GALGOTIAS UNIVERSITY

School of Computing Science and Engineering

Program: B.C.A.

Course Code: BCAS3003

Course Name: Computer Graphics

Course Prerequisites

- Knowledge of Mathematics
- Fundamental knowledge of Computer

Syllabus

Unit 2 – Output Primitives (8 hours)

- Line Drawing Algorithms
- Circle Generation Algorithms
- Ellipse Generating Algorithm
- Pixel Addressing
- Filled-Area Primitives
- Fill Area Function,
- Cell Array, Character Generation

Recommended Books

Text books

- D. Hearn, P. Baker, "Computer Graphics - C Version", 2nd Edition, Pearson Education, 1997

Reference Book

- Heam Donald, Pauline Baker M: "Computer Graphics", PHI 2nd Edn. 1995.
- Harrington S: "Computer Graphics - A Programming Approach", 2nd Edn. Mc GrawHill.
- Shalini Govil-Pai, Principles of Computer Graphics, Springer, 2004

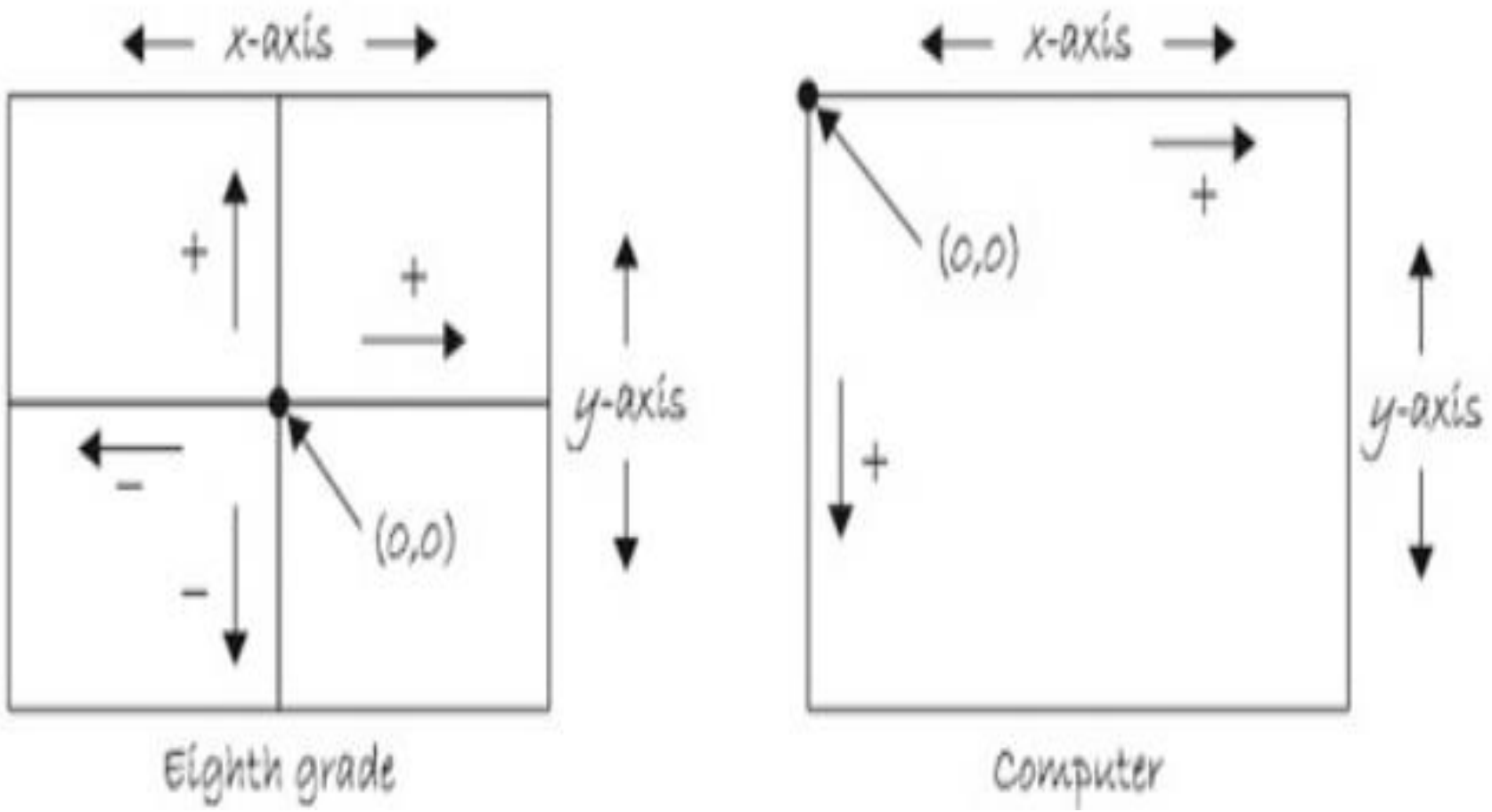
Additional online materials

- Coursera - https://www.coursera.org/learn/fundamentals-of-graphic-design
- https://www.youtube.com/watch?v=fwzYuhduME4&list=PLE4D97E3B8DB8A590
