Line Drawing Algorithms

1. A line in Computer graphics is a portion of straight line that extends indefinitely in opposite direction.
2. It is defined by its two end points.
3. Its density should be independent of line length.

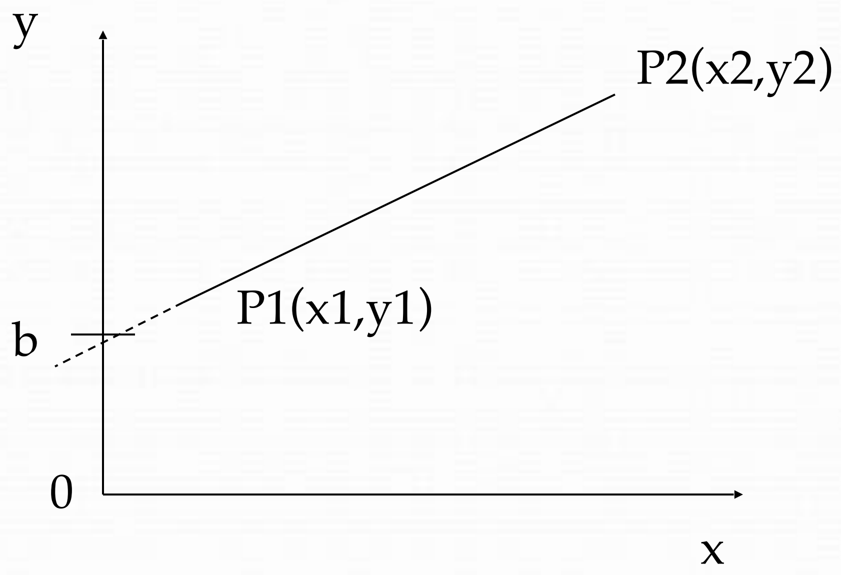
the slope intercept equation for a line:

$$\mathbf{y = mx + b} \qquad (1)$$

where, **m** = Slope of the line

b = the y intercept of a line

The two endpoints of a line segment are specified at positions **(x1,y1)** and **(x2,y2).**



We can determine the value for slope m & b intercept as

m = y2-y1/x2-x1

i.e. m= Δy/ Δx (2)

Example 1 The endpoints of line are(0,0) & (6,18). Compute each value of y as x steps from 0 to 6 and plot the result.

Solution : Equation of line is y= mx +b

m = y2-y1/x2-x1= 18-0/6-0 = 3

Next the y intercept b is found by plugging y1& x1 into the equation y = 3x + b,

0 = 3(0) + b. Therefore, b=0, so the equation for the line is y= 3x.

The challenge is to find a way to calculate the next x,y position by previous one as quickly as possible.

DDA Algorithm

The Digital differential analyzer (DDA) algorithm is an incremental scan-conversion method.

Such an approach is characterized by performing calculations at each step using results from the preceding step.

Algorithm:

(x1,y1) (x2,y2) are the end points and dx, dy are the float variables.

Where dx= abs(x2-x1) and dy= abs(y2-y1)

```
(i)   If dx >=dy then
            length = dx
      else
            length = dy
      endif
```

```
(ii)    dx = (x2-x1)/length
        dy = (y2-y1)/length
(iii)   x = x1 + 0.5
        y = y1 + 0.5
(iv)    i = 0
(v)     Plot ((x), (y))
(vi)      x = x + dx
          y = y + dy
(vii)    i = i + 1
```

(viii) If $i <$ length then go to step (v)

(ix) Stop

Example 2 Scan convert a line having end points (3,2) & (4,7) using DDA.

Solution: dx= x2 - x1 = 4-3 = 1

dy= y2 - y1 = 7-2 = 5

As, $dx < dy$ then

length = y2-y1 = 5

dx = (x2-x1)/ length = 1/5 =0.2

dy = (y2-y1)/ length = 5/5 = 1

x1	y1	x2	y2	L	dx	dy	i	x	y	Result	Plot
3	2	4	7	5	.2	1	0	3.5	2.5	3.5, 2.5	3,2
							1	3.7	3.5	3.7,3.5	3,3
							2	3.9	4.5	3.9,4.5	3,4
							3	4.1	5.5	4.1,5.5	4,5
							4	4.3	6.5	4.3,6.5	4,6
							5	4.5	7.5	4.5,7.5	4,7

Limitations of DDA:

(1) The rounding operation & floating point arithmetic are time consuming procedures.

