- total number of function calls
- total number of draw calls
- total number of triangles sent to OpenGL before any HSR/Culling
- total number of lines sent to OpenGL before any HSR/Culling
- total number of texture loads

Frame Summary	
Function calls	335
Draw calls	3
Triangles	44
Lines	0
Texture Loads	45

Figure 3-2 Frame Summary

### 3.4.2. Frame Function Counts

The Frame Function Counts window serves as a list of all the function calls within the currently selected frame and the number of times each function has been called within that frame. Functions can be selected by left clicking; selected functions can be set as 'watched' by right clicking and clicking 'Add Watch (Selected)'. It is also possible to set a single function as 'watched' by right clicking a single function and clicking 'Add Watch'. 'Watched' functions always appear at the top of the window.

Frame Function Counts	
glGetError	52
glCompressedTexImage2D	30
glTexParameterI	26
glGetString	25
glTexImage2D	15

**Figure 3-3 Frame Function Counts**

### 3.4.3. Data Viewer

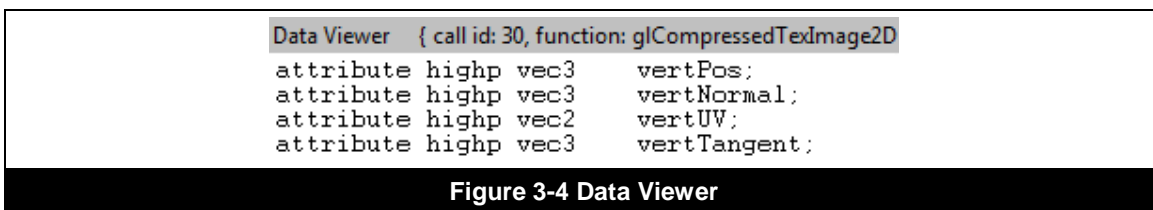


Figure 3-4 Data Viewer

The Data Viewer displays information pertaining to the currently selected function in the Function Call List; the currently selected texture from either the Function Call List or the Textures Tab; or the currently selected shader from either the Function Call List or the Shaders Tab.

### 3.4.4. Render State Tab

The render state tab contains information regarding the current frame's render state. This information changes based upon the position within the current frame selected in the Function Call List.

#### Bind

'Bind' contains a list of the currently bound textures; these textures can be selected and will be displayed in the Data Viewer.

#### Enabled

'Enabled' lists the options that are currently active, having been set through the use of 'glEnable'.

#### Hints

'Hints' lists the options currently set using 'glHint'.

#### Stencil

'Stencil' lists the current settings for the stencil buffer.

#### Misc

This section contains miscellaneous information that does not fit into the various other sections. This covers pixel byte alignment; polygon offset values, and scissoring information along with a variety of other useful information.

#### Program (OpenGL ES 2.0 Only)

'Program' details the settings for the currently running program, including the program's ID, the ID of the vertex and fragment shaders, and what the currently bound attributes and uniforms are. The vertex and fragment shader IDs can be selected and will appear in the Data Viewer.

#### Blend (OpenGL ES 2.0 Only)

'Blend' contains information pertaining to the current blend state, what blending equations and functions are used, the current clear depth and colour mask etc.

#### Clip Planes (OpenGL ES 1.1 Only)

'Clip Planes' details information of the currently set OpenGL clip planes as set with glClipPlane.

#### Lights (OpenGL ES 1.1 Only)

This menu lists all information pertaining to all the available lights within a scene, its specific colours, attenuations etc.

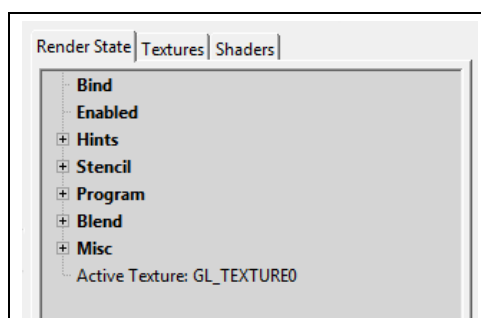


Figure 3-5 Render State Tab

**Tex Env (OpenGL ES 1.1 Only)**

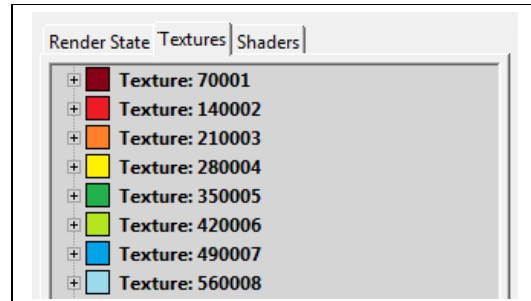
'Tex Env' lists all the information pertaining to each bound texture within the current frame.

