Nvidia OpenGL Configuration mini−HOWTO

Robert B Easter

reaster@comptechnews.com

Revision History

Revision v1.7	2001−01−21	Revised by: rbe

This miniHOWTO is about how to install the OpenGL drivers for Nvidia graphics cards on Linux. In addition to just installing the Nvidia drivers, this mini−HOWTO also explains how to install XFree86, the OpenGL Utility library (part of Mesa), the OpenGL Utility Toolkit (glut), the full set of OpenGL manpages, Qt and its OpenGL extension, and Java and its Java 3D extension so that a user can have a complete runtime and development environment for OpenGL applications on Linux.

Table of Contents

1. Introduction

This miniHOWTO is about how to install the OpenGL drivers for Nvidia graphics cards on Linux. In addition to just installing the Nvidia drivers, this mini−HOWTO also explains how to install XFree86, the OpenGL Utility library (part of Mesa), the OpenGL Utility Toolkit (glut), the full set of OpenGL manpages, Qt and its OpenGL extension, and Java and its Java 3D extension so that a user can have a complete runtime and development environment for OpenGL applications on Linux.

1.1. New Versions of this Document

The lastest version of this mini−HOWTO can be found at:

http://www.linuxdoc.org/HOWTO/mini/Nvidia−OpenGL−Configuration−mini−HOWTO/

1.2. Copyright Information

If you have any questions, please contact <linux−howto@metalab.unc.edu>

1.3. Disclaimer

2. Download the software packages

2.1. Linux Kernel >= 2.2.12 Required

First of all, the OpenGL drivers for the Nvidia cards currently require a system with Linux kernel 2.2.12 or later. If you don't have it, then you will have to upgrade your system's Linux kernel and that is the topic of other HOWTOs! But nevertheless, the first step would be do download a new kernel at:

http://www.kernel.org/

2.2. XFree86 4.0 or later

XFree86 4.0 or later, preferably 4.0.2 as of this writing, is also required. Its installation will be covered later.

Software packages.

XFree86 4.0.2.

XFree86 comes in three files:

- X402src−1.tgz
- X402src−2.tgz
- X402src−3.tgz
- doctools−1.2.tgz

Doctools is something it uses to prepare the X documentation.

ftp://ftp.xfree86.org/

Includes glX, the OpenGL X interface functions. Each window system (MS Windows, X etc) has to provide the platform−specific interfaces between OpenGL and the window system to provide for mapping a GL rendering context to a window system window.

2.3. OpenGL man pages

XFree86 comes with only the glX man pages. If you want a full set of OpenGL man pages, you have to get them yourself.

- mangl.tar.Z
- manglu.tar.Z
- manglx.tar.Z (don't need this one)

ftp://ftp.sgi.com/sgi/opengl/doc/

These man pages are in a format ready to be unpacked into the XFree86 source distributions, see below.

2.4. Mesa

- MesaLib−3.4.tar.gz
- MesaDemos−3.4.tar.gz

http://mesa3d.sourceforge.net/

Mesa provides the foundation for the 3d support included in XFree86, which comes only with the libGL OpenGL core library component. To enable hardware accelerated performance, many 3d hardware manufacturers provide a drop−in replacement for Mesa's standard software−only libGL rendering library. For this document, we are only interested in NVIDIA's libGL drop−in replacement library.

MesaLib provides a software OpenGL implementation consisting of libGL. It also provides libGLU (GL Utility). libGLU is a library built on top of libGL to provide some higher−level functions for applications. OpenGL itself, libGL, is considered a low−level library. GLU is a standard part of most OpenGL installations and many programs make use of it.

MesaDemos provides many OpenGL demo programs and, more importantly, the GL Utility Toolkit (libglut) library. GLUT provides a window system independent interface between OpenGL and any supported window system. For instance, on the X Window System, it hides the details of using glX functions to setup a window. Programmers can write code once and can compile it to work on MS Windows or X, etc provided that a GLUT library is available on the target platform. Like libGLU, libglut is a standard part of most OpenGL installations and is required by many programs even though it is not packaged with XFree86.