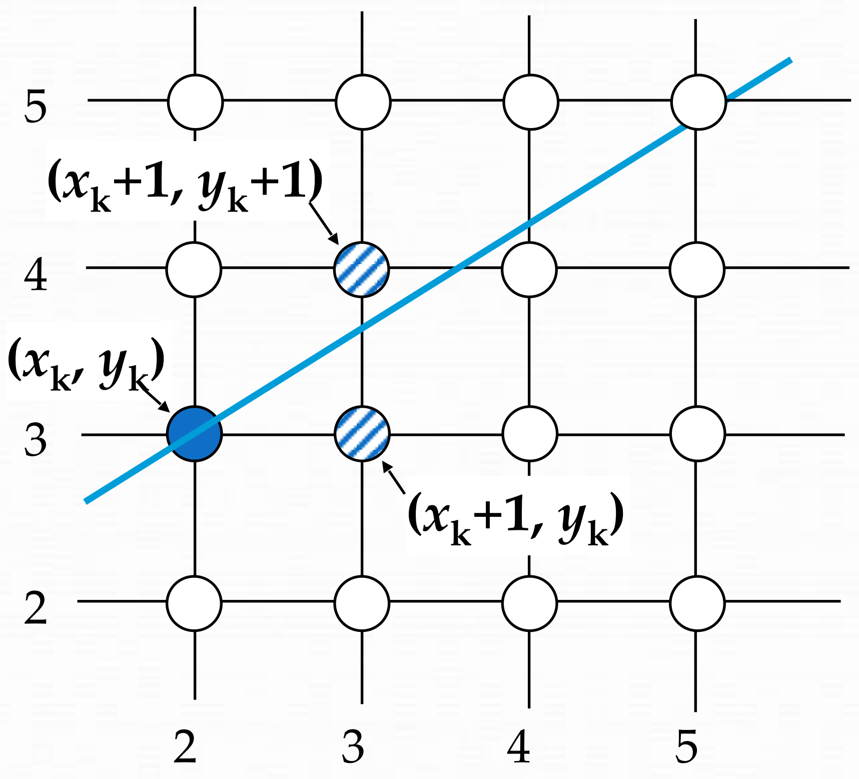
(2) Round-off error can cause the calculated pixel position to drift away from the true line path for long line segment.

The Bresenham Line Algorithm

- The Bresenham algorithm is another incremental scan conversion algorithm
- The big advantage of this algorithm is that it uses only integer calculations

The Big Idea

Move across the *x* axis in unit intervals and at each step choose between two different *y* coordinates

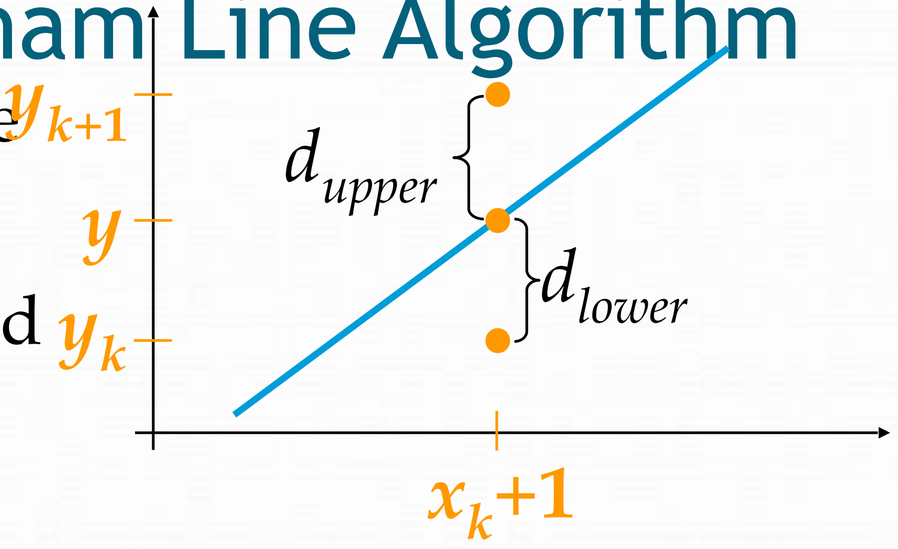


For example, from position (2, 3) we have to choose between (3, 3) and (3, 4)

We would like the point that is closer to

Deriving The Bresenham Line Algorithm

At sample position x_k+1 the vertical separations from the mathematical line are labelled d_{upper} and d_{lower}



The y coordinate on the mathematical line at x_k+1 is:

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

The Bresenham Line Algorithm

BRESENHAM'S LINE DRAWING ALGORITHM

1. Input the two line end-points, storing the left end-point in (x_1, y_1)
2. Calculate the constants Δx *i.e.* dx, Δy *i.e.* dy, $2\Delta y$ and $2\,\Delta x$, get the first value for the decision parameter as:
3. Initialise starting
4. Initialise i=1 as a counter,

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

$$e = e + 2\Delta y$$

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 $(\Delta x - 1)$ times

ACHTUNG! The algorithm and derivation above assumes slopes are less than 1. for other slopes we need to adjust the algorithm slightly.

