SCAN CONVERSION

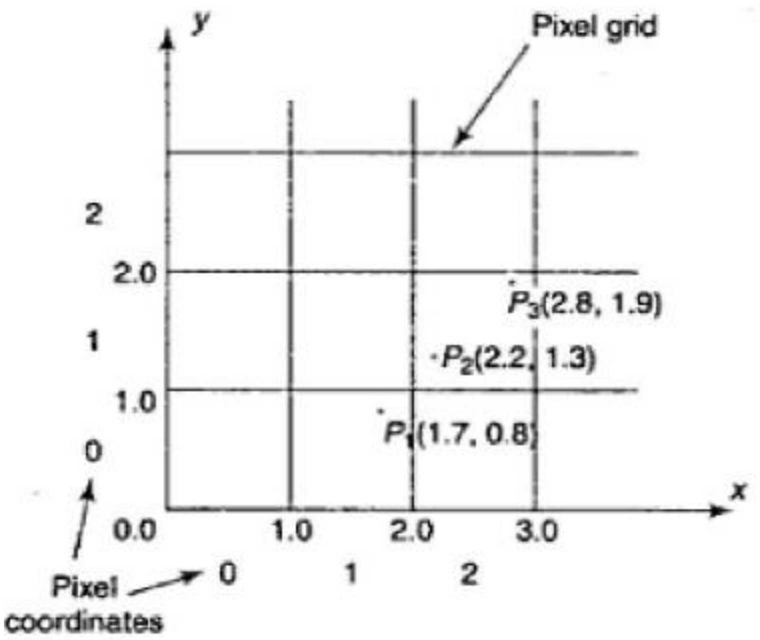
BOOK: COMPUTER GRAPHICS(SCHAUM'S OUTLINE-2ND EDITION), ZHIGANG XIANGM ROY A PLASTOCK

Scan Conversion

- Process of representing graphics objects as a set of pixels.
- Graphics objects are continuous; the pixels used a re discrete.
- Convert each primitive from its geometric definition into a set of pixel is known as scan conversion or rasterization.

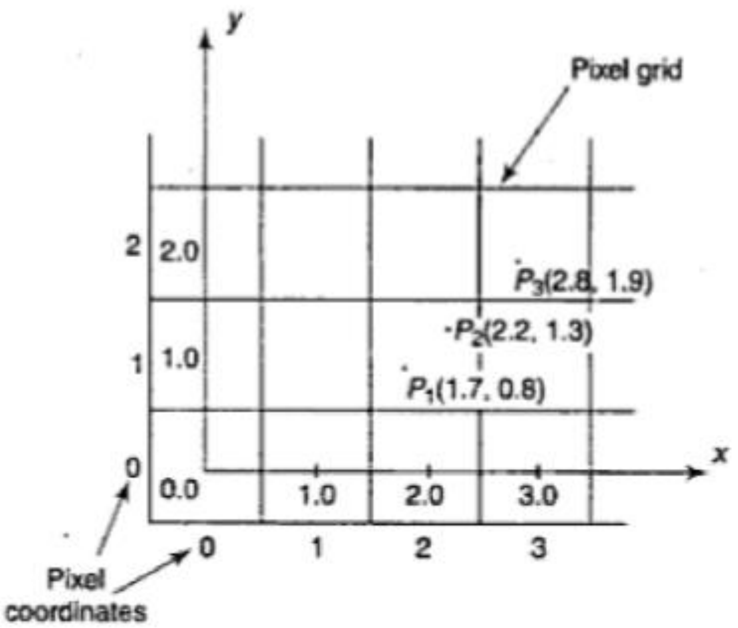
Scan Converting a Point

- A mathematical point (x , y) where a and y are real numbers withing an image area needs to be scan converted to a pixel at location (x' , y').
- x' = Floor (x) and y' = Floor (y)
- The origin of the continuous coordinate system for (x , y) is placed on the lower left corner of the pixel grid in the image space.
- Point $P_1(1.7,0.8)$ is represented by pixel (1,0)



Scan Converting a Point

- Align the integer values in the coordinate system for (x , y) with the pixel coordinates.
- x'=Floor (x+0.5) and y'= Floor (y+0.5)
- The origin of the continuous coordinate system for (x , y) is placed at the center of the pixel grid (0,0) in the image space.
- Point $P_1(1.7,0.8)$ in represented by pixel (2,1).



