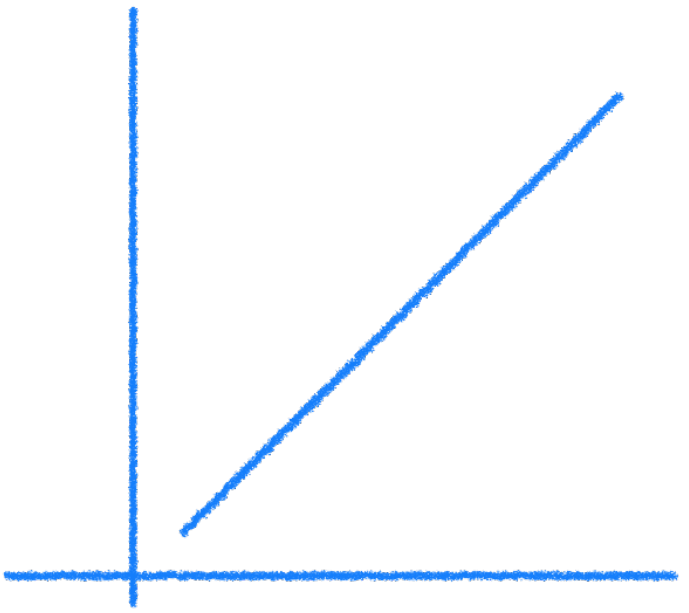
CSE 423 Chap 1

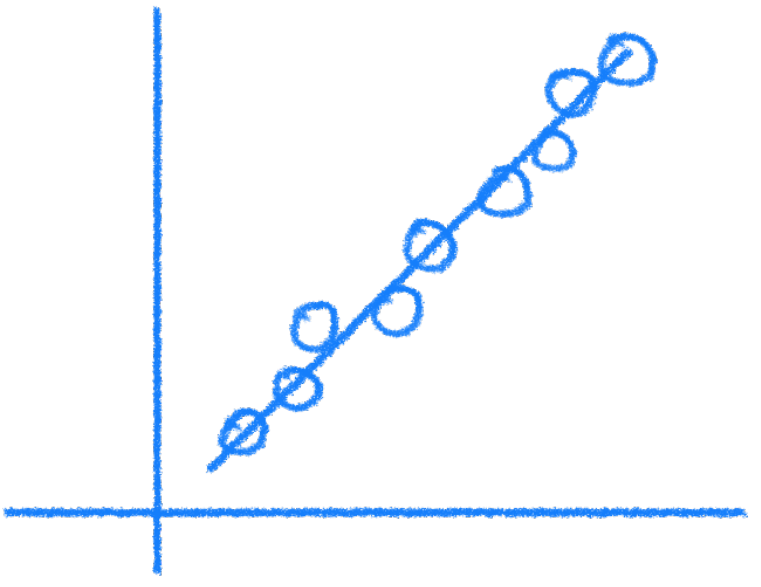
Line drawing Algorithm

- DDA
- MIDPOINT
- MIDPOINT with 8-way Symmetry.



How is a line drawn?

- a line is drawn by moving through pixels.

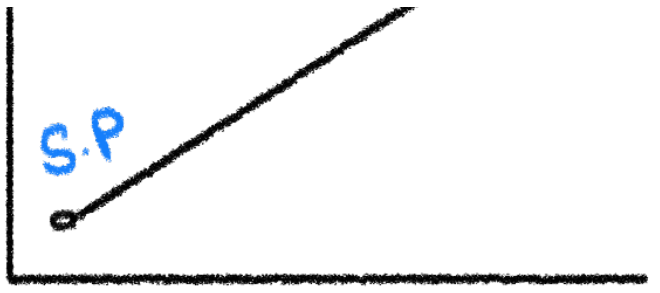


This is similar to the best fit line we use to draw in school & college. Choosing the coordinates that are closer.