Adjustment

For $m>1$, we will find whether we will increment x while incrementing y each time.

After solving, the equation for decision parameter p_k will be very similar, just the x and y in the equation will get interchanged.

Bresenham Example

Let's have a go at this

Let's plot the line from (20, 10) to (30, 18)

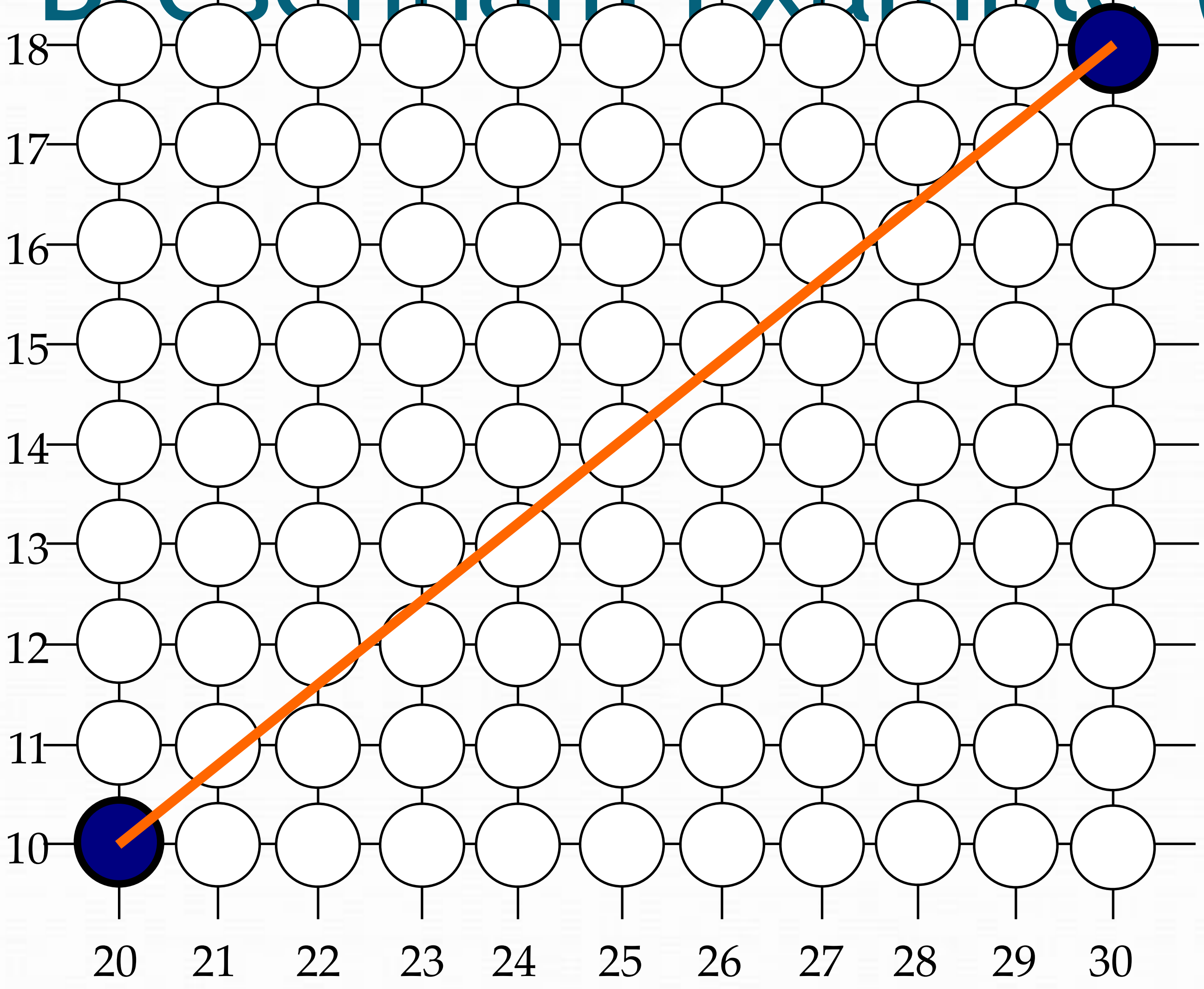
First off calculate all of the constants:

- Δx: 10
- Δy: 8
- $2\Delta y$: 16
- $2\Delta y$ - $2\Delta x$: -4

Calculate the initial decision parameter p_0:

