EXPERIMENT - 2

COMPUTER GRAPICS AND MULTIMEDIA

DDA Algorithm

Write a C program to draw a line using DDA algorithm.

Syeda Reeha Quasar
14114802719
Group - 3C7

EXPERIMENT – 2

AIM:

Write a C program to draw a line using DDA algorithm.

THEORY:

DDA stands for Digital Differential Analyzer. 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.

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.

DDA Algorithm:

Step1: Start Algorithm

Step2: Declare x_1,y_1,x_2,y_2,dx,dy,x,y as integer variables.

Step3: Enter value of x_1,y_1,x_2,y_2.

Step4: Calculate dx = x_2-x_1

Step5: Calculate dy = y_2-y_1

Step6: If ABS (dx) > ABS (dy)
  Then step = abs (dx)
  Else

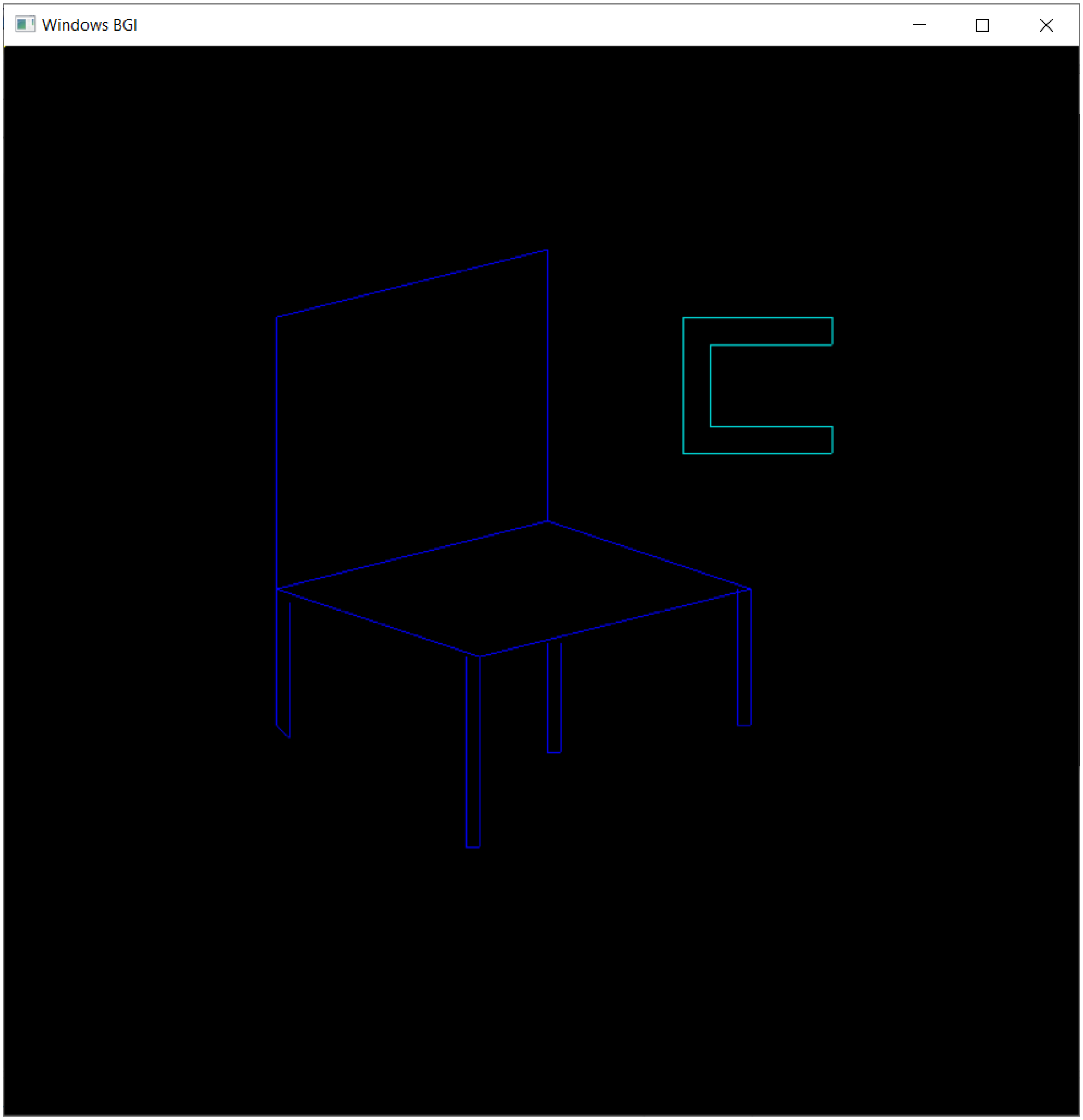
Step7: x_{inc}=dx/step
  y_{inc}=dy/step

```
        assign x = $x_1$
        assign y = $y_1$

Step8: Set pixel (x, y)

Step9: x = x + $x_{inc}$
        y = y + $y_{inc}$
        Set pixels (Round (x), Round (y))

Step10: Repeat step 9 until x = $x_2$

Step11: End Algorithm
```

