VISVESVARAYA TECHNOLOGICAL UNIVERSITY

BELGAUM - 590014.



**A Mini Project Report On**

**“ROTATION OF MULTIPLE 3-D OBJECTS”**

Submitted in the partial fulfillment of the requirements for 6th semester of the Degree of

Bachelor of Engineering in Computer Science and Engineering.

**SUBMITTED BY**

**SWATHI R( 3BR12CS116)**

**TEJASHWINI V(3BR12CS118)**

Under the Guidance of

Prof.Sudhakar Avareddy

Asst.Prof, Dept of CSE



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**BELLARY INSTITUTE OF TECHNOLOGY & MANAGEMENT**

Bellary-Hospet Road, Allipur, Bellary -583101

(Affiliated to Visvesvaraya Technological University, Belgaum)

**BALLARI INSTITUTE OF TECHNOLOGY AND MANAGEMENT**

Bellary-Hospet Road, Allipur, Bellary -583101

(Affiliated to Visvesvaraya Technological University, Belgaum)

Department of Computer Science and Engineering



**CERTIFICATE**

Certified that the project work entitled **“ LUDO GAME ”** carried out by **SWATHI R( 3BR12CS116), TEJASHWINI V(3BR12CS118)**bonafide students of Ballari Institute of Technology & Management, Bellary in partial fulfillment of the **COMPUTER GRAPHICS AND VISUALIZATION LAB (10CSL67**) project of 6th semester of **Bachelor of Engineering in computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the year 2013-2014.

**Prof.Sudhakar Avareddy** **Dr. R N Kulkarni**

**Project guide HOD, Dept of CSE**

**External Viva:**

**Signature of the Examiners:**

**Internal: External:**

**1: 1:**

**2: 2:**

**ACKNOWLEDGEMENT**

The satisfaction and euphoria that accompany the successful completion of any project would be incomplete without mentioning the people who made it possible, whose constant guidance and encouragements crowned my efforts with success. I take this opportunity to express the deepest gratitude and appreciation to all those who held me directly or indirectly towards the successful completion of the project.

It is indeed great pleasure and privilege for me to meet this project report. I wish to take opportunity to pay my profound gratefulness and express my indebtness to my guide SUDHAKAR AVAREDDY, for his keen interest in preparing the project report and also to the lecturer S.Basavaraj Instructor for helping in making out this project.

.

I am very thankful to Dr.R.N.KULKARNI Head of the Department of Computer Science & Engineering for his valuable guidance and suggestion.

I would like to express my sincere gratitude to Dr.V.C.PATIL Principal for providing facilities in the college campus.

I am also thankful for all the Staff members of Computer Science & Engineering dept for their Co-operation of project report.

**CONTENTS**

1. **INTRODUCTION 6-7**
   1. **Overview of openGL**
   2. **Project purpose**
   3. **Project scope**
   4. **Design options**
   5. **Specifications**
2. **REQUIREMENT SPECIFICATION 8**
   1. **Tools used**
   2. **Hardware requirements**
   3. **Software requirements**
3. **FUNCTIONS USED 9-13**
4. **SNAPSHOTS 14-19**

**5. CONCLUSION 20**

**6. BIBLIOGRAPHY 21**

**ABSTRACT**

* + This project is done using OpenGL functions. This project contains information about the project, instructions and game board to play
  + In this project right button of the mouse is used, which shows the menu of different options to play and quit using mouse and keyboard keys.

**1. INTRODUCTION**

OpenGL is a software interface to graphics hardware that allows the programmers to create 2D and 3D graphical images. Interface consists of about 150 distinct commands that can be used to specify the objects and operations needed to produce interactive three-dimensional applications.