- $p0 = 2\Delta y - \Delta x = 6$

Bresenham Example (cont…)

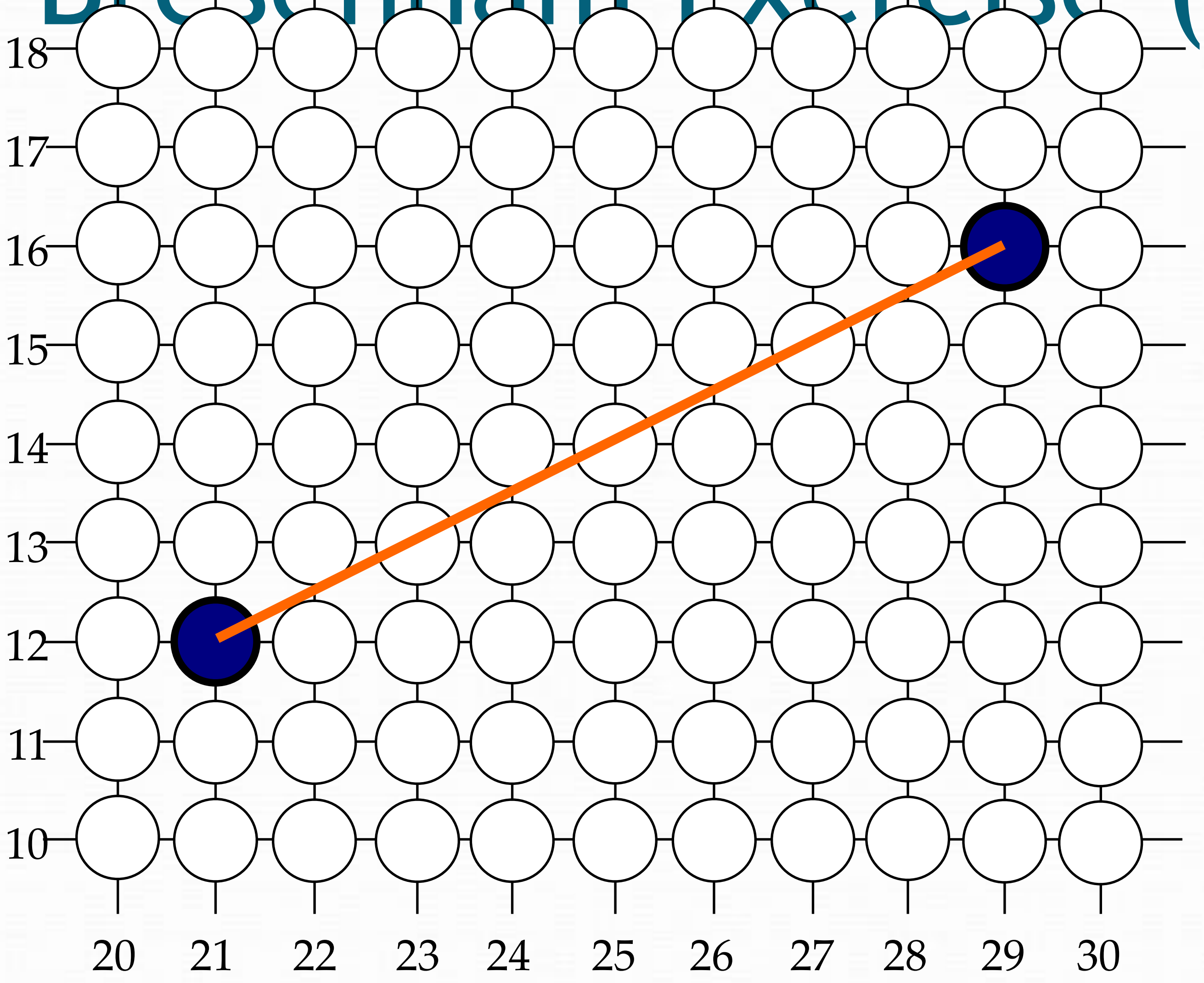


k	p_k	(x_{k+1},y_{k+1})
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		

Bresenham Exercise

Go through the steps of the Bresenham line drawing algorithm for a line going from (21,12) to (29,16)

Bresenham Exercise (cont…)



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

Bresenham Line Algorithm Summary

The Bresenham line algorithm has the following advantages:

- An fast incremental algorithm
- Uses only integer calculations

Comparing this to the DDA algorithm, DDA has the following problems:

- Accumulation of round-off errors can make the pixelated line drift away from what was intended
- The rounding operations and floating point arithmetic involved are time consuming