**3.4.5. Textures Tab**

The Textures tab lists all the textures that are currently loaded. Each texture has a thumbnail and a unique ID, and contains its type, dimension, format (e.g. GL\_RGBA), and storage type (GL\_UNSIGNED\_SHORT\_4\_4\_4\_4). Finally, it is possible to right click on a texture; this will open a menu with the options 'Show Loading Call' and 'Show Binding Call'.

'Show Loading Call' will move the Function Call List to the function call that loads that given texture.

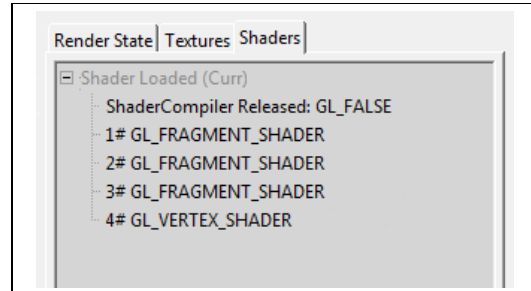
'Show Binding Call' will move the Function Call List to the next instance of glBindTexture that binds that specific texture.



**Figure 3-6 Textures Tab**

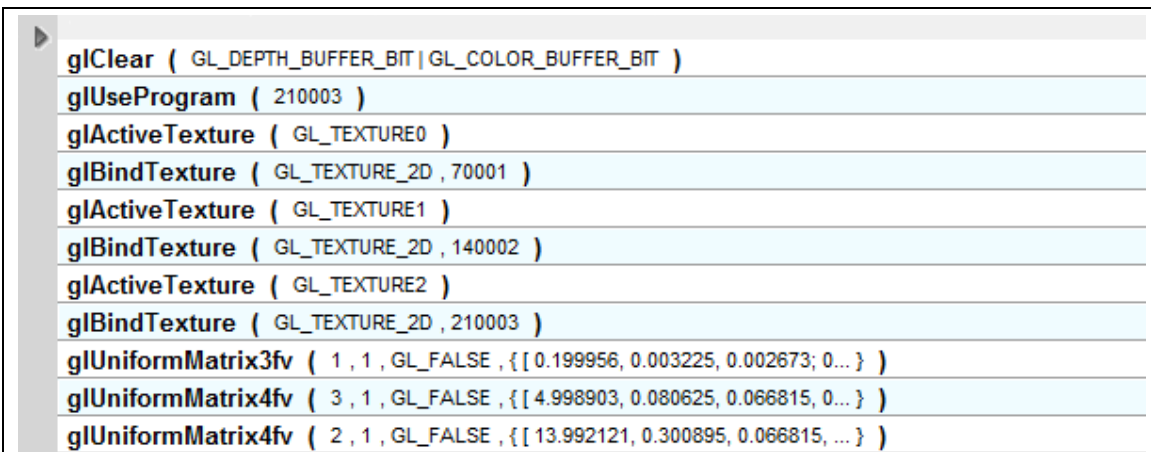
**3.4.6. Shaders Tab**

The Shaders tab lists all the currently loaded shaders. Each shader is given a unique identifier, and if clicked on will open in the Data Viewer.



**Figure 3-7 Shaders Tab**

### 3.4.7. Function Call List



```

▶ glClear ( GL_DEPTH_BUFFER_BIT | GL_COLOR_BUFFER_BIT )
glUseProgram ( 210003 )
glActiveTexture ( GL_TEXTURE0 )
glBindTexture ( GL_TEXTURE_2D , 70001 )
glActiveTexture ( GL_TEXTURE1 )
glBindTexture ( GL_TEXTURE_2D , 140002 )
glActiveTexture ( GL_TEXTURE2 )
glBindTexture ( GL_TEXTURE_2D , 210003 )
glUniformMatrix3fv ( 1 , 1 , GL_FALSE , { [ 0.199956, 0.003225, 0.002673; 0... ] } )
glUniformMatrix4fv ( 3 , 1 , GL_FALSE , { [ 4.998903, 0.080625, 0.066815, 0... ] } )
glUniformMatrix4fv ( 2 , 1 , GL_FALSE , { [ 13.992121, 0.300895, 0.066815, ... ] } )

```

Figure 3-8 Function Call List

The Function Call List shows all of the functions that have been recorded. Each function can be selected; this action adjusts the Render State Tab so that its information matches the render state at the position in the trace marked by the selected function. It is also possible to select shaders and textures from the parameters of a function call within this view; these textures and shaders will appear in the Data Viewer. In addition function parameters can be hovered over; doing this will produce a tooltip which will identify both the contents of the parameter and what the parameter represents (e.g. stride/size/pointer etc.). Finally, function calls can be right clicked; this opens a menu with four available options: 'Add Filter', 'Remove Filters', 'Add/Remove Highlights' and 'Function Quick Help'.

