CSO351: Computer Graphics

Lab Assignment 1.(c):
Dashed Line Generation using Breshenham's Algorithm

Objective:

Write a program in C/C++ for implementation of dashed line generation using Breshenham's algorithm.

Algorithm:

- **Step 1:** Get the input of two end points (x_{beg}, y_{beg}), (x_{end}, y_{end}) and lengths of dash (dlen) and space (slen) of the required line.

- **Step 2:** Calculate the following which would be useful to produce the spaces between the dashes:

  $\text{theta} = \tan^{-1}[(y_{end} - y_{beg}) / (x_{end} - x_{beg})]$
  `s = sin(theta)`
  `c = cos(theta)`

- **Step 3:** Intialize the starting point of the first dash as:

  `x1` = x_{beg}
  `y1` = y_{beg}

- **Step 4:** Calculate the end point (x2, y2) of the current dash using dlen:

  ```
  x2 = x1 + dlen*c
  y2 = y1 + dlen*s
  ```

- **Step 5:** Calculate the difference between two end points of the current dash:

  ```
  dx = x2 - x1
  dy = y2 - y1
  ```

- **Step 6:** Calculate initial decision parameter:

  ```
  param = 2*dy - dx;
  ```

- **Step 7:** Calculate for x_{max} as:

```
if(xa > xb)
{     x = xb;
      y = yb;
      xmax = xa;
}
 else
{     x = xa;
      y = ya;
      xmax = xb;
  }
  plot(x, y);
```

- **Step 8:** Repeat the following until $x < x_{max}$:

```
x = x + 1;
if(param < 0)
      param += 2*dy;
else
{     y = y + 1;
      param += 2*(dy - dx);
}
plot(x, y);
```

- **Step 9:** Calculate the starting point (x1, y1) of new dash using slen:

```
x1 = x2 + slen*c
y1 = y2 + slen*s
```

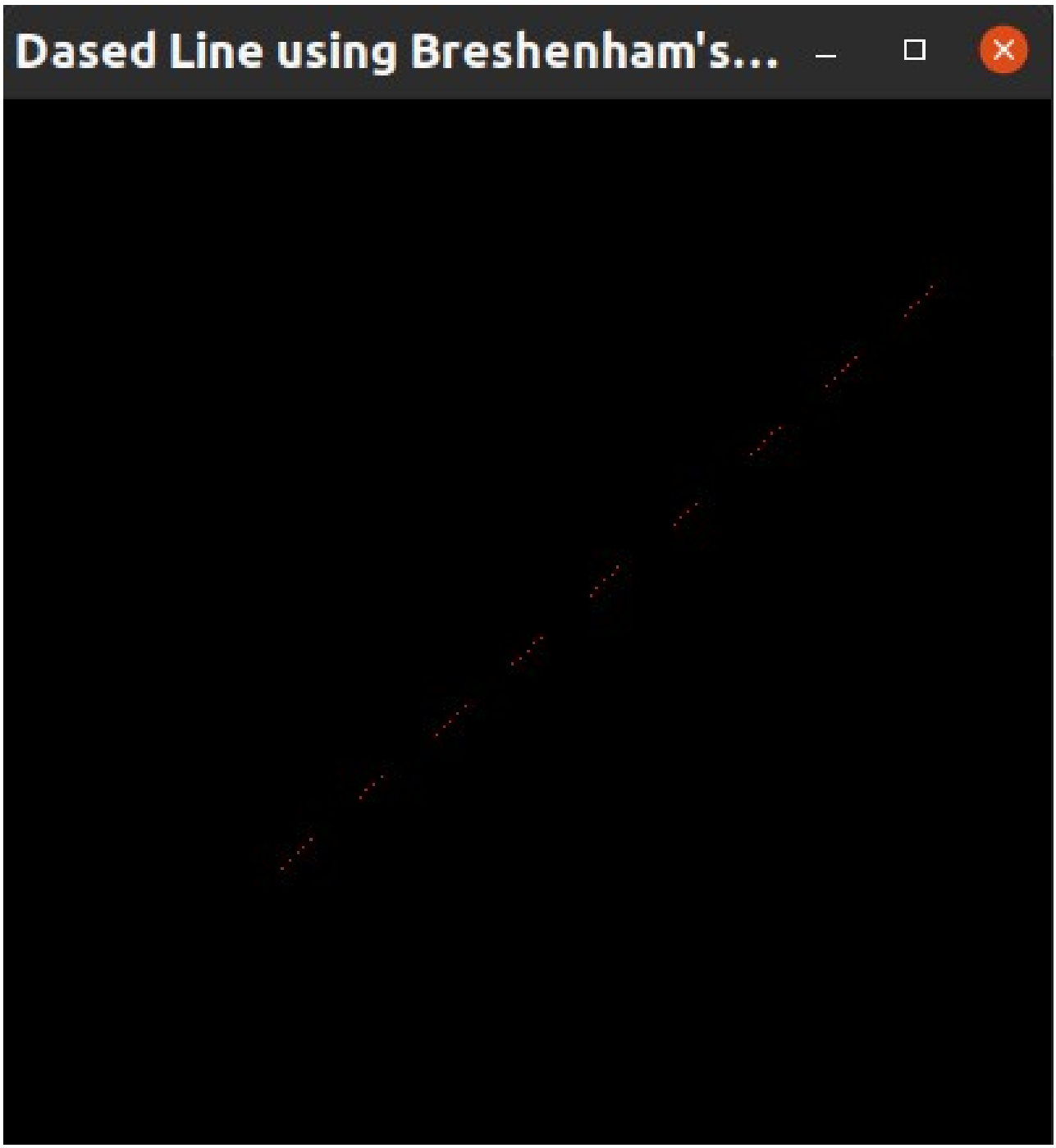
- **Step 10:** Repeat steps 4-9 until $x1 < x_{end}$ and $y1 < y_{end}$.

