Ex. No: 9
Date : 03/11/2022

Karthik D
195001047

3-Dimensional Projections in C++ using OpenGL

Aim

- Write a menu driven program to perform Orthographic and Perspective projection on any 3D object.
- Set the camera to any position on the 3D space. Have (0,0,0) at the center of the screen. Draw X , Y and Z axes.
- Use gluPerspective() to perform perspective projection. Also, can use inbuilt functions for 3D transformations.

Algorithm

- Set the glMatrix mode to GL_Projection
- Use glOrtho() for orthographicl projection
- Use gluPerspective() for perspective projection
- Draw a wired cube in the display
- Set a flag to determine the projection when “space” button is pressed
- Use x_rotate and y_rotate values to rotate the cubes
- Use glRotatef() for to rotate the object according to key presses according to the key presses which include “a”, “w”, “s”, “d”
- Use glutKeyboardFunc() to register the callback function which set the rotate values according to key presses

Program

3dProjections.cpp

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

Ex. No: 9
Date : 03/11/2022

Name: Jagadish R
Reg. no: 195001039

```
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(4);

    //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)
```

Ex. No: 9
Date : 03/11/2022

Name: Jagadish R
Reg. no: 195001039

```
    {
        //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(1, 0, 0, 0.3); //Draw the object
    glutWireCube(1);

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

    glFlush();
}

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

    glColor3d(0, 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:
```

