Bresenham’s Algorithm

Amartya Kundu Durjoy
Lecturer
CSE, UGV

The simple DDA has the disadvantages of using two operations that are expensive in computational time: floating point addition and the round function.

Several good line drawing algorithms avoid these problems by using only integer arithmetic such as Bresenham's line drawing algorithm.

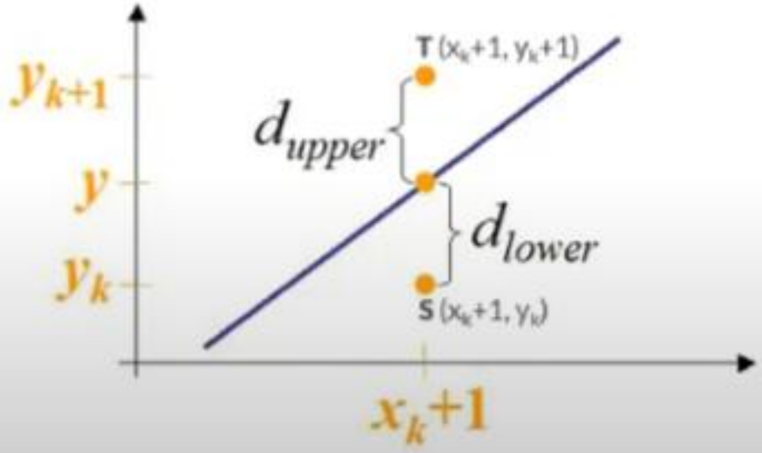
The Bresenham's line drawing algorithm is another incremental scan conversion algorithm.

The big advantage of this algorithm is that it uses only integer calculations.

At sampling position x_k+1, the vertical separations from the mathematical line path are labelled as $d_{upper}(T)$ and $d_{lower}(S)$ as shown in the following figure:

The **y** coordinate on the mathematical line at pixel column $\mathbf{x_k+1}$ is calculated as:

$$\mathbf{y = m(x_k+1) + b}$$



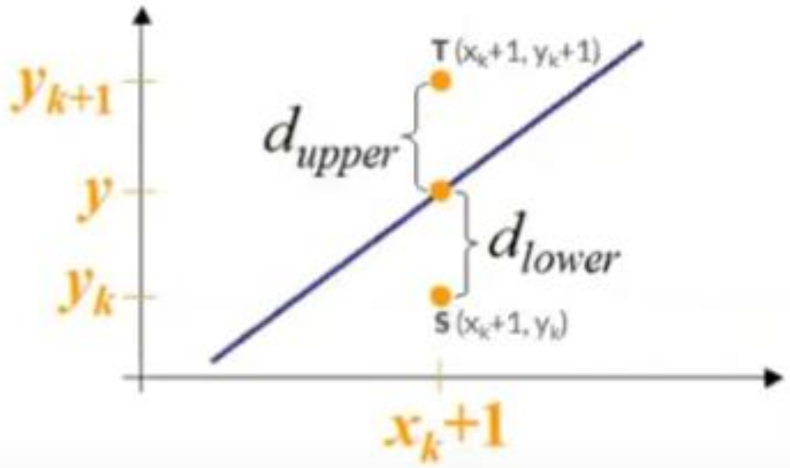
So, $\mathbf{d_{upper}}$ and $\mathbf{d_{lower}}$ are given as follows:

$$d_{lower} = y - y_k = m(x_k + 1) + b - y_k$$

$$d_{upper} = y_{k+1} - y = y_k + 1 - m(x_k + 1) - b$$

The difference between these two separations is

$$d_{lower} - d_{upper} = 2m(x_k + 1) - 2y_k + 2b - 1$$



$$d_{lower} - d_{upper} = 2m(x_k + 1) - 2y_k + 2b - 1$$

Let's substitute m with $\Delta y/\Delta x$ where Δx and Δy are the differences between the end-points:

$$d_{lower} - d_{upper} = 2(\Delta y/\Delta x)(x_k + 1) - 2y_k + 2b - 1$$

$$= (1/\Delta x)[\, 2\Delta y\, x_k + 2\Delta y - 2\Delta x\, y_k + 2\Delta x\, b - \Delta x]$$

$$= (1/\Delta x)[\, 2\Delta y\, x_k - 2\Delta x\, y_k + 2\Delta x\, b - \Delta x + 2\Delta y]$$

