DDA and Bresenham Line

Trisha Shah

SE CMPN A 20102A0004

Algorithm:

DDA:

Step1: Start Algorithm

Step2: Declare x1,y1,x2,y2,dx,dy,x,y as integer variables.

Step3: Enter value of x1,y1,x2,y2.

Step4: Calculate dx = x2-x1

Step5: Calculate dy = y2-y1

Step6: If ABS (dx) > ABS (dy)

Then step = abs (dx)

Else

Step7: xinc=dx/step

yinc=dy/step

assign x = x1

assign y = y1

Step8: Set pixel (x, y)

Step9: x = x + xinc

y = y + yinc

Set pixels (Round (x), Round (y)) Step10: Repeat

step 9 until x = x2

Step11: End Algorithm.

**Bresenham Algorithm:**

Step1: Start Algorithm

Step2: Declare variable x1,x2,y1,y2,d,i1,i2,dx,dy

Step3: Enter value of x1,y1,x2,y2

Where x1,y1are coordinates of starting point

And x2,y2 are coordinates of Ending point

Step4: Calculate dx = x2-x1

Calculate dy = y2-y1

Calculate i1=2\*dy

Calculate i2=2\*(dy-dx)

Calculate d=i1-dx

Step5: Consider (x, y) as starting point and xendas maximum possible value of x.

If dx < 0

Then x = x2

y = y2

xend=x1 If dx > 0

Then x = x1 y =

y1 xend=x2

Step6: Generate point at (x,y)coordinates.

Step7: Check if whole line is generated.

If x > = xend

Stop.

Step8: Calculate co-ordinates of the next pixel

If d < 0

Then d = d + i1

If d ≥ 0

Then d = d + i2

Increment y = y + 1

Step9: Increment x = x + 1

Step10: Draw a point of latest (x, y) coordinates

Step11: Go to step 7

Step12: End of Algorithm

Implementation:

**DDA**

#include <graphics.h>

#include <stdio.h>

#include <math.h>

#include <conio.h>

//////////// DDA LINE DRAWING FUNCTION ////////////

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

{

float x=x1,y=y1,dx,dy;

int step,i;

putpixel(x1,y1,WHITE);

if(abs(x2-x1)>=abs(y2-y1))

step=abs(x2-x1);

else

step=abs(y2-y1);

dx=(float)(x2-x1)/step;

dy=(float)(y2-y1)/step;

for(i=1;i<=step;i++)

{

x=x+dx;

y=y+dy;

putpixel((int) x,(int) y,CYAN);

}

}

main()

{

int x1,y1,x2,y2,gd,gm;

detectgraph(&gd,&gm);

initgraph(&gd, &gm,(char\*)"");

printf("Enter the Co-ordinates of starting point of line: ");

scanf("%d %d",&x1,&y1);

printf("Enter the Co-ordinates of end point of line ");

scanf("%d %d",&x2,&y2);

ddaLine(x1,y1,x2,y2);

getch();

closegraph();

}

**BRESENHAM**

#include<stdio.h>

#include<conio.h>

#include<graphics.h>

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

{

int dx=x2-x1;

int dy=y2-y1;

int p=2\*dy-dx;

int x=x1;

int y=y1;

int i=0;

while(i<=dx)

{

if(p<0)

{

putpixel(x,y,MAGENTA);

x=x+1;

p=p+2\*dy;

}

else

{

putpixel(x,y,RED);

x=x+1;

y=y+1;

p=p+2\*dy-2\*dx;

}

i++;

delay(5000);

}

}

main()

{

int x,y,x1,y1,x2,y2,dx,dy,p,i;

int gd=DETECT,gm;

initgraph(&gd,&gm,(char\*)"");

printf("ENTER COORDINATES OF FIRST POINT");

scanf("%d%d",&x1,&y1);

printf("ENTER COORDINATES OF SECOND POINT");

scanf("%d%d",&x2,&y2);

Bresenham(x1,y1,x2,y2);

