Tutorial 2: Installing OpenSceneGraph under Linux

Franclin Foping
franclin@netcourrier.com

April 25, 2008

Abstract

In the previous tutorial, we have learnt in depth what is OSG. We have also learnt more about scene graphs and graphs. This tutorial will focus on the actual installation process under Linux .

1 Structure of the tutorial

I will start by giving the reasons of my OS choice in section 2 and the installation will be unwound in section 3. There is also the FAQ at the section 4 where I will answer some questions related to the installation of OSG in Linux. The tutorial will be closed by a conclusion.

2 Motivation

A couple of months ago when I first used OSG in Windows I was wondering why I got some weird results. When I convert all my projects to Linux, I quickly realized that I should try OSG on another plateform, therefore Linux was the best candidate.
Secondly, in the official wiki (official website (2007)) the setup process for Windows using Visual Studio is very well documented so there is no need to

write a tutorial about how to install OSG in Windows!
Another reason is the multithreaded architecture of OSG, as you probably know both OSs handle it differently. To be honest, I prefer the Linux method. Another drawback for Windows users, Windows automatically sets the maximum framerate (FPS) of your application to the value of the refresh rate of your monitor. This means that you can never reach a FPS of 60 if your monitor refresh rate is 60. This is even true if you using laptops. If you are unlucky to compile your code in the debug mode of Visual Studio, your program will be slower. Of course, OSG supplied an Optimizer class which will be covered in another tutorial, but still beware of Windows... On the hand hand, you can achieve a framerate of 300 in Linux even if you refresh rate is 50. The proof is shown in figure 1. Amazing isn't it? That is the way Linux goes!

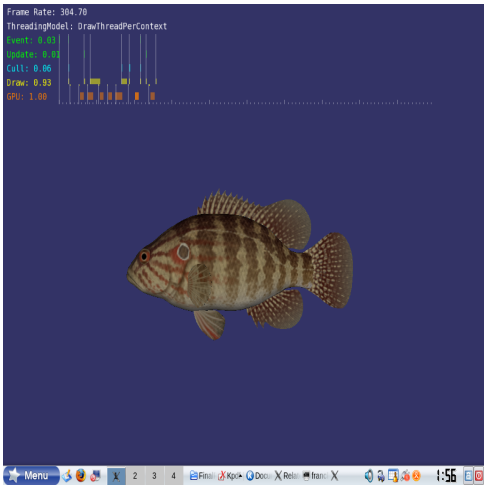

Figure 1: A lovely fish

3 Installing OSG v2 under Linux

Ok, now that you know why I want you to use Linux, it is time to show how you should install OSG on your computer.
If you are using **Ubuntu 7.04: Gutsy**, please skip this section because OSG binaries are already included in that distribution. The only thing you have

to do is to use the package manager (Synaptic for instance) to install OSG and that is it, isn't it?.
If you are using another Linux distribution, you will need to follow me until the end of this tutorial because you have to install OSG from the source code, fortunately enough it is free and open source! You can also install OSG from available *rpm* binary. However I don't really recommend you to do so because of some likely dependency. To be honest, when you compile OSG on your computer, you can easily see what is going on, in case of errors it is easy to tweak some variables and it just works.
Installing from source code is also another way to learn how to compile and run software in Linux, believe it is funny!

3.1 Step 1: Hardware requirements

In the previous tutorial, I described OSG architecture and mentioned its multi-threaded aspect. Therefore, a dual-core processor will be a good boost for your job.
Also a decent graphics card is also needed. For Linux users, I strongly advise you to get an **NVIDIA** graphics card. You may be wondering why I choose nVIDIA, this is because of their Linux drivers. If you are using ATI graphics card, you may experience some issues to install the latest drivers of your card on Linux. The reason being ATI is only concerned about Windows drivers, they don't really spend time on Linux drivers like nVIDIA.
I have used nVIDIA graphics cards on many computers all running Linux and I can confirm that it works.

3.1.1 Step 2: GPU Drivers

I mentioned previously the nature of your graphics cards, don't I? You also know that OSG is built on top of OpenGL so you need to install the **latest** driver of your graphics card and OpenGL.
On Linux, OpenGL header files are contained in the **Mesa** library. So you need to install this package. I am sure it is located on your distribution DVD.
By default, OpenGL header files are located in the following directory:

/usr/include/GL. Make sure this directory exists on your system and contain the following files: *gl.h, glu.h, glext.h and glx.h*. If so, congratulations! you are now ready to install OSG version 2.

