Assignment performed by Gaurav Amarnani from DSE CMPN.
CG LAB 1 - Implement Line Drawing Algorithm.

Gaurav Amarnani DSE CMPN

Computer Graphics
Experiment: 1
Aim: Implement Line Drawing Algorithm

Theory: DDA stands for Digital Differential Analyzer. It is an incremental method of scan conve

Advantages of DDA:

1] It is a faster method than method of using direct use of line equation.
2] This method does not use multiplication theorem.
3] It allows us to detect the change in the value of x and y, so plotting of same point twice is not possible.
4] This method gives overflow indication when a point is repositioned.
5] It is an easy method because each step involves just two addition.

Disadvantage of DDA:

1] It involves floating point additions rounding off is done. Accumulations of round off error cause accumulation of error.
2] Rounding off operations and floating point operations consumes a lot of time.
3] It is more suitable for generating line use the software. But it is less suited for hardware implementation.

Bresenham line drawing:
The algorithm is used for scan converting a line. It was developed by Bresenham. It is an efficient method because it involves only integer addition, subtractions, and multiplication operations. These operations can be performed very rapidly so lines can be generated quickly.

Advantages of Bresenham line drawing

1] It involves only integer arithmetic, so it is simple.
2] It avoids the generation of duplicate points.
3] It can be implemented using hardware because it does not use multiplication and division.
4] It is faster as compared to DDA because it does not involve floating point calculations like DDA algorithm.

Disadvantages

1] This algorithm is meant for basic line drawing only initializing is not a part of Bresenham's line algorithm. ~~do to~~ draw smooth lines, you should want to look into a different algorithm.

Algorithm :

DDA algorithm

Step 1: Start Algorithm

Step 2: Declare $x_1, y_1, x_2, y_2, dx, dy, x, y$ as integer.

Step 3: Enter value of x_1, y_1, x_2, y_2

Step 4: Calculate $dx = x_2 - x_1$

Step 5: Calculate $dy = y_2 - y_1$

Step 6: If ABS(dx) > ABS(dy)
Then step = abs(dx)
Else

Step 7: x_{inc} = dx / step
y_{inc} = dy / step
assign $x = x_1$
assign $y = y_1$

Step 8 : set pixel (x, y)

Step 9 : $x = x + x_{inc}$
$y = y + y_{inc}$
set pixels (Round (x), Round (y))

Step 10 : Repeat step 9 until $x = x_2$

Step 11 : End Algorithm

Page No.
Date

Breenham's line drawing Algorithm :

Step 1 : Start Algorithm

Step 2 : Declare variable $x_1, x_2, y_1, y_2, d, i_1, i_2, dx,$

Step 3 : Enter value of x_1, y_1, x_2, y_2
where x_1, y_1 are coordinates of starting point.
x_2, y_2 are coordinates of ending point.

Step 4 : Calculate $dx = x_2 - x_1$
Calculate $dy = y_2 - y_1$
Calculate $i_1 = 2 \times dy$
Calculate $i_2 = 2*(dy - dx)$
Calculate $d = i_1 - dx$

Step 5 : consider (x, y) as starting point and xend as maximum possible value of x.

If $dx < 0$
Then $x = x_2$
$y = y_2$
$x_{end} = x_1$

If $dx > 0$
Then $x = x_1$
$y = y_1$
$x_{end} = x_2$

Step 6 : generate point at (x, y) coordinates

Step 7: Check if whole line is generated
If $x >= x_{end}$
Stop.

Step 8: Calculate co-ordinates of the next pixel
If $d < D$
Then $d = d + i_1$
If $d \geq D$
Then $d = d + i_2$
Increment $y = y + 1$

Step 9: Increment $x = x + 1$

Step 10: Draw a point of latest (x, y) coordinates

Step 11: Go to Step 7

Step 12: End of Algorithm

Example:

Using DDA line drawing algorithm to rasterize a line from $(1, 1)$ to $(5, 3)$

So, $x_1 = 1$, $y_1 = 1$, $x_2 = 5$ and $y_2 = 3$

$\therefore D_x = (x_2 - x_1) = (5 - 1) = 4$
$D_y = (y_2 - y_1) = (3 - 1) = 2$

As $|Dx| > |Dy|$ the line is of gentle slope category

$\therefore$ steps $= abs(Dx) = 4$

$$x\,increment = \frac{Dx}{steps} = \frac{4}{4} = 1$$

$$y\,increment = \frac{Dy}{steps} = \frac{2}{4} = 0.5$$

$\therefore$ first point is 1.1

$\therefore$ $x_{new} = x_{old} + x_{increment} = 1 + 1 = 2$

$y_{new} = y_{old} + y_{increment} = 1 + 0.5 = 1.5$

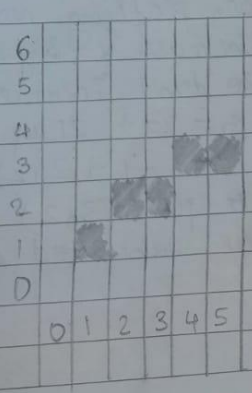
Rounding off 1.5 as 2 for displaying the point.
For next iteration

$x_{new} = x_{old} + x_{increment} = 2 + 1 = 3$

$y_{new} = y_{old} + y_{increment} = 1.5 + 0.5 = 2$

Next steps are tabulated.