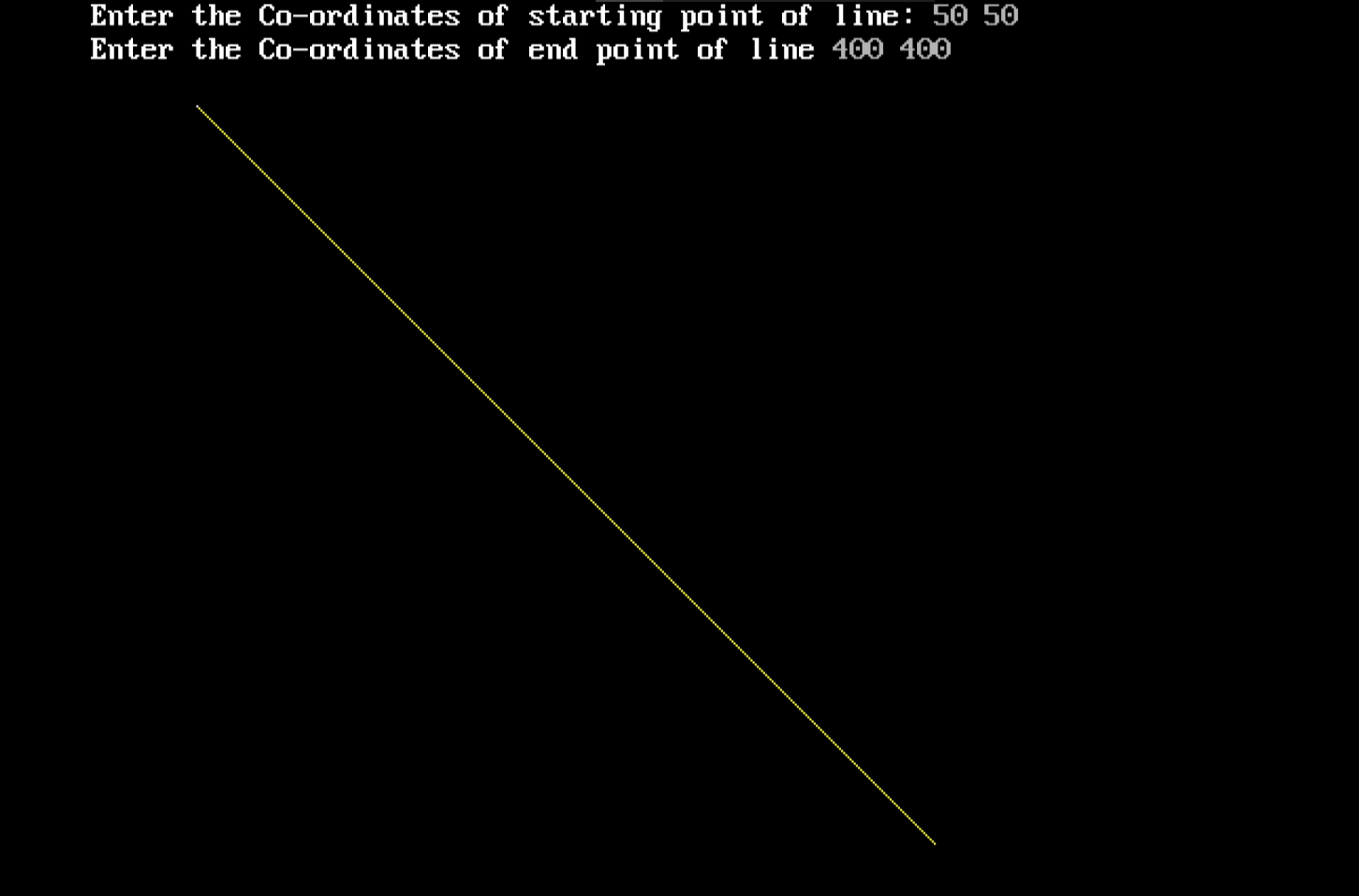
closegraph();

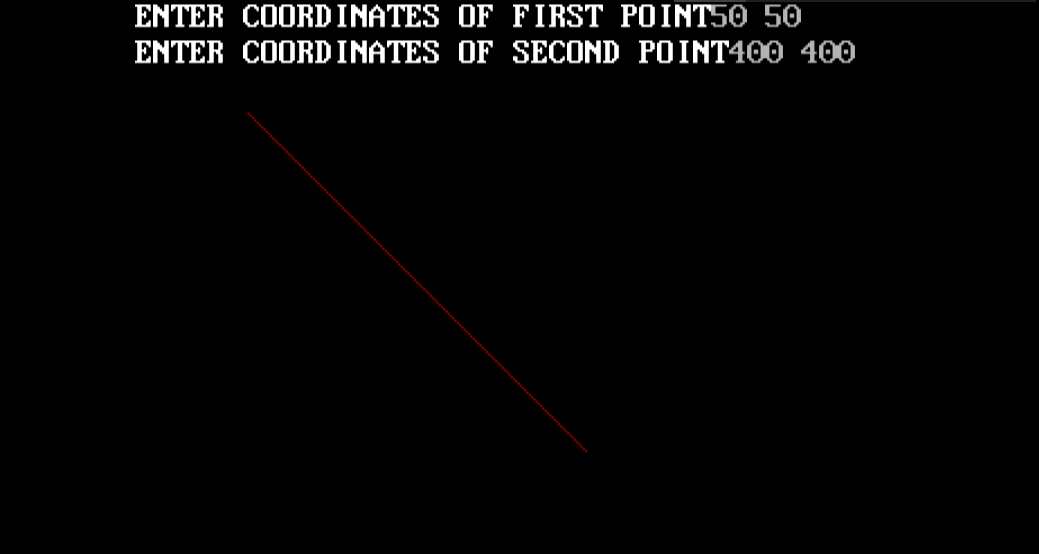
return 0;

}

**Output:**

**DDA**

****



Pros and Cons:

The advantages of DDA Algorithm

1. It is the simplest algorithm and it does not require special skills for implementation.

2. It is a faster method for calculating pixel positions than the direct use of equation y=mx + b. It eliminates the multiplication in the equation by making use of raster characteristics, so that appropriate increments are applied in the x or y direction to find the pixel positions along the line path.

The disadvantages of DDA Algorithm are

1. Floating point arithmetic in DDA algorithm is still time-consuming.

2. The algorithm is orientation dependent. Hence end point accuracy is poor.

3. There is an extra overhead of using round off( ) function.

4. Using round off( ) function increases time complexity of the algorithm.

5. Resulted lines are not smooth because of round off( ) function.

6. The points generated by this algorithm are not accurate.

The advantages of Bresenham Algorithm are

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 (Digital Differential Analyzer) because it does not involve

floating point calculations like DDA Algorithm.

The disadvantages of Bresenham Algorithm are

1. Though it improves the accuracy of generated points but still the resulted line is not smooth.
2. This algorithm is for the basic line drawing.
3. It cannot handle diminishing jaggies.