# Visual Computing and Multimedia

Abel J. P. Gomes

## LAB. 1

## INTRODUCTION TO OpenGL

1. Getting Started
2. Installing Graphics Libraries: OpenGL and GLUT
3. Setting up an IDE to run graphics programs in OpenGL/GLUT
4. A First OpenGL/GLUT Program
5. Programming Exercises

Department of Computer Science and Engineering
University of Beira Interior
Portugal
2011

# Lab. 1

# INTRODUCTION TO OpenGL

## 1. Getting Started

A starting point for learning computer graphics via OpenGL API (Application Programming Interface) is the following wiki:

http://www.opengl.org/wiki/Getting_started

Before starting programming it is good idea to know details about system, say renderer/graphics card and CPU, as well as OpenGL and DirectX versions installed in your computer. Of course, we are here assuming that your computer is a PC running Microsoft Windows XP or Vista.

We can obtain all this information by downloading and installing an OpenGL viewer such as, for example, the OpenGL Extension Viewer (Windows, Windows x64 and MacOS X), which is available from:

http://www.realtech-vr.com/glview/index.html

For more details about other OpenGL viewers, have a look at the wiki aforementioned.

The idea of producing pictures or images using a computer requires two processes: modeling and rendering. **Modeling** has to do with the creation, manipulation, and storage of geometric objects on computer, while rendering just means converting a scene to an image. **Rendering** involves a number of transformations, namely: rasterization, shading, illumination, and animation of the image.

Computer graphics has been widely used in many areas of interest from graphics presentation, paint systems, computer-aided design (CAD), and scientific visualization to simulation of natural phenomena, virtual reality, computer games and, in general, entertainment.

## 2. Installing Graphics Libraries: OpenGL and GLUT

We are going to use the Microsoft Visual C++ Studio in the labs. This IDE (Integrated Development Environment) only runs on Windows XP/Vista machines.

Note that the OpenGL libraries (**`glu32.dll`** and **`opengl32.dll`**) are already installed in Windows. Usually, they are in the directory:

**`C:\WINDOWS\system32`**

In order to run OpenGL programs on any machine (Windows, Linux or Mac OSX), it is necessary to use a supplementary library, called **GLUT** (The OpenGL Utility Toolkit). GLUT is a window system independent toolkit for writing OpenGL programs. It implements a simple windowing API for OpenGL that interfaces to

proprietary windowing systems of specific operating system; hence its platform independence.

GLUT supports:

- Multiple windows for OpenGL rendering
- Callback driven event processing
- Sophisticated input devices
- An 'idle' routine and timers
- A simple, cascading pop-up menu facility
- Utility routines to generate various solid and wire frame objects
- Support for bitmap and stroke fonts
- Miscellaneous window management functions

For more details about GLUT, the reader is referred to:

http://www.opengl.org/resources/libraries/glut/

and

http://www.xmission.com/~nate/glut.html

From this last link of Nate Robbins, download the GLUT toolkit from glut-3.7.6-bin.zip (117 KB). This zip file contains the following files:

- `glut.dll`
- `glut32.dll`
- `glut.h`
- `glut.lib`
- `glut32.lib`

The dlls (**`glut.dll`** and **`glut32.dll`**) must be moved into the folder

**`C:\WINDOWS\system32`**

The libs (**`glut.lib`** and **`glut32.lib`**) must be moved into the folder

**`C:\Program Files\Microsoft Visual Studio\VC98\Lib`**

The include (**`glut.h`**) must be moved into the folder

**C:\Program Files\Microsoft Visual Studio\VC98\Include\GLUT**

## 3. Setting up an IDE To Run Graphics Programs in OpenGL/GLUT

Let us assume that you have already downloaded the GLUT files and installed them in the appropriate folders.

Starting up an OpenGL project on Microsoft Visual C++ Studio involves the following steps:

1. Start VC++ and create a new project.
2. The project has to be a "Win32 Console Application".
3. Type in the name of the project and where it will be on your hard disk drive.
4. Chose an empty project and click next or finish.

