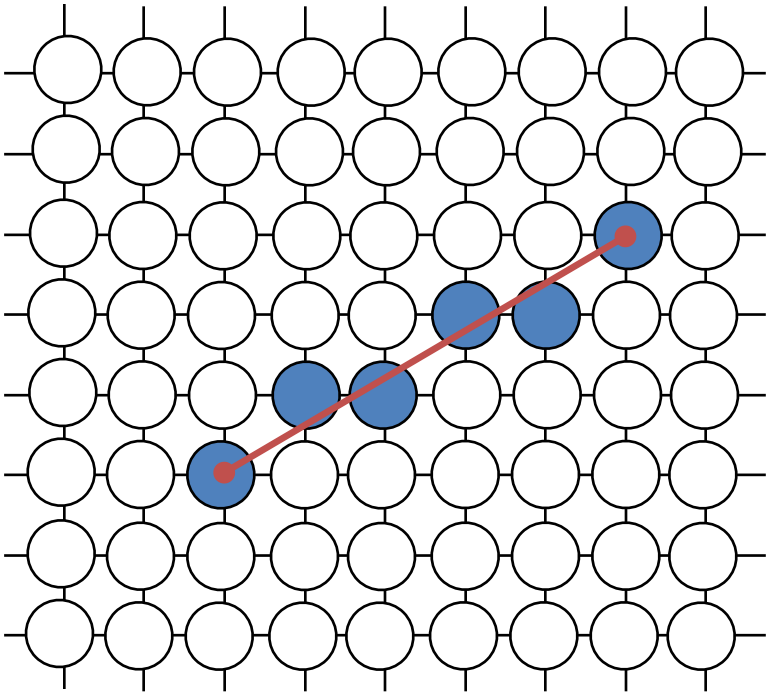
Bresenham's Line Algorithm

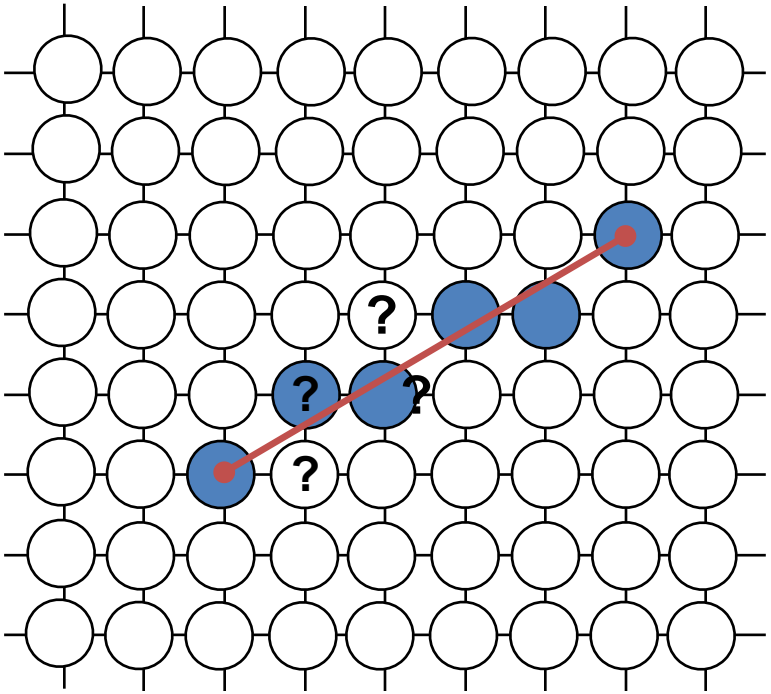
The Problem (cont…)