Source Code:

```
// C program for drawing a chair

#include<stdio.h>

#include<conio.h>

#include<graphics.h>


Dda_function(int x0,int y0,int x1,int y1,int clr)

{

        int i;

        float x,y,dx,dy ,steps;


        dx=x1-x0;

        dy=y1-y0;

        if(abs(dx)>=abs(dy)){

                steps=dx;

        }

        else{

                steps=dy;

        }

        dx=dx/steps;

        dy=dy/steps;
        x=x0;
        y=y0;
        i=1;
        while(i<=steps){
                putpixel(x,y,clr);
                x+=dx;
                y+=dy;
                i=i+1;
        }
}
main(){
        initwindow(800,800);
        // rest
        Dda_function(200, 200, 400, 150, 1); //top
        Dda_function(200, 200, 200, 400, 1); //l line
        Dda_function(400, 150, 400, 350, 1); //r line
        Dda_function(200, 400, 400, 350, 1); //join
        // seat
        Dda_function(200, 400, 350, 450, 1);
        Dda_function(400, 350, 550, 400, 1);
        Dda_function(350, 450, 550, 400, 1);
        // legs
        Dda_function(200, 400, 200, 500, 1);
        Dda_function(210, 410, 210, 510, 1);
        Dda_function(200, 500, 210, 510, 1);



        Dda_function(400, 440, 400, 520, 1);
        Dda_function(410, 440, 410, 520, 1);
        Dda_function(400, 520, 410, 520, 1);

        Dda_function(340, 450, 340, 590, 1);
        Dda_function(350, 450, 350, 590, 1);
        Dda_function(340, 590, 350, 590, 1);

        Dda_function(550, 400, 550, 500, 1);
        Dda_function(540, 400, 540, 500, 1);
        Dda_function(540, 500, 550, 500, 1);

        //c
        Dda_function(500, 200, 500, 300, 3);
        Dda_function(500, 200, 610, 200, 3);
        Dda_function(500, 300, 610, 300, 3);
        Dda_function(610, 280, 610, 300, 3);
        Dda_function(610, 200, 610, 220, 3);
        Dda_function(520, 220, 610, 220, 3);
        Dda_function(520, 280, 610, 280, 3);
        Dda_function(520, 220, 520, 280, 3);

getch();
closegraph();

}
```

OUTPUT



VIVA QUESTIONS:

Q1. Define getpixel and putpixel functions in graphics.h?

Ans. **getpixel():** Function getpixel returns the color of pixel present at point(x, y).

Declarations: int getpixel(int x, int y);

Putpixel(): Function putpixel plots a pixel at a point(x, y) of the specified color.

Declarations: void putpixel(int x, int y, int color);

Q2. What is the difference between DDA algorithm and Bresenham's Algorithm?

Ans.

DDA Algorithm	**Bresenham's Line Algorithm**
1. DDA Algorithm use floating point, i.e., Real Arithmetic.	1. Bresenham's Line Algorithm use fixed point, i.e., Integer Arithmetic
2. DDA Algorithms uses multiplication & division its operation	2.Bresenham's Line Algorithm uses only subtraction and addition its operation
3. DDA Algorithm is slowly than Bresenham's Line Algorithm in line drawing because it uses real arithmetic (Floating Point operation)	3. Bresenham's Algorithm is faster than DDA Algorithm in line because it involves only addition & subtraction in its calculation and uses only integer arithmetic.
4. DDA Algorithm is not accurate and efficient as Bresenham's Line Algorithm.	4. Bresenham's Line Algorithm is more accurate and efficient at DDA Algorithm.
5.DDA Algorithm can draw circle and curves but are not accurate as Bresenham's Line Algorithm	5. Bresenham's Line Algorithm can draw circle and curves with more accurate than DDA Algorithm.

Q3. Give disadvantages of DDA Algorithm?

Ans. It involves floating point additions rounding off is done. Accumulations of round off error cause accumulation of error.

Rounding off operations and floating point operations consumes a lot of time.

It is more suitable for generating line using the software. But it is less suited for hardware implementation.

Q4. Give disadvantages of Bresenham's algorithm?

Ans. This algorithm is meant for basic line drawing only Initializing is not a part of Bresenham's line algorithm. So to draw smooth lines, you should want to look into a different algorithm.