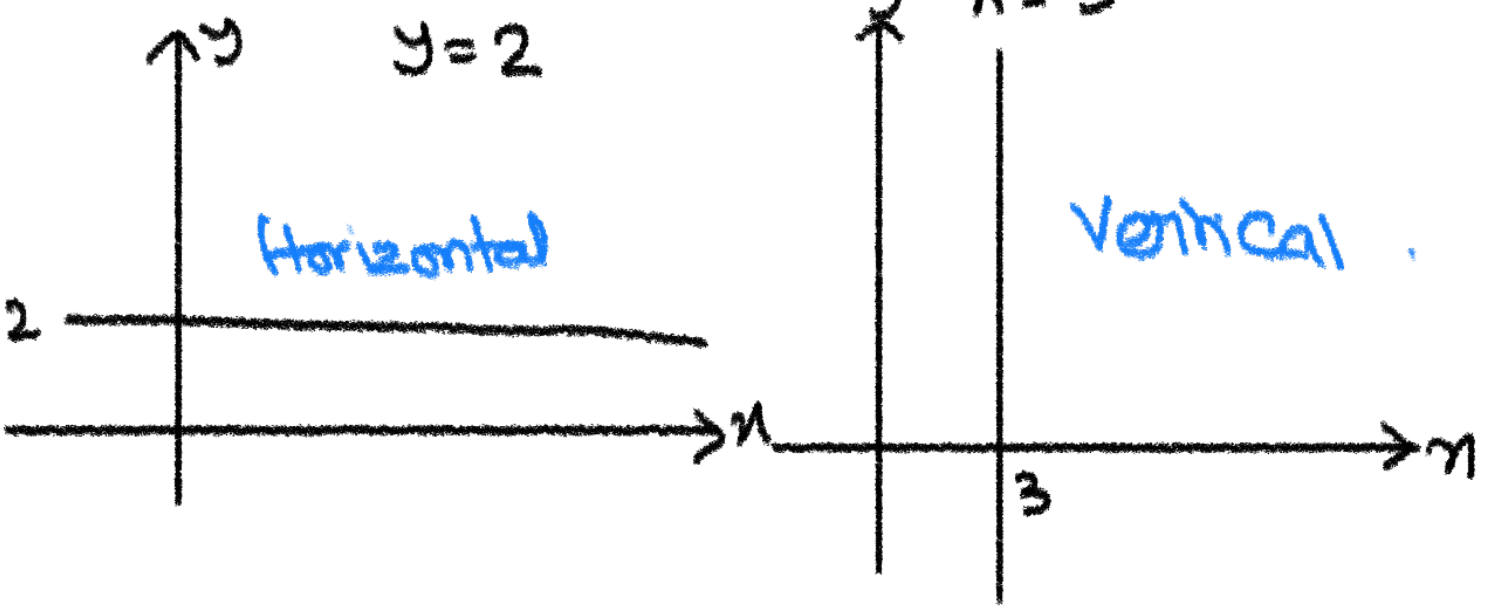
So initially we have,

Simple Approach

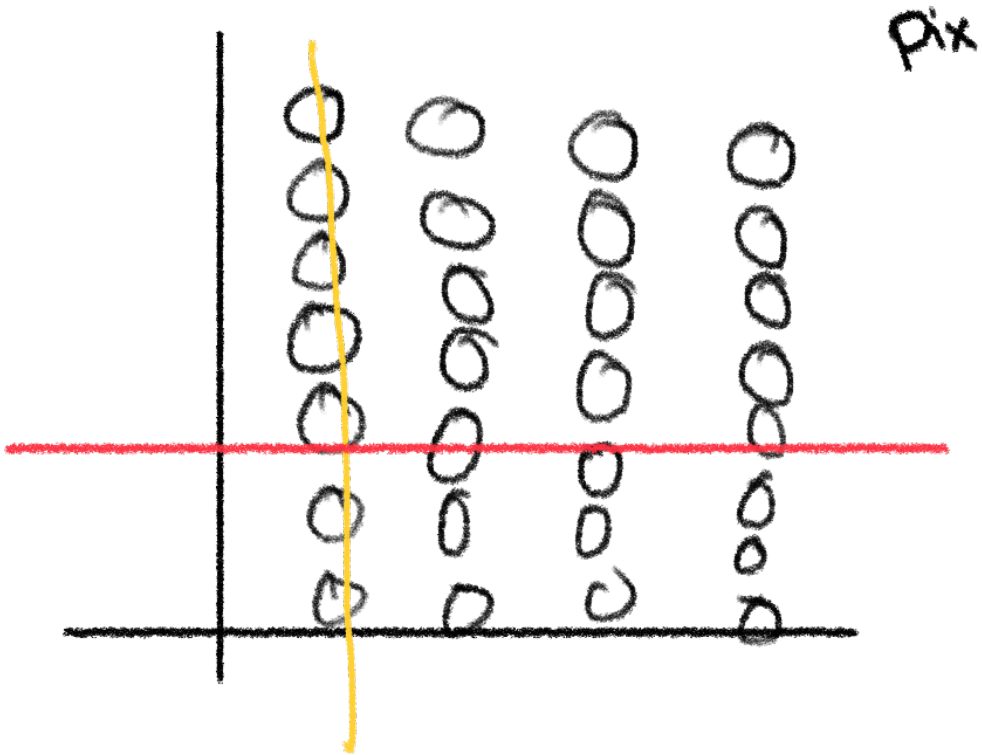
we can draw a line by joining its start and end points/coordinates



IF asked to draw horizontal & vertical line



These are easier to draw as pixels have their own coordinates.