While GLUT is bundled with MesaDemos, it is also available as a separate package from its original project website:

glut−3.7.tar.gz glut_data−3.7.tar.gz

http://reality.sgi.com/mjk/glut3/

You may use *either* the GLUT included with MesaDemos (preferred and easier) or the GLUT from its project website. Don't install both! Its recommended at this time to go with the GLUT packaged with MesaDemos, but instructions on how to install the other GLUT are still provided in the next section as an option. Note that MesaDemos does not include the glut manpages, so you may want to download the project GLUT package just to install its manpages.

2.5. Qt

qt−x11−2.2.3.tar.gz or later version

http://www.troll.no/

Qt is a cross−platform GUI library that makes it easy to create X applications with standard GUI elements (widgets) like menubars, scollbars, dropdown lists, checkboxes, buttons, multiple document interface, and many other GUI things. Using Qt, a program can be compiled for both MS Windows and X without changing any code. Its a very popular GUI library and is used to create the core libraries of KDE (http://www.kde.org/).

Qt has functions (previously as an extension in $QTDIR/extensions/opengl) for OpenGL that provides for creating OpenGL rendering contexts in Qt windows. This provides some alternative to both GLUT and using the glX functions directly, plus the added benefit of full access to the excellent qt widgets and cross−platform portability.

This is useful if you want to compile or develop programs based on Qt (e.g., KDE2 and its apps).

2.6. NVIDIA drivers (Mesa libGL replacement)

- NVIDIA_kernel−0.9−6.tar.gz
- NVIDIA_GLX−0.9−6.tar.gz

Note that XFree86 4.0.1, and later, is required with 0.9−6. If you have XFree86 4.0.0, then you'll have to download the older 0.9−4 version.

http://www.nvidia.com/

> **Tip:** See the updated faq at Nvidia.com while you are downloading. It may have some important information not in this HOWTO.

These make a kernel driver: /lib/modules/2.2.16/video/NVdriver and libGL.so and libGLcore.so files that go into /usr/lib/ to replace and Mesa ones that might be in there. libGL.so is OpenGL. These files are Nvidia's own hardware accelerated OpenGL implementation.

2.7. Java 2 SDK, Java 3D extension, and Java PlugIn for Netscape (optional)

The following files are available at http://www.blackdown.org/:

- j2sdk−1.2.2−FCS−linux−i386−glibc−2.1.3.tar.bz2
- java3d1_2−FCS−linux−i386−sdk.tar.bz2
- JavaPlugIn−1.2.2−FCS−linux−i386−glibc−2.1.3.run

Note that to install these Java files, your system needs to have glibc 2.1.3 or better. To check your version of glibc:

```
13;                ls −l /lib/libc*
```

Java2 1.3.0 FCS can also be used and it includes the JavaPlugin. If you use it, you don't have to get JavaPlugin−1.2.2−FCS. Installation of this Plugin is different and you'll have to see it's docs. Installation of Java2 1.3.0 itself, and Java3D, is the same as with Java2 1.2.2.

The Java 3D media extension contains many 3D demo programs/applets and takes advantage of the OpenGL hardware acceleration on the system. The Java 3D API uses the OpenGL API or, on Windows, the DirectX/3D API internally. The demos run as normal java applications and also as applets inside netscape (4.7x) via the Java PlugIn!

3. Install Software

3.1. Install XFree86

Installation of the software packages requires root login, which can be obtained easily via the superuser/setuser command: "su −" (see, man su).