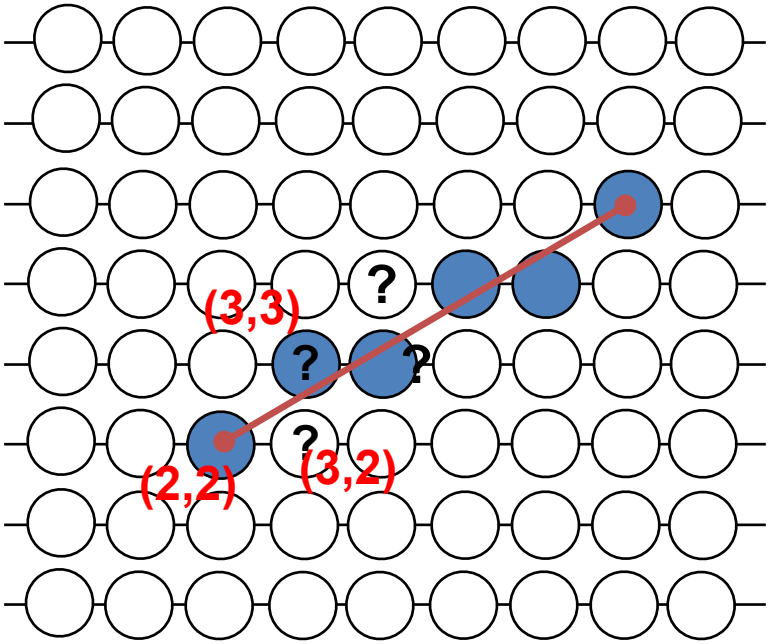
- What happens when we try to draw this on a pixel based display?



How do we choose which pixels to turn on?

- In Bresenham's line drawing algorithm the incremental integer calculations are used to scan convert the lines, so that, the circles and the other curves can be displayed.





- .The next sample positions can be plotted either at (3,2) or (3,3) ?

Advantages of DDA

- ✓ It calculates the pixel positions faster than the calculations performed by using the equation y=mx +b.
- ✓ Multiplication is eliminated as the x and y increments are used to determine the position of the next pixel on a line

Disadvantages of DDA

- ✓ The rounding and floating point operations are time consuming.
- ✓ The round-off error which results in each successive addition leads to the drift in pixel position, already calculated

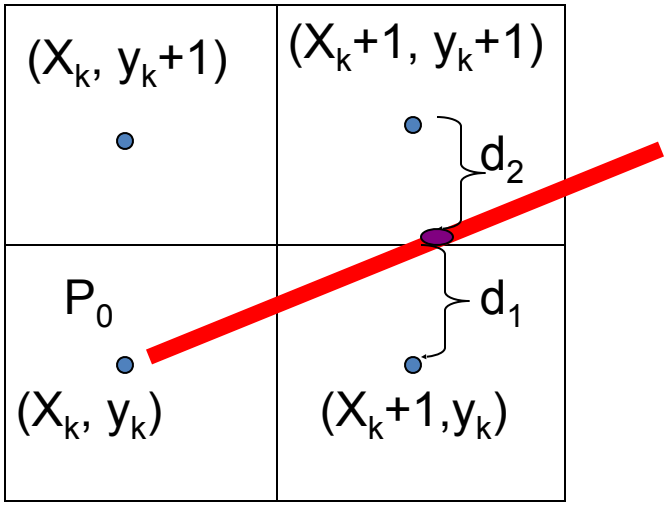
Bresenham's line algorithm

Jack Bresenham worked for 27 years at IBM before entering academia. Bresenham developed his famous algorithms at IBM in the early 1960s

For lines with positive slope m<1

- The pixel positions on line can be identified by doing sampling at unit x intervals.
- The process of sampling begins from the pixel position (X0,Y0) and proceeds by plotting the pixels whose 'Y' value is nearest to the line path.
- If the pixel to be displayed occurs at a position (X_k, Y_k) then the next pixel is either at (X_k+1, Y_k) or (X_k+1, Y_k+1) i.e, (3,2) or (3,3)
- The 'Y' coordinate at the pixel position Xk+1 can be obtained from
- $Y=m(X_k+1)+b$ . . . . eq 1