#### Add Filter

The 'Add Filter' option filters out the currently selected function call from the Function Call List for all frames. This filter can be removed using the Filter Functions dialog, or with the Remove Filters option.

#### Remove Filters

'Remove Filters' removes all filters from the Function Call List that are currently set.

#### Add/Remove Highlights

This option adds/removes highlighting of the selected function from the Function Call List.

#### Function Quick Help

'Function Quick Help' opens the documentation for the selected function at <http://www.khronos.org>.

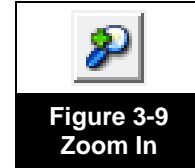
### 3.5. Toolbars

#### 3.5.1. Data Viewer Toolbar

The Data Viewer Toolbar appears above the Data Viewer window when a texture is being displayed. It consists of six items:

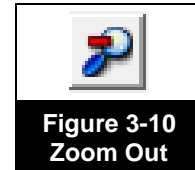
**Zoom In**

Zoom in the currently selected texture.



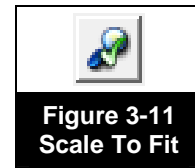
**Zoom Out**

Zoom out of the currently selected texture.



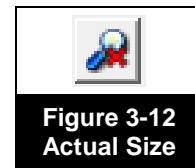
**Scale to Fit**

Scale the currently selected texture to fit the size of the Data Viewer.



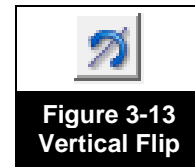
**Actual Size**

Scale the currently selected texture to its actual size.



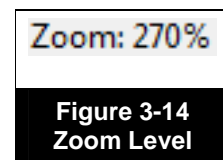
**Vertical Flip**

Flip the currently selected texture vertically.



**Zoom Level**

This option displays the current level of zoom.



### 3.5.2. Function Call List Toolbar

The Function Call List toolbar is a small toolbar that appears above the Function Call List. It consists of four items:

#### Show Draw Calls

'Show Draw Calls' toggles whether to show all functions in the current frame or only the draw calls.



**Figure 3-15**  
**Show Draw**  
**Calls**

#### Toggle Filters

'Toggle Filters' toggles on or off filters set with either the Filter Functions or 'Add Filter' right click option from within the Function Call List.



**Figure 3-16**  
**Toggle Filters**

#### Show Filter Dialog

This option opens the Filter Functions dialog.



**Figure 3-17**  
**Show Filter**  
**Dialog**

#### Find

'Find' opens the Find dialog.



**Figure 3-18** Find

### 3.6. Dialogs

#### 3.6.1. Find

The Find dialog searches the current frame for function calls containing the text entered in the 'Find' box.

'Whole Word' indicates whether the search should only display matches with whole words, or whether the search matches should flag up when the given string is part of a word.

'Match Case' states that the search must match the case of the given string precisely.

'Expression' indicates that the search term is in the form of a regular expression. See Appendix A. Regular Expression Syntax.

'Whole Trace' informs Find that you wish the whole trace file to be searched rather than just the current frame.



Figure 3-19 Find Dialog

#### 3.6.2. Filter Functions

The Filter Functions dialog allows for specific OpenGL ES and EGL functions to be filtered in or out of the trace. A tick next to a given item indicates that the item (and any sub items) will be displayed in the Function Call List; items (and any sub items) without ticks are not displayed. A greyed out box shows that the item does not exist within the current trace (e.g. In an OpenGL ES 2.0 trace, the OpenGL ES 1.1 item, and all its sub items, will appear greyed out).

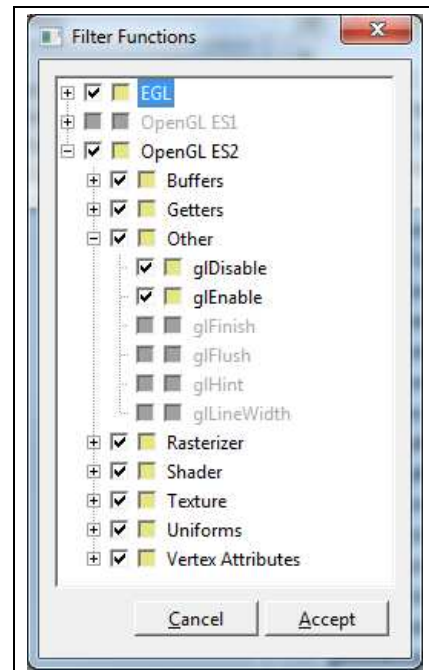


Figure 3-20 Filter Functions

## 3.7. Menus

### 3.7.1. File

#### Open

'Open...' opens an 'open file' dialog; from here a .pvr file can be selected and opened.

