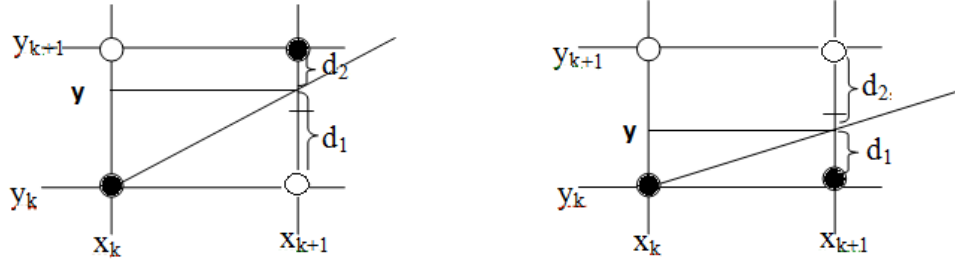


## Bresenham's Line Drawing Algorithm for Lines with Slope $\leq 1$

For the line with slope less than equal to one, the pixel positions are determined by sampling at unit 'x' interval i.e.  $x_{k+1} = x_k + 1$ , with the starting pixel at  $(x_0, y_0)$  from left hand.

For any  $k^{\text{th}}$  step, assuming position  $(x_k, y_k)$  has been selected at previous step, we determine next position  $(x_{k+1}, y_{k+1})$  as either  $(x_{k+1}, y_k)$  or  $(x_{k+1}, y_k + 1)$

At  $x_{k+1}$  label vertical pixel separations from ideal line path as  $d_1$  and  $d_2$ , 'y' coordinate at  $x_{k+1}$  will be  $y = m(x_{k+1}) + c$



The distance of lower pixel from the ideal location  $d_1 = y - y_k$  or  $d_1 = m(x_{k+1}) + c - y_k$

The distance of the ideal location from the upper pixel  $d_2 = y_{k+1} - y$  or  $d_2 = y_{k+1} - m(x_{k+1}) - c$

Thus the difference between the separations of two pixel positions from the actual line path,

$$d_1 - d_2 = m(x_{k+1}) + c - y_k - y_{k+1} + m(x_{k+1}) + c$$

$$d_1 - d_2 = 2m(x_{k+1}) + 2c - 2y_k - 1$$

Substituting  $m = \Delta y / \Delta x$ , we get

$$\Delta x (d_1 - d_2) = 2 \Delta y \cdot x_{k+1} + 2 \Delta x \cdot c - \Delta x \cdot 2y_k - \Delta x$$

$P_k = \Delta x \cdot (d_1 - d_2) = 2 \Delta y \cdot x_{k+1} - 2 \Delta x \cdot y_k + b \dots (i)$  where  $b = 2 \Delta y + \Delta x \cdot 2c - \Delta x$  and  $P_k$  is the decision parameter at the  $k^{\text{th}}$  step

At the  $k+1^{\text{th}}$  step

$$P_{k+1} = 2 \Delta y \cdot x_{k+1} - 2 \Delta x \cdot y_{k+1} + b \dots (ii)$$

Now subtracting (i) and (ii)

$$P_{k+1} - P_k = 2 \Delta y \cdot (x_{k+1} - x_k) - 2 \Delta x \cdot (y_{k+1} - y_k)$$

Since the slope of the line is less than one, we sample in 'x' direction i.e.  $x_{k+1} - x_k = 1$  so,

$$P_{k+1} - P_k = 2 \Delta y - 2 \Delta x \cdot (y_{k+1} - y_k) \dots (iii)$$

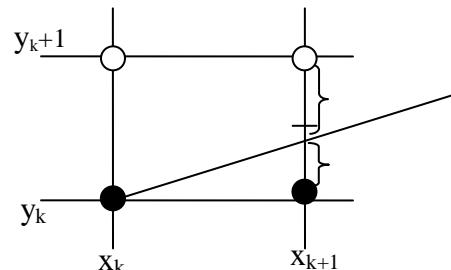
Case 1:

if  $P_k \leq 0$  then the pixel on scanline

' $y_k$ ' is closer to the line path and  $y_{k+1} = y_k$

i.e. from equation (iii)