- the separation between (X_{k+1}, Y_k) and (X_{k+1}, Y) is d1 and the separation between (X_{k+1}, Y) and (X_{k+1}, Y_{k+1}) is d2 then
- $d1 = y - y_k$ and
- $d2 = (Y_k + 1) - Y$



- $Y=m(X_k+1)+b$ . . . . . ……. eq 1
- $d1=y - y_k$
- $d1=m(x_k+1)+b - Y_k$ (from eqn (1) …..(2)
- And $d2= (Y_k+1) - Y$
- $=(Y_K+1) - [m(X_k+1)+b]$
- $=(Y_K+1) - m(X_k+1) - b$ …….(3)
- The difference is given as
- d1-d2 =
- $= m(X_k+1)+b-Y_k-[(Y_k+1)-m(X_k+1)-b]$
- $=m(X_k+1)+b-Y_k-(Y_k+1)+m(X_k+1)+b$
- $d1-d2 = 2m(X_k+1)-2Y_k+2b - 1$ ……(4)

Contd..

- A decision parameter P_k can be obtained by substituting m= dy/dx in equation 4
-
- $d1 - d2 = 2m(X_k+1)-2Y_k+2b - 1$
- $= 2\ dy/dx\ (X_k+1) - 2Y_k + 2b - 1$
- $= \dfrac{2\ dy(X_k+1)-2\ dx.Y_k + 2b.dx - dx}{dx}$
- $dx(d1-d2) = 2\ dy(X_k+1)-2\ dx.Y_k + 2b.dx - dx$
- $= 2\ dyX_k+2\ dy-2\ dx.Y_k + 2b.dx - dx$
- $= 2\ dyX_k- 2\ dx.Y_k + c$
- Where, $dx(d1-d2) = Pk$ and
- $c= 2\ dy+ dx(2b-1)$
- $Pk = 2\ dyX_k- 2\ dx.Y_k + c$ . . . . . (5)
-
- The value of c is constant and is independent of the pixel position. It can be deleted in the recursive calculations, of for P_k

- if d1 < d2 (i.e, Yk is nearer to the line path than Yk+1) then, P_k is negative.
- If Pk is -ve, a lower pixel (Y_k)is plotted else, an upper pixel (Y_k+1)is plotted.

- At k+1 step, the value of Pk is given as
- $P_{K+1} = 2\ dyX_{k+1}- 2\ dx.Y_{k+1} + c$ .....(6) (from 5)

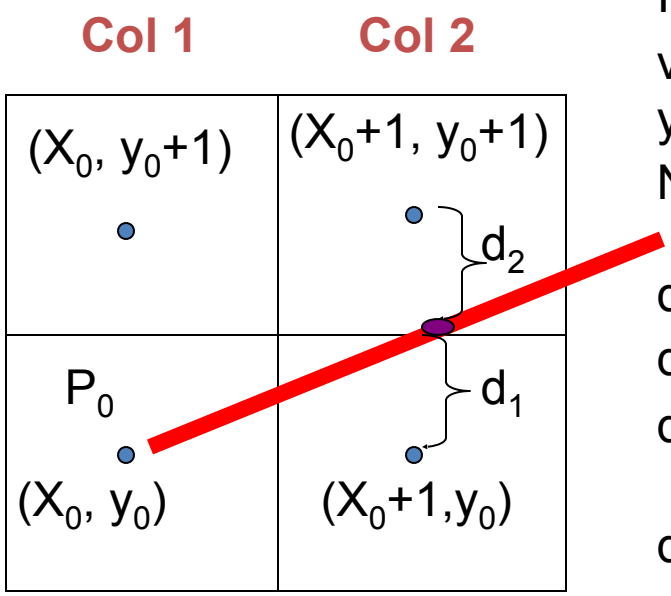
M

- Eq 6 – eq 5
- $Pk+1 - Pk = (2\,dyX_{k+1} - 2\,dx.Y_{k+1} + c) - (\mathbf{2\,dyX_k + 2\,dx.Y_k + c})$
- $= 2dy(X_{k+1} - X_k) - 2\,dx(Y_{k+1} - \mathbf{Y_k})$ **.....(7)**