- NPTEL - https://nptel.ac.in/courses/106/106/106106090/
- https://www.coursera.org/learn/research-methods
- https://www.coursera.org/browse/physical-science-and-engineering/research-methods

Pixel Addressing

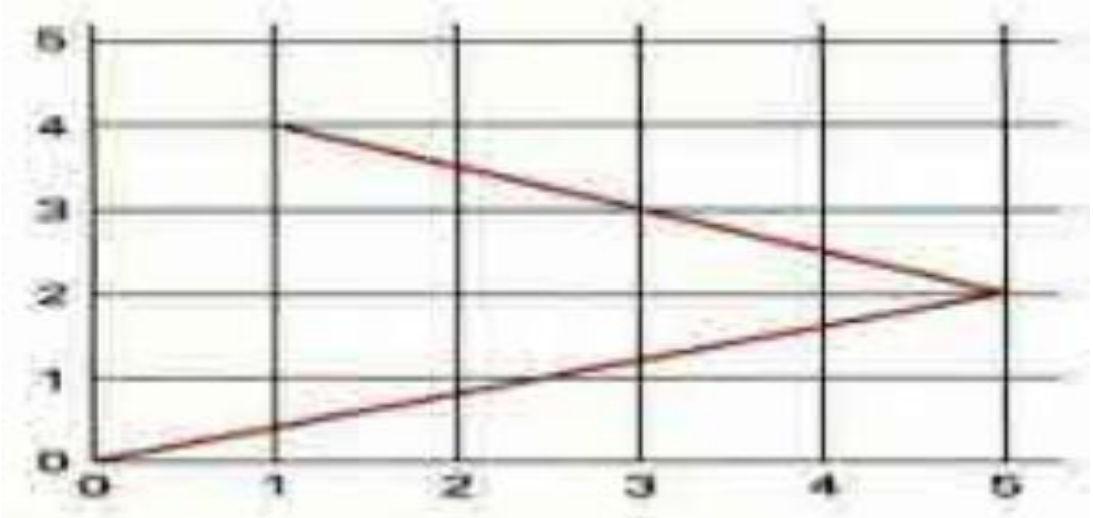
- When an object is scan converted into the frame buffer, the input description is transformed to pixel coordinates.
- So, the displayed image may not correspond exactly with the relative dimensions of the input object.
- To preserve the specified geometry of world objects, we need to compensate for the mapping of mathematical input points to finite pixel area, we use one of the two ways:
  - Adjust the dimensions of displayed objects to account for the amount of overlap of pixel areas with the object boundaries. (i.e. a rectangle with 40 cm width, will be displayed in 40 pixel)
  - Map world coordinates onto screen positions between pixels, so that we align objects boundaries with pixel boundaries instead of pixel centers.