$$= (1/\Delta x)[\, 2\Delta y\, x_k - 2\Delta x\, y_k + c] \qquad \text{,Where } c = 2\Delta x\, b - \Delta x + 2\Delta y$$

$$\Delta x\,(d_{lower} - d_{upper}) = 2\Delta y\, x_k - 2\Delta x\, y_k + c$$

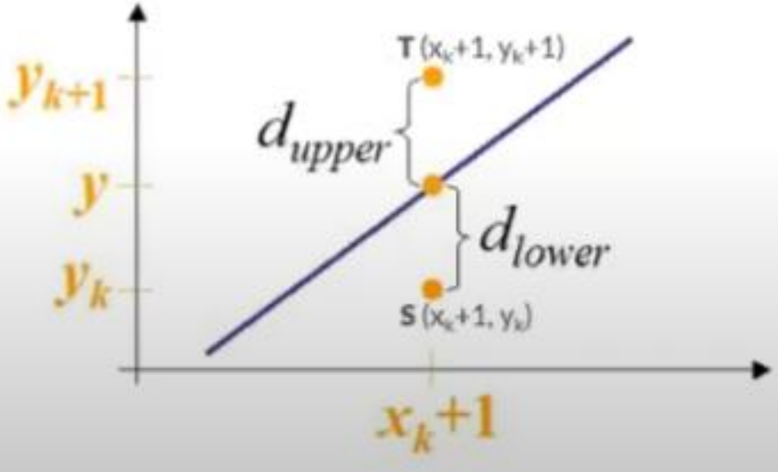
$$\Delta x\,(d_{lower} - d_{upper}) = 2\Delta y\, x_k - 2\Delta x\, y_k + c$$

This equation is used as a simple decision about which pixel is closer to the mathematical line.
So, a decision parameter p_k for the k^{th} step along a line is given by:

$$p_k = \Delta x\,(d_{lower} - d_{upper}) = 2\Delta y\, x_k - 2\Delta x\, y_k + c$$

The sign of the decision parameter p_k is the same the sign of $d_{lower} - d_{upper}$,

When $p_k < 0$, we have $d_{lower} < d_{upper}$ and pixel S is chosen.
When $p_k \geq 0$, we have $d_{lower} \geq d_{upper}$ and pixel T is chosen.



At step (k+1)th decision parameter is given as:

$$p_{k+1} = 2\Delta y\, x_{k+1} - 2\Delta x\, y_{k+1} + c$$

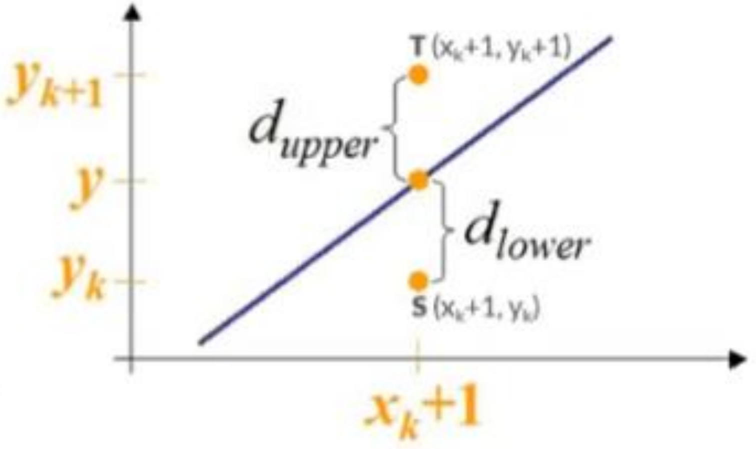
Subtracting p_k from this we get:

$$p_{k+1} - p_k = 2\Delta y(x_{k+1} - x_k) - 2\Delta x(y_{k+1} - y_k)$$

But, $x_{k+1} = x_k + 1$ so that

$$p_{k+1} = p_k + 2\Delta y - 2\Delta x(y_{k+1} - y_k)$$

Where **$(y_{k+1} - y_k)$** is either 0 or 1, depending on the sign of parameter p_k



If **S** is chosen pixel(meaning $p_k < 0$) than $y_{k+1} = y_k$ and so,

$$p_{k+1} = p_k + 2\Delta y$$

If **T** is chosen pixel(meaning $p_k \geq 0$) than $y_{k+1} = y_k + 1$ and so,

$$p_{k+1} = p_k + 2\Delta y - 2\Delta x$$

The recursive calculation of decision parameters is performed at each integer **x** position, starting at the left coordinate endpoint of the line.