- Since $X_{k+1} = X_k + 1$ The eqn 7 becomes

- $Pk+1 - Pk = 2dy(X_k + 1 - X_k) - 2\,dx(Y_{k+1} - Y_k)$
- $= 2dy - 2\,dx(Y_{k+1} - Y_k)$
- $Pk+1 = Pk + 2dy - 2\,dx(Y_{k+1} - Y_k)$ **.....(8)**

- Where (Yk+1 – Yk) is either 0 or 1 based on the sign of Pk.
- **The starting parameter P0 at the pixel position (X0,Y0) is given as**
- **P0 = 2dy - dx** **.....(9)**



y=mx+c is the eq of line

In col 2 the line is passing through x_0+1 so the y value is given by

$y=m(x_0+1)+c$ (green dot)

Now we need to find out the values of d1 and d2

$d1= y-y_0$

$d2=(y_0+1)-y$

$d1=m(x_0+1) – y_0$ and $d2= (y0+1) - m(x_0+1)$

$d1-d2=[mx_0+m – y_0] - [(y_0+1) - mx_0-m]$

$=mx_0+m – y_0 - y_0-1 + mx_0+m$

$= 2mx_0+2m –2y_0-1$

$= 2mx_0+2m –2mx_0-1$ (y=mx+c passes thr (x0, y0) so we can say $y_0=mx_0+c$)

$d1-d2 = 2mx_0+2m –2mx_0-1$

$d1-d2=2m-1$ ($m= \Delta y / \Delta x$)

$$\Delta x(d1-d2) = 2\Delta y - \Delta x$$

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

Bresenham's algorithm

- Step 1: Enter the 2 end points for a line and store the left end point in (X0,Y0).
- Step 2: Plot the first point be loading (X0,Y0) in the frame buffer.
- Setp 3: determine the initial value of the decision parameter by calculating the constants dx, dy, 2dy and 2dy-2dx as
- P0 = 2dy –dx
- Step 4: for each Xk, conduct the following test, starting from k= 0