However, it is difficult to draw slant lines perfectly.

For this we need pixels.

∴ our target is to find out all the necessary pixels to draw line.

Things to Consider:

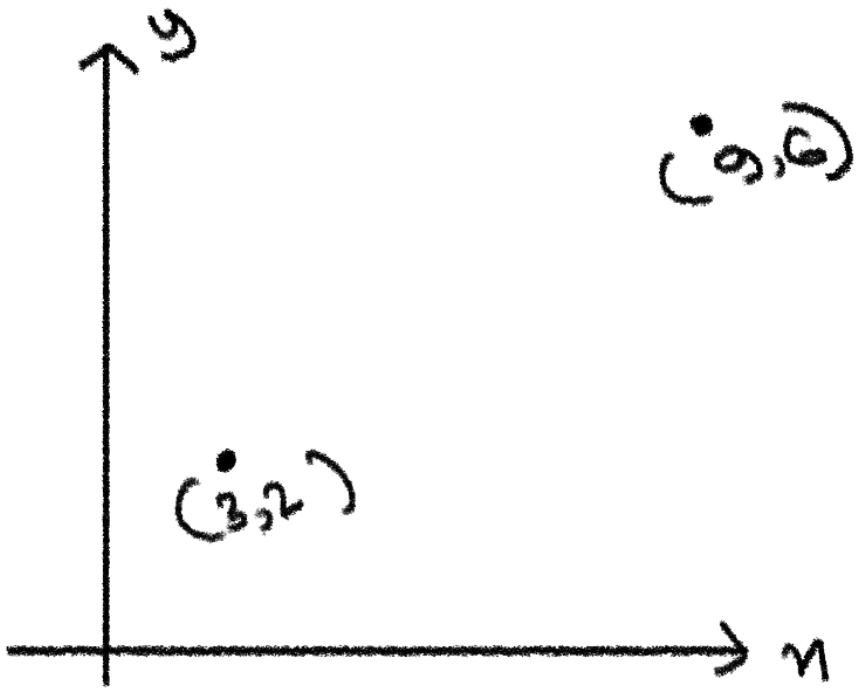
→ Avoid jaggies

→ Computation should be faster.

Simple Approach

Two points are given and we need to draw line between them.

For ex : (3,2) & (9,6)



How do we draw a line when two points are given?