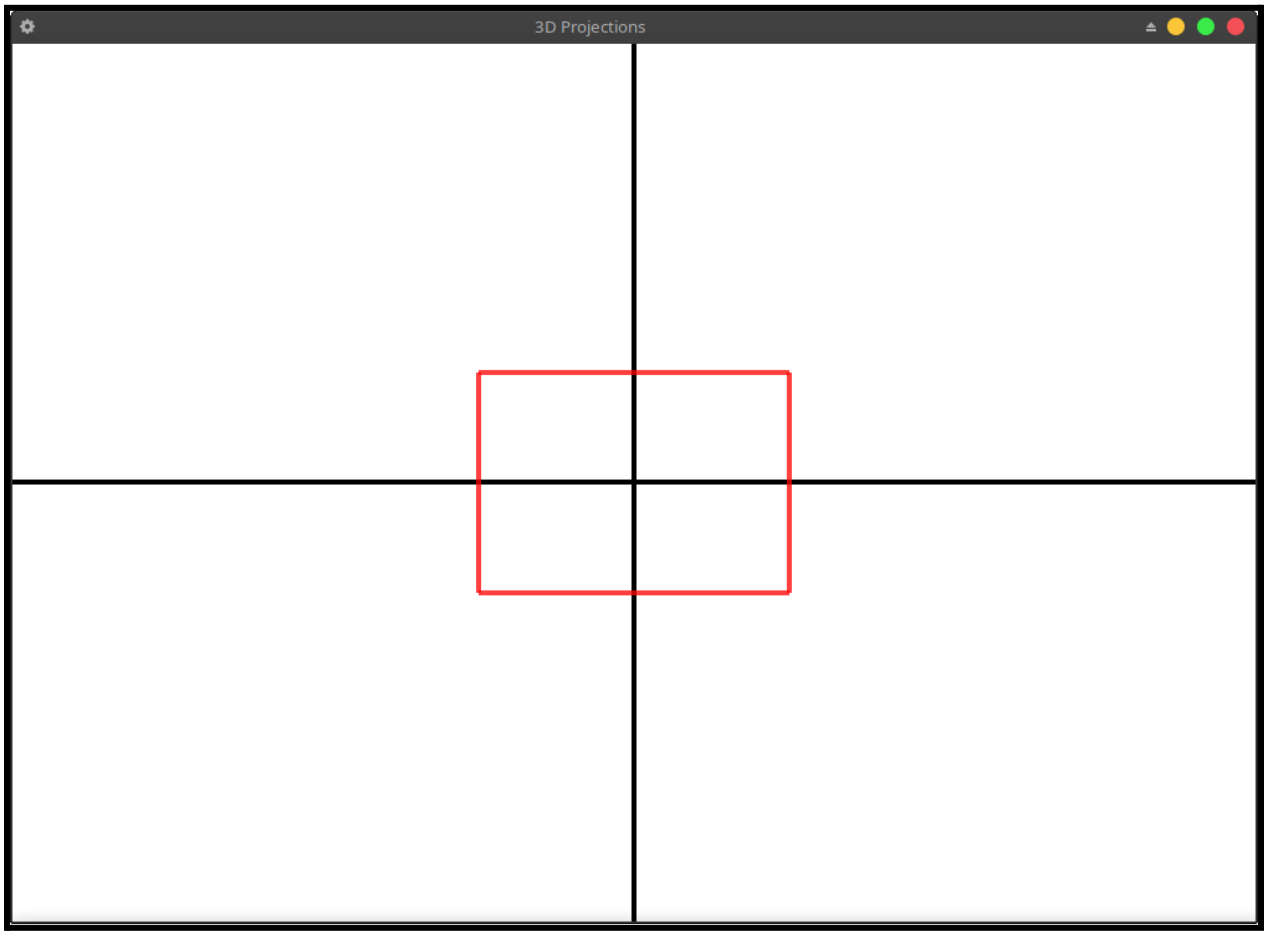
Ex. No: 9
Date : 03/11/2022

Name: Jagadish R
Reg. no: 195001039

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

Output

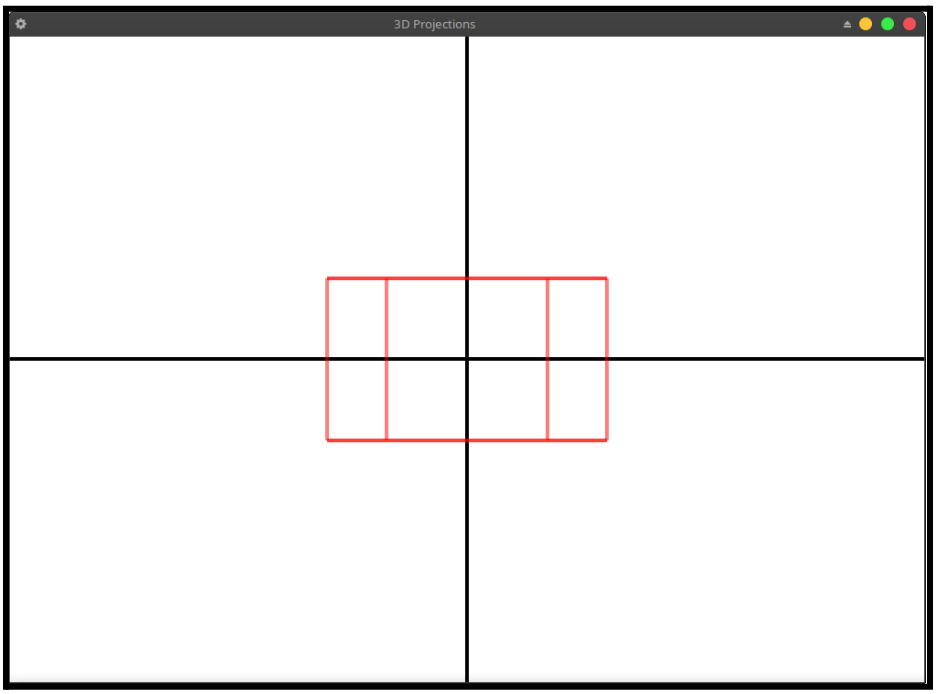
Orthographic Projection of Cube

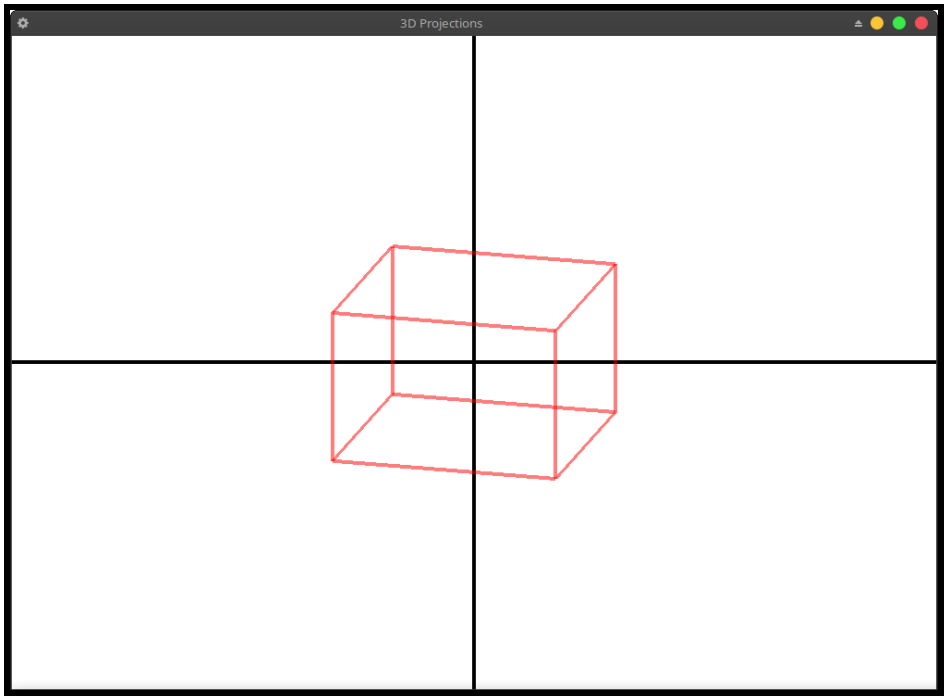


Ex. No: 9
Date : 03/11/2022

Name: Jagadish R