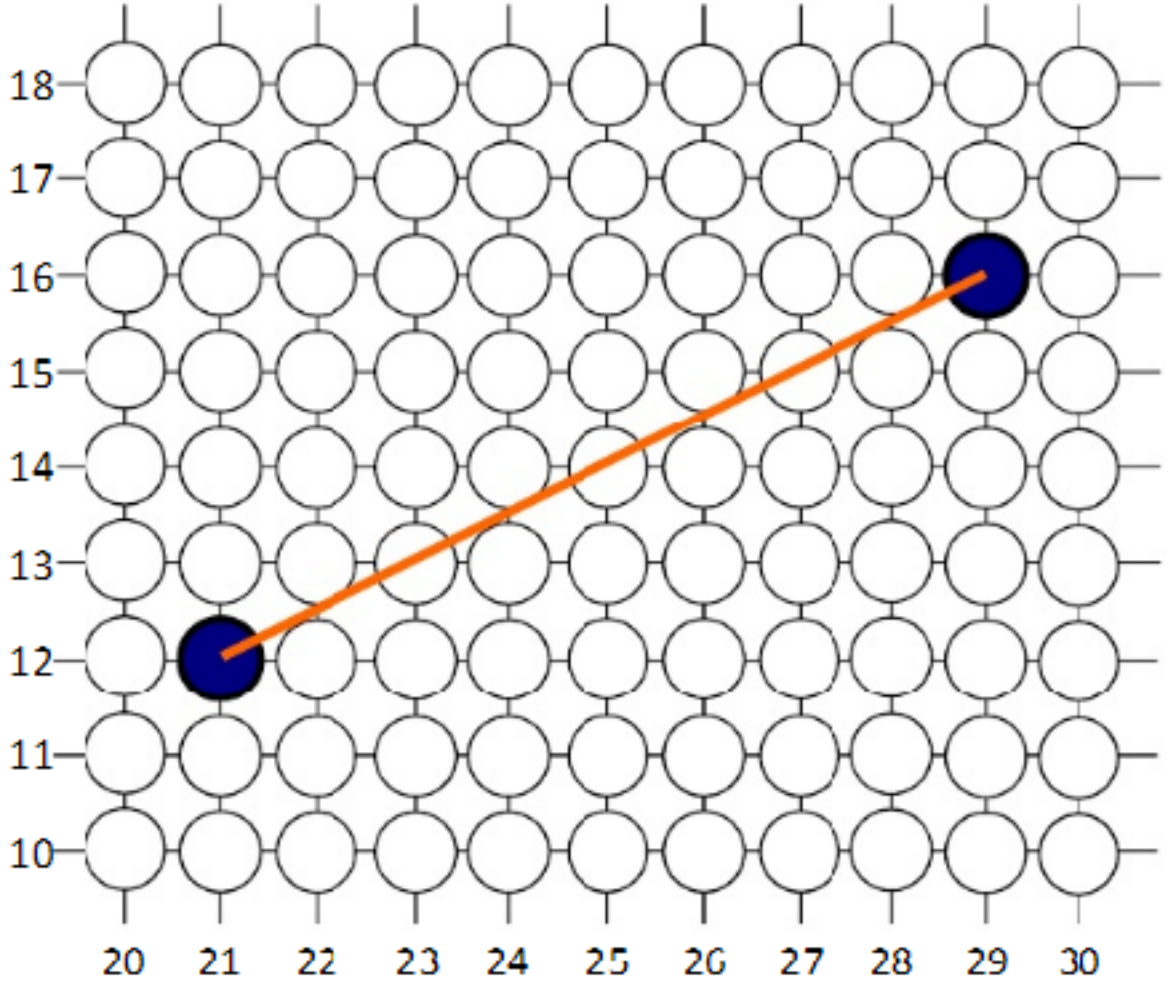
- If Pk <0, then the next point to be plotted is at (Xk+1, Yk) and
- Pk+1 = Pk + 2dy
- Else, the next point is (Xk+1, Yk+1) and
- Pk+1 = Pk + 2dy –2dx (step 3)
- Step 5: iterate through step (4) dx times.

Example

- Let the given end points for the line be (30,20) and (40, 28)
- $M = \frac{dy}{dx} = \frac{y2 - y1}{x2 - x1} = \frac{28 - 20}{40 - 30} = \frac{8}{10}$
- m = 0.8
- dy = 8 and dx = 10

- The initial decision parameter P0 is
- P0 = 2dy – dx = 2(8) – 10 = 16 – 10 = 6
- P0 = 6
- The constants 2dy and 2dy-2dx are
- 2dy = 2(8) = 16 2dy-2dx = 2(8)- 2(10) =16 – = 20
- 2dy = 16 2dy – 2dx = - 4

Bresenham Exercise (cont…)



k	p_k	(x_{k+1}, y_{k-1})
0		
1		
2		
3		
4		
5		
6		
7		
8		

The starting point (x0, y0)=(30,20) and the successive pixel positions are given in the following table