By Solving $y = mx + c$

↓ gradient

$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

$$m = \frac{6-2}{9-2}$$

$$= \frac{4}{6} = 2/3$$

Eqn of straight line,

$$y - y_1 = m(x - x_1)$$

$$y - 2 = \frac{2}{3}(x - 3)$$

$$y = \frac{2}{3}x - 2 + 2$$

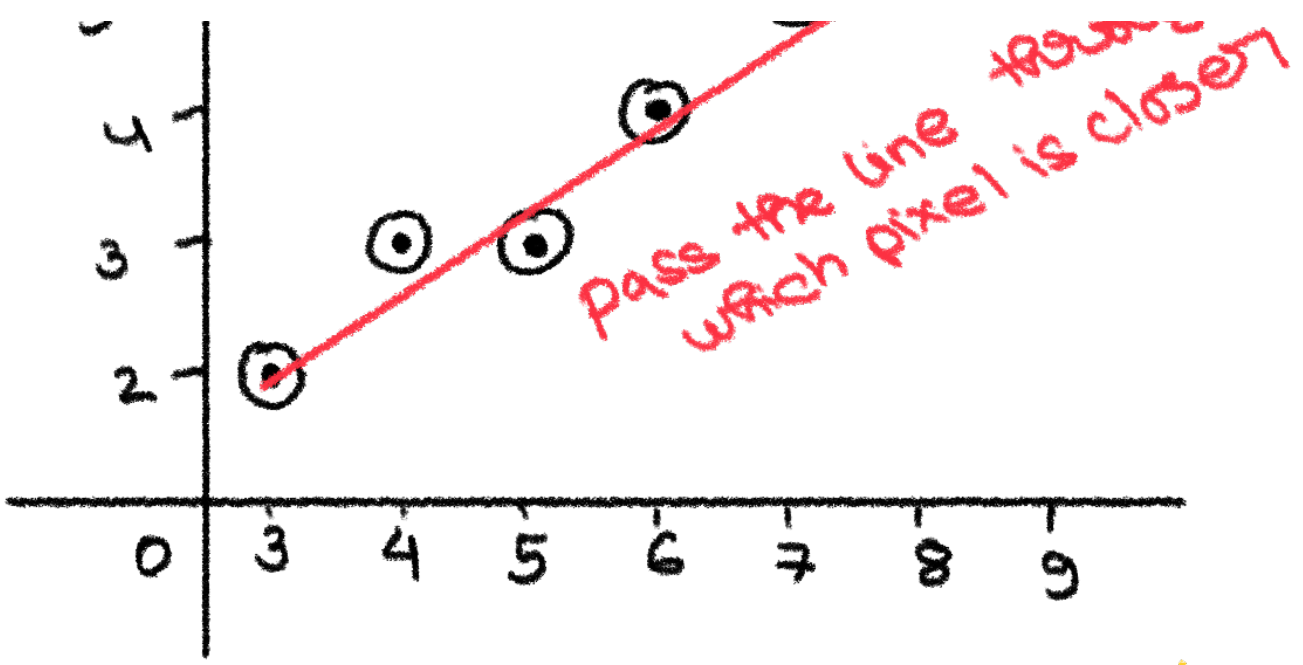
$$y = \frac{2}{3}x$$

→ S.P → E.P

x	3	4	5	6	7	8	9
y	2	3	3	4	5	5	6

x	y cal.	y	Round off
3	$y = \frac{2}{3} \times 3$	2	2
4	$y = \frac{2}{3} \times 4$	2.7	3
5	$y = \frac{2}{3} \times 5$	3.3	3
6	$y = \frac{2}{3} \times 6$	4	4
7	$y = \frac{2}{3} \times 7$	4.6	5
8	$y = \frac{2}{3} \times 8$	5.3	5
9	$y = \frac{2}{3} \times 9$	6	6

Plot in graph



What if you are asked to find from (3,2) to (1000, 100)?

- It will take all day!
- As we are rounding off we are not getting the accurate line.

∴ PROBLEMS

↳ we need to multiply with m then do add.

→ The approach is too slow.
→ We are not getting 100% accuracy.
→ Need to find y continously.

∴ This is NOT A FEASIBLE APPROACH.

DDA [Digital Differential Analyzer]

Algo:

If m is between $-1 < m < 1$

$$x_{k+1} = x_k + 1$$

$$y_{k+1} = y_k + m$$

else,

$$y_{k+1} = y_k + 1$$

$$x_{k+1} = x_k + 1/m$$

Then, round off to the nearest pixel.

***

In simple approach we took x values & found y values

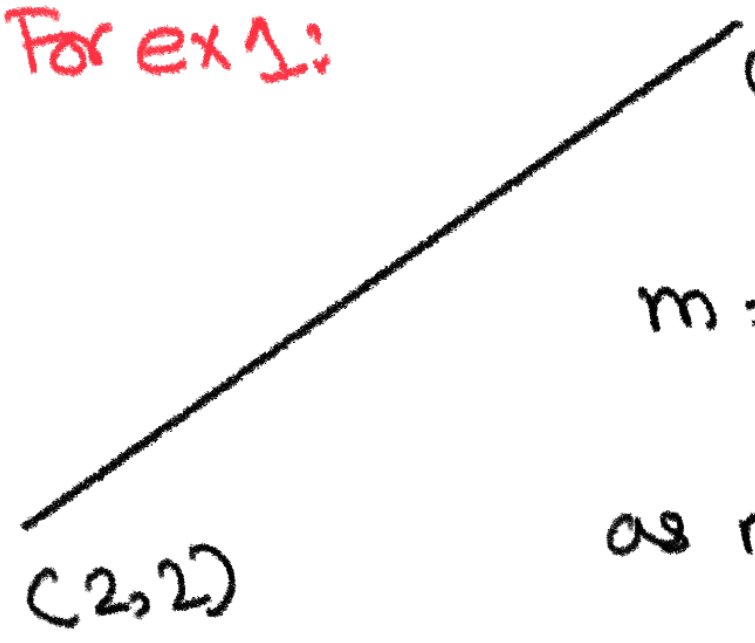
$x = 3, 4, 5, 6, 7, 8, 9$

However, we could have also used y values to find x

$y = 2, 3, 4, 5, 6$

For this we use DDA as it optimized. Based on gradient it decides which value to make integer & which to find.