#### Open Recent

'Open Recent' contains a list of the most recently opened files; these list items can be selected and opened.

#### Save

'Save -> All Function Calls' saves the contents of the Function Call List, broken down frame by frame, to a .txt file.

'Save -> Frames Summary' saves the contents of Frame Summary for each frame, as a .csv file.

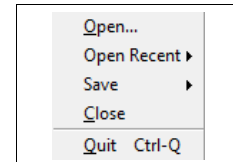
'Save -> Calls Summary' saves the contents of Frame Function Counts for each frame as a .csv file.

#### Close

'Close' closes the current trace.

#### Quit

'Quit' quits the application.



**Figure 3-21 File Menu**

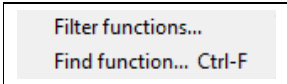
### 3.7.2. Tools

#### Filter Functions

'Filter Functions...' opens the Filter Functions dialog.

#### Find Function

'Find function...' opens the Find dialog.



**Figure 3-22 Tools Menu**

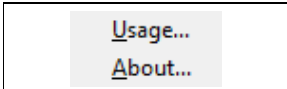
### 3.7.3. Help

#### Usage

'Usage...' opens this document.

#### About

'About...' opens an about page containing version information, contact details etc.



**Figure 3-23 Help Menu**



## 4. Related Materials

### Software

- PVRTune

### Documentation

- PVRTune User Manual

## 5. Contact Details

For further support contact:

[devtech@imgtec.com](mailto:devtech@imgtec.com)

PowerVR Developer Technology  
Imagination Technologies Ltd.  
Home Park Estate  
Kings Langley  
Herts, WD4 8LZ  
United Kingdom

Tel: +44 (0) 1923 260511

Fax: +44 (0) 1923 277463

Alternatively, you can use the PowerVR Insider forums:

[www.imgtec.com/forum](http://www.imgtec.com/forum)

For more information about PowerVR or Imagination Technologies Ltd. visit our web pages at:

[www.imgtec.com](http://www.imgtec.com)

## Appendix A. Regular Expression Syntax

<b>Special Constructs</b>	
(?i X )	Match sub pattern, case insensitive
(?I X )	Match sub pattern, case sensitive
(?n X )	Match sub pattern with newlines
(?N X )	Match sub pattern with no newlines
( X )	Capturing parentheses (use with back references, see below)
(?: X )	Non-capturing parentheses
(?= X )	Zero width positive look ahead
(?! X )	Zero width negative look ahead
(?<= X )	Zero width positive look behind
(?<! X )	Zero width negative look behind
(?> X )	Atomic grouping (possessive match)
<b>Logical Operators</b>	
X Y	X followed by Y
X   Y	Either X or Y
<b>Quantifiers</b>	
X *	Match 0 or more
X +	Match 1 or more
X ?	Match 0 or 1
X { }	Match 0 or more
X {n }	Match n times
X { ,m }	Match no more than m times
X {n, }	Match n or more
X {n,m }	Match at least n but no more than m times
These quantifiers are greedy. By following them with ‘?’ you can turn them into lazy quantifiers, or follow them by ‘+’ for possessive (non-backtracking) quantifiers.	
<b>Boundary Matching</b>	
^	Match begin of line [if at begin of pattern]
\$	Match end of line [if at end of pattern]
\<	Begin of word
\>	End of word
\b	Word boundary
\B	Word interior
\A	Match only beginning of file
\Z	Match only end of file

<b>Character Classes</b>	
[ abc ]	Match a, b, or c
[ ^abc ]	Match any but a, b, or c
[ a-zA-Z ]	Match upper- or lower-case a through z
[ ] ]	Matches ]
[ - ]	Matches -
<b>Predefined Character Classes</b>	
.	Match any character
\d	Digit [0-9]
\D	Non-digit
\s	Space
\S	Non-space
\w	Word character [a-zA-Z_0-9]
\W	Non-word character
\l	Letter [a-zA-Z]
\L	Non-letter
\h	Hex digit [0-9a-fA-F]
\H	Non-hex digit
\u	Single uppercase character
\U	Single lowercase character
\p	Punctuation (not including '_')
\P	Non punctuation
<b>Characters</b>	
\\	Back slash character
\033	Octal
\x1b	Hex
\t	Tab
\n	Newline
<b>Back References</b>	
\1 to \9	Reference to 1 <sup>st</sup> to 9 <sup>th</sup> capturing group

Imagination Technologies, the Imagination Technologies logo, AMA, Codescape, Enigma, IMGworks, I2P, PowerVR, PURE, PURE Digital, MeOS, Meta, MBX, MTX, PDP, SGX, UCC, USSE, VXD and VXE are trademarks or registered trademarks of Imagination Technologies Limited. All other logos, products, trademarks and registered trademarks are the property of their respective owners.