Reg. no: 195001039

Rotations

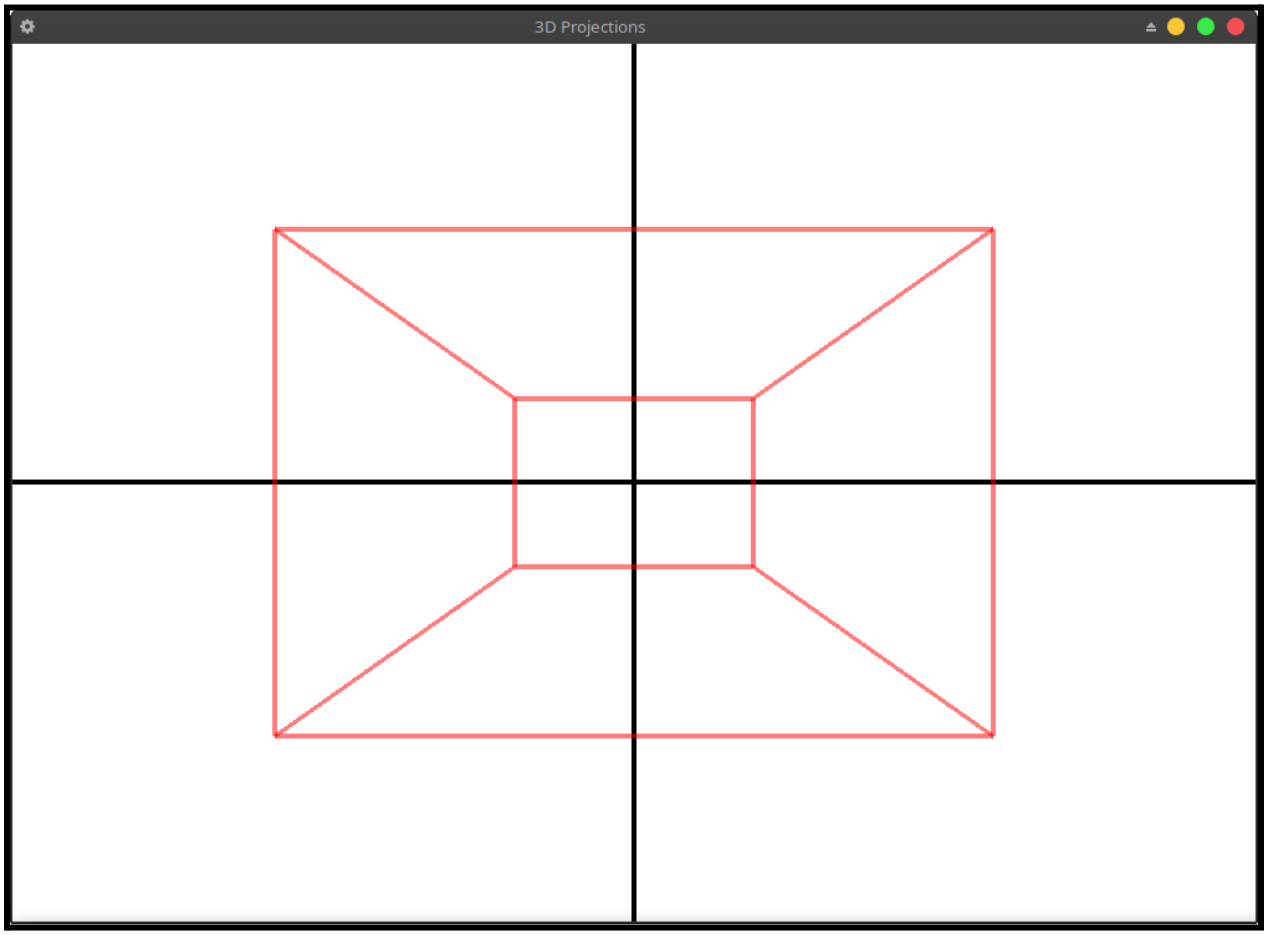




Ex. No: 9
Date : 03/11/2022

Name: Jagadish R
Reg. no: 195001039

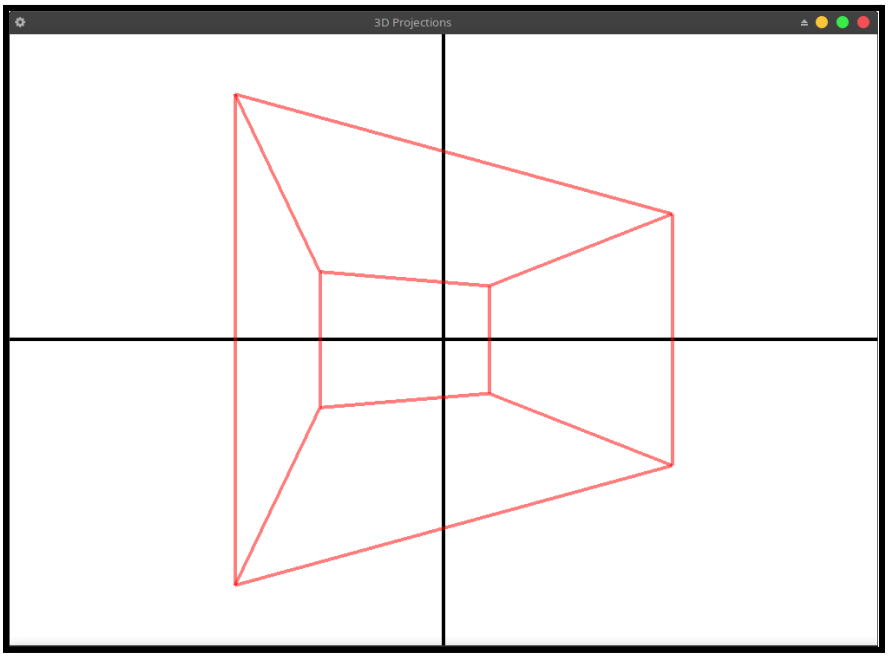
Perspective Projection of Cube

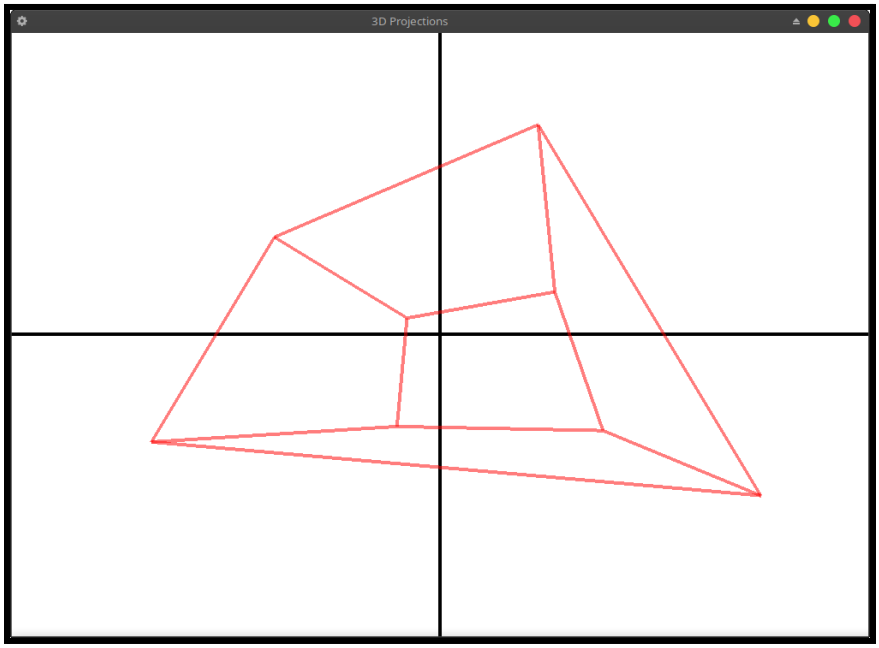


Ex. No: 9
Date : 03/11/2022

Name: Jagadish R
Reg. no: 195001039

Rotations





Ex. No: 9
Date : 03/11/2022

Name: Jagadish R
Reg. no: 195001039

Learning Outcomes

- Learned about the inbuilt functions for 3D objects and transformations
- Learned to perform orthographic and perspective projections for the 3d objects
- Learned about the keyboard events and their applications in the openGL