$$P_{k+1} = P_k + 2 \Delta y$$



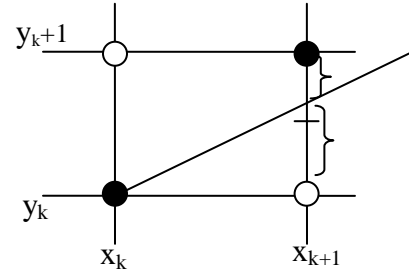
Case 2:

if  $P_k > 0$  then the pixel on scanline

' $y_k + 1$ ' is closer to the line path and  $y_{k+1} = y_k + 1$

i.e. from equation (iii)

$$P_{k+1} = P_k + 2\Delta y - 2\Delta x$$



The Initial Decision Parameter  $P_0 = ?$

We have,

$$d_1 - d_2 = 2m(x_k + 1) + 2c - 2y_k - 1$$

if the line passes thru  $(x_0, y_0)$  then

$$\begin{aligned} d_1 - d_2 &= 2m(x_0 + 1) + 2c - 2y_0 - 1 \\ &= 2mx_0 + 2c - 2y_0 + 2m - 1 \end{aligned}$$

$$\text{or } d_1 - d_2 = 2m - 1 \quad \text{since, } 2mx_0 + 2c - 2y_0 = 0$$

$$\text{or } P_0 = \Delta x (d_1 - d_2) = 2\Delta y - \Delta x$$

## Algorithm

*For*  $|m| \leq 1$

- i. Read  $x_a, y_a, x_b, y_b$  (Assume  $-1 \leq m \leq 1$ )
- ii. Load  $(x_0, y_0)$  into the frame buffer (i.e. plot the first point)
- iii. Calculate constants  $\Delta y, \Delta x, 2\Delta y$  and  $2\Delta y - 2\Delta x$   
Obtain the first decision parameter  $p_0 = 2\Delta y - \Delta x$
- iv. At each  $x_k$  along the line starting at  $k = 0$  perform the following tests:  
If  $p_k < 0$  then the next point to plot is  $(x_k + 1, y_k)$  and  
$$p_{k+1} = p_k + 2\Delta y$$
  
else the next point to plot is  $(x_k + 1, y_k + 1)$  and  
$$p_{k+1} = p_k + 2\Delta y - 2\Delta x$$
- v. Repeat step iv  $\Delta x$  times

Here first we initialize the decision parameter and set the first pixel. Next, during each iteration, we increment 'x' to the next horizontal position, then use the current value of the decision parameter to select the bottom or top pixel (increment y) and update the decision parameter and at the end set the chosen pixel.

### Advantages

It is a faster incremental algorithm that makes use of integer arithmetic calculations only, avoids floating point computations

So it uses faster operations such as addition/subtraction and bit shifting

CPU intensive rounding off operations are avoided

### Disadvantages

It is used for drawing basic lines, antialiasing is not a part of this algorithm so for drawing smooth lines it is not suitable

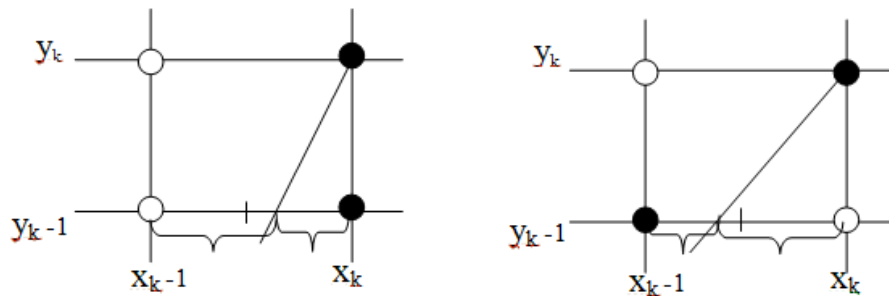
## Bresenham's Line Drawing Algorithm for Lines with Negative slope and Magnitude of the slope greater than one $(-2, -6)$ $(-4, -9)$

For the line with slope greater than equal to one and negative slope, the pixel positions are determined by sampling at unit 'y' interval i.e.  $y_{k+1} = y_k + 1$ , with the starting pixel at  $(x_0, y_0)$  from right hand.