**1.1. Overview of OpenGL**

OpenGL is a hardware-independent, operating system independent, vendor neutral graphics API specification. OpenGL is both a standard API and the implementation of that API. OpenGL has been designed using a client/server paradigm, allowing the client application and the graphics server controlling the display hardware to exist on the same or separate machines. The network is transparent to the application.OpenGL is window system independent, and therefore contains no windowing operations or mechanisms for user input. Also, OpenGL does not provide direct support for complex geometrical shapes, such as cubes or spheres. These must be built up from supported primitives.

The program is OpenGL functions in a client/server environment. That is, the application program producing the graphics may run on a machine other than the one on which the graphics are displayed. The server part of OpenGL, which runs on the workstation where the graphics are displayed, can access whatever physical graphics device or frame buffer is available on that machine, fully functional, including everything that users usually want out of 3D graphicsSome features of OpenGL include the following:

* Geometric and raster primitives
* RGBA or color index mode
* Display list or immediate mode
* Viewing and modeling transformations
* Lighting and Shading
* Hidden surface removal (Depth Buffer)
* Anti-aliasing
* Texture Mapping
* Feedback and Selection
* Accumulation Buffer
* Wireframes
* Motion

OpenGL’s rendering commands however are “primitive”. We can tell the program to draw points, lines and polygons, and we have to build more complex entities upon these. OpenGL is designed to work efficiently even if the computer that displays the graphics you create, isn't the computer that runs your graphics program. OpenGL is hardware-independent. Many different vendors have written implementations that run on different hardware. These implementations are all written to the same OpenGL standard and are required to pass strict conformance tests.

**1.2. Project Purpose:**

The project demonstrates Ludo Game . It also includes movements in Bicycle like rotation of handle in clockwise and anticlockwise,increament and decrement of speed,zoom-in and zoom-out. It is a user-friendly interface that is menu oriented.

**1.3. Project Scope:**

With every user-friendly interface this editor helps in showing the overview of the 3D Bicycle. The 3-D graphics package designed here provides an interface for the users to move the Bicycle.

**1.4. Design Options:**

The Graphics Package is designed using the inbuilt graphics library. The objects, which can be drawn using the editor, are stored as functions that can be used according to the requirements.

Few header files available in OpenGL are:

* #include<GL/gl.h>

This includes the OpenGL core header file. It is required by all OpenGL applications.

* #include<GL/glu.h>

This includes the OpenGL Utility Library header file. It is required if ‘glu’ functions are used.

* #include<GL/glut.h>

This includes the OpenGL Utility Toolkit header file. This statement automatically includes header

files gl.h and glu.h

**1.5. Specifications:**

OpenGL is implemented using Dev-C++. The GL/glut.h header file is included. C++ programming language is used to implement the entire code. The user interface is done with the help of input devices like mouse and keyboard.

**2.REQUIREMENT SPECIFICATION**

**2.1. Tools used**

* **Window:**

A data structure describes a sub-rectangle of the screen (possibly the entire screen). You can write to a window as thought it were a miniature screen, scrolling independently of other windows on the physical screen.

* **Screens:**

A subset of windows which area as large as the terminal screen, i.e., they start at the upper left hand corner and encompass the lower right hand corner.

* **Terminal Screen:**

The package’s ideal of what the terminal display currently looks like, i.e., what the user sees now.

* **Graphics Library:**

The graphics library provides a lot of functions and routines for drawing and for doing graphics operations.

* **Output:**

The project outputs about the game, instructions about the game and winner of the game.

* **Debugging:**

Debug () function can be used explicitly to see if any error occurs during the level of execution. Tracing can be done using breakpoints and single step execution.

**2.2. Hardware requirements:**

* OS: Windows xp/windows 7
* RAM: 512 MB
* Processor : p4, core5 II generation or any new versions of processor
* HDD : 15GB

**2.3. Software requirements:**

* Dev c++ 4.9.9.2v
* Glut 3.7.6

**3. Functions Used**

The project makes use of the following built-in and user-defined functions.