3.2 Step 3: Generating makefiles

In previous release of OSG, there was no need to generate makefiles because they were already supplied with the source code. However, in the latest release (version 2.2) there is a cross plateform build system generator called **cmake**.
Therefore you should first install **cmake** on your system.

3.2.1 Download cmake

Get yourself a copy of the latest cmake source code. Click on `www.cmake.org`. Please note that in order to compile OSG source files, you need a version released after the 2.4.6. version.

3.2.2 Installing cmake

Installing cmake on your system is the easiest job on the earth: only 3 steps are needed:

1. unzip the archive file and change your current directory to the newly created directory. For instance you should type:
   tar xvzf cmake-2.4.7.tar.gz; cd cmake-2.4.7

2. You now need to configure your environment by using the following command:
   ./configure

3. Finally you need to compile and install cmake by typing
   make && make install

You can check the return code of each step by typing **echo $?**, if it returns 0, congratulations! everything was ok! if not, an error occured. My advice

to you is to always check the return code of every step.
By default, cmake is installed in the following directory: **/usr/local/bin** and if it is included in your PATH environment variable, you can easily test that cmake was successfully installed by just typing **cmake** in the prompt or **man cmake** if you see something, congratulations! you are now ready to compile OSG. Now in order to generate OSG makefiles, unzip OSG source code and change the working directory to the newly created directory.
unzip OSG.zip ; cd OSG ; cmake . Note that the full stop after the cmake command is **compulsory**! Check that everything went ok and if yes, congratulations! You can go on to the next step.

3.3 Compiling and installing OSG

All you need to do now is to type the following commands:
make && make install. Please note that the compiling process takes a while, in fact you are compiling about 151 projects so go and get a cup of tea. Even if you are using a powerful computer, you still need to be patient. This is also true for Windows users! The process will first create osg shared libraries and then executables for example codes. Shared libraries are Linux equivalent to Windows' DLLs.

3.4 Verifying your installation

The easiest way to test OSG is to use a built-in example that doesn't make use of the X server. There is an example specially designed for that purpose: **osgversion**. So change your working directory to the OSG bin directory:
cd OSG- bin ; ./osgversion
If you see the following text:
OpenSceneGraph Library 2.0.0
CONGRATULATIONS!!! Everything happens as expected, go and get yourself a bottle of beer!
Now to test a graphic project, I advise you to use the built-in example **osglogo**. So as in the previous case, you only have to type on the prompt **cd**

OSG- bin ; ./osglogo

Figure 2 shows the result: the beautiful OSG logo!

Figure 2: OSG logo

4 FAQ

1. **I cannot build the examples, why?**
   This happens because of the configuration of CmakeCache.txt *file, unfortunately if you generate it without a GUI front-end for Cmake, you will have to change some flags in that file. By default, Cmake turns off the corresponding flag to build examples. Therefore you have to edit that file manually. There is an easy way around it, though. Open the CmakeCache.txt file with a text editor and then go at the lines 29 and 30, you should see the following text:* ***//Enable to build OSG Examples BUILD_OSG_EXAMPLES:BOOL=OFF****, the only thing you should do is to replace OFF by ON. Save your modification and run* ***make****, you should now be able to build all examples.*

2. **During compilation, I got a message *unable to find OPENGL_INCLUDE_DIR*, what is wrong?**
   This is an indication that the GNU C compiler (gcc or g++) cannot find OpenGL header files. Recall from section 3 that you have to use the latest driver of your GPU and also you have to install the MESA *package which is found on your DVD distribution! OpenGL header files are usually copied at the right directory (/usr/include/GL) when you installed the GPU driver. Beware of it!*

3. **When I run an example or and OSG program, it says that it cannot find osg libraries!**
   This is because the loader cannot find OSG shared libraries. By default, the OSG installation directory for libraries is /usr/local/lib. However, you need to tell the Runtime Linker the location of those libraries. So log in as root and edit the file /etc/ld.so.conf. *You should add the following directory at the end of that file* /usr/local/lib. *Close the editor and type* ***ldconfig****. That is it!*

5 Conclusion

Throughout this tutorial, I talk about installing OSG on Linux, hopefully everything went ok with you, if not please feel free to drop me an email. Even in this tutorial there is no exercise, we will soon start coding. It is important to install OSG on your Linux before we go on otherwise you will be lost in the subsequent tutorials. The next tutorial will focus on some important C++ skills for OSG developers. These skills are so important that they deserve a separate tutorial. Fasten your belt because we will start hardcore coding in the next tutorial. It will also be the first tutorial with some exercises for you to practise, remember practice makes perfect!

References

official website, O. (2007), 'OSG Official Website', [Available online: `http://www.openscenegraph.org`].