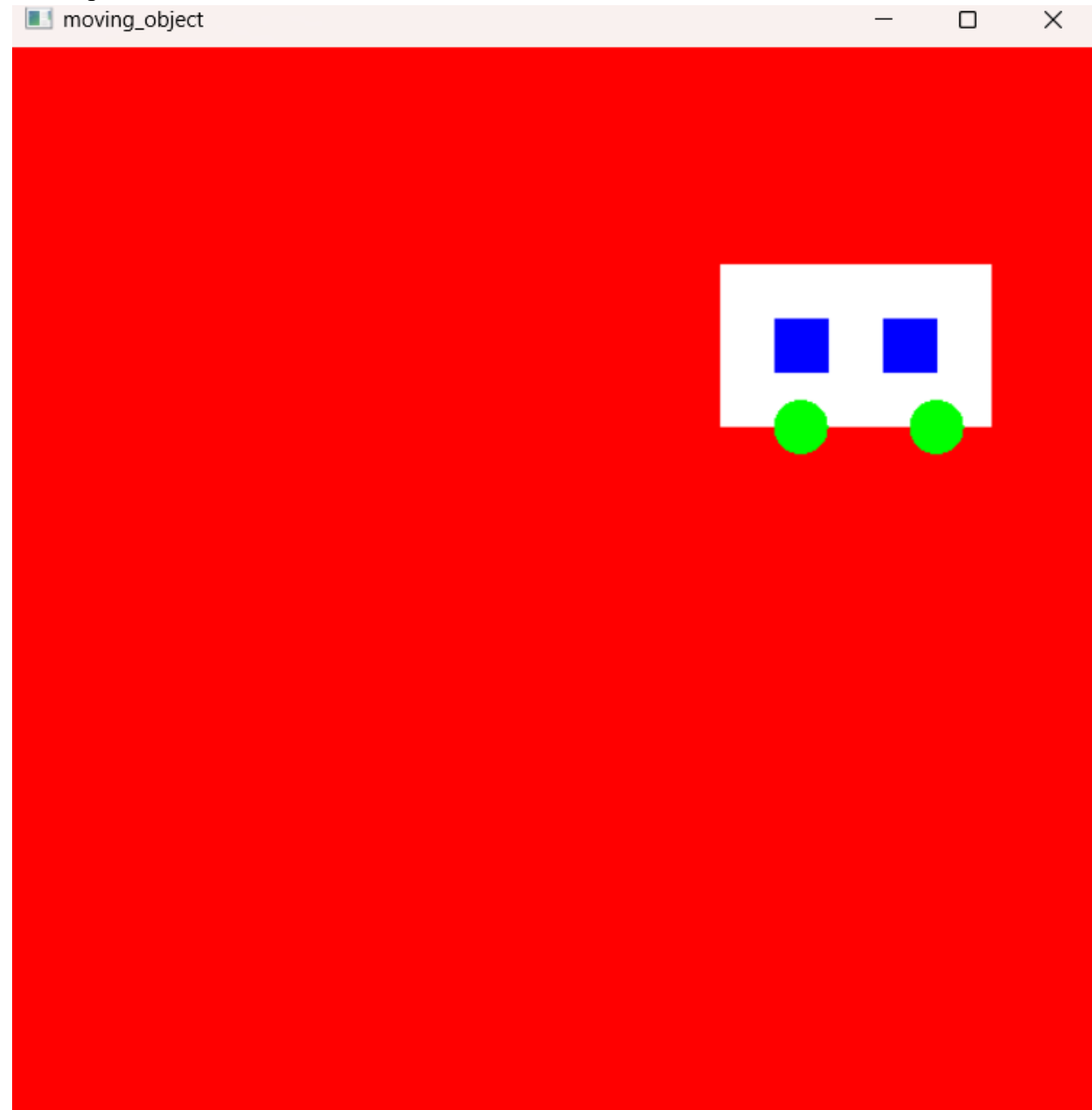


Title

2d Shape



movement with translation.

Introduction

In this lab task, I used OpenGL with GLUT to draw a Scenery with a tree, sun, cloud and traffic light shape using basic geometric primitives. The program demonstrates how to create 2D graphics by combining Circles 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.
- 'glVertex2f()' – defines shape corners.
- 'glFlush()' – displays the drawing.

2. Shapes used:

- Circle – Bus wheels.
- Polygons (rectangles) – bus windows.
- Quads (rectangles) – bus body

Code

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

#include<windows.h>

#ifdef APPLE

#else

#include <GL/glut.h>

#endif

#include <stdlib.h>

#include <math.h>

float p=6.0;

void drawCircle(int h, int k, int rx, int ry)

{

    glColor3f(0.0, 1.0, 0.0);

    glBegin(GL_POLYGON);

    for (int i = 0; i <= 360; i++) {

        float angle = 3.14159 * i / 180.0;
```

```

        glVertex2f(h + rx * cos(angle), k + ry * sin(angle));
    }
    glEnd();
}

void display(void)
{
    glClear(GL_COLOR_BUFFER_BIT);

    if(p<=10)
        p=p-.0005;
    else
        p=6;

    glutPostRedisplay();
    glBegin(GL_QUADS);
    glVertex2f(p-1,3);
    glVertex2f(p-1,6);
    glVertex2f(p+4,6);
    glVertex2f(p+4,3);
    glEnd();

    glColor3f(0.0, 1.0, 0.0);
    glBegin(GL_POLYGON);
    for (int i = 0; i <= 360; i++) {

        float angle = 3.14159 * i / 180.0;

        glVertex2f(p+.5 + 0.5 * cos(angle), 3 + 0.5 * sin(angle));
    }
    glEnd();