Result:

Input:

```
swaraj@shiv-raj-75:~/Documents/Assignments/Sem5/CG$ ./1.iii
Enter x1 and y1 : -10 -10
Enter x2 and y2 : 80 70
Enter lengths of dash and space: 5 10
```

Output:



Conclusion:

- In Breshenham's line generation algorithm, next pixel calculated is that one which has the least distance from true line, i.e. more accurate than DDA.
- It involves only integer arithmetic, i.e. easy and fast.
- With the help sine and cosine of $\tan^{-1}$ (slope) i.e. theta dashes and spaces are generated.

Appendix: Code

```
#include<GL/glut.h>
#include<stdio.h>
#include<stdlib.h>
#include<math.h>

GLint xbeg, ybeg, xend, yend, dlen, slen;

void init(void)
{
     glClearColor(0.0, 0.0, 0.0, 0.0);
     glMatrixMode(GL_PROJECTION);
     glLoadIdentity();
     gluOrtho2D(-50, 100, -50, 100);
}

void display (int xa, int ya, int xb, int yb)
{
     GLint dx = abs (xb - xa),
          dy = abs (yb - ya),
          param = 2 * dy - dx,
          x, y, xMax;
     GLfloat p[2];
     if (xa > xb)
     {
          x = xb;
          y = yb;
          xMax = xa;
     }
     else
     {
          x = xa;
          y = ya;
          xMax = xb;
     }
     p[0] = x; p[1] = y;
     glVertex2fv (p);
     while (x < xMax)
     {
          x++;
```

```
            if (param < 0)
                param += 2 * dy;
            else
            {
                y++;
                param += 2 * (dy - dx);
            }
            p[0] = x;p[1] = y;
            glVertex2fv (p);
        }
}

void showGraphic (void)
{
        glClear (GL_COLOR_BUFFER_BIT);
        glColor3f(1.0, 0.0, 0.0);
        glBegin (GL_POINTS);
        GLdouble x1, y1, x2, y2,
            theta = atan2 (yend - ybeg, xend - xbeg),
            s = sin (theta),
            c = cos (theta);
        x1 = xbeg;
        y1 = ybeg;

        while (x1 < xend && y1 < yend)
        {
            x2 = x1 + dlen * c;
            y2 = y1 + dlen * s;
            display (x1, y1, x2, y2);
            x1 = x2 + slen * c;
            y1 = y2 + slen * s;
        }
        glEnd ();
        glFlush ();
}

int main(int argc, char **argv)
{
        GLfloat slope;
        do {
            printf("Enter x1 and y1 : ");
            scanf("%d %d", &xbeg, &ybeg);
            printf("Enter x2 and y2 : ");
            scanf("%d %d", &xend, &yend);
            slope = (yend - ybeg) / (xend - xbeg);
        } while (slope < 0 || slope > 1);
        printf("Enter lengths of dash and space: ");
        scanf("%d %d", &dlen, &slen);
        glutInit (&argc, argv);
        glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
        glutInitWindowSize (400, 400);
        glutInitWindowPosition (0, 0);
        glutCreateWindow ("Dased Line using Breshenham's Algorithm");
        init();
        glutDisplayFunc(showGraphic);
        glutMainLoop();
        return 0;
}
```