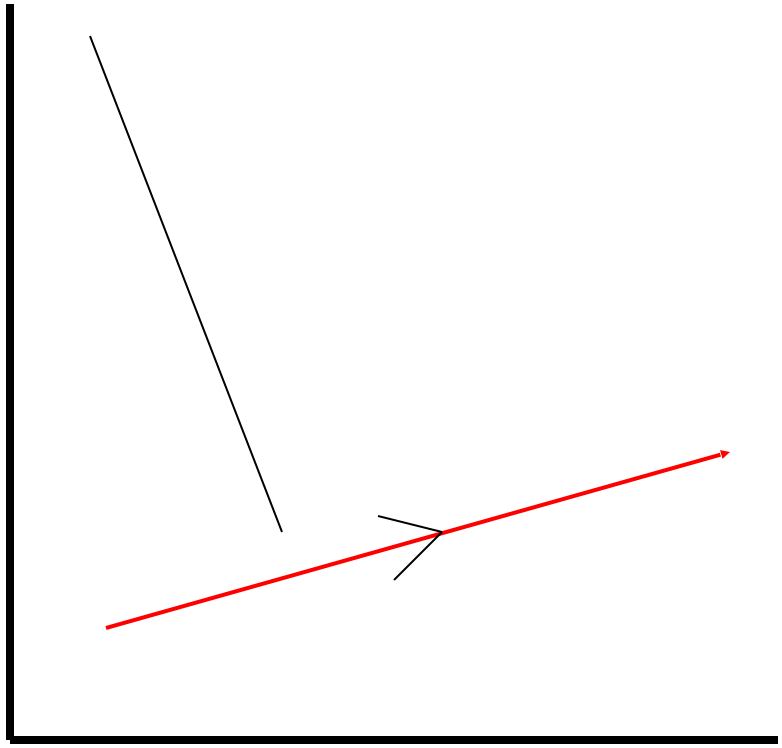
K	P_k	(X_{k+1}, Y_{k+1})
0	6	(31,21)
1	2	(32,22
2	-2	33,22
3	14	34,23
4	10	35,24
5	6	36,25
6	2	37,26
7	-2	38,26
8	14	39,27
9	10	40,28

- End points (30,20) (40,28)
- dx=x2-x1=40-30=10 dy= y2-y1=28-20=8
- P0=2dy-dx=2X8-10 = 6 >0 pt(31,21)
- Pk+1= pk+2dy-2dx= 6+16-20 =2 >0 (32,22)
- Pk+1=2+16-20 =-2 <0 (33,22)
- Pk+1= pk+2dy =-2+16 = 14 >0 (34,23)
- Pk+1= pk+2dy-2dx=14+16-20=10>0 (35,24)
- Pk+1=10+16-20=6 >0 (36,25)
- Pk+1= 6+16-20=2>0 (37,26)
- Pk+1=2+16-20=-2 <0 (38,26)
- Pk+1= pk+2dy=-2+16=14>0 (39,27)
- Pk+1=pk+2dy-2dx= 14+16-20=10>0 (40,28)

- In bresenham's algorithm, if the positive slope of a line is greater than 1, the roles of x and y are interchanged.
- For positive slope lines:

1. If the initial position of a line is the right end point, then both x and y are decremented as we move from right to left.
2. If d1 = d2 then always select the upper or the lowet candidate pixel.

- For negative slope lines:
- One coordinate increases and the other coordinate decreases.
- Special cases:
- the vertical lines dx = 0, horizontal lines dy = 0 and diagonal lines |dx| = |dy| can be directly loaded into the frame buffer.



Positive slope less than one

```
Void LineBres( int x1, int y1, int x2, int y2)
{
 int dx =abs(x2-x1), dy= abs(y2-y1)
 int p=2* dy- dx ;
 int x, y, Xend;     //  Xends is similar to steps in DDA
   If(x1>x2)
   {
       x=x2;y=y2;Xend=x1;
    }
  else
   {
     x=x1;y=y1;Xend=x2;
   }
 setPexel( x, y );
```

```
While( x < Xend)
{
   x++;
    if( p < 0)
       p=p+2dy;
    else
     {
        y++;
        p=p+2dy-2dx;
     }
    setPixel(x,y);
  }   // End While

}   //End LineBres
```

Advantages of Bresenham's Alg.

- ➢It uses only integer calculations
- ➢So, it is faster than DDA