Experiment No 1

Title Install and explore the OpenGL.

Computer Graphics(Basic and how to install OpenGL/Glut)

INTRODUCTION

What is Computer Graphics?

Branch of Computer Science.

Computer + Graphs + Pics = Computer Graphics.

Drawing line, Chart, Graphs etc. on the screen using Programming language is computer Graphics.

Book Definition

Computer graphics is the branch of computer science that deals with generating images with the aid of

computers. It displays the information in the from of graphics objects such as picture, charts, graphs and

diagrams instead of simple text. We can say computer graphics makes it possible to express data in pictorial

form.

OpenGL

It is cross-platform, cross-language API for rendering 2D and 3D Graphics(Vector Graphics).

Refer https://www.khronos.org/opengl/

GLUT

http://freeglut.sourceforge.net/docs/api.php#Introduction

https://www.opengl.org/resources/libraries/glut/spec3/node10.html

https://www.opengl.org/resources/libraries/glut/spec3/node11.html

https://www.opengl.org/resources/libraries/glut/spec3/node12.html

https://www.opengl.org/resources/libraries/glut/spec3/node13.html

https://www.opengl.org/resources/libraries/glut/spec3/node15.html

How to Setup Glut on Windows

https://medium.com/swlh/setting-opengl-for-windows-d0b45062caf

How to install Ubuntu in Windows Virtual Box

https://youtu.be/x5MhydijWmc

How to install OpenGL / Glut (Ubuntu)

Step 1: sudo apt-get update

To update your basic packages

Step 2: sudo apt-get install build-essential

For installing essential packages.

Step 3: sudo apt-get install freeglut3 freeglut3-dev

Step 4: sudo apt-get install binutils-gold

Step 5: sudo apt-get install g++ cmake

Step 6: sudo apt-get install mesa-common-dev mesa-utils

Step 7: sudo apt-get install libglew-dev libglew1.5-dev libglm-dev

Step 8: glxinfo | grep OpenGL

Create cpp file and write your code

Run on terminal:

```
g++ MyProg.cpp -lGL -lGLU -lglut (for C++ program)
./a.out
```

Basic Code # Triangle

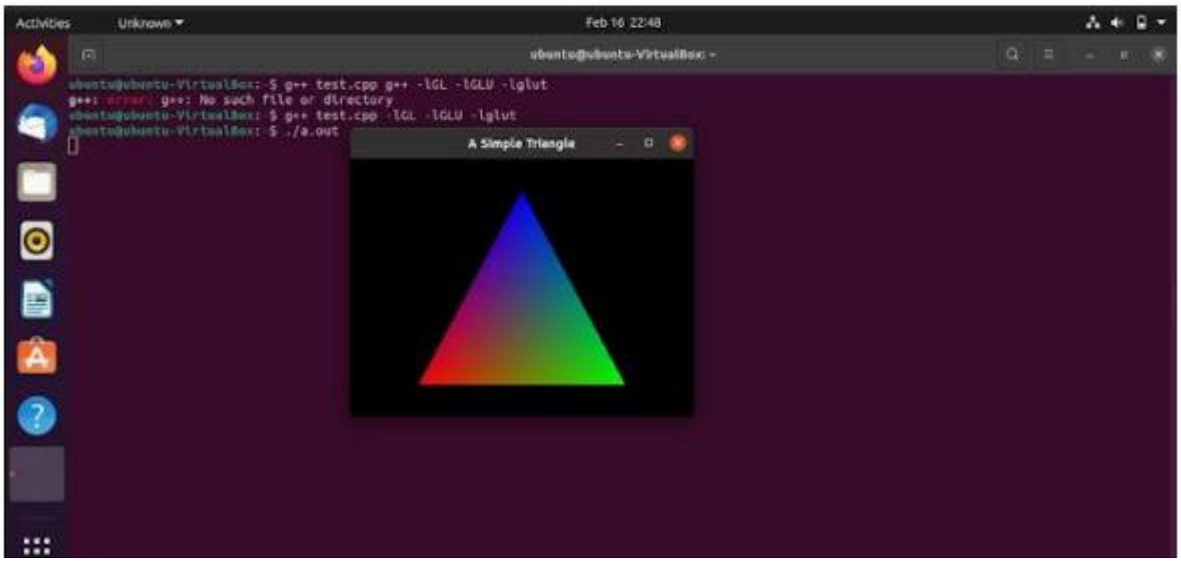
In C++

```
// A simple introductory program; its main window contains a static picture
// of a triangle, whose three vertices are red, green and blue. The program
#include <GL/glut.h>
// Clears the current window and draws a triangle.
void display() {
 // Set every pixel in the frame buffer to the current clear color.
 glClear(GL_COLOR_BUFFER_BIT);
 // Drawing is done by specifying a sequence of vertices. The way these
 // vertices are connected (or not connected) depends on the argument to
 // glBegin. GL_POLYGON constructs a filled polygon.
 glBegin(GL_POLYGON);
 glColor3f(1, 0, 0); glVertex3f(-0.6, -0.75, 0.5);
 glColor3f(0, 1, 0); glVertex3f(0.6, -0.75, 0);
 glColor3f(0, 0, 1); glVertex3f(0, 0.75, 0);
 glEnd();
 // Flush drawing command buffer to make drawing happen as soon as possible.
 glFlush();
}
// Initializes GLUT, the display mode, and main window; registers callbacks;
// enters the main event loop.
int main(int argc, char** argv) {
 // Use a single buffered window in RGB mode (as opposed to a double-buffered
 // window or color-index mode).
 glutInit(&argc, argv);
 glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
 // Position window at (80,80)-(480,380) and give it a title.
 glutInitWindowPosition(80, 80);
 glutInitWindowSize(400, 300);
 glutCreateWindow("A Simple Triangle");
 // Tell GLUT that whenever the main window needs to be repainted that it
 // should call the function display().
 glutDisplayFunc(display);
 // Tell GLUT to start reading and processing events. This function
 // never returns; the program only exits when the user closes the main
 // window or kills the process.
 glutMainLoop();
}
```

How to run glut/ OpenGL in g++ (Ubuntu)

```
g++ MyProg.cpp -lGL -lGLU -lglut
```

OUTPUT



```
ubuntu@ubuntu-VirtualBox:~$ g++ test.cpp g++ -lGL -lGLU -lglut
g++: error: g++: No such file or directory
ubuntu@ubuntu-VirtualBox:~$ g++ test.cpp -lGL -lGLU -lglut
ubuntu@ubuntu-VirtualBox:~$ ./a.out
```