Setting up the OpenGL and GLUT libraries for the project involves the following steps:

1. The first thing you need to do when the project opens up is to click on the "Project" menu item from the top.
2. Chose "Settings" (a window will come up).
3. On the left side of the window there are tabs, chose the "Link" tab.
4. The string field labeled "Object/library modules" has a few lib files already set in it.
5. Go to the end of this string field and enter:

   **opengl32.lib glut32.lib glu32.lib**
6. Chose "OK" and that will include all the OpenGL libraries that you need.
7. Now all you need to do is include <GL\glut.h> and you are ready to go.

## 4. A First OpenGL/GLUT Program

The following program draws a line segment between two points, P(5.0,1.5) and Q(9.3,7.2) in the 2-dimensional Euclidean space.

```
1  #include <GLUT/glut.h>        // Header file for GLUT and OpenGL
2  #include <stdlib.h>
3
4  //------------------------------------------------------------
5  // DRAWING CALLBACK FUNCTION
6  //------------------------------------------------------------
7
```

```
8  void draw(){
9       // background colour: yellow
10   glClearColor( 100, 100, 0, 0 );
11   glClear ( GL_COLOR_BUFFER_BIT );
12
13      // Sets up the DOMAIN (xmin,xmax,ymin,ymax) in R^2.
14      // Let (xmin,xmax,ymin,ymax)=(0.0,20.0,0.0,20.0)
15   glMatrixMode(GL_PROJECTION);
16   glLoadIdentity();
17   gluOrtho2D(0.0,20.0,0.0,20.0);
18
19      // Let us now define the line segment in red
20      // Endpoints: (5.0,1.5) and (9.3,7.2)
21   glMatrixMode(GL_MODELVIEW);
22   glLoadIdentity();
23   glColor3f( 1, 0, 0 );
24   glBegin(GL_LINES);
25     glVertex2f(5.0,1.5);
26     glVertex2f(9.3,7.2);
27   glEnd();
28
29      // display line segment
30   glutSwapBuffers();
31 }
32
33 //------------------------------------------------------------------
34 // KEYBOARD CALLBACK FUNCTION
35 //------------------------------------------------------------------
36
37 void keyboard(unsigned char key,int x,int y)
38 {
39      // press ESC key to quit
40   if(key==27) exit(0);
41 }
42
43 //------------------------------------------------------------------
44 // MAIN FUNCTION
45 //------------------------------------------------------------------
46
47 int main(int argc,  char ** argv)
48 {
49   glutInit(&argc, argv);
50      // Double Buffered RGB display
51   glutInitDisplayMode( GLUT_RGB | GLUT_DOUBLE);
52      // Set window size
53   glutInitWindowSize( 500,500 );
54   glutCreateWindow("Line Segment");
55      // Declare callback functions
56   glutDisplayFunc(draw);
```

```
57    glutKeyboardFunc(keyboard);
58       // Start the main loop of events
59    glutMainLoop();
60    return 0;
61 }
```

**Comments**:

(1) The previous program illustrates the basic structure of an OpenGL/GLUT program:

a. First, we define the scene domain (or scene world) where we are working on. This is given by the statement

```
gluOrtho2D(0.0,20.0,0.0,20.0);
```

This is the world of the **geometry**. This is usually done in the drawing callback function.

b. Second, we define the image domain (or image world) where we are going to display our rasterized scene in pixels. This is given by the statement

```
glutInitWindowSize(500,500);
```

This is the world of the image. This is usually done in the main function.

## 5. Programming Exercises

1. Write a program to draw the following points: (0.0,0.0), (20.0,0.0), (20.0,20.0), (0.0,20.0) and (10.0,25.0). For this purpose, use the **GL_POINTS** primitive.
2. Write a program to draw the sinc function. This function is given by

$$\mathrm{sinc}(x) = \frac{\sin(x)}{x}$$

with $x \in [-10.0,10.0]$. For this purpose, use the **GL_LINE_STRIP** primitive. The sampling step is 0.25 when $x$ varies in the given interval. Note that sinc(0.0)=1.0.

3. Write a program to draw a circle centered at ($c_x$,$c_y$) with a given radius *r*, whose points are given by the following equations

$$\begin{cases} x = c_x + r.\cos(\theta) \\ y = c_y + r.\sin(\theta) \end{cases}$$