For ex 1:



$$m = \frac{5-2}{7-2} = \frac{3}{5} = 0.6$$

as m is between

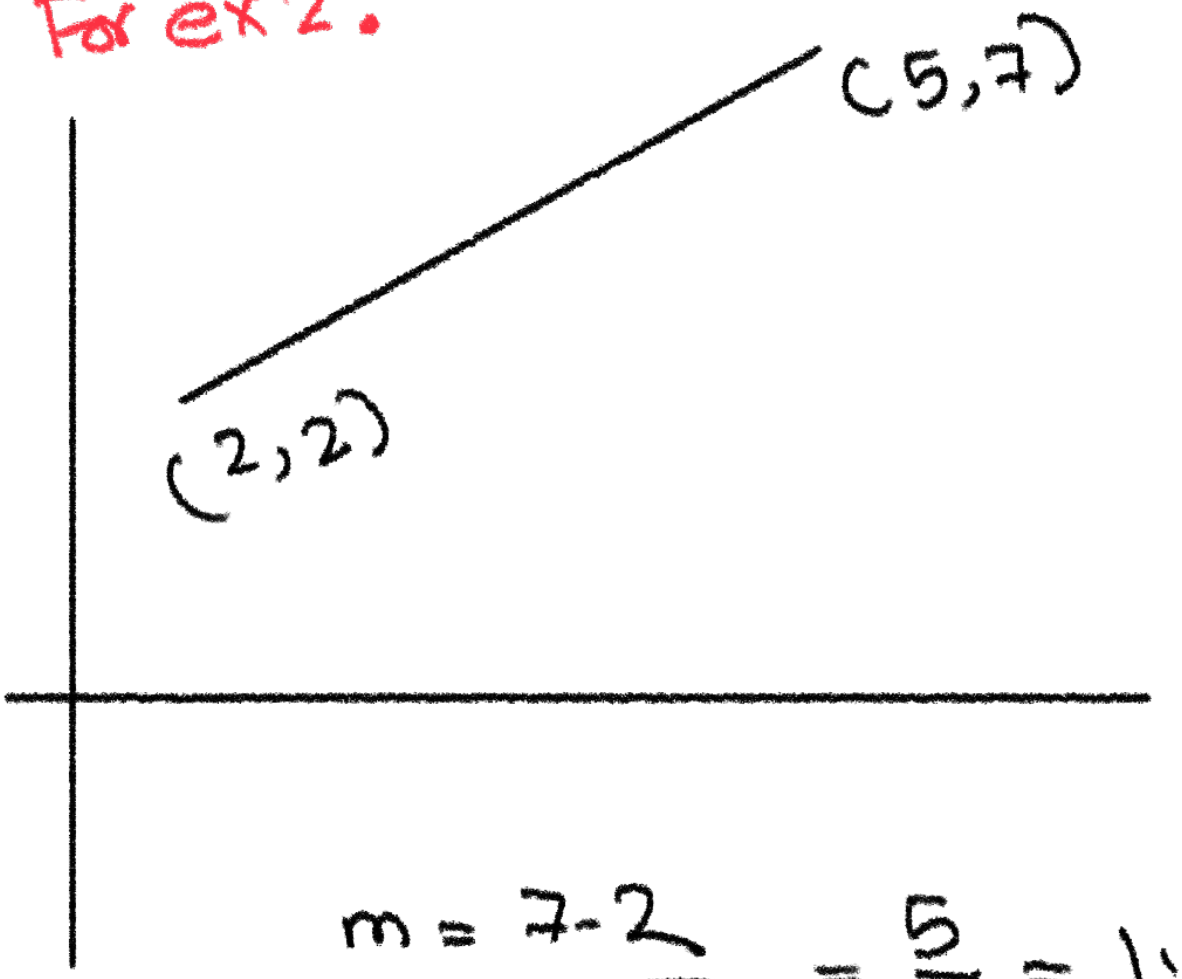
$-1 < m < 1$

$m = 0.6$

x	y	y (round off)	pixel

		2	(2,2)
2+1 = 3	2+0.6 = 2.6	3	(3,3)
3+1 = 4	2.6+0.6 = 3.2	3	(4,3)
4+1 = 5	3.2+0.6 = 3.8	4	(5,4)
5+1 = 6	3.8+0.6 = 4.4	4	(6,4)
6+1 = 7	4.4+0.6 = 5	5	(7,5)

For ex2:



$$m = \frac{7-2}{5-2} = \frac{5}{3} = 1.7$$

$$y_{k+1} = y_k + 1$$

$$x_{k+1} = x_k + 1/m$$

y	x	Round off	Pixel
2	2	2	(2,2)
$2+1=3$	$2+\frac{1}{5/3}=2.58$	3	(3,3)
$3+1=4$	$2.58+\frac{1}{5/3}=3.16$	3	(3,4)
$4+1=5$	$3.16+\frac{1}{5/3}=3.74$	4	(4,5)
$5+1=6$	$3.74+\frac{1}{5/3}=4.32$	4	(4,6)
$6+1=7$	$4.32+\frac{1}{5/3}=5$	5	(5,7)

Problems:

→ Even here we are rounding off values which is computationally expensive.

→ $m = 1.6666666667$

we might use

$m \approx 2$ we will deviate from original line.

