Bresenham’s Line Drawing Algorithm

Nancy Victor
Assistant Professor
SITE, VIT University,
Vellore

DDA Algorithm Drawbacks

- DDA is the simplest line drawing algorithm
  - Not very efficient
  - Round off operation is expensive
- Optimized algorithms typically used.
  - Integer DDA
  - E.g.Bresenham algorithm
- Bresenham algorithm
  - Incremental algorithm: current value uses previous value
  - Deals with integer addition, subtraction and multiplication by two.

Bresenham's Line Drawing Algorithm

- Basic Principle:

  To find optimum raster locations to represent straight lines.

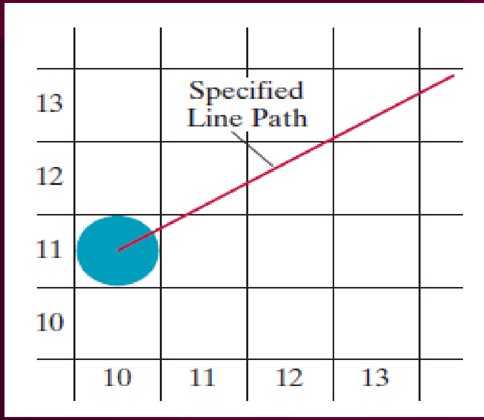
- This algorithm always increments x or y by one unit depending on the slope of the line.
- Increment in the other variable is found on the basis of the distance between the actual line location and the nearest pixel. This distance is called as the decision variable.
- The vertical axis show scan line positions and horizontal axis identify pixel columns.

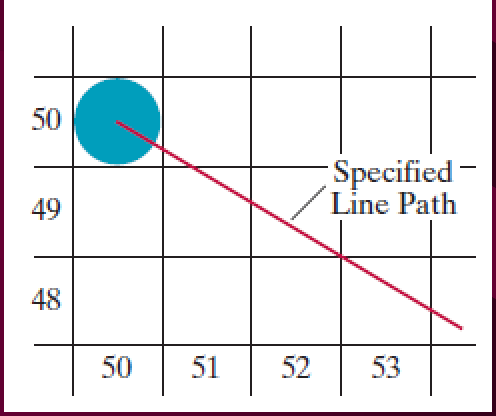
Bresenham's Line Drawing Algorithm

- Given two endpoints (x1, y1) and (x2, y2), we can chose the start point (xk, yk).
- The choice is purely arbitrary, it can be either of (x1, y1) and (x2, y2) points.
- From this start point or pixel, we have eight possible choices for the next pixel in the line, since each pixel is surrounded by 8 other pixels (except border pixels).
- If we consider the scan conversion process for lines with positive slope less than 1, we can isolate two choices out of these 8 choices.

Bresenham's Line Drawing Algorithm

- We need to determine at the next sample position whether to plot the pixel at position (11, 11) or the one at (11, 12).
- Do we select the next pixel position as (51, 50) or as (51, 49)?
- We can find this by testing the sign of the integer parameter, that provides a measure of the relative distances of the two pixels from the actual position on a given line.

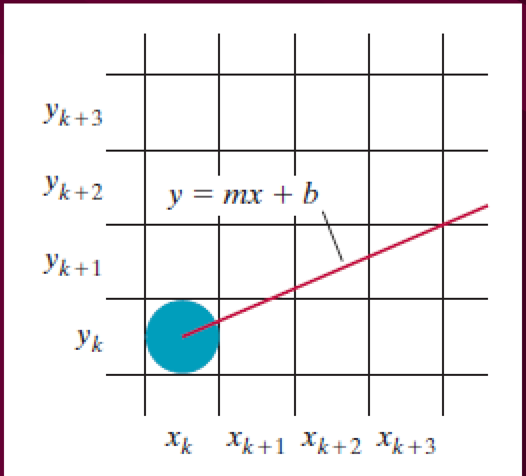




Bresenham's Line Drawing Algorithm

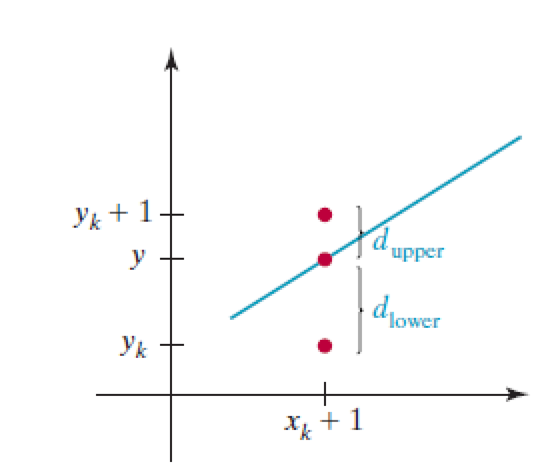
- For lines with positive slope less than 1, pixel positions are determined by sampling at unit x intervals.
- Starting from (x_0,y_0) of a given line, we step to each x position and plot the pixel whose scan line y value is closest to the line path.
- Assuming we have determined that the pixel at (xk , yk) is to be displayed, we next need to decide which pixel to plot in column

