Experiment No: 9

Experiment Name:

Projection of 3D Cube

Objective:

To display a **3D cube** on a 2D screen using **projection techniques** in OpenGL, and to understand how 3D objects are represented in 2D space.

Theory:

In computer graphics, projection is the technique of converting 3D coordinates into 2D coordinates for visualization on a screen.

There are two main types of projection:

1. Orthographic Projection: Parallel projection lines; no perspective effect.
2. Perspective Projection: Objects farther away appear smaller; gives a realistic depth effect.

In this experiment:

- A 3D cube is drawn using vertices and quads.
- Perspective projection is used via `gluPerspective()` to show depth.
- Each face is colored for better visualization.
- Depth testing ensures proper visibility of faces.

Requirements:

Software:

- Code::Blocks /C++
- OpenGL, GLUT library

Procedure / Description of Code:

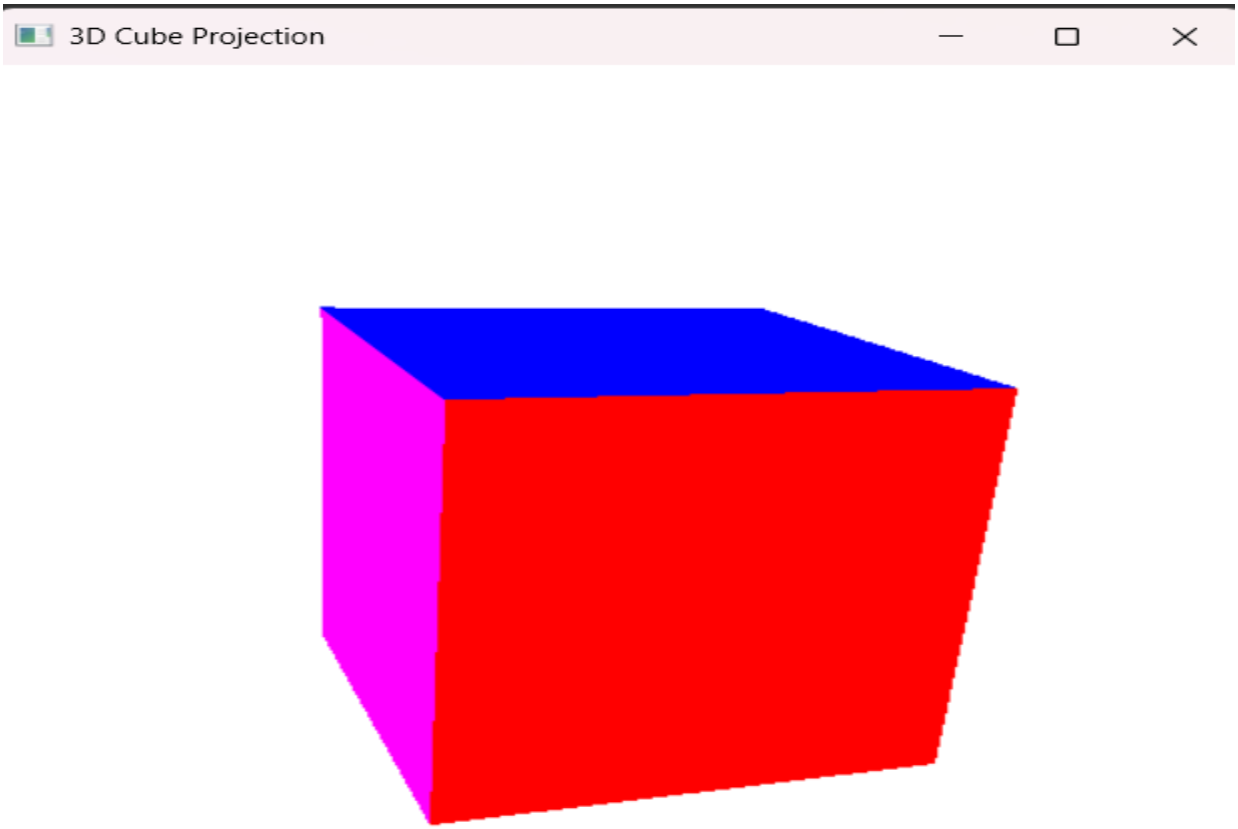
1. Initialize OpenGL and create a window using GLUT.
2. Enable depth testing to handle overlapping faces.
3. Set perspective projection using `gluPerspective()` for realistic 3D visualization.
4. Define the vertices of a cube and draw its 6 faces using `GL_QUADS`.
5. Assign different colors to each face for better visualization.
6. Translate and rotate the cube to make it viewable from an angle.
7. Display the cube using `glutSwapBuffers()` for double buffering.

SOURCE CODE:

```
#include <GL/glut.h>

void display()

{

   glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

   glLoadIdentity();

   glTranslatef(0.0f, 0.0f, -6.0f);


   // Rotate cube for better view

   glRotatef(30, 1, 1, 0);

   glBegin(GL_QUADS);


      // Front face

      glColor3f(1, 0, 0);

      glVertex3f(-1, -1,  1);

      glVertex3f( 1, -1,  1);

      glVertex3f( 1,  1,  1);

      glVertex3f(-1,  1,  1);


      // Back face
```

```
glColor3f(0, 1, 0);
glVertex3f(-1, -1, -1);
glVertex3f(-1,  1, -1);
glVertex3f( 1,  1, -1);
glVertex3f( 1, -1, -1)
// Top face
glColor3f(0, 0, 1);
glVertex3f(-1,  1, -1);
glVertex3f(-1,  1,  1);
glVertex3f( 1,  1,  1);
glVertex3f( 1,  1, -1);
// Bottom face
glColor3f(1, 1, 0);
glVertex3f(-1, -1, -1);
glVertex3f( 1, -1, -1);
glVertex3f( 1, -1,  1);
glVertex3f(-1, -1,  1);

// Right face
glColor3f(0, 1, 1);
glVertex3f( 1, -1, -1);
glVertex3f( 1,  1, -1);
glVertex3f( 1,  1,  1);
glVertex3f( 1, -1,  1);
// Left face
glColor3f(1, 0, 1);
glVertex3f(-1, -1, -1);
glVertex3f(-1, -1,  1);
glVertex3f(-1,  1,  1);
```

```
        glVertex3f(-1,  1, -1);
    glEnd();
    glutSwapBuffers();
}
void init()
{
    glClearColor(1, 1, 1, 1); // white background
    glEnable(GL_DEPTH_TEST);

    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();

    // Perspective projection
    gluPerspective(45.0, 1.0, 1.0, 100.0);

    glMatrixMode(GL_MODELVIEW);
}
int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
    glutInitWindowSize(500, 500);
    glutCreateWindow("3D Cube Projection");
    init();
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}
```



Conclusion / Discussion:

- This experiment demonstrates 3D to 2D projection in computer graphics.
- Perspective projection simulates real-world depth**,** making objects farther away appear smaller.
- Depth testing ensures correct visibility and overlapping of cube faces.
- OpenGL provides a simple way to draw 3D objects and project them onto 2D screens**.**