    //drawCircle(p+3, 3, 1, 1);

```

```

    glBegin(GL_POLYGON);
    for (int i = 0; i <= 360; i++) {
        float angle = 3.14159 * i / 180.0;
        glVertex2f(p+3 + 0.5 * cos(angle), 3 + 0.5 * sin(angle));
    }
    glEnd();
glColor3f(0.0, 0.0, 1.0);
glBegin(GL_QUADS);
    glVertex2f(p,4);
    glVertex2f(p,5);
    glVertex2f(p+1,5);
    glVertex2f(p+1,4);
    glEnd();
glColor3f(0.0, 0.0, 1.0);
glBegin(GL_QUADS);
    glVertex2f(p+2,4);
    glVertex2f(p+2,5);
    glVertex2f(p+3,5);
    glVertex2f(p+3,4);
    glEnd();
glColor3f(1.0, 1.0, 1.0);
glFlush();
}

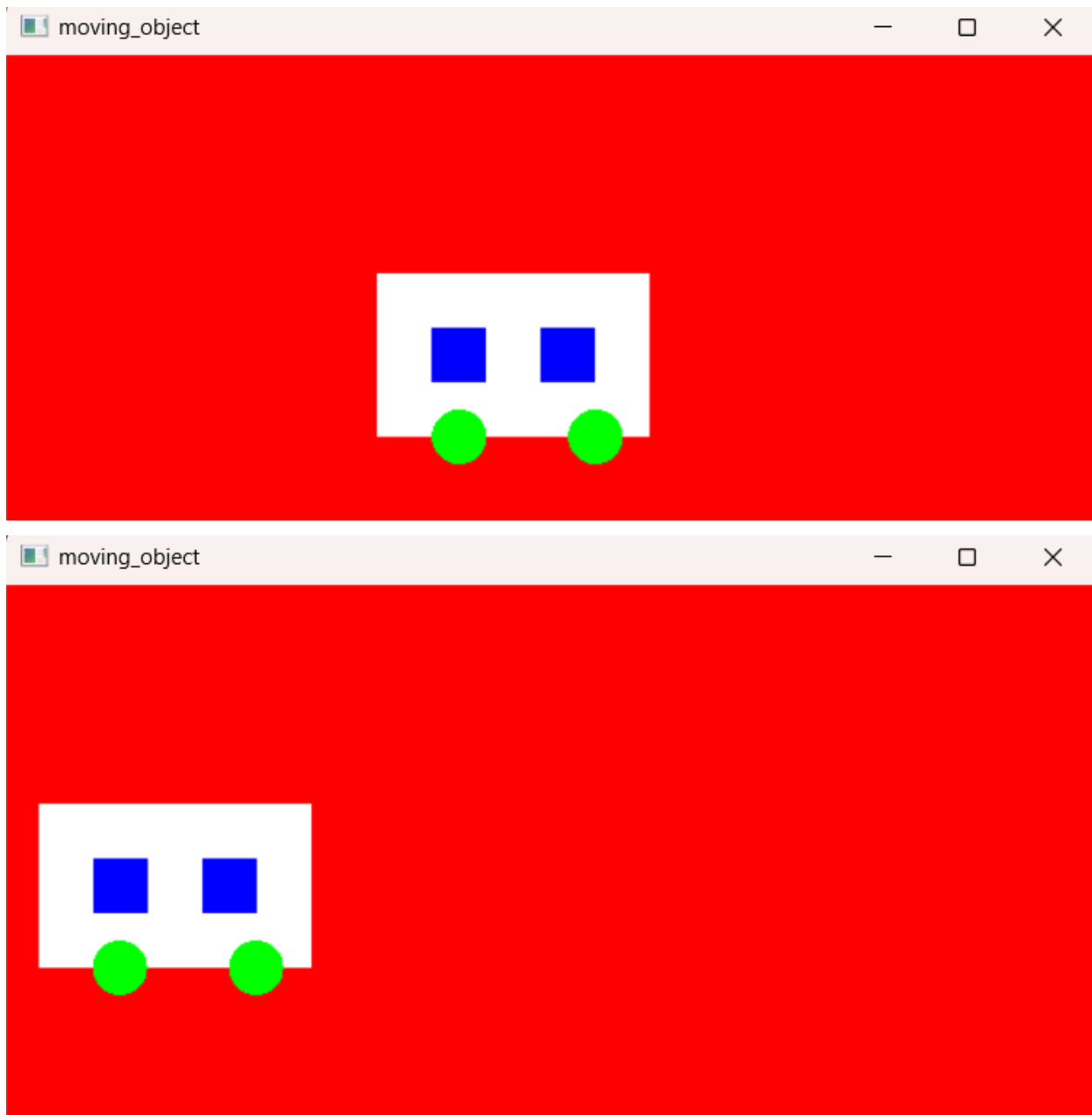
void init(void)
{
    glClearColor (1.0, 0.0, 0.0, 0.0);
    glOrtho(-10.00,10.00,-10.00,10.00,-10.00,10.00);

```

```
}  
  
int main(int argc, char** argv)  
{  
    glutInit(&argc, argv);  
    glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);  
    glutInitWindowSize (600, 600);  
    glutInitWindowPosition (100, 100);  
    glutCreateWindow ("moving_object");  
    init();  
    glutDisplayFunc(display);  
    glutMainLoop();  
    return 0;  
}
```

Output





Discussion

In this lab task, I successfully created a robot using basic OpenGL functions. The Scenery was drawn by combining simple shapes such as circles and rectangles for the traffic lights, cloud, sun, tree leaves and polygons for the tree body, traffic light body. 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.