  xk+1 = xk + 1.
- Our choices are the pixels at positions (xk + 1, yk) and (xk + 1, yk + 1).



Bresenham's Line Drawing Algorithm

- At sampling position xk + 1, we label vertical pixel separations from the mathematical line path as dlower and dupper .



- The y coordinate on the mathematical line at pixel column position xk + 1 is calculated as

$$y = m(xk + 1) + b$$

Bresenham’s Line Drawing Algorithm

$$
\begin{aligned}
d_{\text{lower}} &= y - y_k \\
&= m(x_k + 1) + b - y_k
\end{aligned}
$$

$$
\begin{aligned}
d_{\text{upper}} &= (y_k + 1) - y \\
&= y_k + 1 - m(x_k + 1) - b
\end{aligned}
$$

- To determine which of the two pixels is closest to the line path, we can set up an efficient test that is based on the difference between the two pixel separations:

  dlower − dupper = 2m(xk + 1) − 2yk + 2b − 1

Bresenham's Line Drawing Algorithm

- A decision parameter pk for the kth step in the line algorithm can be obtained by rearranging the equation so that it involves only integer calculations.
- We accomplish this by substituting m = dy/dx, where dy and dx are the vertical and horizontal separations of the endpoint positions, and defining the decision parameter as

$$
\begin{aligned}
p_k &= \Delta x(d_{\text{lower}} - d_{\text{upper}}) \\
&= 2\Delta y \cdot x_k - 2\Delta x \cdot y_k + c
\end{aligned}
$$

Bresenham's Line Drawing Algorithm

- The sign of p_k is the sameas the sign of dlower − dupper, since $dx > 0$ for our example.
- Parameter c is constant and has the value $2dy + dx(2b - 1)$,which is independent of the pixel position and will be eliminated in the recursive calculations for p_k.
- If the pixel at y_k is "closer" to the line path than the pixel at $y_k + 1$ (that is, dlower < dupper), then decision parameter p_k is negative.
- In that case, we plot the lower pixel; otherwise we plot the upper pixel.

Bresenham's Line Drawing Algorithm

- At step k + 1, the decision parameter is

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

- Let's get rid of multiplications
  - $P_{k+1} - P_k = 2\ \Delta Y\ (x_{k+1} - x_k) - 2\ \Delta X\ (y_{k+1} - y_k)$ (get rid of the constants)

$$P_{k+1} = P_k + 2\ \Delta Y - 2\ \Delta X\ (y_{k+1} - y_k)$$

- $P_{k+1} \quad = P_k + 2\Delta Y$ or

  $= P_k + 2(\Delta Y - \Delta X)$

with $(y_{k+1} - y_k) = 0$ or 1 depending on P_k sign

Bresenham's Line Drawing Algorithm

- This recursive calculation of decision parameters is performed at each integer x position, starting at the left coordinate endpoint of the line.
- The first parameter, p0, is evaluated from the equation at the starting pixel position (x0, y0) and with m evaluated as dy/dx:

$$
\begin{aligned}
P_0 &= \Delta X\,(d_1 - d_2) \\
&= \Delta X[2m(x_0+1) - 2y_0 + 2b - 1] \\
&= \Delta X[2(mx_0 + b - y_0) + 2m - 1]
\end{aligned}
$$

m=Δy/x

0

$$\boxed{P_0 = 2\Delta Y - \Delta X}$$

Bresenham's Line Drawing Algorithm

Bresenham's Line-Drawing Algorithm for $|m| < 1.0$

1. Input the two line endpoints and store the left endpoint in (x_0, y_0).
2. Set the color for frame-buffer position (x_0, y_0); i.e., plot the first point.
3. Calculate the constants Δx, Δy, $2\Delta y$, and $2\Delta y - 2\Delta x$, and obtain the starting 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. Perform step 4 $\Delta x - 1$ times.