Scan Converting a Line

- A line in a computer graphics typically refers to a line segment which is a portion of a straight line that extends indefinitely in the opposite direction.
- It is defined by its two end points and the line equation y=mx + b where m is the slope and b is the y intercept of the line.
- The line equation describes the coordinated of all points that lie between the two end points.

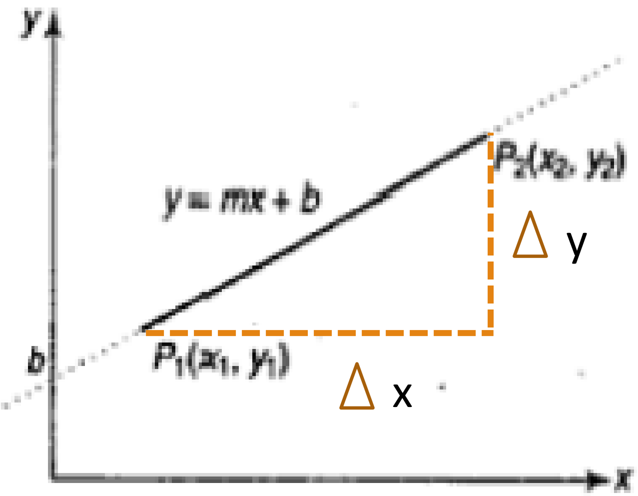


Fig. 3.2 *Defining a Line*

Algorithm for Scan Converting a Line

- Direct use of the line equation (Self Study)
- DDA Algorithm
- Bresenham's Line Algorithm

DDA Algorithm

- The Digital Differential Analyzer (DDA) is an incremental scan conversion method.
- Such an approach is characterized by performing calculation at each step using results from preceding steps.

m	x_{k+1}	y_{k+1}
$m < 1$	$x_{k+1} = x_k + 1$	$y_{k+1} = y_k + m$
$m > 1$	$x_{k+1} = x_k + (1/m)$	$y_{k+1} = y_k + 1$
$m = 1$	$x_{k+1} = x_k + 1$	$y_{k+1} = y_k + 1$

DDA Algorithm (Cont..)

Two end points are given: (x_1,y_1) and (x_2,y_2)

1. Compute dx and dy and compute m.
2. compare between dx and dy.
3. If |dx| is greater than |dy|, step size will be |dx|; otherwise step size will be |dy|.
4. Compute x_{inc} =dx/step and y_{inc} =dy/step.
5. Plot the pixel (x_1,y_1) and then increment x_1 and y_1 $x_1=x_1+x_{inc}$ and $y_1=y_1+y_{inc}$.
6. Continue this process until the iteration reaches to step size.

Problem Solving Using DDA Algorithm

- The end points of a line are (5,4) and (12,7). Computer each value of x and y and plot the results.
- The end points of a line are (5,7) and (10,5). Computer each value of x and y and plot the results.
- The end points of a line are (12,9) and (17,14). Computer each value of x and y and plot the results.
- The end points of a line are (17,14) and (12,9). Computer each value of x and y and plot the results.

[For solution pls check the class lecture]

