**TITLE: - IMPLEMENTATION OF DDA ALGORITHM**

**OBJECTIVE: -**

* To implement DDA (Digital Differential Analysis) Algorithm

**THEORY: -**

DDA (Digital Differential Analysis) Algorithm is a basic line drawing algorithm used to rasterize or draw lines in computer screen. It is simple and efficient algorithm which draw line by calculating the intermediate points from star to end coordinates. It uses the concept of Digital Differential in determining the change in x and y coordinates in each interval.

**CASE I: Positive Slope, |m| <1**

For this case, we sample points at unit x interval i.e. Δx=1 and compute each successive y coordinates.

We have,

…………………………… (i)

…………………………… (ii)

Subtracting equations (i) from (ii);

But, Δx=1, i.e. .

**So,**

**CASE II: Positive Slope, |m| >1**

For this case, we sample points at unit y interval i.e. Δy=1 and compute each successive x coordinates.

We have,

…………………………… (i)

…………………………… (ii)

Subtracting equations (i) from (ii);

But, Δy=1, i.e. .

**So,**

**CASE III: Negative Slope, |m| <1**

For this case, we sample points at unit x interval i.e. Δx=1 and compute each successive y coordinates.

We have,

…………………………… (i)

…………………………… (ii)

Subtracting equations (i) from (ii);

But, Δx=1, i.e. .

**So,**

**CASE IV: Positive Slope, |m| > 1**

For this case, we sample points at unit x interval i.e. Δy=1 and compute each successive x coordinates.

We have,

…………………………… (i)

…………………………… (ii)

Subtracting equations (i) from (ii);

But, Δy=1, i.e. .

**So,**

**CASE V: m = 1**

For this case, both x and y are incremented equally. Since, for slope (m) =1; Δx= Δy=1.

**Advantages of DDA Algorithm:**

It is faster than basic or direct method as it eliminates the floating point calculations by rounding up the floating coordinates.

**Disadvantages of DDA Algorithm:**

It has less precision and is not optimized since, it required rounding up of the floating coordinates.

**Summary of DDA Algorithm:**

Step 1: Input two end points (x1,y1) and (x2,y2); and set x=x1 and y=y1.

Step 2: Calculate dx and dy as: dx=x2-x1 and dy=y2-y1.

Step 3: If (|dx|>|dy|) then step=dx; else step=dy.

Step 4: Calculate: X-increment=dx/steps and Y-increment=dy/steps.

Step 5: Plot the point (x,y) with desired color.

Step 6: Beginning from k=0 to k<step; x=x +X-increment and y=y + Y- increment.

Step 7: Plot (x,y) with desired color.

**# Program to illustrate DDA Algorithm.**

#include<iostream.h>

#include<graphics.h>

#include<conio.h>

#include<string.h>

#include<math.h>

#include<stdlib.h>

#include<stdio.h>

int xc, yc;

float step;

int tablex, tabley, row\_height=20, column\_width=70;

void setcursor(int x, int y){

cout<<"\033["<<y<<";"<<x<<"H";

}

void drawtable(int row,int column){

int p\_row=tablex, p\_col=tabley;

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

line(p\_row,p\_col,p\_row+column\_width\*column,p\_col);

p\_col=p\_col+row\_height;}

p\_row=tablex, p\_col=tabley;

for(int j=0; j<=column; j++){

line(p\_row,p\_col,p\_row,p\_col+row\_height\*row);

p\_row=p\_row+column\_width;}

outtextxy(tablex+column\_width/2,tabley+row\_height/2,"k");

outtextxy(tablex+column\_width+column\_width/2,tabley+row\_height/2,"xk");

outtextxy(tablex+2\*column\_width+column\_width/2,tabley+row\_height/2,"yk");

outtextxy(tablex+3\*column\_width+5,tabley+row\_height/2,"round xk");

outtextxy(tablex+4\*column\_width+5,tabley+row\_height/2,"round yk");

