Lab Report

Course Code: CSE 422

Course Title: Computer Graphics Lab

Report No: 02

Title: Drawing a robot using different OpenGL functions.

Submitted To:

Syada Tasmia Alvi

Senior Lecturer

Department of Computer Science and Engineering (C.S.E)

Submitted by:

Name: Mahmudul Hasan Piash

ID: 221-15-5606

Section: 61_D1

Department of Computer Science and Engineering (C.S.E)

Date of submission: 20th October, 2025

Title

Drawing a robot using different OpenGL functions.

Introduction

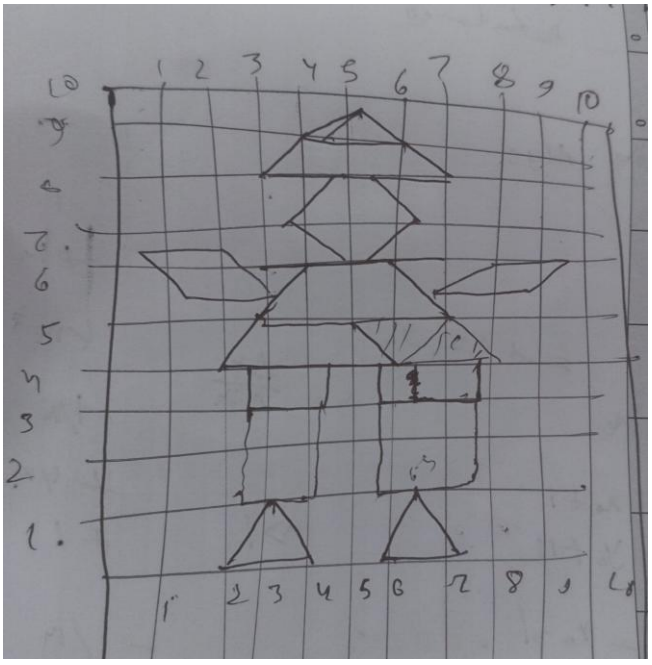
In this lab task, I used OpenGL with GLUT to draw a Robot shape using basic geometric primitives. The program demonstrates how to create 2D graphics by combining triangles and polygons with different colors. Through this task, we learned how to initialize an OpenGL window, set background and object colors, and use functions like glBegin(), glVertex3f(), and glFlush() to display objects on the screen. This experiment helps in understanding the fundamentals of computer graphics and how shapes are formed using coordinates in OpenGL.

Contents

In this lab task :

1. Functions used
   - ‘glClear()’ – clears the screen.
   - ‘glColor3f()’ – sets color.
   - ‘glBegin()’/ ‘glEnd()’ – start and end shape drawing.
   - ‘glVertex3f()’ – defines shape corners.
   - ‘glFlush()’ – displays the drawing.
2. Shapes used:
   - Triangles – rocket top and two wings.
   - Polygons (rectangles)– rocket body, window, and blast.

Graph



Code

```
#include <GL/gl.h>
#include <GL/glut.h>
void display(void)
{
/* clear all pixels */
glClear (GL_COLOR_BUFFER_BIT);
/* draw white polygon (rectangle) with corners at
* (0.25, 0.25, 0.0) and (0.75, 0.75, 0.0)
*/
    //left wing
    glColor3f (0, 1.0, 1.0);
    glBegin(GL_POLYGON);
    glVertex3f (.65, .55, 0);
    glVertex3f (.75, .65, 0);
    glVertex3f (.9, .65, 0);
    glVertex3f (.8, .55, 0);
    glEnd();


    //right wing
    glColor3f (0, 1.0, 1.0);
    glBegin(GL_POLYGON);
    glVertex3f (.15, .55, 0);
    glVertex3f (.05, .65, 0);
    glVertex3f (.20, .65, 0);
    glVertex3f (.3, .55, 0);
```

```
glEnd();


 //hat 1
 glColor3f (0, 0, 1.0);
glBegin(GL_TRIANGLES);
glVertex3f (.35, .9, 0);
glVertex3f (.475, 1.0, 0);
glVertex3f (.6, .9, 0);
glEnd();

 //hat 2
 glColor3f (0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f (.25, .8, 0);
glVertex3f (.35, .9, 0);
glVertex3f (.6, .9, 0);
glVertex3f (.7, .8, 0);
glEnd();

 //head
 glColor3f (1.0, 1.0, 1.0);
glBegin(GL_POLYGON);
glVertex3f (.38, .6, 0);
glVertex3f (.28, .7, 0);
glVertex3f (.38, .8, 0);
glVertex3f (.58, .8, 0);
```

```
    glVertex3f (.68, .7, 0);
    glVertex3f (.58, .6, 0);
    glEnd();

// belly
glColor3f (0, 1.0, 1.0);
    glBegin(GL_POLYGON);
    glVertex3f (.25, .5, 0);
    glVertex3f (.35, .6, 0);
    glVertex3f (.60, .6, 0);
    glVertex3f (.7, .5, 0);
    glEnd();
// left thigh
glColor3f (0, 0, 1.0);
    glBegin(GL_POLYGON);
    glVertex3f (.15, .4, 0);
    glVertex3f (.25, .5, 0);
    glVertex3f (.5, .5, 0);
    glVertex3f (.6, .4, 0);
    glEnd();
    //right thigh 1
    glColor3f(0.0, 1.0, 0.0);
    glBegin(GL_TRIANGLES);
    glVertex3f (.5, .5, 0);
    glVertex3f (.7, .5, 0);
    glVertex3f (.6, .4, 0);
    glEnd();
```