If you have a version of XFree86 installed already, you may want to move it or delete it. However, installing over an existing X is generally OK and preserves any programs or libraries you might have installed into the X directories (not that you should really do that):

```
13;     cd /usr
        mv X11R6 X11R6−old
        cd /etc
        mv X11 X11−old

        −− you may have an X directory in /var also
        cd /var
        mv X11R6 X11R6−old
```

If these locations are not correct for your distribution of Linux, you will have to look around your filesystem a bit − try looking in /var

```
13;     cd /usr/src
        mkdir release
        cd release
        tar −xvzf X402src−1.tgz
        tar −xvzf X402src−2.tgz
        tar −xvzf X402src−3.tgz
        tar −xvzf doctools−1.2.tar.gz

        −− unpack the man pages (actually, glx pages are already present)
        cd /usr/src
        tar −xvzf mangl.tar.Z
        tar −xvzf manglu.tar.Z
```

A file has to be edited to allow these man pages to compile/install with the rest of the distribution:

```
13;     cd /usr/src/release/xc/doc/man/GL
        Edit the file: Imakefile
                SUBDIRS = glx gl glu
```

When you unpacked the man*.tar.Z files above, two new directories where added: gl glu

```
13;     cd /usr/src/release
        cd doctools

        −− Having this variable set confuses the sgml docs build.
        −− With it unset, the build uses the proper defaults.
        unset $SGML_CATALOG_FILES

        make
```

```
        make install
        cd ..
        cd xc/config/cf
        vi host.def
        −− add the following two lines to host.def:
        −−      #define HasSgmlFmt YES
        −−      #define BuildAllDocs YES
        −− See the README file in doctools.

        −− Note: doctools installs the perl program sgmlfmt to
        −− /usr/local/bin.  It looks for the perl executable
        −− at /usr/local/bin/perl.  If perl is installed
        −− on your system at /usr/bin/perl, then it will not
        −− find perl and the sgml docs build will fail!
        −− Make a symlink if needed (or edit the script):
        cd /usr/local/bin
        ln −s /usr/bin/perl perl

        cd /usr/src/release/xc
        make World
        −− before installing, make sure you have moved
        −− or deleted prior installation of X
        −− unless you are sure you want to just overwrite
        make install
        make install.man

        −− make symlinks
        cd /usr/include
        ln −s ../X11R6/include/DPS DPS
        ln −s ../X11R6/include/GL GL
        ln −s ../X11R6/include/X11 X11
        ln −s ../X11R6/include/bitmaps bitmaps
        cd ..
        ln −s X11R6 X11
```

Add /usr/X11R6/lib to your /etc/ld.so.conf file, then run "ldconfig" to update /etc/ld.so.cache so the libraries will be visible.

The GL/GLX/GLU HTML documentation is located at /usr/src/release/xc/doc/hardcopy/GL. This directory can be copied as follows:

```
13;               cd /usr/src/release/xc/doc/hardcopy
               cp −r GL /usr/X11R6/lib/X11/doc/html
```

The index.html file in the docs points to manindex5x.html, but the filename may actually be manindex5.html. Just make a symlink to fix it:

```
13;               cd /usr/X11R6/lib/X11/doc/html
               ln −s manindex5.html manindex5x.html
```

When X is up and running (later), try using the xman program to see that the gl,glx,glu and glut man pages are in section 3. If you have KDE2, khelpcenter allows manpage browsing.

3.2. Install Mesa

Note: This gives you the libGLU* and libglut* files that are missing in XFree86. XFree86 only comes with the OpenGL core library, libGL (based on Mesa). This also installs Mesa's libGL, but we will delete that since it is to be replaced by the Nvidia libGL.

To completely uninstall any Mesa libs that may have come with Slackware:

```
13;             removepkg mesa
```

Procedures vary for other distributions. If there is no clear way to uninstall an existing Mesa, then at least confirm where it is installed: normally either under /usr or /usr/local. The example below assumes that Mesa is installed (or going to get installed) under /usr. Installing over an old version is probably harmless. Look for /usr/lib/libMesa* or /usr/local/lib/libMesa* and delete them unless you have programs that need them.

```
13;     −− IF you are going to use the project GLUT distribution of GLUT, then
        −− unpack the Glut−3.7 packages ...
        −− Mesa's compile looks for it
        cd /usr/src
        tar −xvzf glut−3.7.tar.gz
        tar −xvzf glut_data−3.7.tar.gz
        −− IF you are using this GLUT, use the −−with−glut=/usr/src/glut−3.7
        −− parameter with Mesa's ./configure below in addition to the −−prefix

        cd /usr/src
        tar −xvzf MesaLib−3.4.tar.gz
        tar −xvzf MesaDemos−3.4.tar.gz
        cd Mesa−3.4
        ./configure −−prefix=/usr
        make
        make install
        ldconfig
```