Limitation of DDA

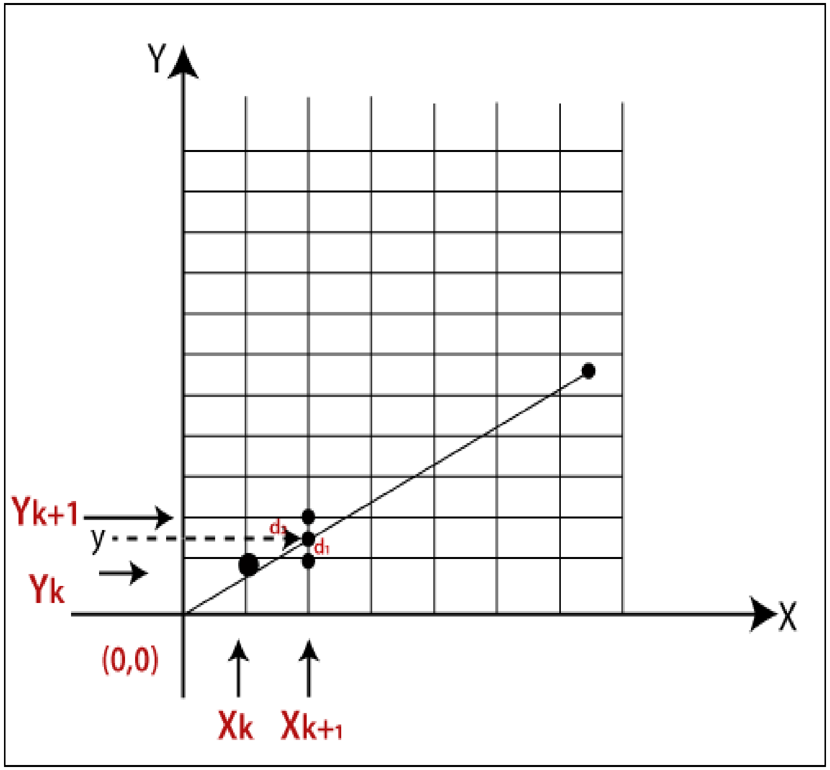
- This algorithm is faster than direct use of line equation since it calculates points on the line without any floating point multiplication.
- It deals with the rounding off operation and floating point arithmetic so it has high time complexity.
- As it is orientation dependent, so it has poor endpoint accuracy.
- Due to the limited precision in the floating point representation it produces cumulative error.

Bresenham's Line Algorithm

- Bresenham's Line Algorithm is a highly efficient incremental method for scan converting lines.
- It produces mathematically accurate results using only integer addition, subtraction and multiplication by 2, which can be accomplished by a simple arithmetic shift operation.

Bresenham’s Line Algorithm

- For scan-converting the line shown in the figure here 0<m<1.
- Start with the first pixel of the line.
- Then select subsequent pixels to the right, one pixel at a time in the horizontal direction towards the end pixel.
- Once a pixel is chosen at any step, the pixel is either the one to its right or the one to its right and up.
- Choose the pixel who has minimum distance from its true path between the first and last pixel of the line.



Bresenham's Line Algorithm

We know that m= (dy/dx) and initial decision variable, p= 2dy-dx (for m<1) and p=2dx-dy(m>1).

m	p	x_{k+1}	y_{k+1}	Updated value of p
m<1	p<0	$x_{k+1}=x_k +1$	$y_{k+1}=y_k$	p = p+2dy
	p≥0	$x_{k+1}=x_k +1$	$y_{k+1}=y_k +1$	p = p+2dy-2dx
m>1	p<0	$x_{k+1}=x_k$	$y_{k+1}=y_k +1$	p = p +2dx
	p≥0	$x_{k+1}=x_k +1$	$y_{k+1}=y_k +1$	p = p+2dx-2dy

** Pls follow class lecture for details explanation

Bresenham's Line Algorithm for |m|<1

1. Given two end points: (x_1,y_1) and (x_2,y_2)
2. Start with first point (x_1,y_1) and plot the first pixel.
3. Compute dx, dy, m and initial value of decision parameter that is p= 2dy-dx.
4. At each x_k, along the line, starting at k=1, perform the following:
   - If p_k <0 , the next point to plot is (xk+1,yk) and $p_{k+1}= p_k +2dy$.
   - Otherwise, the next point to plot is (x_{k+1}, y_{k+1}) and $p_{k+1}= p_k +2dy-2dx$.
5. Repeat this process until the value of x reaches to x_2.