For any  $k^{\text{th}}$  step, assuming position  $(x_k, y_k)$  has been selected at previous step, we determine next position  $(x_{k+1}, y_{k+1})$  as either  $(x_k, y_k - 1)$  or  $(x_k - 1, y_k - 1)$

At  $x_{k+1}$  label vertical pixel separations from ideal line path as  $d_1$  and  $d_2$ , 'y' coordinate at  $x_{k+1}$  will be

$$(y_k - 1) = m x_k + c$$



The distance of right pixel from the ideal location  $d_1 = x - x_k$  or  $d_1 = ((y_k - 1) + c)/m - x_k$

The distance of the ideal location from the left pixel  $d_2 = x_k - 1 - x$  or  $d_2 = x_k - 1 - ((y_k - 1) - c)/m$

Thus the difference between the separations of two pixel positions from the actual line path,

$$d_1 - d_2 = ((y_k - 1) - c)/m - x_k - x_k + 1 + ((y_k - 1) - c)/m$$

$$d_1 - d_2 = 2((y_k - 1) - c)/m - 2x_k + 1$$

$$d_1 - d_2 = 2 \Delta x y_k - 2 \Delta x - 2c \Delta x - \Delta y 2x_k + \Delta y$$

Substituting  $m = \Delta y / \Delta x$ , we get

$$\Delta y (d_1 - d_2) = 2 \Delta x \cdot y_k - 2 \Delta y \cdot x_k + \Delta y - 2 \Delta x - \Delta x \cdot 2c$$

$P_k = \Delta y \cdot (d_1 - d_2) = 2 \cdot \Delta x \cdot y_k - 2 \cdot \Delta y \cdot x_k + b \dots (i)$  where  $b = -2\Delta x + \Delta x \cdot 2c + \Delta y$  and  $P_k$  is the decision parameter at the  $k^{th}$  step

At the  $k+1^{th}$  step

$$P_{k+1} = 2 \cdot \Delta x \cdot y_{k+1} - 2 \cdot \Delta y \cdot x_{k+1} + b \dots (ii)$$

Now subtracting (i) and (ii)

$$P_{k+1} = P_k + 2\Delta x \cdot (y_{k+1} - y_k) - 2\Delta y \cdot (x_{k+1} - x_k)$$

Since the slope of the line is greater than one, we sample in decreasing 'y' direction i.e.  $y_{k+1} - y_k = -1$  so,

$$P_{k+1} = P_k - 2\Delta x - 2\Delta y (x_{k+1} - x_k) \dots (iii)$$

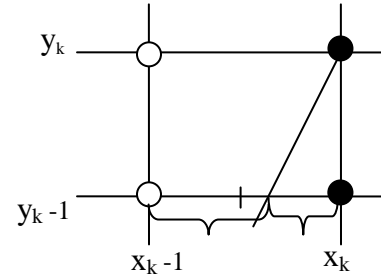
Case 1:

if  $P_k < 0$  then the pixel on scanline

' $x_k$ ' is closer to the line path and  $x_{k+1} = x_k$

i.e. from equation (iii)

$$P_{k+1} = P_k - 2\Delta x$$



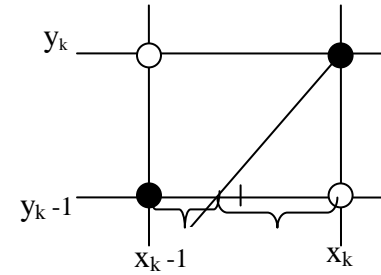
Case 2:

if  $P_k \geq 0$  then the pixel on scanline

' $x_k - 1$ ' is closer to the line path and  $x_{k+1} = x_k - 1$

i.e. from equation (iii)

$$P_{k+1} = P_k - 2\Delta x + 2\Delta y$$



The Initial Decision Parameter  $P_0 = ?$

We have,

$$d_1 - d_2 = 2 \cdot ((y_k - 1) - c)/m - 2x_k + 1$$

if the line passes thru  $(x_0, y_0)$  then

$$d_1 - d_2 = 2 \cdot ((y_0 - 1) - c)/m - 2x_0 + 1$$

$$= 2y_0/m - 2/m - c/m - 2x_0 + 1$$

$$\text{or } d_1 - d_2 = -2/m + 1 \quad \text{since, } 2y_0/m - c/m - 2x_0 = 0$$

$$\text{or } P_0 = \Delta y (d_1 - d_2) = -2\Delta x + \Delta y$$