Round off error.

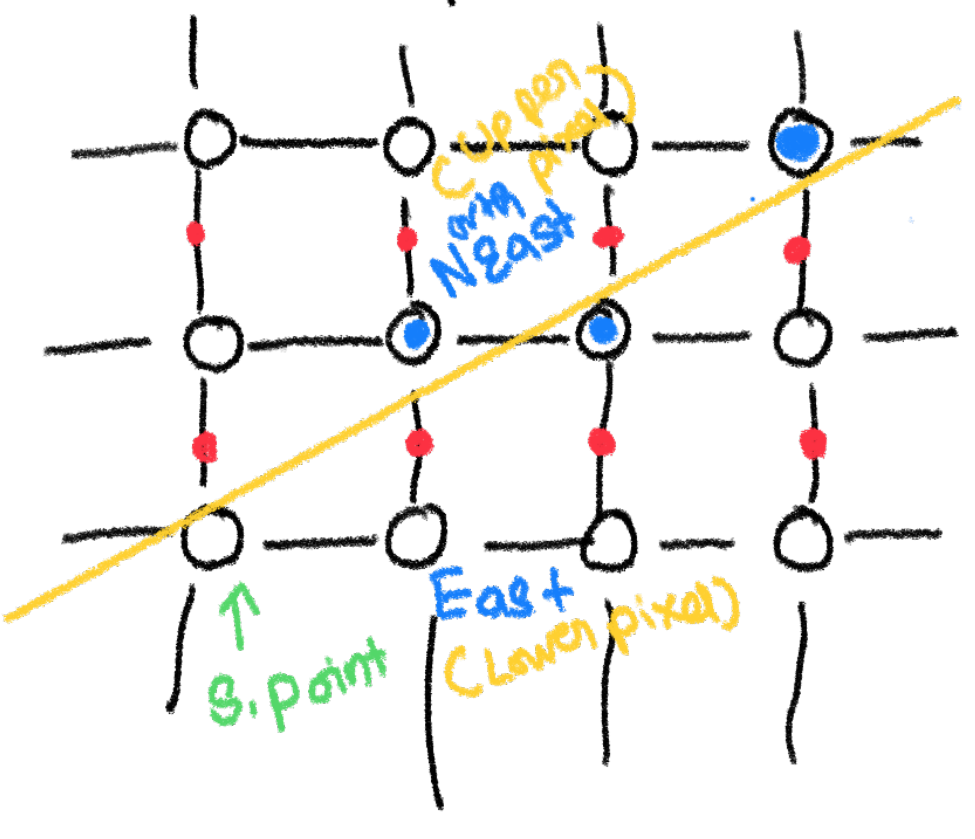
Advan:

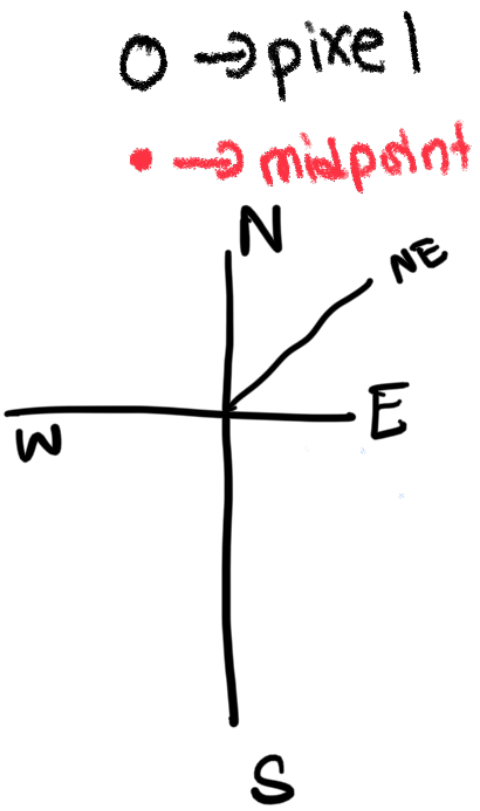
→ Faster

→ No need to multiply

MIDPOINT LINE ALGORITHM

→ Creates midpoint and checks if the line passes above the midpoint or below the midpoint.
The closest pixel is coloured.