I	x	y	plot
1	1	1	(1,1)
2	2	1.5=2	(2,2)
3	3	2	(3,2)
4	4	2.5=3	(4,3)
5	5	3	(5,3)

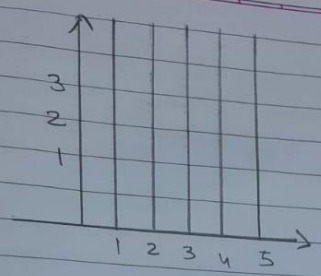


Bresenhams Example

1) Plot a line using Bresenham's line generation algorithm from (1, 1) to (5, 3)

$x_1 = 1, \; y_1 = 1$
$x_2 = 5, \; y_2 = 3$
$\therefore \; D_x = x_2 - x_1 = 4$
$D_y = y_2 - y_1 = 2$
$G = 2D_y - D_x$
$= 2(2) - (4)$
Plot 1st point here as $|D_x| > |D_y|$ that means line is a gentle slope, so here we have to move on x till x_2 i.e 5.
After plotting 1st point as (1, 1) increase x by 1.
$\therefore \; x = 2$
Here $a = 0$
we have to increase y by 1 and update G as
$G = G + 2(D_y - D_x)$
$= 0 + 2(2-4) = -4$
So, plotting next point as (2, 2) then again increase x by. Now $x = 3$. Here $G = -4$
so don't increase y just update G only as
$G = G + 2D_y$
$= -4 + 2(2) = 0$
So plot (3, 2) and moving on all the points are tabulated below

X	Y
1	1
2	2
3	2
4	3
5	3



Comparision of DDA and Bresenhams Line Drawing Algorithms:

DDA Line Algorithm	Bresenham Line Algorithm
1. DDA stands for Digital Differential Analyzer.	While it has no full form.
2. DDA algorithm is less efficient than the other.	While it is more efficient Than DDA.
3. It is costlier than Bresenham algorithm.	While Bresenham is cheaper than DDA line algorithm.
4. The calculation speed of DDA algorithm is less than Bresenham line algorithm.	While this is calculation speed of Bresenham is faster than DDA.
5. It has less precision or accuracy.	While it has more precision of accuracy.

PROGRAM:

```
#include<stdio.h>
#include<graphics.h>
#include <stdlib.h>
#include<math.h>

//Performed by Gaurav Amarnani DSE CMPN.

void bresenham(int X0, int Y0, int X1, int Y1){
   int dx=abs(X1-X0);
   int dy=abs(Y1-Y0);
   int x=X0, y=Y0, i=1;
   int e=2*dy-dx;
   do{
      putpixel(x,y,15);
      while(e>=0){
         y=y+1;
         e=e-2*dy;
      }
      x=x+1;
      e=e+2*dy;
      i=i+1;
   } while (i<=dx);
}

void dda(int X0, int Y0, int X1, int Y1) {
   int dx = X1 - X0;
   int dy = Y1 - Y0;

   int steps = abs(dx) > abs(dy) ? abs(dx) : abs(dy);

   float Xinc = dx / (float) steps;
   float Yinc = dy / (float) steps;
   float X = X0;
   float Y = Y0;

   int i;
   for (i = 0; i <= steps; i++) {
      putpixel(X,Y,WHITE);
      X += Xinc;
      Y += Yinc;
```

```
    }
}

void main() {
    int gd = DETECT, gm,ch,x1,y1,x2,y2;

    initgraph (&gd, &gm, "c:\\turboc3\\bgi");

    printf("1. DDA. \n 2. BRESENHAM. \n 3. Exit.");
    scanf("%d",&ch);
    printf("Enter x1, y1, x2, y2:");
    scanf("%d%d%d%d",&x1,&y1,&x2,&y2);

    switch(ch){
        case 1:
            dda(x1, y1, x2, y2);
            printf("Performed by Gaurav Amarnani DSE CMPN.");
            break;
        case 2:
            bresenham(x1, y1, x2, y2);
            printf("Performed by Gaurav Amarnani DSE CMPN.");
            break;
        case 3:
            printf("Performed by Gaurav Amarnani DSE CMPN.");
            exit(0);
        default:
            printf("Wrong Input.\n Performed by Gaurav Amarnani DSE
CMPN.");
            exit(0);
    }
    getch();
}
```

OUTPUT:


1. DDA.
2. BRESENHAM.
3. Exit.1
Enter x1, y1, x2, y2:100 150 190 250
Performed by Gaurav Amarnani DSE CMPN.


1. DDA.
2. BRESENHAM.
3. Exit.2
Enter x1, y1, x2, y2:100 150 190 250
Performed by Gaurav Amarnani DSE CMPN.

CONCLUSION:

Conclusion :

Learned about DDA & Bresenham Algorithm to draw lines and wrote code to implement it.