Pixel Addressing



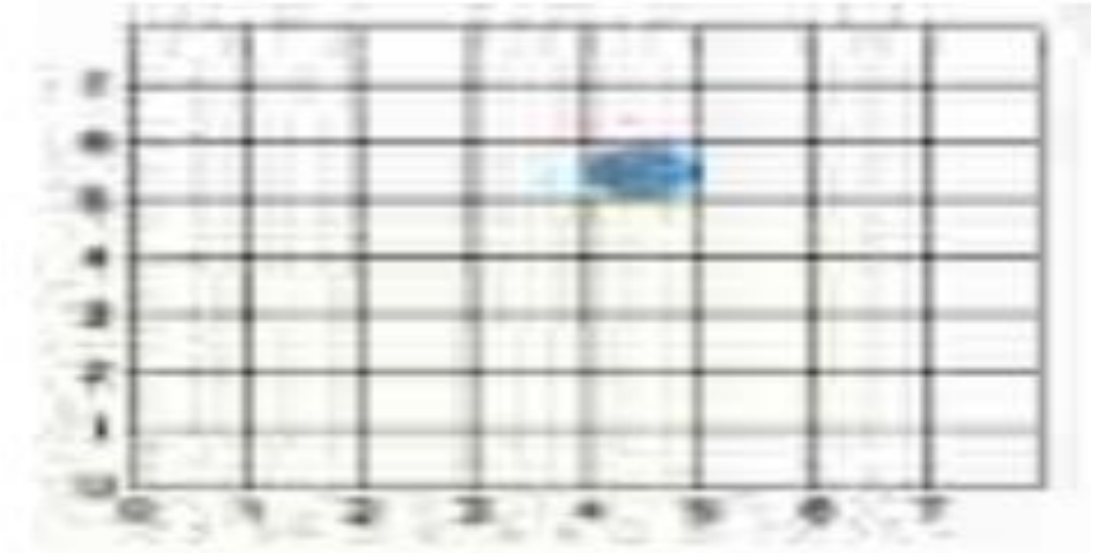
Screen Grid Coordinates

- An alternative to addressing display positions in terms of pixel centers is to reference screen coordinates with respect to the grid of horizontal and vertical pixel boundary lines spaced one unit a part.
- Screen coordinate position is then the pair of integer values identifying a grid intersection position between two pixels. For example, the mathematical line path for a polyline with screen endpoints (0, 0), (5, 2), and (1,4) is shown beside.



Screen Grid Coordinates

- With the coordinate origin at the lower left of the screen, each pixel area can be referenced by the integer grid coordinates of its lower left corner.
- The following figure illustrates this convention for an 8 by 8 section of a raster, with a single illuminated pixel at screen coordinate position (4, 5).



Screen Grid Coordinates

- The previous algorithms for drawing line, circle, …etc are still valid when applied to input positions expressed as screen grid coordinates.
- The decision parameter Pk is a measure of screen grid separation differences rather than separation differences from pixel centers.
  - A circle of radius 5 and center position (10, 10), for instance, would be displayed by the midpoint circle algorithm using screen grid coordinate positions.
  - But the plotted circle has a diameter of 11, To plot the circle with the defined diameter of 10, we can modify the circle algorithm to shorten each pixel scan line and each pixel column.

Antialiasing Techniques

- Antialiasing is a technique that is used to smooth jagged distortions in curves and diagonal lines so they appear smoother.
- There are three main classes of antialiasing algorithms.
- As aliasing problem is due to low resolution, one easy solution is to increase the resolution causing sample points to occur more frequently. This increases the cost of image production.
- The image is created at high resolution and then digitally filtered. This method is called supersampling or postfiltering and eliminates high frequencies which are the source of aliases.
- The image can be calculated by considering the intensities over a particular region. This is called prefiltering.

Antialiasing Techniques

Postfiltering

- Supersampling or postfiltering is the process by which aliasing effects in graphics are reduced by increasing the frequency of the sampling grid and then averaging the results down.
- This process means calculating a virtual image at a higher spatial resolution than the frame store resolution and then averaging down to the final resolution. It is called postfiltering as the filtering is carried out after sampling.

Disadvantage of Postfiltering

- There is a technical and economic limit for increasing the resolution of the virtual image.
- A continuous image I(x,y) is sampled at n times the final resolution. The image is calculated at n times the frame resolution. This is a virtual image.

Antialiasing Techniques

Prefiltering

- Prefiltering methods treat a pixel as an area, and compute pixel color based on the overlap of the scene's objects with a pixel's area.
- These techniques compute the shades of gray based on how much of a pixel's area is covered by a object.
- For example, a modification to Bresenham's algorithm was developed by Pitteway and Watkinson. In this algorithm, each pixel is given intensity depending on the area of overlap of the pixel and the line.
- Prefiltering thus amounts to sampling the shape of the object very densely within a pixel region. For shapes other than polygons, this can be very computationally intensive.