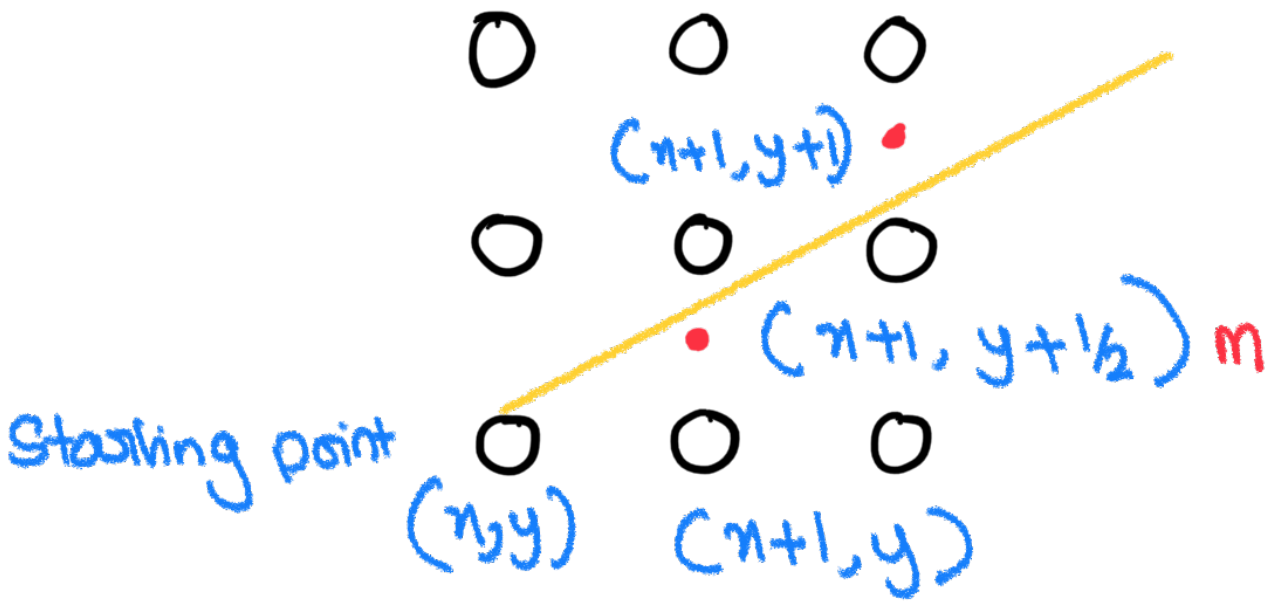


How do we know line is above or below pixel?

$$f(x,y) = ax + by + c$$

Conditions:

f(midpoint)	Line's position	pixel
(-ve)	line below midpoint	lower pixel (E)
(+ve)	line above midpoint	Upper pixel (NE)



New midpoint value is dependent on previous midpoint value & also NE/E.

Now, we need to find $f(M)$

$$f(M) = f(x+1, y+1/2)$$

$$d_{old} = a(x+1) + b(y+1/2) + c$$

Case 1 NE UPPER PIXEL

NE $(x+1, y+1)$

$$f(M_1) = f(x+1+1, y+1/2+1)$$

$$= f(x+2, y+3/2)$$

$$= a(x+2) + b(y+3/2) + c$$

we know new midpoint is dependent on old midpoint So let's find the relation.

$$f(M_1) - f(M) = ax + 2a + by + 3/2\,b + c$$
$$- \quad ax + a + by + \tfrac{1}{2}b + c$$

$$f(M_1) - f(M) = a + b$$

$$f(M_1) = f(M) + a + b$$

$$d_{new} = d_{old} + a + b$$

CASE 2 Lower Pixel (E)

$$f(M_1) = f(x+1+1, y+1/2)$$

$$= f(x+2, y+1/2)$$

$$= a(x+2) + b(y+1/2) + c$$

$$f(M_1) - f(M) = ax + 2a + by + 1/2\,b + c$$
$$- \; ax + a + by + 1/2\,b + c$$
$$\overline{\qquad\qquad\qquad\qquad\qquad\qquad}$$
$$a$$

$$f(M_1) = f(M) + a$$

$$d_{new} = d_{old} + a$$

Now Let's find a & b

$$y = mx + c \qquad m = \frac{\Delta y}{\Delta x}$$

$$y = \frac{\Delta y}{\Delta x} x + c$$

$$\Delta x\, y = \Delta y\, x + \Delta x\, c$$

$$\Delta x\, y - \Delta y\, x - \Delta x\, c = 0$$

[multiply by -ve]

$$-\Delta x\, y + \Delta y\, x + \Delta x\, c = 0$$

$$\Delta y\, x - \Delta x\, y + \Delta x\, c = 0 \quad [ax + by + c = 0]$$

$$a = \Delta y$$
$$b = -\Delta x$$
$$c = \Delta x$$

Lets put the value.

$$d_{old} = a(x+1) + b(y+1/2) + c$$
$$= ax + a + by + 1/2\,b + c$$
$$= \underbrace{ax + by + c}_{0} + a + 1/2\,b$$
$$= a + 1/2\,b$$
$$= \Delta y - 1/2\,\Delta x$$

we need to convert to integers from floating point.