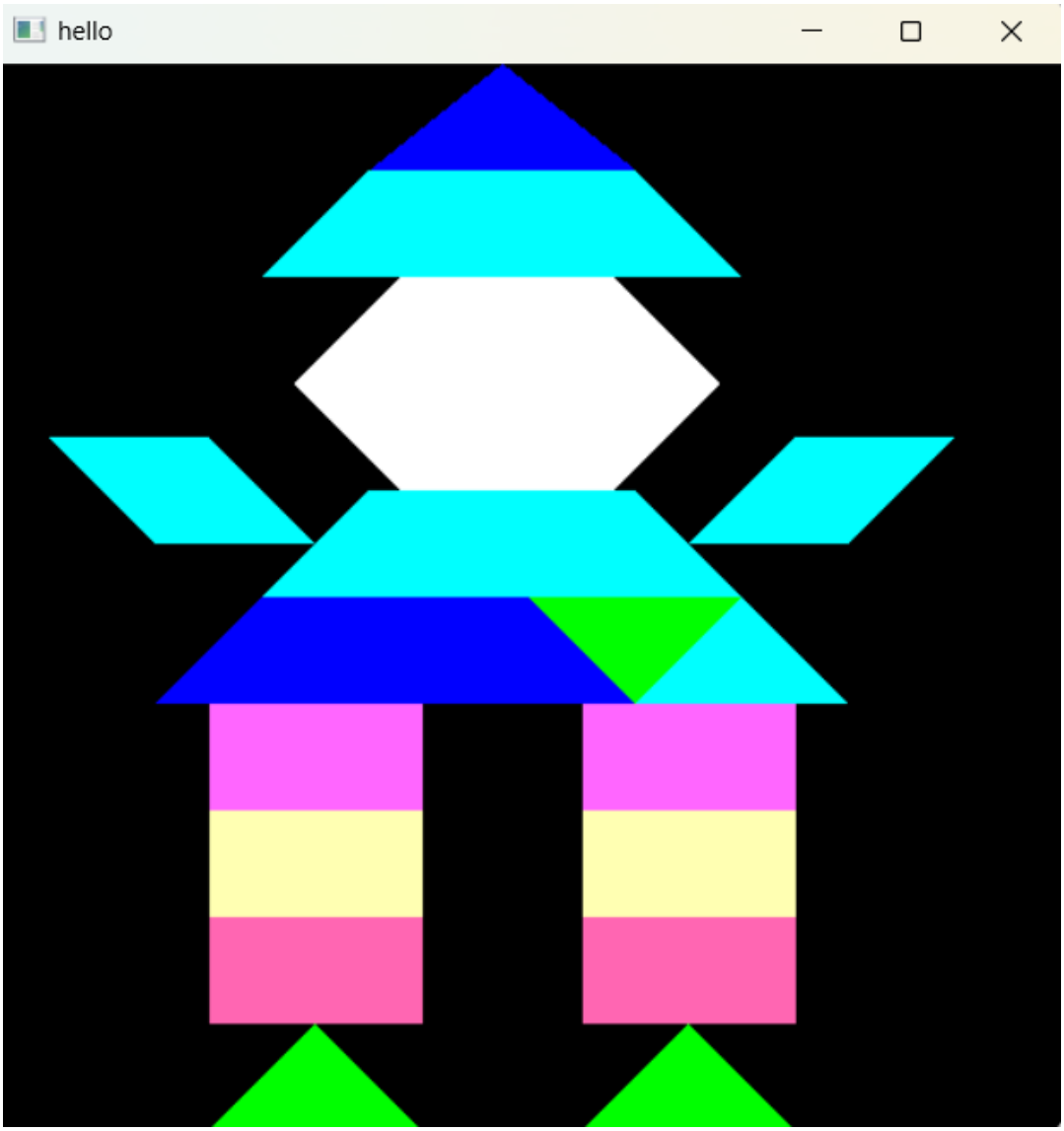
```
    glColor3f(0.0, 1.0, 3.0);
    glBegin(GL_TRIANGLES);
    glVertex3f (.6, .4, 0);
    glVertex3f (.7, .5, 0);
    glVertex3f (.8, .4, 0);
    glEnd();

// left leg
        glColor3f(1.0, 0.4, 0.7);
        glBegin(GL_POLYGON);
        glVertex3f (.2, .1, 0);
        glVertex3f (.2, .2, 0);
        glVertex3f (.4, .2, 0);
        glVertex3f (.4, .1, 0);
        glEnd();

        glColor3f(1.0, 1.4, 0.7);
        glBegin(GL_POLYGON);
        glVertex3f (.2, .2, 0);
        glVertex3f (.2, .3, 0);
        glVertex3f (.4, .3, 0);
        glVertex3f (.4, .2, 0);
        glEnd();

        glColor3f(1.0, 0.4, 1.7);
        glBegin(GL_POLYGON);
```

```
    glVertex3f (.2, .3, 0);
    glVertex3f (.2, .4, 0);
    glVertex3f (.4, .4, 0);
    glVertex3f (.4, .3, 0);
    glEnd();
//rigth leg

    glColor3f(1.0, 0.4, 0.7);
    glBegin(GL_POLYGON);
    glVertex3f (.55, .1, 0);
    glVertex3f (.55, .2, 0);
    glVertex3f (.75, .2, 0);
    glVertex3f (.75, .1, 0);
    glEnd();

    glColor3f(1.0, 1.4, 0.7);
    glBegin(GL_POLYGON);
    glVertex3f (.55, .2, 0);
    glVertex3f (.55, .3, 0);
    glVertex3f (.75, .3, 0);
    glVertex3f (.75, .2, 0);
    glEnd();

    glColor3f(1.0, 0.4, 1.7);
    glBegin(GL_POLYGON);
    glVertex3f (.55, .3, 0);
    glVertex3f (.55, .4, 0);
```

```
        glVertex3f (.75, .4, 0);
        glVertex3f (.75, .3, 0);
        glEnd();
    //left foot

    glColor3f(0.0, 1.0, 0.0);

    glBegin(GL_TRIANGLES);
    glVertex3f (.2, 0, 0);
    glVertex3f (.3, .1, 0);
    glVertex3f (.4, .0, 0);
    glEnd();
