UCS1712 – GRAPHICS AND MULTIMEDIA LAB

Assignment 09 - 3D Projections

Name:	Mahesh Bharadwaj K
Reg No:	185001089
Semester:	VII
Date:	October 27, 2021

---

1 Question 1

Aim:

Write a C++ program using OPENGL to perform 3D Projections – Orthographic and Perspective.

Algorithm

- Create a cpp file
- Import the libraries required for OPEN GL
- Initialise the display by setting the dimensions 640×480.
- Init 3D object
- Project using orthographic and perspective projections

Program

```
/*
 To demonstrate Orthographic Parallel and Perspective Projection using OpenGL
 and to also use keyboard functions and show different object views, along with
 setting the camera position.
*/

#include <iostream>
#include <cstring>
#include <GL/glut.h>
#include <math.h>

using namespace std;

//Global constants
const float WINDOW_WIDTH = 1000;
const float WINDOW_HEIGHT = 1000;
const float X_MIN = -500;
const float X_MAX = 500;
const float Y_MIN = -500;
const float Y_MAX = 500;
const int FPS = 60;

//Global variables to handle rotation
double x_rotate = 0;
double y_rotate = 0;
```

```
//Global variable for projection
bool isOrthoProjection = true;

void initializeDisplay();
void keyboardKeys(unsigned char key, int x, int y);
void drawAxes();

int main(int argc, char **argv)
{

    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(WINDOW_WIDTH, WINDOW_HEIGHT);
    glutCreateWindow("3D Projections");

    //Register the callback functions
    glutDisplayFunc(initializeDisplay);
    glutKeyboardFunc(keyboardKeys);

    //Change to projection mode before applying glOrtho()/gluPerspective()
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();

    glutMainLoop();

    return 0;
}

void initializeDisplay()
{
    //Initialize display parameters

    glClearColor(1, 1, 1, 1);
    glClear(GL_COLOR_BUFFER_BIT);

    //Translucency
    glEnable(GL_BLEND);
    glBlendFunc(GL_SRC_ALPHA, GL_ONE_MINUS_SRC_ALPHA);

    //Line width
    glLineWidth(3);

    //Apply the transformations & drawing on the model view matrix
    glMatrixMode(GL_MODELVIEW);

    //Draw the X and Y axis
    drawAxes();

    //Transform only the drawn object, so use the matrix stack accordingly
    glPushMatrix();

    if (isOrthoProjection)
    {
        //Parallel Projection
        glOrtho(-2, 2, -2, 2, -2, 2);
    }
    else
    {
        //Perspective Projection
        gluPerspective(120, 1, 0.1, 50); //FoVy = 120, Aspect Ratio = 1
```

```
    }

    gluLookAt(0, 0, 1, 0, 0, 0, 0, 1, 0); //Camera, Center & Up Vector
    glRotatef(x_rotate, 1, 0, 0);          //Keyboard based rotations
    glRotatef(y_rotate, 0, 1, 0);

    glColor4f(0, 0, 1, 0.3); //Draw the object
    glutWireTeapot(0.5);

    glPopMatrix(); //Pop the matrix back into the model view stack

    glFlush();
}

void drawAxes()
{
    //To draw X and Y axis

    glColor3d(1, 0, 0);

    glBegin(GL_LINES);

    glVertex2f(-2, 0);
    glVertex2f(2, 0);

    glVertex2f(0, -2);
    glVertex2f(0, 2);

    glEnd();
    glFlush();
}

void keyboardKeys(unsigned char key, int x, int y)
{
    //Callback function for keyboard interactivity

    key = tolower(key);

    switch (key)
    {
    case 'w':
    {
        x_rotate += 5;
        break;
    }
    case 's':
    {
        x_rotate -= 5;
        break;
    }
    case 'd':
    {
        y_rotate += 5;
        break;
    }
    case 'a':
    {
        y_rotate -= 5;
        break;
    }
    case 32:
```

```
    {
        //Spacebar for changing projections
        isOrthoProjection = !isOrthoProjection;
        break;
    }
    }

    //Update the display
    glutPostRedisplay();
}
```