$$2ax + 2by + 2c = 0$$
$$f(x,y) = 2(ax + by + c) = 0$$
$$d_{old} = 2\{a(x+1) + b(y+1/2) + c\}$$
$$d_{old} = 2\{a + 1/2\,b\}$$
$$d_{old} = 2a + b$$
$$d_{old} = 2\Delta y - \Delta x$$

For NE,

$$d_{new} = d_{old} + \boxed{2a + 2b} \quad NE$$

For E, $d_{new} = d_{old} + \boxed{2a}$ E

$$dNE = 2(a+b)$$

$$\boxed{dNE = 2(\Delta y - \Delta x)}$$

$$dE = 2a$$

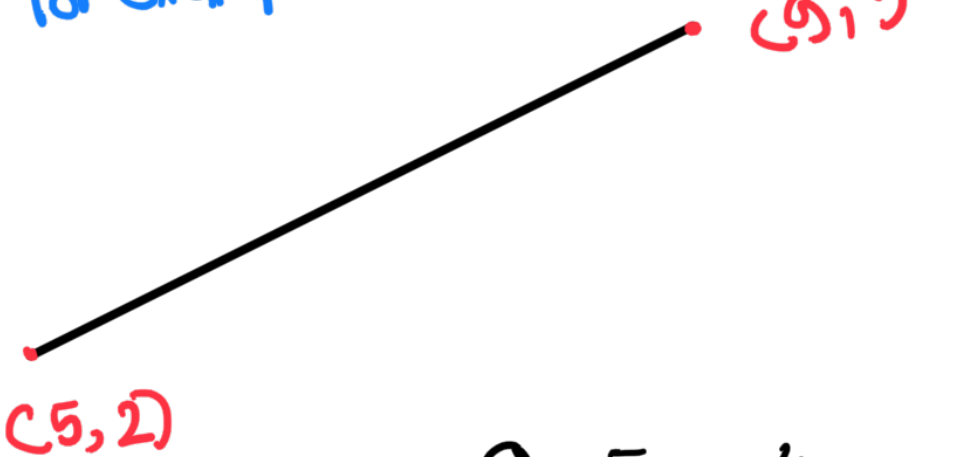
$$\boxed{dE = 2\Delta y}$$

Algorithm :

```
Void drawline(int x0, int y0, int x1, int y1) {

int Δx = x1 - x0 ;
int Δy = y1 - y0 ;
int dold = 2Δy - Δx ;
int dE = 2 * Δy
int dNE = 2 * (Δy - Δx) ;
int x = x0 , int y = y0
    draw pixel (x,y) ;
    while (x < x1 ) {
if (dold < 0) { ΔE
    x++ ;
    dold = dold + dE ; }
else,          { ΔNE
    x++ ;
    y++ ;
    dold = dold + dNE ;
    }
draw pixel (x, y) ,
    }
```

For example :



$\Delta x = 9 - 5 = 4$

$\Delta y = 5 - 2 = 3$

$d_0 = 2 \times 3 - 4$

$= 2$

$dE = 2 * 3 = 6$

$dNE = 2(3-4)$

$= -2$

$x = 5$, $y = 2$ (pixel)

$5 < 9$

d_0 = +ve $\therefore$ NE

① $\Delta NE \longrightarrow$ $x = 5 + 1 = 6$

$y = 3$

$d = 2 - 2 = 0$

$x = 6$, $y = 3$ (pixel)

② $\Delta NE \longrightarrow$ $x = 6 + 1 = 7$

$y = 3 + 1 = 4$

$d = 0 - 2 = -2$

$x = 7$, $y = 4$ (pixel)

③ $\Delta E \rightarrow$ $x = 8$

$y = 4$

$d = -2 + 6 = 4$

$x = 8$, $y = 4$ (pixel)

) $\Delta NE =$ $x = 9$

$y = 5$

$d = 4 - 2 = 2$

$x = 9$, $y = 5$ (pixel)

Zone 0

(2,2)

```
Void draw_line(int x0, int y0, int x1, int y1)
int Δx = x1 - x0 ;
int Δy = y1 - y0 ;
int d_old = 2Δy - Δx ;
int dE = 2 * Δy ;
int dNE = 2 * (Δy - Δx) ;
int x = x0 , int y = y0
draw pixel (x, y) ;
while (x < x1) {
    if (d_old < 0) {        ΔE
        x++ ;
```

```
        dold = dold + dE ;
    else {                    ΔNE
        x++ ;
        y++ ;
        dold = dold + dNE ;
    }
    drawpixel (x , y) ;
  }
}
end.
```