

## Assignment No. 3

Title - Line drawing with line patterns.

Problem statement -

Write a c++ program for line drawing using DDA or bresenham's algorithm with pattern such as solid, dotted, dashdot, dashed & thick line.

Objectives -

- To understand & study line generation algorithm.
- To study different line patterns.
- To understand concepts & features of OOP.
- To understand & study different manipulation facilities in QT creator using c++.

Outcomes -

To implement a line drawing algorithm with different line patterns.

Software & hardware requirement -

Linux based OS.

QT creator.

## Theory :-

Line drawing algorithm - to generate a line 2 different algorithm are used.

- i) DDA
- ii) Bresenham

DDA algorithm	Bresenham algorithm
<ul style="list-style-type: none"><li>• Based on incremental method</li><li>• Slower</li><li>• Uses floating point arithmetic.</li><li>• Less accuracy.</li></ul>	<ul style="list-style-type: none"><li>• based on incremental method</li><li>• Faster</li><li>• Uses integer calculations.</li><li>• efficient &amp; accurate</li></ul>

## DDA algorithm :-

- i) DDA stands for digital differential analyzer.
- ii) It is the simplest line drawing algorithm in computer graphics which is used floating point or integer arithmetic.
- iii) It works on incremental method. It plots the point from source to destination by incrementing source co-ordinate.



iv) It takes unit step along one co-ordinate & computes the corresponding value along other co-ordinate.

### algorithm -

Integer: integer function

i.e. Integer  $(-8.5) = -9$

Sign: returns  $-1, 0, 1$  for arguments.

Step 1.) Read end points  $(x_1, y_1)$   $(x_2, y_2)$

Step 2.) Approximate the length of line  
if  $(\text{abs}(x_2 - x_1) > \text{abs}(y_2 - y_1))$  then  
length =  $(\text{abs}(x_2 - x_1))$   
else  
length =  $(\text{abs}(y_2 - y_1))$

Step 3.) Select raster unit

$$\Delta x = (x_2 - x_1) / \text{length}$$

$$\Delta y = (y_2 - y_1) / \text{length}$$

Step 4.) Round the values

$$x = x_1 + 0.5 + \text{sign}(\Delta x)$$

$$y = y_1 + 0.5 + \text{sign}(\Delta y)$$

Step 5.) plot the pixel  
 $i = 1$

while  $(i \leq \text{length})$   
{

setpixel (Integer(x), Integer(y))

$$x = x + \Delta x; \quad y = y + \Delta y;$$


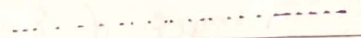
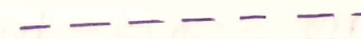
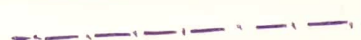

$$i++;$$

}

### ① Line patterns :-

The DDA algorithm can be modified to generate the different line patterns

The lines are classified as follows :-

1. Solid line 
2. Dotted line 
3. Dashed line 
4. Dash-Dot line 
5. Thick line 

### 1. Solid line :-

The solid line can be generated using the DDA algorithm as it is.

### 2. Dotted line :-

Dotted line can be drawn using dda algorithm just leave space after every pixel to make the dotted line.

```
code: while (i ≤ len)
{
    if (i % 2 == 0)
        setpixel (x, y, color)
    x = x + xinc
    y = y + yinc
    i++
}
```



### 3 Dashed line

Dashed line can be drawn using the DDA algorithm, first leave space after every pixel to make dash.

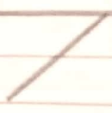
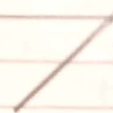






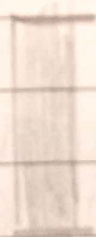
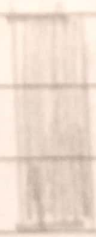
```
code: while (i ≤ len)
{
    if (i % 6 == 0 || i % 6 == 1 || i % 6 == 2)
        setpixel (x, y, color)
    end if
    x = x + xinc
    y = y + yinc
    i++
} //end while.
```

### Dash-Dot line:

Dashed-Dot line contain Dot in between the dashed line.

```
code: while (i ≤ len)
{
if (i % 6 == 0 || i % 6 == 1 || i % 6 == 2 ||
i % 6 == 3)
    setpixel (x, y, color)
    end if
    x = x + xinc
    y = y + yinc
}
```

Test cases =

Line	inp	O/p	Expected O/p	Result
Solid line	$x_1=0, y_1=0$ $x_2=4, y_2=4$			Success
Dotted line	$x_1=0, y_1=4$ $x_2=0, y_2=8$			Success
Dashed line	$x_1=0, y_1=5$ $x_2=0, y_2=8$			Success
Dash-Dot-Dash	$x_1=0, y_1=5$ $x_2=0, y_2=10$			Success
Thick line	$x_1=5, y_1=10$ $x_2=10, y_2=0$			Success

### 5. Thick line

- To generate a thick line segment, we can run the vector generation algorithm in parallel to find the pixel along the line edges.
- For a gentle slopping the line between  $(x_a, y_a)$  &  $(x_b, y_b)$  with thickness  $w$ , we would have a top boundary bet<sup>n</sup> the point  $(x_a, y_a + w_y)$  &  $(x_b, y_b + w_y)$  & lower left boundary bet<sup>n</sup>  $(x_a, y_a - w_y)$  &  $(x_b, y_b - w_y)$  where  $w_y$  given by,

$$w_y = \left[ \left( w - \frac{1}{2} \right) \cdot \left\{ \frac{\sqrt{(x_b - x_a)^2 + (y_b - y_a)^2}}{2} \right\} / \text{abs}(x_2 - x_1) \right]$$

code: if  $(y_2 - y_1) > (x_2 - x_1)$

$$\{ \text{float } w_y = \left[ w - \frac{1}{2} \cdot \left\{ \frac{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}{2} \right\} / \text{abs}(x_2 - x_1) \right] \}$$

for  $(i = 0; i < w_y; i++)$

dda  $(x, y_1 + i, x_2, y_2 + i)$

dda  $(x, y_1 - i, x_2, y_2 - i)$

end for

~~else~~ }

$$\text{float } w_x = \left[ w - \frac{1}{2} \cdot \left\{ \frac{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}}{2} \right\} / \text{abs}(y_2 - y_1) \right]$$



```
for (i=0; i<wx; i++)  
    dda (x1+i, y1, x2+i, y2)  
    dda (x2-i, y2, x2-i, y1)  
end for
```

}

### Conclusion:-

We have learn & implement different line drawing patterns using DDA line drawing algorithm.