Output

Figure 1: **Orthographic**

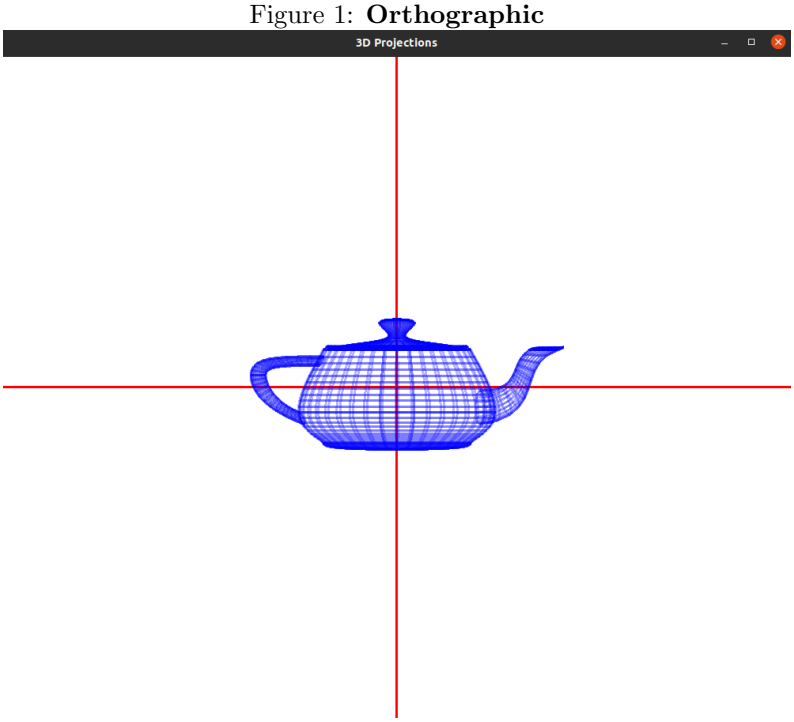
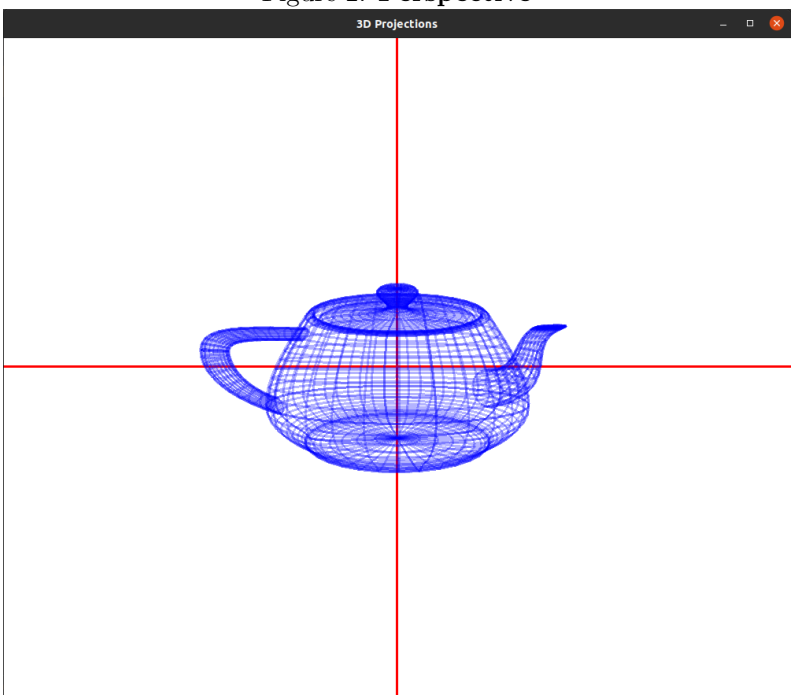


Figure 2: **Perspective**



Result

- OPENGL programs to perform 3D projections was designed and implemented successfully.