Example

- To illustrate the algorithm, we digitize the line with endpoints (20,10) and (30,18). This line has slope of 0.8, with

  $\Delta x = 10$

  $\Delta y = 8$

- The initial decision parameter has the value

  $p_0 = 2\Delta y - \Delta x = 6$

  and the increments for calculating successive decision parameters are

  $2\,\Delta y = 16$

  $2\,\Delta y - 2\,\Delta x = -4$

Example

- We plot the initial point (x_0 , y_0)=(20,10) and determine successive pixel positions along the line path from the decision parameter as

K	p_k	(x_{k+1}, y_{k+1})	**K**	p_k	(x_{k+1}, y_{k+1})
0	6	(21,11)	5	6	(26,15)
1	2	(22,12)	6	2	(27,16)
2	-2	(23,12)	7	-2	(28,16)
3	14	(24,13)	8	14	(29,17)
4	10	(25,14)	9	10	(30,18)

Example

A plot of the pixels generated along this line path is shown in Fig.

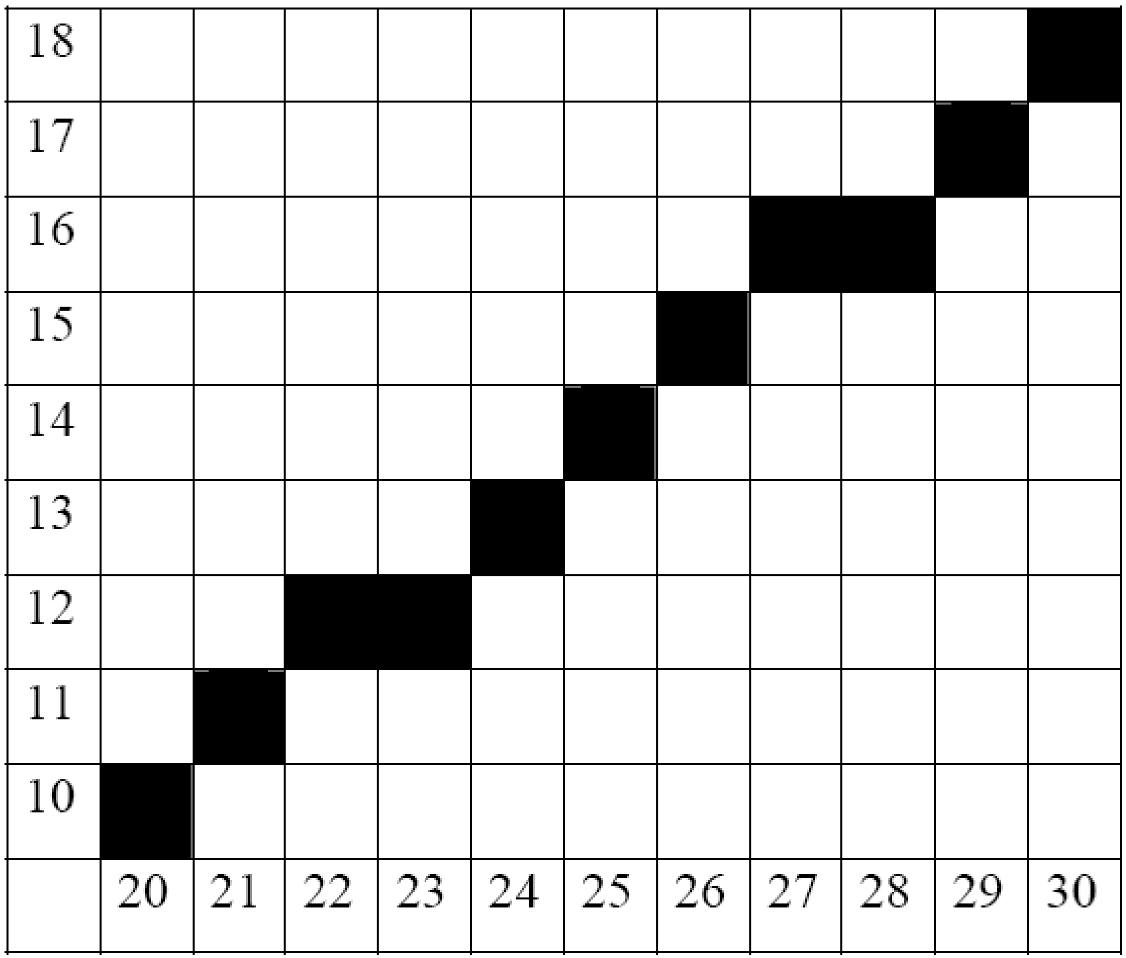


Figure: The Bresenham line from point (20,10) to point (30,18)

Thank You…