*Built-in functions:*

1. glClearColor()
2. glEnable()
3. gluInitDisplayMode()
4. glutBitMapCharacter()
5. glutPostRedisplay()
6. glClear()
7. glPushMatrix()
8. glRotated()
9. glPopMatrix()
10. glOrtho()
11. glRasterPos3f()
12. glutSolidCube()
13. glRotatef()
14. glFlush()
15. glColor3f()
16. glPointSize()
17. glViewPort()
18. glMatrixMode()
19. glLoadIdentity()
20. glutInit()
21. glutDisplayMode()
22. glutInitWindowSize()
23. glutInitWindowPosition()
24. glutCreateWindow()
25. glutDisplayFunc()
26. glEnd()
27. glVertex2f()
28. glBegin()
29. glLineWidth()
30. glutReshapeFunc()
31. glutKeyboardFunc()
32. glutMouseFunc()
33. glutCreateMenu()
34. glutAddMenuEntry()
35. glutAttachMenu()
36. glutMainLoop()

***User defined functions***

* Void dice()
* Void player1()
* Void player2()
* Void call()
* Void mymouse()
* Void display2()
* Void wait()
* Void display1()
* Void key()
* Void about()
* Void about1()
* Void about2()
* Void check\_cond()
* Void circle()
* Void pl1\_pl2()
* Void arrow()
* Void display\_msg()
* Void display\_about()
* Void display\_about\_game()
* Void display\_inter\_guide()
* Void enter\_display2()
* Void square()
* Void display\_enter1()
* Void options()

Project makes use of following ***header files*:**

1.stdlib.h

2.math.h

3.GL/glut.h.

4.stdio.h

**Description of the built-in functions:**

* **glClear** (GLbitfield mask);

This function clears the specified buffers to their current clearing values. The ‘mask’arguement is bitwise-

OR.. It can be one of the following buffers.

**Buffer Name**

Color buffer GL\_COLOR\_BUFFER\_BIT

Depth buffer GL\_DEPTH\_BUFFER\_BIT

Accumulation buffer GL\_ACCUM\_BUFFER\_BIT

* **glMatrixMode** (GLenum mode);

This function specifies which matrix will be affected by the subsequent transformations. Mode can be

GL\_MODELVIEW, GL\_PROJECTION or GL\_TEXTURE.

* **glLoadIdentity** ();

This function sets the current transformation matrix to an identity matrix.

* **glPushMatrix** ();

We can save the present values of matrices by pushing them onto the top of the stack.

* **glPopMatrix** ();

We can recover the previous values of matrices by popping the contents from the stack. If the stack contains

a single matrix then calling this function will generate an error.

* **glRotatef** (TYPE angle, TYPE dx, TYPE dy, TYPE dz);

Alters the current matrix by rotating it through ‘angle’ degrees about the axis (dx,dy,dz). TYPE is either GLfloat or GLdouble.

* **glFlush ();**

This function empties all commands in these buffers and forces all pending commands to be executed immediately without waiting buffers are full. It guarantees that all OpenGL commands made up to that point will complete executions in a finite amount time after calling this function.

* **glutPostRedisplay** ();

Using this function ensures that the display will be drawn only once each time the program goes through

the event loop.

* **glEnable** (GLenum feature);

This function enables an OpenGL feature. Features that can be enabled include GL\_DEPTH\_TEST which enables the hidden surface removal and GL\_LIGHTING which

is used to perform lighting calculations.

* **glViewPort** (int x, int y, GLsizei width, GLsizei height);

This specifies a width\*height viewport in pixels whose lower left corner is at (x,y) measured from the origin of the window

* **glOrtho** (GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far);

This function defines an orthographic viewing volume with all parameters measured from the center of the projection plane.

* **glutInit** (int \*argc, char \*\* argv);

It initializes GLUT. The arguments from the main are passed in and can be used by the application.

* **glutCreateWindow** (char \*title);

This creates a window on the display. The string title can be used to label the window.