Antialiasing Techniques

Prefiltering

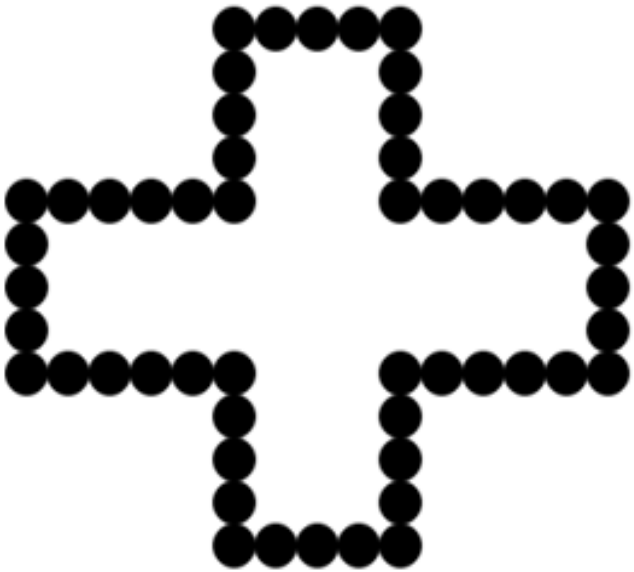
- Without antialiasing, the jaggies are harshly evident.
- Along the character's border, the colors are a mixture of the foreground and background colors

Filled Area Primitives

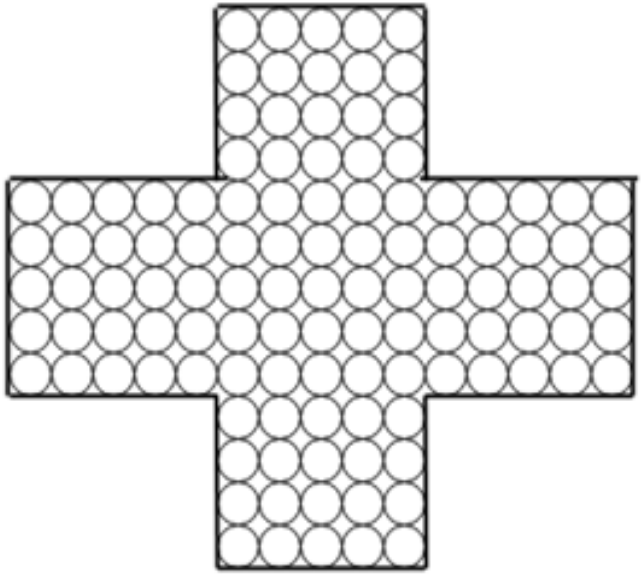
- Boundary Filled Algorithm
- Flood Fill Algorithm
- Scan Line Polygon Fill Algorithm:

Boundary Filled Algorithm

- Region filling is the process of filling image or region. Filling can be of boundary or interior region as shown in fig. Boundary Fill algorithms are used to fill the boundary and flood-fill algorithm are used to fill the interior.



