# Contents

* [1. Introduction](http://titan.cs.ukzn.ac.za/opengl/opengl-d5/trant.sgi.com/opengl/tutorials/siggraph_crsenotes/html/brianp/hardcopy.htm#Intro)
* [2. Bitmap-based Output](http://titan.cs.ukzn.ac.za/opengl/opengl-d5/trant.sgi.com/opengl/tutorials/siggraph_crsenotes/html/brianp/hardcopy.htm#Bitmap)
* [3. Vector-based Output](http://titan.cs.ukzn.ac.za/opengl/opengl-d5/trant.sgi.com/opengl/tutorials/siggraph_crsenotes/html/brianp/hardcopy.htm#Vector)
* [4. Microsoft Windows OpenGL Printing](http://titan.cs.ukzn.ac.za/opengl/opengl-d5/trant.sgi.com/opengl/tutorials/siggraph_crsenotes/html/brianp/hardcopy.htm#Windows)

**1. Introduction**

OpenGL was designed for realtime 3-D raster graphics, which is very different from 2-D printed copy. Nevertheless, many OpenGL applications need hardcopy output. There are basically two approaches:

1. raster/bitmap-based
2. vector-based

The following two sections describe the raster and vector approaches. Microsoft OpenGL users may elect to use the built-in printing support described in the last section.

**2. Bitmap-based Output**

A simple solution to OpenGL hardcopy is to simply save the window image to an image file, convert the file to Postscript, and print it. Unfortunately, this usually gives poor results. The problem is that a typical printer has much higher resolution than a CRT and therefore needs higher resolution input to produce an image of reasonable size and fidelity.

For example, a raster image of size 1200 by 1200 pixels would more than fill the typical 20-inch CRT but only result in a printed image of only 4 by 4 inches if printed at 300 dpi.

To print an 10 by 8-inch image at 300 dpi would require a raster image of 3000 by 2400 pixels. This is a situation in which off-screen, tiled rendering is useful. For more information see [OpenGL/Mesa Offscreen Rendering](http://titan.cs.ukzn.ac.za/opengl/opengl-d5/trant.sgi.com/opengl/tutorials/siggraph_crsenotes/html/brianp/offscrn.htm) and [TR](http://titan.cs.ukzn.ac.za/opengl/opengl-d5/trant.sgi.com/opengl/tutorials/siggraph_crsenotes/html/brianp/tr.html), a tile rendering utility library for OpenGL.

Once you have a raster image in memory it needs to be written to a file. If printing is the only intended purpose for the image than directly writing an Encapsulated Postscript file is best.

Mark Kilgard's book Programming OpenGL for the X Window System contains code for generating Encapsulated Postscript files. The source code may be downloaded from <ftp://ftp.sgi.com/pub/opengl/opengl_for_x/xlib.tar.Z>.

**3. Vector-based Output**

In general, high quality vector-style hardcopy is difficult to produce for arbitrary OpenGL renderings. The problem is OpenGL may generate arbitrarily complex raster images which have no equivalent vector representation. For example, how are smooth shading and texture mapping to be converted to vector form?

Getting the highest quality vector output is application dependant. That is, the application should probably generate vector output by examining its scene data structures.

If a more general solution is desired there are at least two utilities which may help:

[GLP](http://dns.easysw.com/~mike/glp/) (http://dns.easysw.com/~mike/glp/) is a C++ class library which uses OpenGL's feedback mechanism to generate Postscript output. GLP is distributed with a GNU copyright.

[GLPrint](http://www.ceintl.com/products/GLPrint/) (http://www.ceintl.com/products/GLPrint/) from Computational Engineering International, Inc. is a utility library OpenGL printing. The product is currently in beta release.

**4. Microsoft Windows OpenGL Printing**

Microsoft's OpenGL support printing of OpenGL images via metafiles. The basic steps are:

1. Call StartDoc to associate a print job to your HDC handle
2. Call StartPage to setup the document
3. Create a rendering context with wglCreateContext
4. Bind the context with wglMakeCurrent
5. Do your OpenGL rendering
6. Unbind the context with wglMakeCurrent(NULL, NULL)
7. Call EndPage to finish the document
8. Call EndDoc to finish the print job

This procedure is raster-based and may require much memory. To circumvent this problem, printing is done in bands. This however takes more time.