The first decision parameter $\mathbf{p_0}$ is evaluated at the starting pixel position $\mathbf{(x_0, y_0)}$:

$$\mathbf{P_k = \Delta x(d_{lower} - d_{upper})}$$

$$P_k = \Delta x[2m(x_k + 1) - 2y_k + 2b - 1]$$

$$P_0 = \Delta x[2mx_0 + 2m - 2y_0 + 2b - 1]$$

$$= \Delta x[2(mx_0 + b - y_0) + 2m - 1] \qquad [\, y=mx+b \text{ or } mx+b-y = 0 \,]$$

$$= \Delta x(2m - 1)$$

$$\mathbf{P_0 = 2\Delta y - \Delta x}$$

Bresenham's Line Algorithm

1. Input the two line end-points, storing the left end-point in (x_0, y_0)
2. Plot the point (x_0, y_0)
3. Calculate the constants Δx, Δy, 2Δy, and (2Δy - 2Δx) and get the first value for the decision parameter as:

$$P_0 = 2\Delta y - \Delta x$$

4. At each x_k along the line, starting at **k=0**, perform the following test:

   If $p_k < 0$, the next point to plot is (x_k+1, y_k) and

$$p_{k+1} = p_k + 2\Delta y$$

   Otherwise, the next point to plot is (x_k+1, y_k+1) and

$$p_{k+1} = p_k + 2\Delta y - 2\Delta x$$

5. Repeat step-4 (Δx) times

To illustrate the algorithm, we digitize the line with endpoints **(2,0)** and **(7, 4)**. This line has a slope of 0.8, with

$$\Delta x = 5, \Delta y = 4$$

The initial decision parameter has the value $P_0 = 2\Delta y - \Delta x = 3$

And the increments for calculation successive decision parameters are:

$$2\Delta y = 8,$$

$$2\Delta y - 2\Delta x = -2$$

We plot the initial point **(2, 0)**, and determine successive pixel positions along the line path from the decision parameters as:

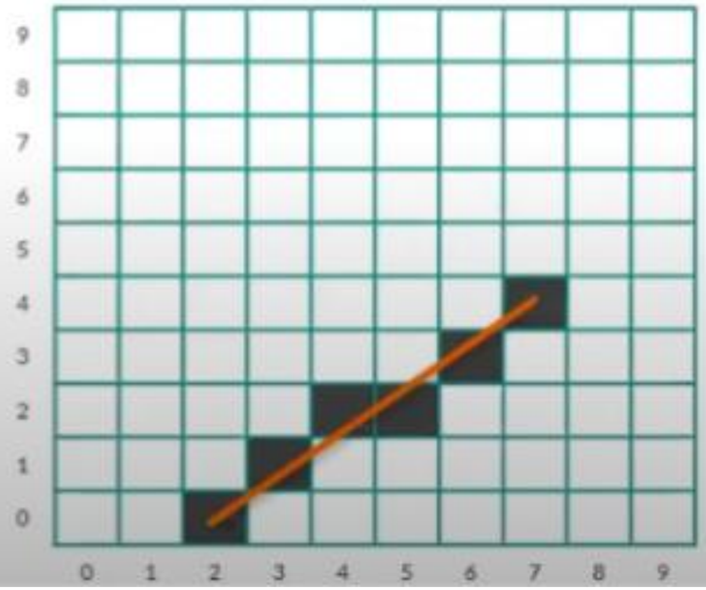
If $p_k < 0$, the next point to plot is (x_k+1, y_k) and

$$p_{k+1} = p_k + 2\Delta y = p_k + 8$$

Otherwise, the next point to plot is (x_k+1, y_k+1) and

$$p_{k+1} = p_k + 2\Delta y - 2\Delta x = p_k - 2$$

k	p_k	(x_{k+1}, y_{k+1})
0	3	(3,1)
1	1	(4,2)
2	-1	(5,2)
3	7	(6,3)
4	5	(7,4)



Self Study

- Mid point circle algorithm
- Bresenham’s circle algorithm,