At this point, Mesa installed its own version of the glx.h include files over the ones that XFree86 installed. This will cause some programs to fail to compile and is corrected by copying the XFree86 GL include files from the X source back to your system:

```
13;     cp /usr/src/release/xc/include/GL/*.h /usr/X11R6/include/GL
```

3.3. Install Nvidia OpenGL drivers

```
13;     −− delete the libGL.* files that come with XFree86 / Mesa 3.4 ...
        −− the nvidia libGL.* should replace them
        cd /usr/X11R6/lib
        rm libGL.*
        cd modules/extensions
        rm libGL*
        rm libglx*
        cd /usr/lib
        rm libGL.*
```

```
cd /usr/src
tar −xvzf NVIDIA_kernel−0.9−6.tar.gz
tar −xvzf NVIDIA_GLX−0.9−6.tar.gz
cd NVIDIA_kernel−0.9−6
```

Tip: If you experience problems starting X, see the files TNT_USERS_README and M64_USERS_README. These files explain how to tweak the kernel driver. They were written to fix problems with TNT and TNT2 M64 cards but these tweaks are reported to help the GeForce2 MX also. Try bypassing the BIOS as explained in M64_USERS_README.

```
make
cd ..
cd NVIDIA_GLX−0.9−6
make
ldconfig

−− Make a basic XF86Config file using the "nv" driver:
cd /etc/X11
xf86config
−− Follow the prompts and fill in the information xf86config asks for.
−− Select the Nvidia GeForce or appropriate name.
−− You can test X with this XF86Config file, or continue for OpenGL:

−− You must edit XF86Config and set the following:
vi XF86Config
        Load "glx"
        Replace 'driver "nv"' with 'driver "nvidia"'
        Put "1600x1200" first (or your preferred screen resolution)
        Copy ttf fonts from Windows into a font directory and add a font path.
        Use ttmkfdir (check freshmeat.net) to a fonts.dir file.
        A good place to keep your own fonts is /usr/local/share/fonts ...

−− Nvidia drivers do NOT use the dri module − don't load it.

−− You may like to edit /etc/X11/xinit/xinitrc to have run "startkde"
−− or "gnome−session" instead of twm.

−− Note: /usr/include/GL should be a symlink to /usr/X11R6/include/GL
```

Specifying "nvidia" for the driver in the XF86Config makes that take effect each time you startx. But the NVdriver kernel driver will have to be loaded each time your system boots using:

```
13;             insmod NVdriver
```

You can place this command in one of the system startup files, like /etc/rc.d/rc.modules. But this may not be necessary if the following line is present in /etc/modules.conf:

```
13;               alias char−major−195 NVdriver
```

If this line is present, NVdriver is loaded automatically when X is started (autoclean). You can check if its loaded using the command, lsmod.

3.4. Install Qt

```
13;        −− for Qt, there is no "make install", just place the source
        −− where you want it to live:
        cd /usr/local
        tar −xvzf qt−x11−2.2.3.tar.gz
        ln −s qt−2.2.3 qt
        cd qt
```

Read the INSTALL file about environment variables to setup before you try to build Qt. You can add the following to /etc/profile:

```
13;               QTDIR=/usr/local/qt
               PATH=$PATH:$QTDIR/bin
               MANPATH=$MANPATH:$QTDIR/man
               LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$QTDIR/lib
               export QTDIR PATH MANPATH LD_LIBRARY_PATH
```

