GL2PS: an OpenGL to PostScript printing library

Christophe Geuzaine

Version 0.5, 19 November 2001

Contents

1 Introduction

GL2PS is a library for creating PostScript output from any OpenGL application. Though it was primarily designed for three-dimensional geometry, mesh and post-processing visualization, it may be useful every time high quality vector output is desired. The main difference between GL2PS and other similar libraries is the use of sorting algorithms capable of handling intersecting and stretched polygons, as well as non manifold objects.

The library, written in C, is released under GNU Library General Public License (see `http://www.gnu.org/` for more details), and is available at `http://www.geuz.org/gl2ps/`. Any corrections, questions or suggestions should be e-mailed to `Christophe.Geuzaine@advalvas.be`.

The interface consists of seven functions, all beginning with the prefix `gl2ps`. All the data structures and the symbolic constants peculiar to GL2PS begin with `GL2PS`.

2 Usage

2.1 gl2psBeginPage and gl2psEndPage

2.1.1 Specification

```
void gl2psBeginPage( char *title, char *producer, GLint format,
                     GLint sort, GLint options, GLint colormode,
                     GLint colorsize, GL2PSrgba *colortable,
                     GLint buffersize,
                     FILE *stream, char *filename )

void gl2psEndPage( void )
```

2.1.2 Description

`gl2psBeginPage` and `gl2psEndPage` delimit the OpenGL commands that will be caught in the feedback buffer and output to `stream`. The parameters given to `gl2psBeginPage` determine the way primitives are handled:

`title` Specifies the plot title. For PostScript output, this string is placed in the `%%Title` field.

`producer` Specifies the plot producer. For PostScript output, this string is placed in the `%%For` field.

`format` Specifies the output format, chosen among:

- `GL2PS_PS` The output stream will be a PostScript file.
- `GL2PS_TEX` The output stream will be a LaTeX file, containing only the text strings of the plot (cf. section 2.2), as well as an `\includegraphics` command permitting to include a graphic file having the same basename as `filename`.[1]

`sort` Specifies the sorting algorithm, chosen among:

- `GL2PS_NO_SORT` The primitives are not sorted, and are output in `stream` in the order they appear in the feedback buffer.
- `GL2PS_SIMPLE_SORT` The primitives are sorted according to their barycenter. This can be sufficient for simple scenes.

[1] The two steps to generate a LaTeX plot with GL2PS are thus:

1. generate the PostScript file (e.g. `file.ps`) with no text strings, using the `GL2PS_PS` format combined with the `GL2PS_NO_TEXT` option;
2. generate the LaTeX file `file.tex`, using the `GL2PS_TEX` format and specifying `file.tex` as the `filename` argument to `gl2psBeginPage`.

You can of course combine the LaTeX output with other graphic formats than PostScript. For example, you may transform `file.ps` into `file.pdf` and use pdfLaTeX with the same `file.tex` as for PostScript. You may also use a bitmap image (e.g. Jpeg or png) and still combine it with `file.tex`.

`GL2PS_BSP_SORT` The primitives are inserted in a BSP tree. The tree is then traversed back to front in a painter-like algorithm. This should be used for complex three-dimensional scenes, but keep in mind that the BSP tree algorithm is quite memory hungry...

`options` Sets global plot options, chosen among (multiple options can be combined with the bitwise inclusive OR symbol `|`):

`GL2PS_NONE` No option.

`GL2PS_DRAW_BACKGROUND` The background frame is drawn.

`GL2PS_SIMPLE_LINE_OFFSET` A small offset is added in the z-buffer to all the lines in the plot. This is a simplified version of the `GL2PS_POLYGON_OFFSET_FILL` functionality (cf. section 2.3), putting all the lines of the rendered image slightly in front of their actual position. This thus performs a simple anti-aliasing solution, e.g. for finite element like meshes.

`GL2PS_SILENT` All the messages written by GL2PS on the error stream are suppressed.

`GL2PS_BEST_ROOT` The construction of the BSP tree is optimized by choosing the root primitives leading to the minimum number of splits. This is (really) not efficient yet.

`GL2PS_NO_TEXT` All the text strings are suppressed from output. This is useful to produce the image part of a LaTeX output.

`colormode` Specifies the color mode: `GL_RGBA` or `GL_COLOR_INDEX`.

`colorsize` Specifies the size of the colormap if `colormode` is `GL_COLOR_INDEX`.

`colortable` Contains the colormap if `colormode` is `GL_COLOR_INDEX`. This colormap must contain `colorsize` elements of type `GL2PSrgba`.

`buffersize` Specifies the size of the feedback buffer.

`stream` Specifies the stream to which data is printed.

`filename` Specifies a name for the stream to which data is printed.