// right foot
    glColor3f(0.0, 1.0, 0.0);
    glBegin(GL_TRIANGLES);
    glVertex3f (.55, 0, 0);
    glVertex3f (.65, .1, 0);
    glVertex3f (.75, 0, 0);
    glEnd();



/* don't wait!
* start processing buffered OpenGL routines
*/
    glFlush ();
}
void init (void)
```

```
{
/* select clearing (background) color */
    glClearColor (0.0, 0.0, 0.0, 0.0);
    /* initialize viewing values */
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(0.0, 1.0, 0.0, 1.0, -1.0, 1.0);
}
/*
* Declare initial window size, position, and display mode
* (single buffer and RGBA). Open window with "hello"
* in its title bar. Call initialization routines.
* Register callback function to display graphics.
* Enter main loop and process events.
*/
int main(int argc, char** argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize (500, 500);
    glutInitWindowPosition (100, 100);
    glutCreateWindow ("hello");
    init ();
    glutDisplayFunc(display);
    glutMainLoop();
    return 0; /* ISO C requires main to return int. */
}
```

Output



Discussion

In this lab task, I successfully created a robot using basic OpenGL functions. The rocket was drawn by combining simple shapes such as triangles for the nose and wings, and polygons for the body, window, and burst. Each part was given a different color using glColor3f() to make the figure more visually clear. The program used glBegin() and glEnd() to define shapes, while glVertex3f() specified their coordinates. Overall, this lab demonstrated how OpenGL can be used to design graphical objects using coordinating geometry & color control.