LD_LIBRARY_PATH is optional if you include an entry in /etc/ld.so.conf for the library path: /usr/local/qt/lib, then run "ldconfig" to update /etc/ld.so.cache.

```
13;        −− note: configure has some options you can try, to see them
        −− see ./configure −−help
        ./configure

        −− NOTE: when you run make as suggested on the next line, you may
        −− encounter a make error that halts the build IF you run make
        −− from outside X.  The program $QTDIR/bin/uic (the User Interface Compiler)
        −− may Segmentation Fault when run from a Linux console.  You can run
        −− "startx" and use the twm (tiny window manager) and xterm (or whatever you
        −− might have setup for X) to run the rest of the Qt build.  If for some
        −− reason twm is not even available, then you can run "XFree86 38;", use
        −− "CTRL−ALT−F1" to get to a console, start an xterm as
        −−  "xterm −display localhost:0.0 38;", then switch back to X with "ALT−F7".

        make

        −− Only for old versions of Qt before 2.1.0 or so ...
        −− compile the opengl extension
        −− Note that in qt 2.2.0 on, the OpenGL support has been moved out of extensions
        −− and is now a standard part of the library that is installed if configure
        −− finds OpenGL installed on your system.  If you were to not want OpenGL
        −− support in Qt, you'd have to pass the −no−opengl option to configure.
        cd extensions/opengl/src
        −− Check the Makefile and ensure there are not Mesa references.
        make


        ldconfig
        cd ../examples
        −− Try compiling and running the examples.
```

3.5. Install GLUT 3.7 Distribution (optional)

If you installed the MesaDemos package along the Mesa 3.4, then you have already installed GLUT 3.7 since it is included with MesaDemos. However, you may be interested in installing the GLUT manpages and you can skip right to the "Install GLUT manual pages", below ...

Installing GLUT is a bit tricky. I'm not too familiar with imake, the program that it uses to manage the Makefiles, and didn't quite see how to get GLUT to install to where I wanted it (/usr/lib, but MesaDemos will do this without any trouble though). It can be done manually anyhow:

```
13;     cd /usr/src
        tar −xvzf glut−3.7.tar.gz
        cd glut−3.7

        Read the file: README.linux
        cd linux
        READ the file: README
        cp Glut.cf ..
        cd ..
        Edit Glut.cf: remove any Mesa references.
        Replace any −lMesaGL −lMesaGLU with −lGL −lGLU if needed.
        In particular, replace:
                OPENGL = $(TOP)/../lib/libMesaGL.so
                GLU = $(TOP)/../lib/libMesaGLU.so
        with:
                OPENGL = −lGL
                GLU = −lGLU

        ./mkmkfiles.imake
        cd lib/glut
        cp /usr/src/glut−3.7/linux/Makefile .
        Edit the Makefile: remove any Mesa references.
        Replace any −lMesaGL −lMesaGLU with −lGL −lGLU if needed.
        In particular, replace:
                OPENGL = $(TOP)/../lib/libMesaGL.so
                GLU = $(TOP)/../lib/libMesaGLU.so
        with:
                OPENGL = −lGL
                GLU = −lGLU

        make
        ln −s libglut.so.3.7 libglut.so
        ln −s libglut.so.3.7 libglut.so.3
        cp −d libglut.* /usr/lib
        cd ..
        cd gle
        −− make a shared lib for libgle
        make
        gcc −shared −o libgle.so.3.7 *.o
        ln −s libgle.so.3.7 libgle.so
        ln −s libgle.so.3.7 libgle.so.3
        cp −d libgle.* /usr/lib
        cd ..
        cd mui
        −− make a shared lib for libmui
        make
        gcc −shared −o libmui.so.3.7 *.o
        ln −s libmui.so.3.7 libmui.so
        ln −s libmui.so.3.7 libmui.so.3
```

```
        cp −d libmui.* /usr/lib

        −− Install the GLUT manual pages (not included with MesaDemos)
        cd /usr/src/glut−3.7
        make SUBDIRS=man Makefile
        cd man/glut
        make install.man
        ldconfig

        cd ../../progs/demos/ideas
        −− edit the Makefile, change OPENGL = −lGL and GLU = −lGLU
        make
        ./ideas
        −− test compiling some demos
        −− take a look at which libraries have to be linked (−lX11 ...) in
        −− the Makefiles.  Qt's tmake program available at www.troll.no
        −− is a quick way to make a Makefile but you have to edit it
        −− and add the −l needed.
```

3.6. Install Java 3D (optional)

If you already have a Java JDK/SDK or JRE, that is, a Java/Software Development Kit or Java Runtime Environment, installed, then you may have to take care to uninstall them or leave them alone!