2.2 gl2psText

2.2.1 Specification

```
void gl2psText( char *string, char *fontname, GLint fontsize )
```

2.2.2 Description

`gl2psText` permits to include text in the PostScript or LaTeX output in a very simple way. The text is inserted at the current raster position (set by one of the `glRasterPos` OpenGL commands). Beware that text will be sorted according to the position of the leftmost element of the string only. The parameters are:

`string` Specifies the text string to print.

`fontname` Specifies the name of a valid PostScript font (for example `"Times"` or `"HelveticaBoldItalic"`). This parameter has no influence for LaTeX output.

`fontsize` Specifies the size of the font. This parameter has no influence for LaTeX output.

2.3 `gl2psEnable` and `gl2psDisable`

2.3.1 Specification

```
void gl2psEnable( GLint mode )

void gl2psDisable( GLint mode )
```

2.3.2 Description

`gl2psEnable` and `gl2psDisable` delimit OpenGL commands to which a local `mode` is applied. These modes are:

`GL2PS_POLYGON_OFFSET_FILL` Tries to emulate the `GL_POLYGON_OFFSET_FILL` functionality. The value of the offset is taken as the current value of the corresponding OpenGL offset (set with `glPolygonOffset`). Not fully functional yet.

`GL2PS_POLYGON_BOUNDARY` Not implemented yet.

`GL2PS_LINE_STIPPLE` Tries to emulate the `GL_LINE_STIPPLE` functionality.

2.4 `gl2psPointSize` and `gl2psLineWidth`

2.4.1 Specification

```
void gl2psPointSize( GLfloat value )

void gl2psLineWidth( GLfloat value )
```

2.4.2 Description

`gl2psPointSize` and `gl2psLineSize` emulate the standard `glPointSize` and the `glLineWidth` functions. They are necessary since the point sizes and line widths are not saved in the OpenGL feedback buffer.

3 Example

Here is a typical calling sequence to produce BSP sorted PostScript output in the file `"MyFile"`, with all the lines slightly shifted front in the z-buffer. The `draw()` function contains all the OpenGL commands.

```
FILE *fp = fopen("MyFile", "w");
int buffsize = 0, state = GL2PS_OVERFLOW;

while( state == GL2PS_OVERFLOW ){
  buffsize += 1024*1024;
  gl2psBeginPage ( "MyTitle", "MySoftware",
                   GL2PS_PS, GL2PS_BSP_SORT,
                   GL2PS_SIMPLE_LINE_OFFSET | GL2PS_SILENT,
                   GL_RGBA, 0, NULL, buffsize, fp, NULL );
  draw();
  state = gl2psEndPage();
}

fclose(fp);
```

To output the text `"MyText"` at the current raster position, the `draw()` function should contain something like:

```
gl2psText("MyText", "Courier", 12);
```

4 Contributors

Michael Sweet (`mike@easysw.com`) for the original implementation of the feedback buffer parser; Marc Umé (`marc.ume@digitalgraphics.be`) for the original list code; Jean-François Remacle (`remacle@scorec.rpi.edu`) for plane equation fixes; Bart Kaptein (`B.L.Kaptein@lumc.nl`) for memory leak fixes; Quy Nguyen-Dai (`quy@vnilux.com`) for output file size optimization; Sam Buss (`sbuss@ucsd.edu`) for the new smooth shaded triangle code.

Projects similar to `GL2PS` include: Michael Sweet's GLP library (`http://dns.easysw.com/~mike/opengl/index.html`); Mark J. Kilgard's rendereps (`http://reality.sgi.com/opengl/tips/Feedback.html`); the GLpr library from CEI international (`http://www.ceintl.com/`).

5 Versions

0.1 (Feb 12, 2000) First distributed version.

0.2 (Feb 20, 2000) Added `GL2PS_POLYGON_BOUNDARY` and `GL2PS_BEST_ROOT`. Changed arguments of `gl2psBeginPage` and `gl2psText`. Corrected some memory allocation stuff. First version of this user's guide.

0.21 (Mar 16, 2000) Initialization fixes.

0.3 (Jul 29, 2000) Code cleaning. Added `GL2PS_LINE_STIPPLE`.

0.31 (Aug 14, 2000) Better handling of erroneous primitives.

0.32 (May 23, 2001) Fixed memory leaks.

0.4 (Jun 12, 2001) Added `gl2psPointSize` and `gl2psLineWidth`. Some code cleaning to allow easier generation of vector file formats other than postscript.

0.41 (Aug 6, 2001) Fixed string allocation (1 char too short). Set smaller default line width.

0.42 (Oct 8, 2001) Optimization of output file size. PostScript header cleaning. Better line width computation.

0.5 (Nov 19, 2001) New `format` and `filename` arguments for `gl2psBeginPage`. Better PostScript handling of smooth shaded primitives. Fix handling of zero-length strings. New options for LaTeX output. Changed (again) the line width computation.