* **glutInitDisplayMode** (unsigned int mode);

It requests the display with the properties in mode. The value of the mode is determined by the logical-OR of options including the color model (GLUT\_RGB,GLUT\_INDEX) and buffering(GLUT\_SINGLE,GLUT\_DOUBLE).

* **glutInitWindowPosition** (int x, int y);

It specifies the initial position of the top left corner of the window in pixels

* **glutInitWindowSize** (int width, int height);

It specifies the initial height and width of the window in pixels.

* **glutKeyboardFunc** (void \*f(char key, int width, int height));

It registers the keyboard callback function f. The callback function returns the ASCII code of the key pressed and the position of the mouse.

* **glutMouseFunc** (void \*f(int button, int state, int x, int y);

It registers the mouse callback function f. The callback function returns the button(GLUT\_LEFT\_BUTTON,GLUT\_RIGHT\_BUTTON,GLUT\_MIDDLE\_BUTTON),the state of the button after the event(GLUT\_UP,GLUT\_DOWN) and the position of the mouse relative to the top left corner of the window.

* **glutReshapeFunc** (void \*f(int width, int height));

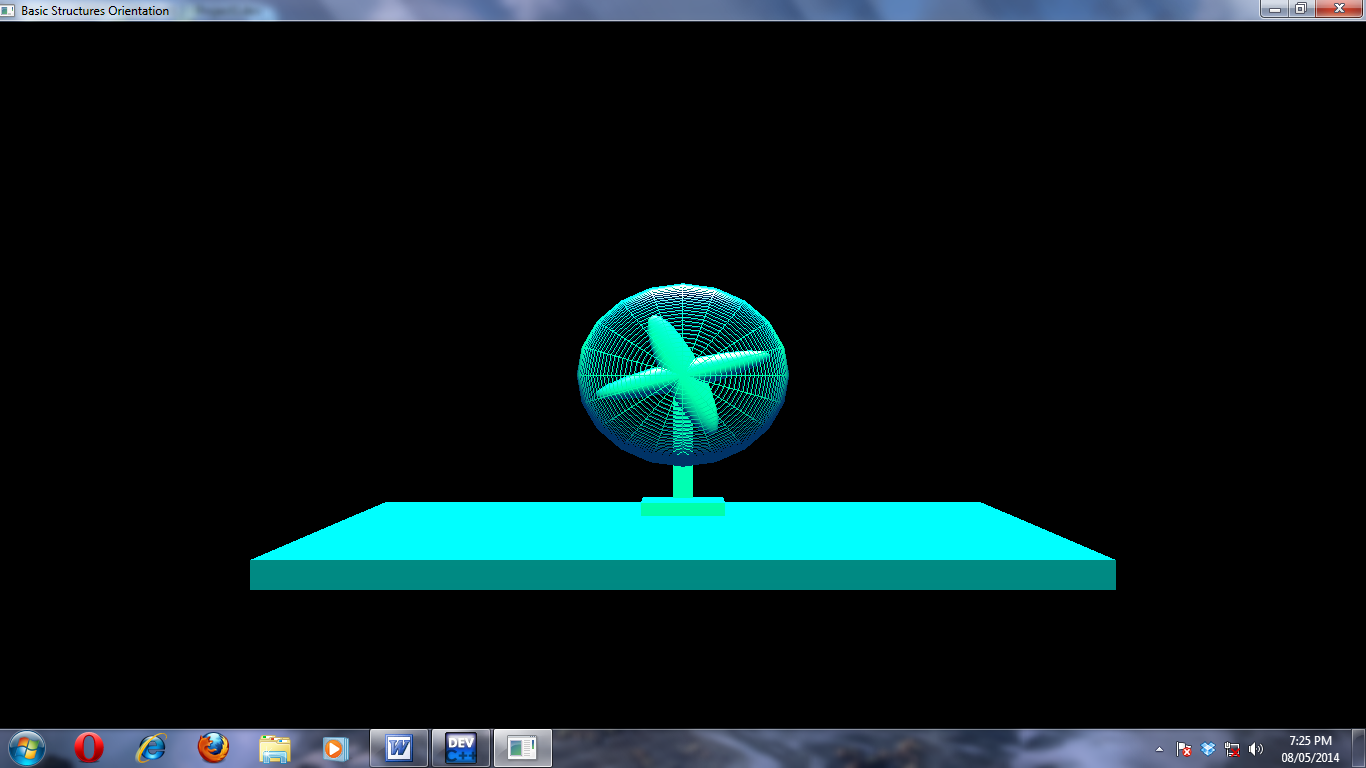
It registers the reshape callback function f. callback function returns the width and height of the new window.This function invokes the display callback.