It is recommended that you have the lastest version of Netscape 4.7x, which at this time of writing, is 4.76, if you plan to install the Java PlugIn for netscape. It works, but you may (or may not) experience Segmentation Faults when leaving a page that contained a Java 3D applet.

Assuming your are logged in as root and have downloaded the Java packages from blackdown.org into the root home directory, /root, do:

Install the Java 2 SDK (1.2.2) and Java 3D (1.2) extension:

```
13;               cd /usr/local
                tar −xvyf ~/j2sdk−1.2.2−FCS−linux−i386−glibc−2.1.3.tar.bz2
                ln −s jdk1.2.2 jdk
                cd jdk
                tar −xvyf ~/java3d1_2−FCS−linux−i386−sdk.tar.bz2
                cd jre/lib/ext
                cp j3d* ..
                cp vecmath.jar ..
                cd /usr/local
                chown −R root:root jdk1.2.2
```

Edit /etc/profile, add:

```
13;               JAVA_HOME=/usr/local/jdk
               PATH=$PATH:$JAVA_HOME/bin
               export JAVA_HOME PATH
```

This completes the installation of the Java 2 SDK, which includes the JRE, and the Java 3D 1.2 extension.

Install the Java PlugIn for netscape:

```
                −− source profile to set JAVA_HOME and PATH
                source /etc/profile
                cd
                chmod u+x JavaPlugIn−1.2.2−FCS−linux−i386−glibc−2.1.3.run
                ./JavaPlugIn−1.2.2−FCS−linux−i386−glibc−2.1.3.run
                −− each user has to run this file to install the plugin
                −− it is per user, not global
                netscape 38;
```

When netscape loads, go to Edit/Preferences/Advanced and Enable Java and Enable Java Plugin, then exit netscape.

The next step is to configure the Java PlugIn. It comes with a configuration applet.

```
13;                netscape ~/.netscape/java/ControlPanel.html 38;
                −− the Control Panel for JavaPlugIn should load
```

Again, netscape will load. Click the "Advanced" tab, select "Other ..." and type for Path:

```
13;                /usr/local/jdk1.2.2/jre
```

Then click "Apply" and exit Netscape. By changing this, it tells the Plugin to use the system Java Runtime Environment instead of the JRE it installed under ~/.netscape. The system JRE is where the Java 3D API extension is installed.

Test Java 3D demos:

```
13;                cd $JAVA_HOME/demo/java3d/GearTest
                java GearBox 38;
                −− runs as normal java application
                netscape GearBox_plugin.html 38;
                −− runs in netscape as an applet
```

If you experience trouble with Java, you can try deleting ~/.java and related files in your home directory, then try again. These files left over from a prior Java installation can cause problems.

If all works well, you should have a complete Java Developement and Runtime Environment for both normal apps and high−performance 3D apps. See http://java.sun.com/ for further information about Java and the Java 3D extension.

---

4. Final comments

I believe that is about it! At this point you should have a fully functioning OpenGL system for running and developing OpenGL apps.

You can try building the demos in /usr/src/Mesa−3.4/{demos, xdemos} by using the Makefile.X11 as Makefile and running "make targets" or "make teapot" etc. They should build and link with the hardware accelerated libGL and run very fast! Qt has a OpenGL example in $QTDIR/examples/gear, that you should be able to run as simply as "make;./gear".

Just about any standard GL/GLX/GLU/glut app should run fine, such as WolfGL, GLQuake, glqwcl.glx (GLQuakeWorld), quake2, and of course ... quake3!!!

HAVE FUN!