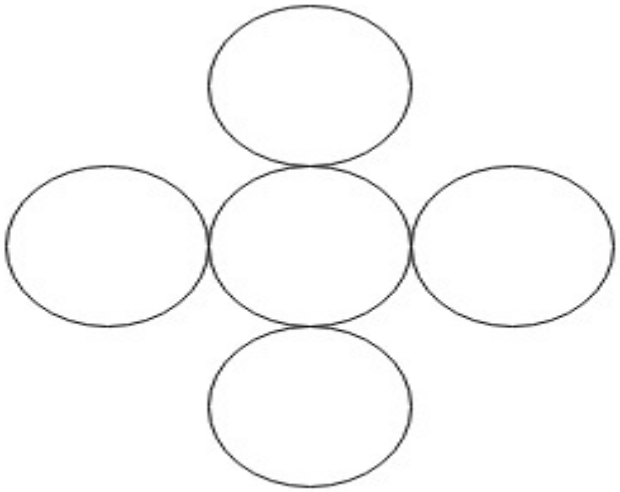
Boundary Filled Region



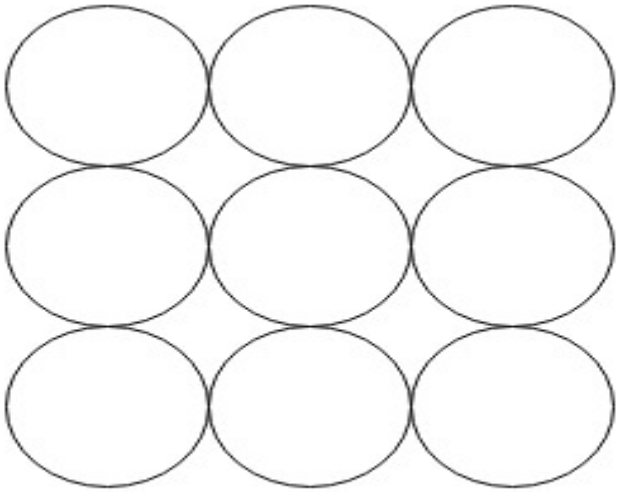
Interior or Flood Filled Region

Boundary Filled Algorithm

- This algorithm uses the recursive method.
- First of all, a starting pixel called as the seed is considered. The algorithm checks boundary pixel or adjacent pixels are colored or not.
- If the adjacent pixel is already filled or colored then leave it, otherwise fill it.
- The filling is done using four connected or eight connected approaches.



Four Connected



Eight Connected

Boundary Filled Algorithm

- Four connected approaches is more suitable than the eight connected approaches.
- **Four connected approaches:** In this approach, left, right, above, below pixels are tested.
- **Eight connected approaches:** In this approach, left, right, above, below and four diagonals are selected.
- Boundary can be checked by seeing pixels from left and right first. Then pixels are checked by seeing pixels from top to bottom. The algorithm takes time and memory because some recursive calls are needed.

Drawback

- It may not fill regions sometimes correctly when some interior pixel is already filled with color. The algorithm will check this boundary pixel for filling and will found already filled so recursive process will terminate. This may vary because of another interior pixel unfilled.

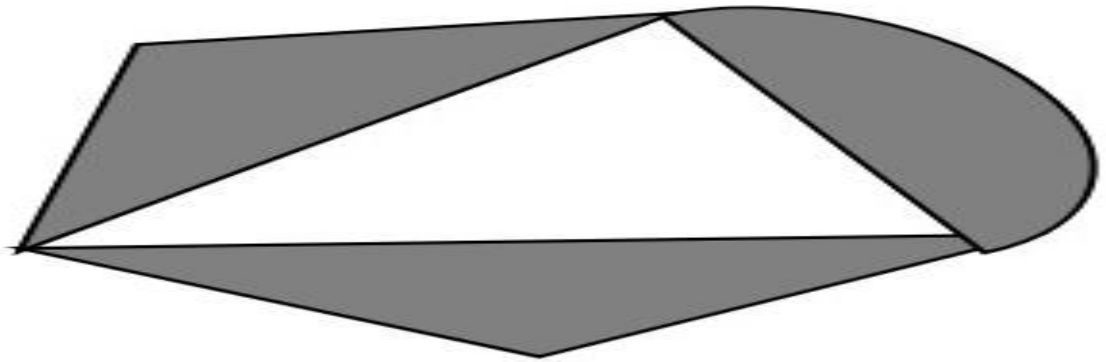
Boundary Filled Algorithm

Algorithm

```
Procedure fill(x, y, color, color1)
        int c;
        c = getpixel(x, y);
        if (c!=color) (c!=color1)
                {
                   setpixel(x, y, color)
                   fill (x+1, y, color, color 1);
                    fill (x-1, y, color, color 1);
                   fill (x, y+1, color, color 1);
                   fill (x, y-1, color, color 1);
                }
```

Flood Fill Algorithm

- In this method, a point or seed which is inside region is selected. This point is called a seed point.
- Then four connected approaches or eight connected approaches is used to fill with specified color.
- The flood fill algorithm has many characters similar to boundary fill. But this method is more suitable for filling multiple colors boundary.
- When boundary is of many colors and interior is to be filled with one color we use this algorithm.



Flood Fill Algorithm

- In fill algorithm, we start from a specified interior point (x, y) and reassign all pixel values are currently set to a given interior color with the desired color.
- Using either a 4-connected or 8-connected approaches, we then step through pixel positions until all interior points have been repainted.

Disadvantage

- Very slow algorithm
- May be fail for large polygons
- Initial pixel required more knowledge about surrounding pixels.