Problem Solving using Bresenham's Algo

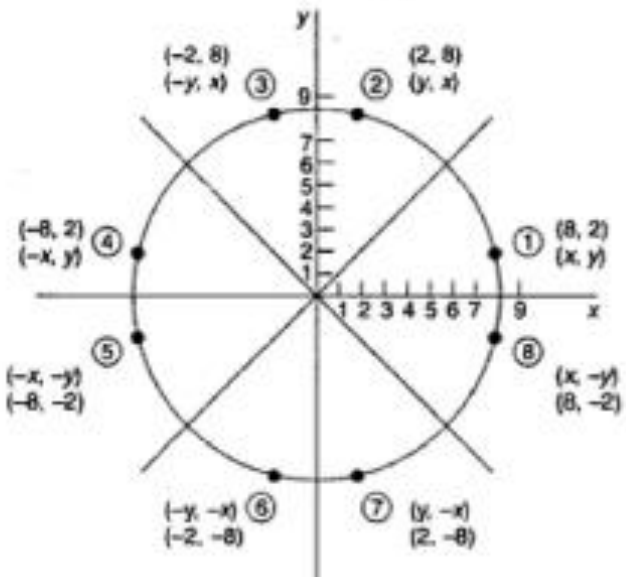
Indicate which raster location would be chosen by Bresenham's algorithm when a scan converting line from pixel coordinate (1,1) to pixel coordinate (8,5).

[For solution, follow the class lecture]

Scan Converting a Circle

- A circle is a symmetrical figure.
- Any circle generating algorithm can take advantage of the circle’s symmetry to plot eight points for each value that the algorithm calculates.
- Eight way symmetry is used by reflecting each calculated point around each 45 degree axis.
- For example, if point 1 in figure was calculated with a circle algorithm, seven more points could be found by reflection.

Scan Converting a Circle



$P_1 = (x, y)$

$P_2 = (y, x)$

$P_3 = (-y, x)$

$P_4 = (-x, y)$

$P_5 = (-x, -y)$

$P_6 = (-y, -x)$

$P_7 = (y, -x)$

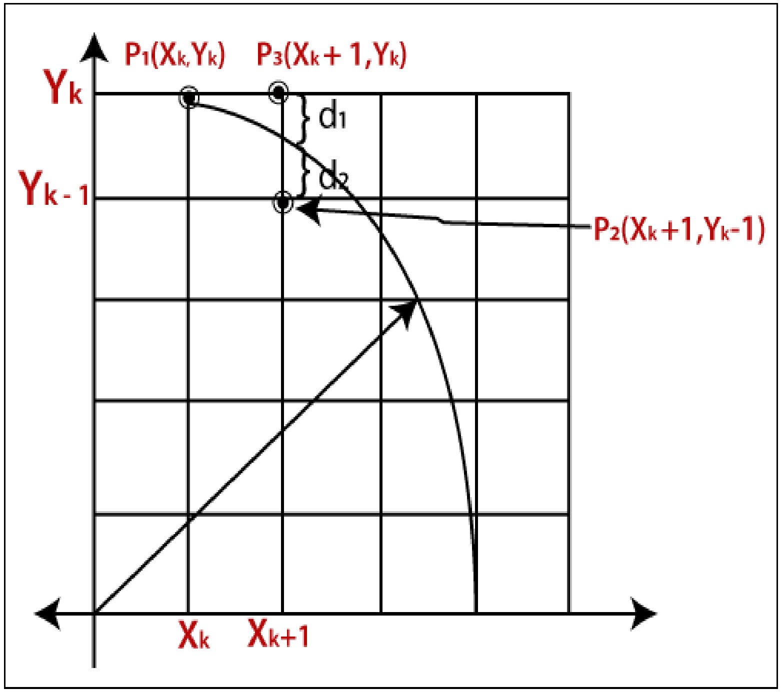
$P_8 = (x, -y)$

Algorithm for Scan Converting a Circle

- Generate circle using polynomial method [Self Study]
- Generate circle using Trigonometric method [Self Study]
- Bresenham’s Circle Algorithm
- Mid Point Circle Algorithm

Bresenham’s Circle Algorithm

- Highly efficient as it produces mathematically accurate results using only integer addition, subtraction and multiplication by 2, which can be accomplished by a simple arithmetic shift operation.
- If the eight-way symmetry of a circle is used to generate a circle, points will be only have to be generated through a 45 degree angle.
- If points are generated from 90 degree to 45 degree, moves will be made only in the +x and –y direction.