}

void main(){

int gd=DETECT, gm=0;

initgraph(&gd, &gm, "..\\bgi");

xc=getmaxx()/2;

yc=getmaxy()/2;

tablex=xc-318;

tabley=yc-120;

float x1,y1,x2,y2; //Line end point coordinates

int k=0;

rectangle(xc+40,yc-110,getmaxx(),yc-43);

rectangle(xc+40,yc-45,getmaxx(),yc+100);

outtextxy(xc+150,yc-107,"INPUT");

line(xc+40,yc-97,getmaxx(),yc-97);

outtextxy(xc+150,yc-40,"OUTPUT");

line(xc+40,yc-30,getmaxx(),yc-30);

setcursor(47,10); cout<<"Enter Point to start: ";

setcursor(68,10); cin>>x1;

setcursor(78,10); cin>>y1;

setcursor(47,11); cout<<"Enter the end point: ";

setcursor(68,11); cin>>x2;

setcursor(78,11); cin>>y2;

int dx=(x2-x1), dy=(y2-y1);

if(abs(dx)>abs(dy))

step=abs(dx);

else

step=abs(dy);

char x\_value[10], y\_value[10],round\_x[10], round\_y[10],SN[10];

int x\_round, y\_round;

int text\_x=tablex+column\_width/2-10;

int text\_y=tabley+row\_height+row\_height/2;

while(k<=step){

x\_round=x1+0.5;

y\_round=y1+0.5;

sprintf(x\_value,"%.2f",x1);

sprintf(y\_value,"%.2f",y1);

itoa(x\_round,round\_x,10);

itoa(y\_round,round\_y,10);

itoa(k,SN,10);

outtextxy(text\_x,text\_y,SN);

outtextxy(text\_x+column\_width,text\_y,x\_value);

outtextxy(text\_x+column\_width\*2,text\_y,y\_value);

outtextxy(text\_x+column\_width\*3,text\_y,round\_x);

outtextxy(text\_x+column\_width\*4,text\_y,round\_y);

putpixel(xc+120+x1,yc+y1,WHITE);

text\_y=text\_y+row\_height; //Displays the output pixels

x1=x1+(dx/step);

y1=y1+(dy/step);

k++;

}

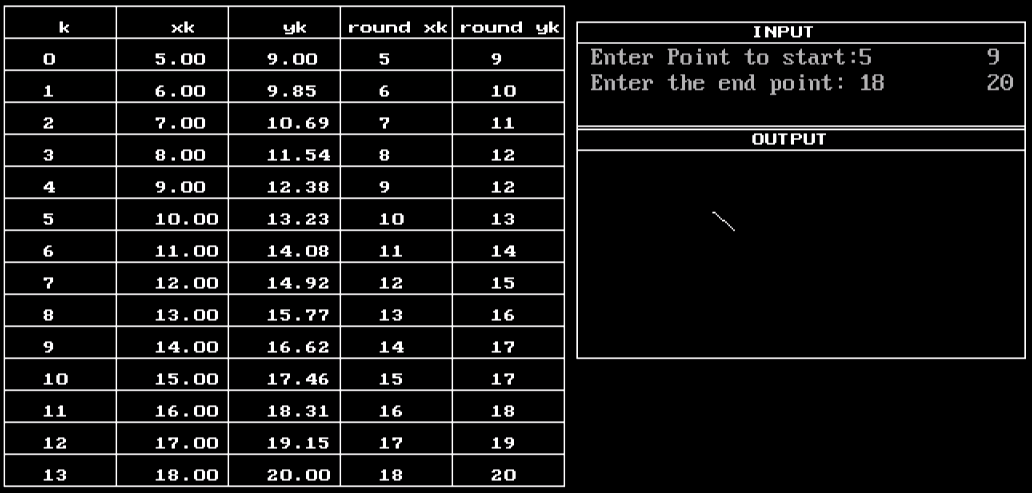
drawtable(step+2,5);

getch();

closegraph();

}

**OUTPUT**: -



**CONCLUSION**:

In this way, we implement DDA algorithm through writing code in C++ programming language and analysed its precision and way of calculation.