Flood Fill Algorithm

Algorithm

- Procedure floodfill (x, y, fill_color,old_color)

```
If (getpixel (x, y)=old_color)
        {
          setpixel (x, y, fill_color);
          floodfill (x+1, y, fill_color, old_color);
          floodfill (x-1, y, fill_color, old_color);
          floodfill (x, y+1, fill_color, old_color);
          floodfill (x, y-1, fill_color, old_color);
        }
```

Flood Fill Algorithm

Algorithm

- Procedure floodfill (x, y, fill_color,old_color)

```
If (getpixel (x, y)=old_color)
        {
          setpixel (x, y, fill_color);
          floodfill (x+1, y, fill_color, old_color);
          floodfill (x-1, y, fill_color, old_color);
          floodfill (x, y+1, fill_color, old_color);
          floodfill (x, y-1, fill_color, old_color);
        }
```

Flood Fill Algorithm

Algorithm with 4-connected

```
#include<stdio.h>
#include<conio.h>
#include<graphics.h>
#include<dos.h>
void flood(int,int,int,int);
void main()
{
   intgd=DETECT,gm;
   initgraph(&gd,&gm,"C:/TURBOC3/bgi");
   rectangle(50,50,250,250);
   flood(55,55,10,0);
   getch();  }
```

Flood Fill Algorithm

Algorithm with 4-connected

```
void flood(intx,inty,intfillColor, intdefaultColor)
{
   if(getpixel(x,y)==defaultColor)
   {
      delay(1);
      putpixel(x,y,fillColor);
      flood(x+1,y,fillColor,defaultColor);
      flood(x-1,y,fillColor,defaultColor);
      flood(x,y+1,fillColor,defaultColor);
      flood(x,y-1,fillColor,defaultColor);
   }
}
```

Flood Fill Algorithm

Algorithm with 8-connected

```
#include<stdio.h>
#include<graphics.h>
#include<dos.h>
#include<conio.h>
void main()
{
        intgd=DETECT,gm;
        initgraph(&gd,&gm,"C:\\TURBOC3\\BGI");
        rectangle(50,50,150,150);
        floodfill(70,70,0,15);
        getch();
        closegraph(); }
```

Flood Fill Algorithm

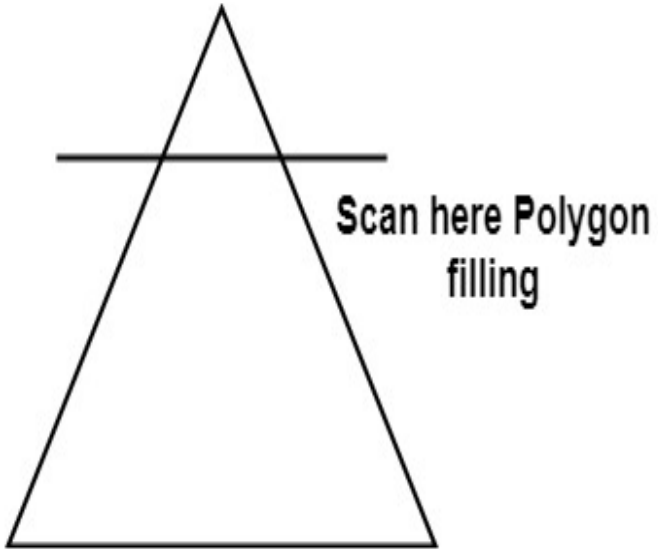
Algorithm with 8-connected

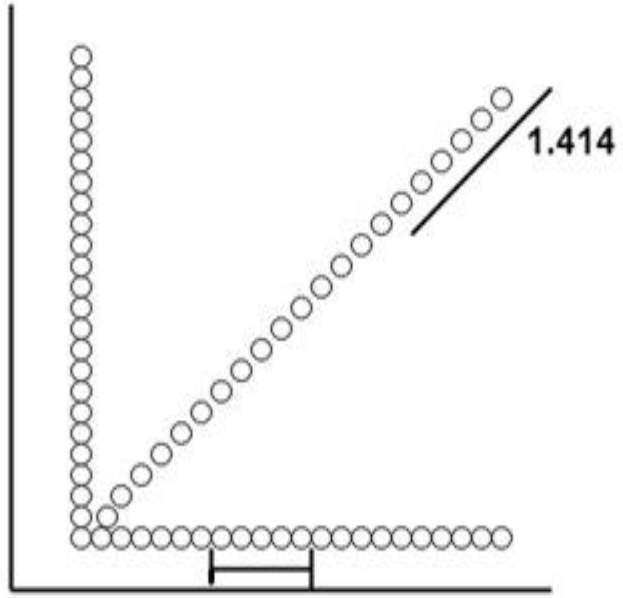
```
void floodfill(intx,inty,intold,intnewcol)
{
        int current;
        current=getpixel(x,y);
        if(current==old)  {
                delay(5);
                putpixel(x,y,newcol);
                floodfill(x+1,y,old,newcol);     floodfill(x-1,y,old,newcol);
                floodfill(x,y+1,old,newcol);    floodfill(x,y-1,old,newcol);
                floodfill(x+1,y+1,old,newcol);
                floodfill(x-1,y+1,old,newcol);
                floodfill(x+1,y-1,old,newcol);
                floodfill(x-1,y-1,old,newcol);          }  }
```