Bresenham’s Circle Algorithm

- Initial decision variable, p=3-2r.

p	x_{k+1}	y_{k+1}	Updated value of p
$p \geq 0$	$x_{k+1} = x_k + 1$	$y_{k+1} = y_k - 1$	$p=p+4(x_k - y_k)+10$
$p<0$	$x_{k+1} = x_k + 1$	$y_{k+1} = y_k$	$p=p+4x_k+6$

Bresenham's Circle Algorithm

- Plot initial point (x_k, y_k) such that $x_k=0$ and $y_k=r$.
- Compute the initial decision variable, $p_k=3-2r$.
- If $p_k<0$ then
  $x_{k+1}=x_k+1$
  $y_{k+1}=y_k$
  $p_{k+1}=p_k+4x_k+6$
- If $p_k\geq 0$ then
  $x_{k+1}=x_k+1$
  $y_{k+1}=y_k-1$
  $p_{k+1}=p_k+4(x_k-y_k)+10$
- Repeat step 3 and 4 until $x\geq y$.
- Plot the points of other seven octant of the circle using eight way symmetry .

Mid Point Circle Algorithm

- Midpoint circle algorithm is another incremental circle algorithm that is very similar to Bresenham's approach.

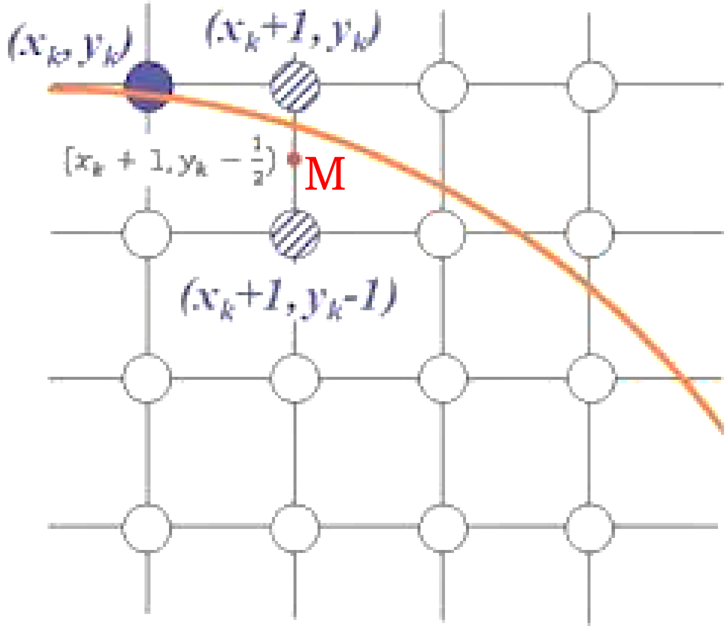
- It is based on the following functionality for testing the spatial relationship between an arbitrary point (x, y) and a circle of radius r centered at the origin.

$$f(x, y) = x^2+y^2-r^2$$

f(x, y)=0 means (x, y) on the circle

f(x,y)<0 means (x,y) inside the circle

f(x,y)>0 means (x,y) outside the circle



Mid Point Circle Algorithm

- Initial decision variable, p= 1-r or p=(5/4)-r

p	x_{k+1}	y_{k+1}	Updated value of p
$p \geq 0$	$x_{k+1} = x_k + 1$	$y_{k+1} = y_k - 1$	$p=p+2(x_k - y_k)+5$
$p<0$	$x_{k+1} = x_k + 1$	$y_{k+1} = y_k$	$p=p+2x_k+3$

Mid Point Circle Algorithm

- Plot initial point (x_k, y_k) such that $x_k=0$ and $y_k=r$.
- Compute the initial decision variable, $p_k=(5/4)-r$ or $p=1-r$.
- If $p_k<0$ then

  $x_{k+1}=x_k+1$

  $y_{k+1}=y_k$

  $p_{k+1}=p_k+2x_k+3$
- If $p_k\geq 0$ then

  $x_{k+1}=x_k+1$

  $y_{k+1}=y_k-1$

  $p_{k+1}=p_k+2(x_k-y_k)+5$
- Repeat step 3 and 4 until $x\geq y$.
- Plot the points of other seven octant of the circle using eight way symmetry .