* **glutDisplayFunc** (void (\*func) (void));

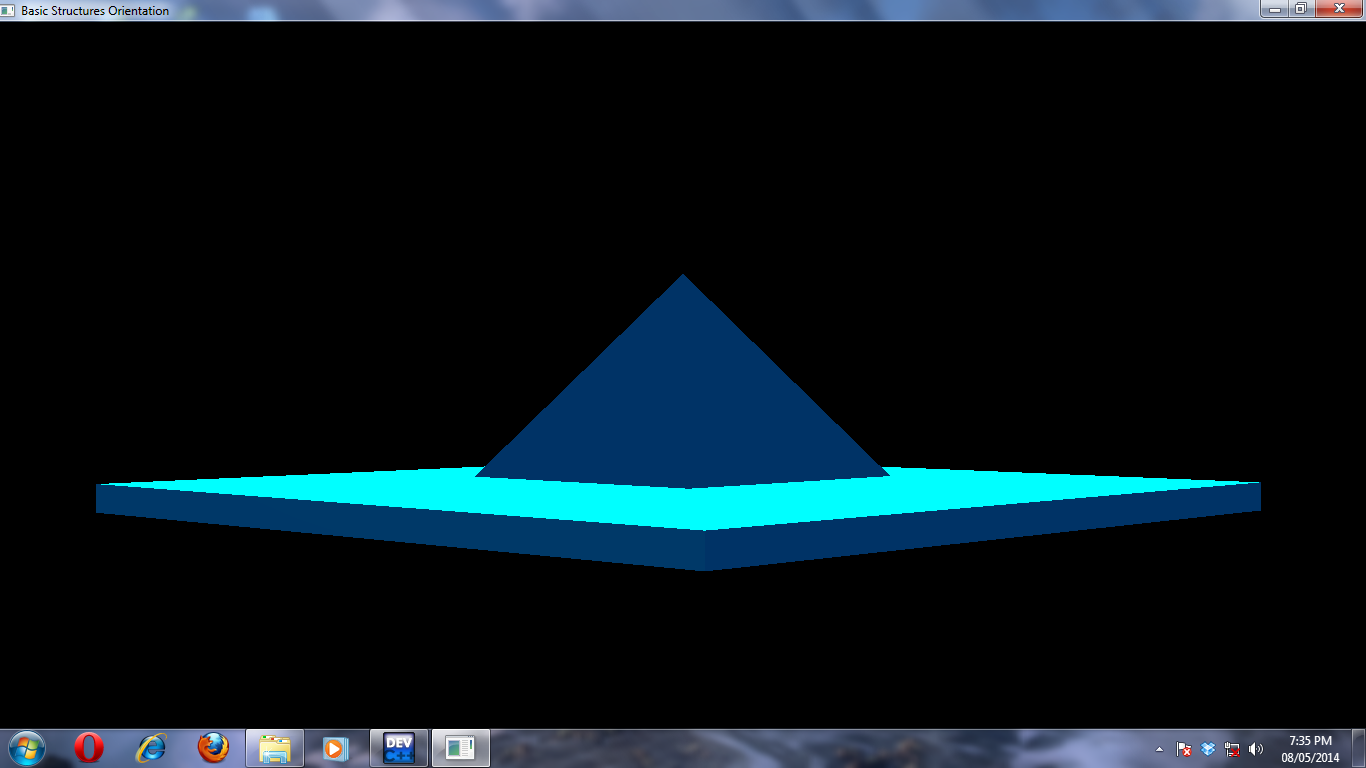
It registersdisplay function func that is executed when the window needs to be redrawn.