Scan Line Polygon Fill Algorithm

- This algorithm lines interior points of a polygon on the scan line and these points are done on or off according to requirement. The polygon is filled with various colors by coloring various pixels.
- In the figure, polygon and a line cutting polygon in shown.
- Scanning is done using raster scanning concept on display device.
- The beam starts scanning from the top left corner of the screen and goes toward the bottom right corner as the endpoint.
- The algorithms find points of intersection of the line with polygon while moving from left to right and top to bottom.
- The various points of intersection are stored in the frame buffer. The intensities of such points is keep high.
- Concept of coherence property is used. According to this property if a pixel is inside the polygon, then its next pixel will be inside the polygon.

Scan Line Polygon Fill Algorithm





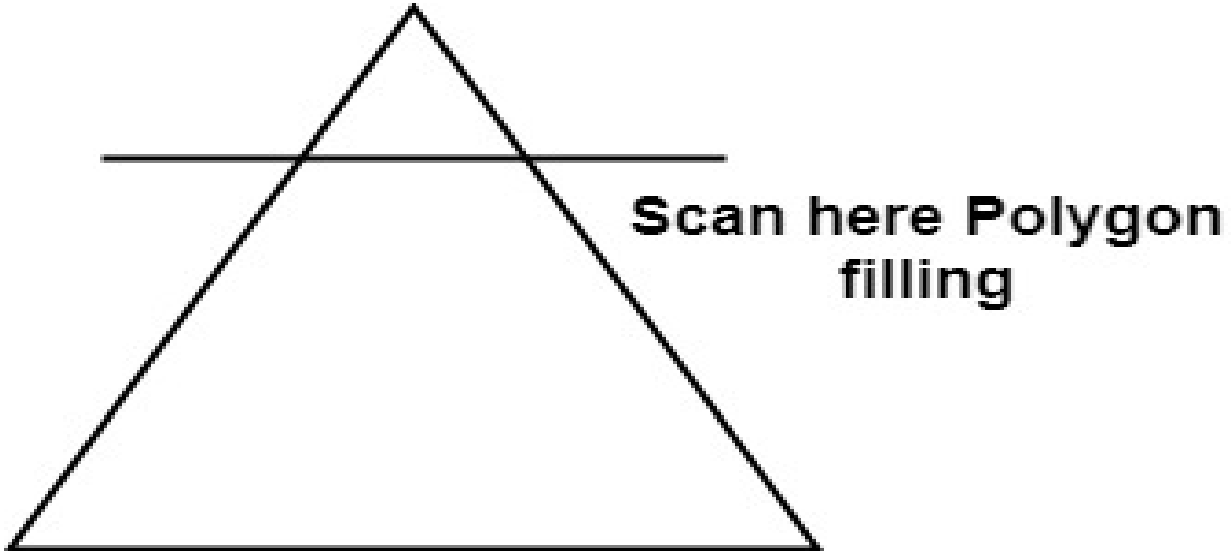
Pixels along with horizontal line are 1 unit apart and vertical.

Pixels along diagonal line are 1.414 units.

Scan Line Polygon Fill Algorithm

Side effects of Scan Conversion

- Staircase or Jagged: Staircase like appearance is seen while the scan was converting line or circle.
- Unequal Intensity: It deals with unequal appearance of the brightness of different lines. An inclined line appears less bright as compared to the horizontal and vertical line.



Questions

- Explain Filed Area Primitives.
- Explain Flood fill algorithm with examples.
- Explain Boundary fill algorithm with examples.


Thank You