Problem

Plot the first octant of a circle centered at origin, having radius 10 unit using Midpoint Circle algorithm or Bresenham's Circle Algorithm.

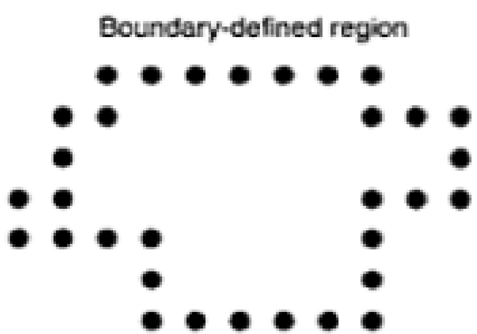
[Pls follow the class lecture for solution]

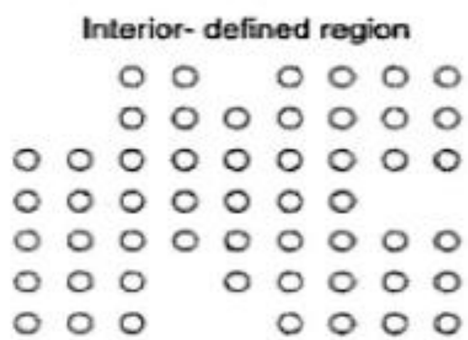
Region Filling

- Region filling is the process of "coloring in" a definite image area or region.
- Regions may be defines at the pixel level.
- At the pixel level, region is described in 2 ways:
  - Boundary-defined region
  - Interior-defined region

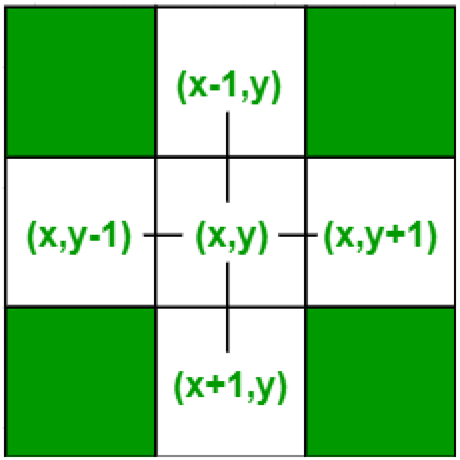
Region Filling

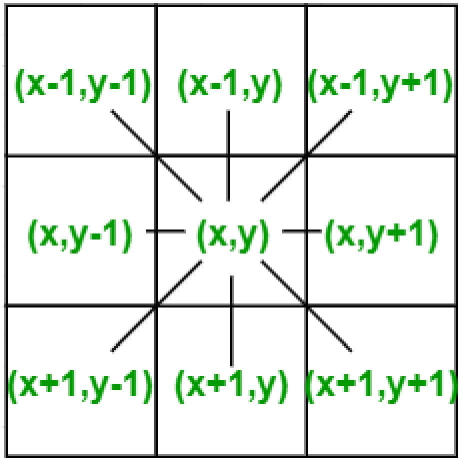
- Boundary-Defined Region: Region is described in term of the bounding pixel that outlines it.
- Interior-Defined Region: Region is described as the totality of pixels that comprise it.