**5.SNAPSHOTS**

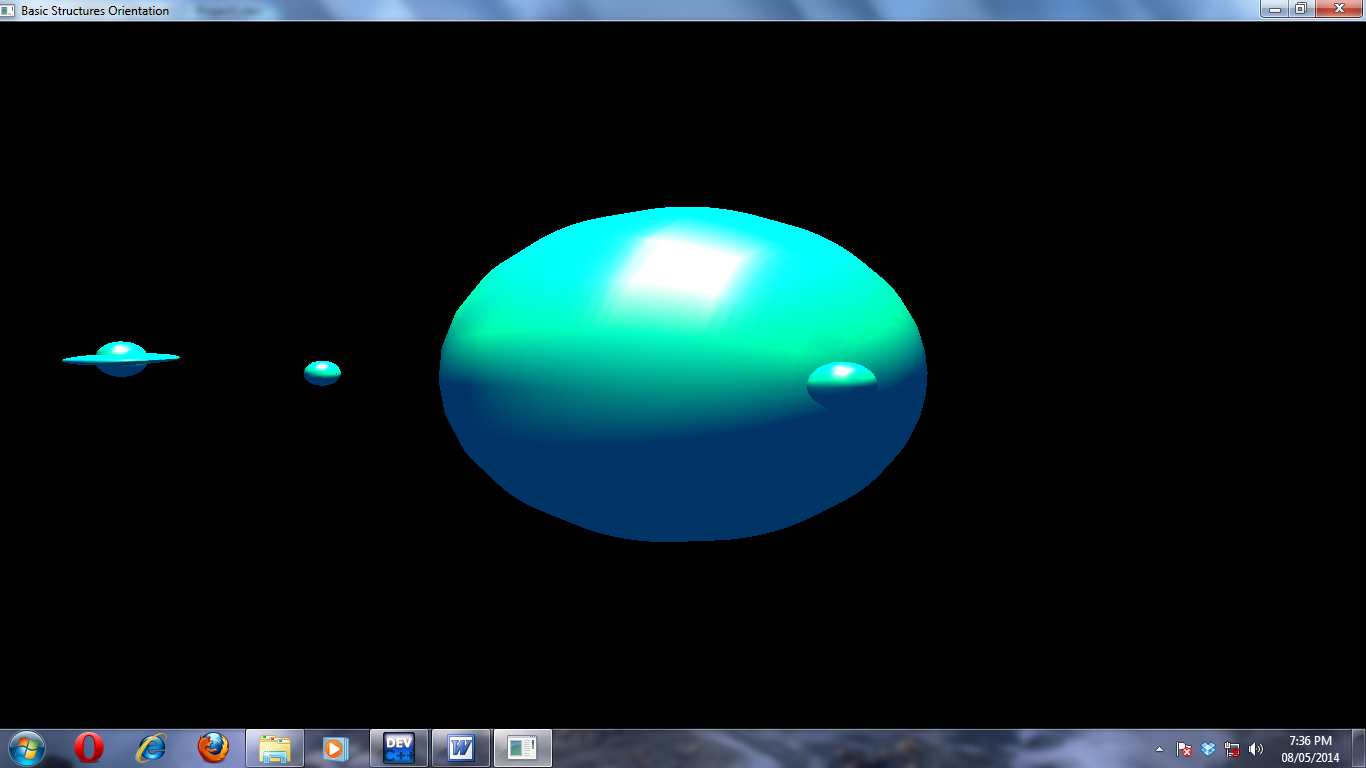
FIG 1: TABLE FAN

****

**FIG 2:PYRAMID**



**FIG 3:SOLAR SYSTEM**



**FIG 4:FLYING BALL**

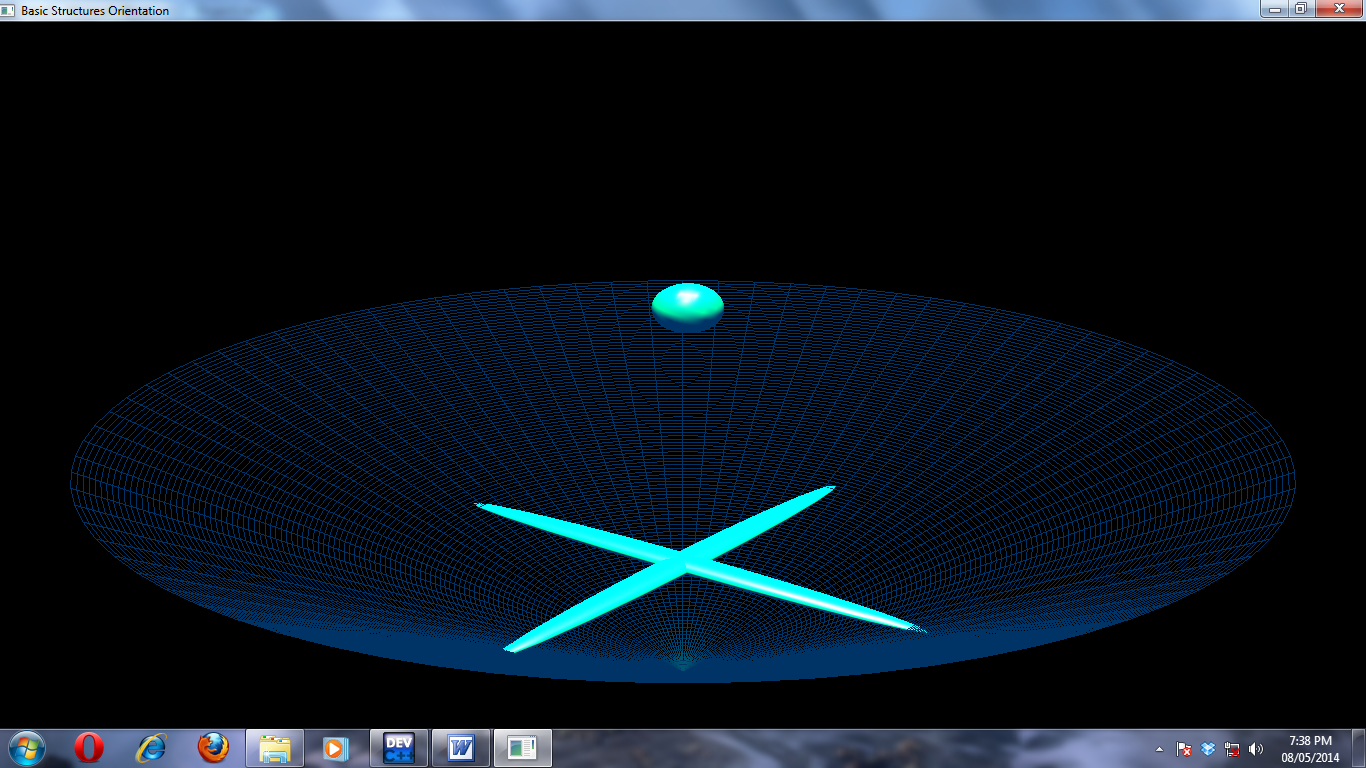
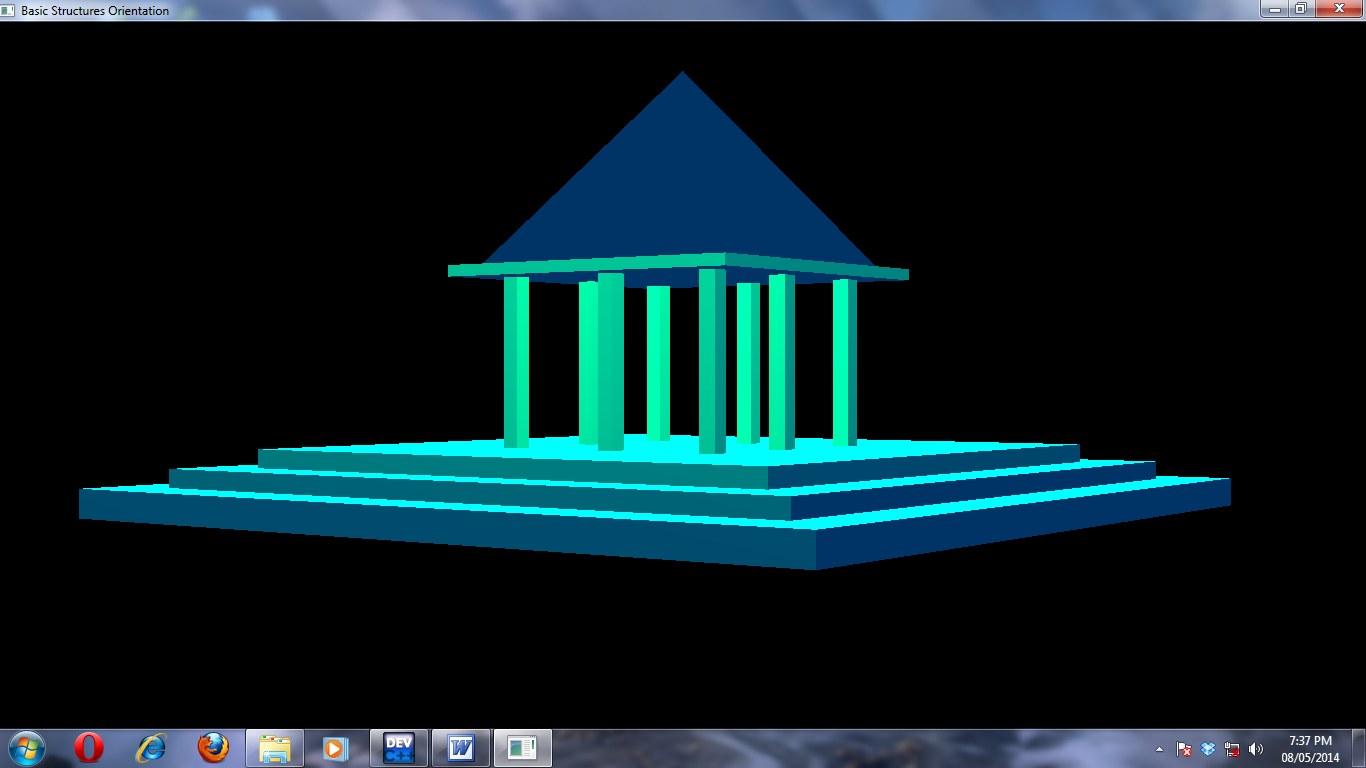


FIG 5:FAN



**FIG 6:TEMPLE**



**7.CONCLUSION**

* This project is as sincere effort at learning the development of a simple OpenGL software package, which is the building block of the current day high-level graphical applications. During the development of this package , a humble effort was made to understand the intricacies of the various functions and operations of the Glut library and their integration to develop a fair application. This project was good learning to platform to understand the basis of OpenGL, which will help in developing complex applications.

**8. BIBLIOGRAPHY**

* **“*Computer graphics” using OpenGL”-* F.S.HILL, Jr-2nd Edition, Pearson education , 2001.**
* **“*Interactive computer graphics”* A Top-Down Approach with OpenGL Edward Angel-2nd Edition, Addison – Wesley, 2000.**
* [**http://www.swiftless.com**](http://www.swiftless.com/)
* [**www.opengl.org**](http://www.opengl.org/)