DEERWALK INSTITUTE OF TECHNOLOGY

Tribhuvan University

Faculties of Computer Science

Bachelors of Science in Computer Science and Information Technology (BSc. CSIT)

Course: Computer Graphics (CSC214)

Year/Semester: II/III

A Lab report on: Implementation of DDA and Graphics Functions

Submitted by:
Name: Arun Mainali
Roll: 1307

Submitted to:
Binod Sitaula
Department of Computer Science

❖ LAB 1

OBJECTIVE:

1. Introduction to Graphics Handling Functions.
2. Implementation of DDA for:
   i. Left to Right
   ii. Right to Left

THEORY:

Graphic Functions:

Function line()
This function draws a line in the graphics mode.

Function arc()
This function draws an arc from start to the end point. It uses the radius and the angle inputted in the function parameter.

Function bar()
The bar() functions draws a rectangular bar on the screen. It takes four parameters of type int which are in fact the points on the graphics screen, and fills the bar with the defined fill-pattern.

Function bar3d()
As the name suggests that this function will draw the 3 Dimensional rectangular bar on the screen. Bar3d function takes six parameters of which four are the points , fifth one is the depth of the bar and sixth is the flag to indicate the 3D view. Here is the prototype of the bar3d function.

Function circle()
Draws the circle on the screen using a point(x,y) and the radius for the circle.

Function drawpoly()
This function draws an outline of a polygon. Here numpoints is the number of end points in the polygon. And points is the integer array which contains the x and y coordinates of the points.

Function ellipse()
Draws an ellipse using the current drawing color.

Function floodfill()
This function fills an object drawn on the graphics screen with the defined color and fill pattern. Now if the x,y coordinates lie within the boundries of the object then it will fill the interior of the object otherwise out side the object.

Function outtext()
Displays a text string on the screen in graphics mode.

Function outtextxy()
Displays the text at a specified position in graphics mode.

Function putpixel()
Prints a pixel at the specified point on the screen.

Function rectangle()
Draws a rectangular box on the screen using the points given in the parameters.

Digital Differential Analyzer (DDA):

It is an incremental method of scan conversion of line. In this method calculation is performed at each step but by using results of previous steps.

Suppose at step i, the pixels is (x_i,y_i)

The line of equation for step i

$y_i=mx_{i+b}$.....................equation 1

Next value will be

$y_{i+1}=m_{xi+1}+b$................equation 2

$m = \frac{\Delta y}{\Delta x}$

$y_{i+1}-y_i=\Delta y$......................equation 3

$y_{i+1}-x_i=\Delta x$.....................equation 4

$y_{i+1}=y_i+\Delta y$

$\Delta y=m\Delta x$

$y_{i+1}=y_i+m\Delta x$

$\Delta x=\Delta y/m$

$x_{i+1}=x_i+\Delta x$

$x_{i+1}=x_i+\Delta y/m$

Case1: When |M|<1 then (assume that x1<x2)

x= x1,y=y1 set Δx=1

yi+1=y1+m, x=x+1

Until x = x2</x

Case2: When |M|<1 then (assume that y1<y2)

```
x= x1,y=y1 set Δy=1
xi+1=$\frac{1}{m}$,    y=y+1
Until y → y2</y
```

Advantage:

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 additions.

Disadvantage:

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 using the software. But it is less suited for hardware implementation.

ALGORITHM:

```
Step 1: Start Algorithm
Step 2: Declare $x_1$,$y_1$,$x_2$,$y_2$,dx,dy,x,y as integer variables.
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
```

PROGRAM CODE 1:

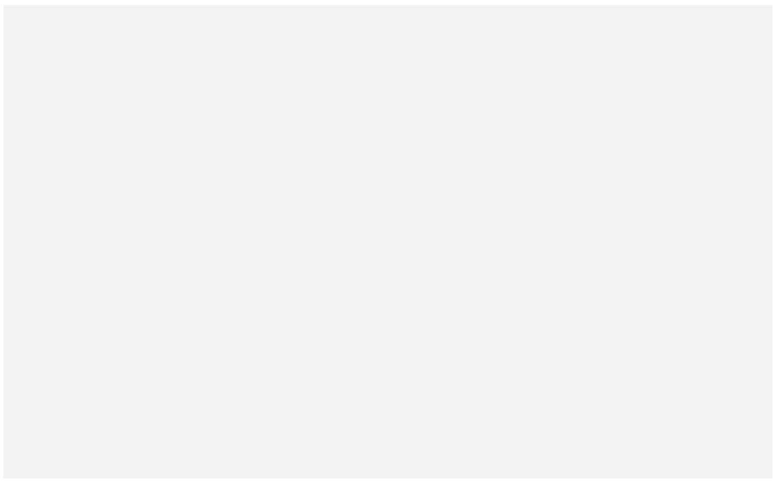
```
#include<stdio.h>
#include<conio.h>
#include<math.h>
#include<graphics.h>

int main()
{
	float y,x,x1,x2,y1,y2,dx,dy,step;
	int i,gd=DETECT,gm;
	//initgraph(&gd,&gm,"C:\\turboc3\\bgi");
	initgraph(&gd,&gm,"");
	printf("Enter initial points(FORMAT: x,y): ");
	scanf("%f,%f",&x1,&y1);
	printf("Enter final points(FORMAT: x,y): ");
	scanf("%f,%f",&x2,&y2);

	dx = abs(x2-x1);
	dy = abs(y2-y1);

	if(dx>=dy)
	{
		step = dx;
	}
	else
	{
		step = dy;
	}

	dx = dx/step;
	dy = dy/step;
	x = x1;
	y = y1;

	i=1;
	while(i<=step)
	{
```

```
                putpixel(x,y,WHITE);
                x = x + dx;
                y = y + dy;
                i = i + 1;
                delay(100);
        }
        closegraph();
        getch();
}
```

PROGRAM CODE 2:

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

int main() {
   int gd = DETECT, gm;
   int ballX = 100, ballY = 100;
   int dirX = 2, dirY = 3;
   int radius = 30;

   initgraph(&gd, &gm, "");

   if (graphresult() != grOk) {
      printf("Graphics initialization error.\n");
      return 1;
   }

   setbkcolor(BLACK);
   cleardevice();

   while (!kbhit()) {
      cleardevice();

      setcolor(YELLOW);
      setfillstyle(SOLID_FILL, YELLOW);
      fillellipse(ballX, ballY, radius, radius);

      ballX += dirX;
      ballY += dirY;

      if (ballX - radius <= 0 || ballX + radius >= getmaxx()) {
         dirX = -dirX;
      }
      if (ballY - radius <= 0 || ballY + radius >= getmaxy()) {
            dirY = -dirY;
        }

        delay(10);
    }

    closegraph(gd);
    return 0;
}
```

SAMPLE OUTPUT:



CONCLUSION:

From this lab, we were able to draw a line using Digital Differential Analyzer (DDA) Algorithm. We also got to learn various graphics functions supported by C programming and use them.