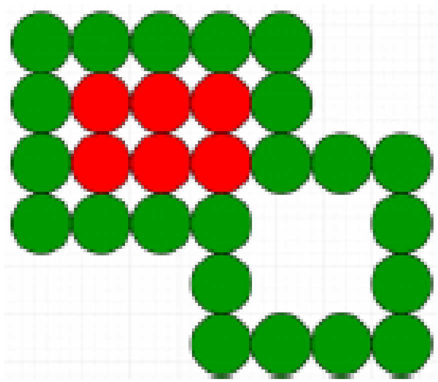
4 connected vs 8 connected





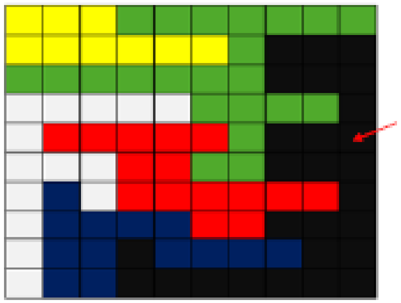
Boundary Fill Algorithm

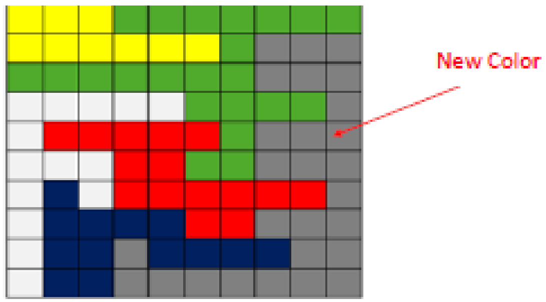
- This is a recursive algorithm that begins with a starting pixel, called a seed, inside a region.
- If the color is not equal to the fill color and boundary color, then it filled with fill color and the function is called for all the neighbors of the seed.
- If a point is found to be of fill color or of boundary color, the function does not call its neighbors and returns.
- This process continues until all points up to the boundary color for the region have been tested.



Flood Fill Algorithm

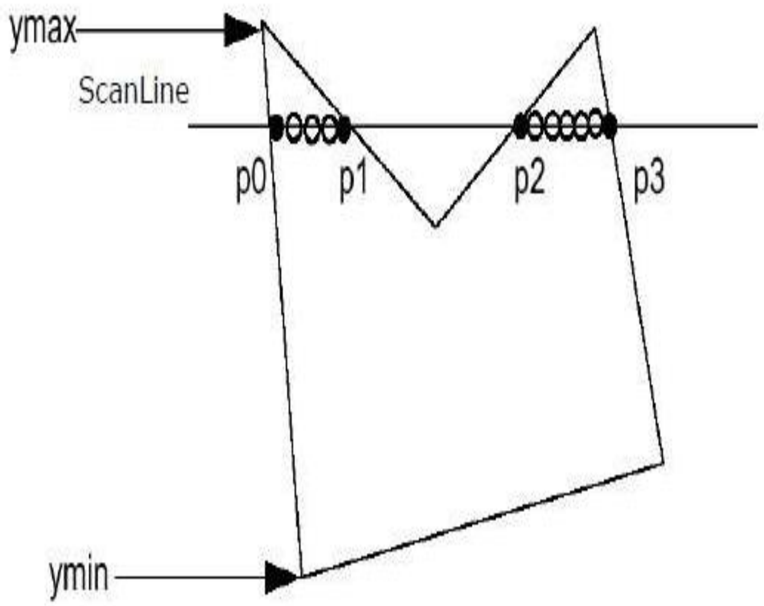
- This is a recursive algorithm that begins with a starting pixel, called a seed, inside a region.
- If the color is equal to current color, it fills the pixel with new color and the function call its neighbors.
- If it is not then the algorithm simply returns to its caller.



