Home Task

CSE422: Computer Graphics Lab

Submitted To

Syada Tasmia Alvi

Senior Lecturer

Department of CSE

Daffodil International University

Submitted By

Student ID: 221-15-5277

Student Name: Arnob Das Shacha

Section: 61_D1

Department of CSE

Daffodil International University

Title:

Drawing a House, Cloud and Sun Using Bresenham Line Drawing Algorithm in OpenGL.

Introduction:

In this project, I have implemented the Bresenham Line Drawing Algorithm in OpenGL to draw a simple house scene. The scene includes a house structure, a door, two windows, a roof, a sun, and a cloud. Each element of the scene is constructed using straight lines generated through the Bresenham algorithm. The purpose of this project is to understand and practice computer graphics line drawing algorithms and how to apply them to form complex shapes.

Content:

In this project I have used the following functions and shapes:

Functions

- **bresenham()** – implements Bresenham's line drawing algorithm to draw pixels between two endpoints.
- **display()** – contains all drawing instructions and graphical objects.
- **init()** – initializes background color and coordinate system.
- **main()** – creates window and connects display function.

Shapes Used

- **Rectangle** → used for house walls and windows
- **Triangle** → used for house roof and sun rays
- **Straight lines** → used to form clouds and all structural edges

Code:

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

#include <math.h>


void bresenham(int x1,int y1,int x2,int y2)

{

   int dx = abs(x2 - x1);
```

```
int dy = abs(y2 - y1);

int s1 = (x2 - x1) >= 0 ? 1 : -1;
int s2 = (y2 - y1) >= 0 ? 1 : -1;

int swap = 0;
if(dy > dx){
   int temp = dx;
   dx = dy;
   dy = temp;
   swap = 1;
}

int p = 2*dy - dx;

int x = x1;
int y = y1;

glBegin(GL_POINTS);
for(int i = 0; i <= dx; i++)
{
   glVertex2i(x,y);

   if(p >= 0){
      if(swap) x += s1;
      else y += s2;
      p -= 2*dx;
```

```
        }
        if(swap) y += s2;
        else x += s1;

        p += 2*dy;
    }
    glEnd();
}

void display()
{
    glClear(GL_COLOR_BUFFER_BIT);

    glPointSize(3);

    // HOUSE OUTLINE
    glColor3f(1,1,1);

    // base rectangle
    bresenham(150,150,350,150);
    bresenham(350,150,350,300);
    bresenham(350,300,150,300);
    bresenham(150,300,150,150);

    // roof
    bresenham(150,300,250,370);
    bresenham(250,370,350,300);
```

```
//  Door
glColor3f(0.5,0.3,0.0);
bresenham(230,150,230,240);
bresenham(270,150,270,240);
bresenham(230,240,270,240);

//  Windows
glColor3f(0.2,0.6,1.0);

// left window
bresenham(165,230,205,230);
bresenham(165,230,165,260);
bresenham(205,230,205,260);
bresenham(165,260,205,260);

// right window
bresenham(295,230,335,230);
bresenham(295,230,295,260);
bresenham(335,230,335,260);
bresenham(295,260,335,260);

//  SUN
glColor3f(1.0,0.8,0.0);
bresenham(400,400,430,400);
bresenham(400,400,415,430);
bresenham(415,430,430,400);
```

```
    // CLOUD
    glColor3f(0.8,0.8,0.8);
    bresenham(60,380,110,380);
    bresenham(90,400,130,400);
    bresenham(70,410,120,410);

    glFlush();
}

void init()
{
    glClearColor(0,0,0,0);
    gluOrtho2D(0,500,0,500);
}

int main(int argc,char** argv)
{
    glutInit(&argc,argv);
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
    glutInitWindowSize(500,500);
    glutCreateWindow("House Using Bresenham");
    init();
    glutDisplayFunc(display);
    glutMainLoop();
}
```

Output:

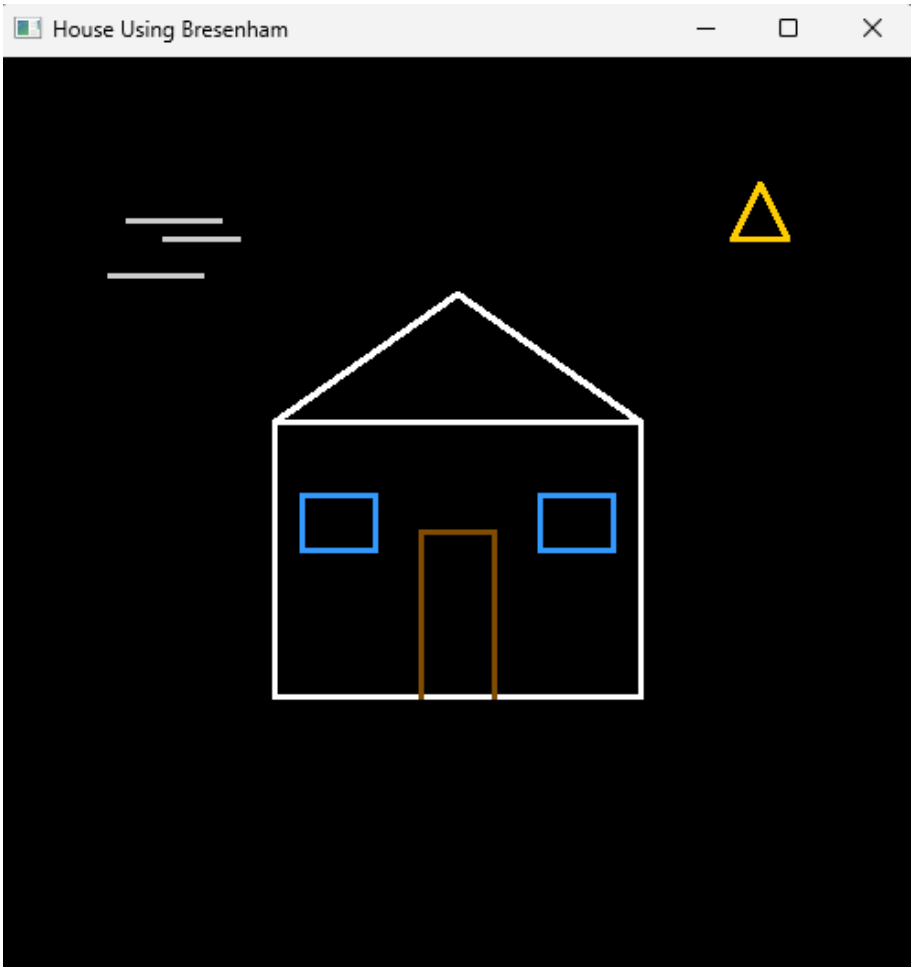

Fig: House, clouds and sun

Discussion:

In this task, the main objective was to use the Bresenham Line Drawing Algorithm instead of OpenGL in-built line primitives.

By calculating the pixel points step-by-step using decision parameters, I drew multiple connected lines to form:

- the rectangular base of the house
- the slanted roof
- a door in the center
- two windows
- a sun in the top corner
- a floating cloud shape

Color functions were applied using `glColor3f()` so the scene looks more visually attractive. All coordinates were manually defined so that each part fits properly in the overall structure. This project helped me understand how individual straight lines combine to form a complete figure.

Conclusion:

This project successfully demonstrates the use of the Bresenham Line Drawing Algorithm to create a complete graphical scene in OpenGL. By combining multiple lines, I was able to construct a house, sun, and cloud. Through this implementation, I gained practical experience in computer graphics rendering, coordinate design, and pixel-level drawing using an efficient line generation algorithm.