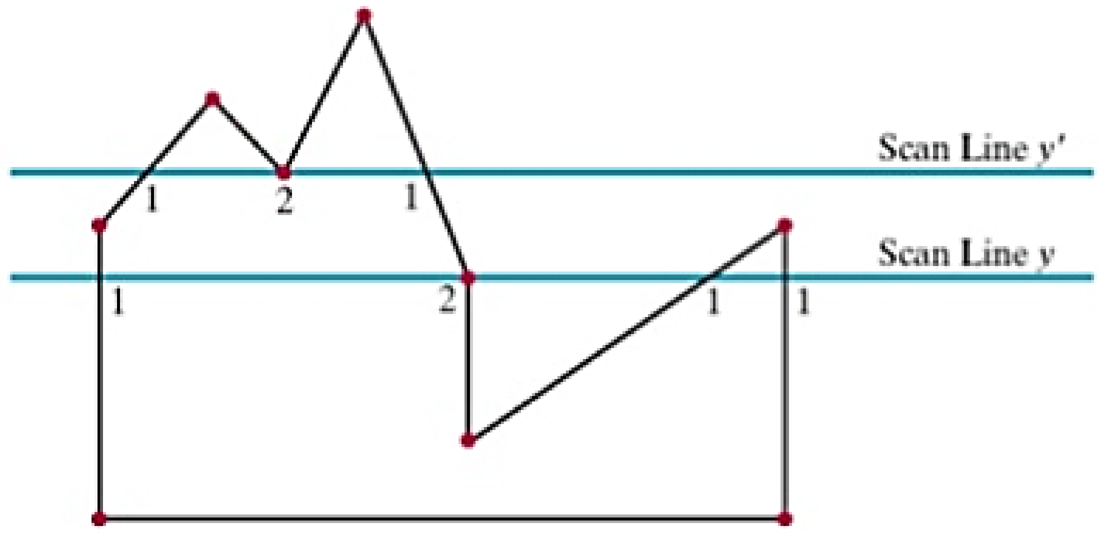


Scan Line Algorithm

- Find out the y_{min} and y_{max} from the given polygon.
- Scanline intersects with each edge of the polygon from y_{min} to y_{max}. Name each intersection point of the polygon.
- Sort the intersection point in the increasing order of X coordinate.
- Fill all those pair of coordinates that are inside polygons and ignore alternate pairs.

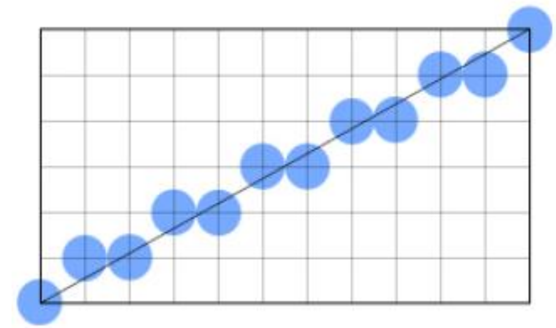


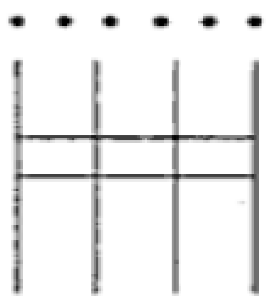
Scan Line Algorithm: Special Cases

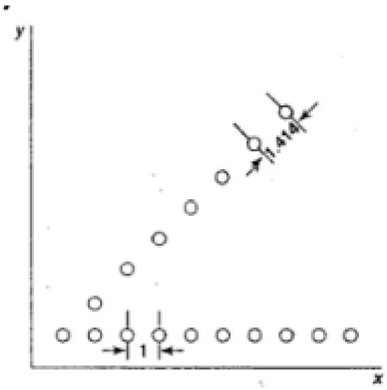


Aliasing Effect

- Stair Case:
  - When scan converting a primitive such as line or circle.
- Unequal Brightness:
  - A slanted line appears dimmer than a horizontal or vertical line, although all are presented at the same intensity level.
- Picket Fence
  - It occurs when an object is not aligned with or does not fit into the pixel grid properly.







Anti Aliasing

- It is the technique that can greatly reduces aliasing effects and improve the appearances of images without increasing the resolution.
- There are two types of anti aliasing techniques:
  - Pre-filtering: Filtering before sampling (Area Sampling)
  - Post-filtering: Sampling before filtering (Super Sampling)

** Self Study: Area Sampling and Super Sampling

Mathematical Problem

** Solve the exercise problems of Chapter 3 the end