Chapter 2

Scan Conversation

2.1 Scan Conversation:

The process of representing continuous graphics objects as a collection of discrete pixels is called scan conversion.

2.1.1 Benefit of Scan Conversation:

Algorithms can generate graphics objects at a faster rate. Using algorithms memory can be used efficiently. Algorithms can develop a higher level of graphical objects.

2.1.2 Side Effect of Scan Conversation:

- Aliasing.
- Unequal intensity of diagonal lines.
- Over striking in photographic applications.
- Local or Global aliasing

2.2 Rasterisation:

Rasterisation (or rasterization) is the task of taking an image described in a vector graphics format (shapes) and converting it into a raster image. The rasterised image may then be displayed on a computer display, video display or printer, or stored in a bitmap file format. Rasterisation may refer to the technique of drawing 3D models, or the conversion of 2D rendering primitives such as polygons, line segments into a rasterized format.

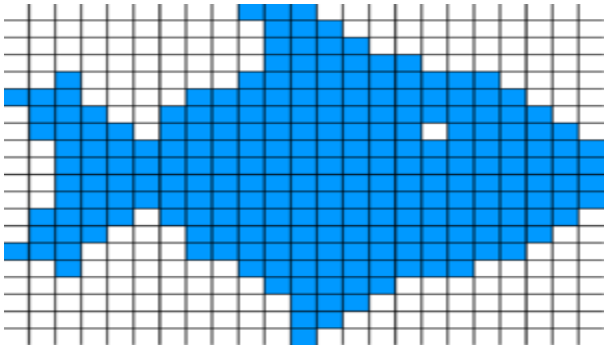


Figure 2.1 Rasterisation

2.3 Line Drawing Algorithm

A line drawing algorithm is a graphical algorithm for approximating a line segment on discrete graphical media. On discrete media, such as pixel-based displays and printers, line drawing requires such an approximation (in nontrivial cases). Basic algorithms rasterize lines in one color. A better representation with multiple color gradations requires an advanced process, spatial anti-aliasing.

2.3.1 Characteristics of Line Drawing Algorithm

The characteristics of a good algorithm are:

Precision – the steps are precisely stated(defined).

Uniqueness – results of each step are uniquely defined and only depend on the input and the result of the preceding steps.

Finiteness – the algorithm stops after a finite number of instructions are executed

2.3.2 Requirement of line drawing/incremental algorithm

- Integer Pixel Grid
- Slope
- Disjoint Pixel

2.4 DDA line incremental algorithm

DDA stands for Digital Differential Analyzer. It is an incremental method of scan conversion of line. In this method calculation is performed at each step but by using results of previous steps.

Advantage:

- It is a faster method than method of using direct use of line equation.
- This method does not use multiplication theorem.
- It allows us to detect the change in the value of x and y ,so plotting of same point twice is not possible.
- This method gives overflow indication when a point is repositioned.
- It is an easy method because each step involves just two additions.

Disadvantage:

- It involves floating point additions rounding off is done. Accumulations of round off error cause accumulation of error.
- Rounding off operations and floating point operations consumes a lot of time.
- It is more suitable for generating line using the software. But it is less suited for hardware implementation.

Finding Pixels:

1st pixel:

x1=x;

y1=y;

2nd/onward pixel:

If m<1

x1=x+1;

y1=round(y+m)

if m>1

x1 = round(x + 1 /m)

y1 = y + 1

if m=1

x1=x+1

y1=y+1

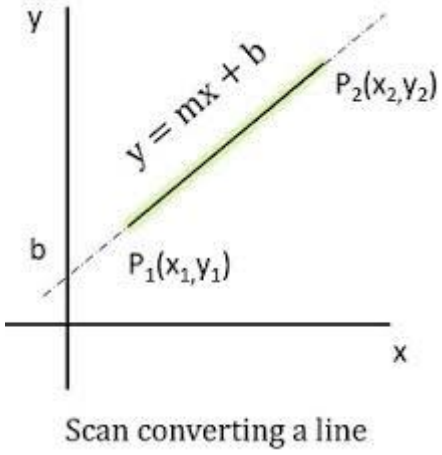


Figure: 2.2 line (slope)

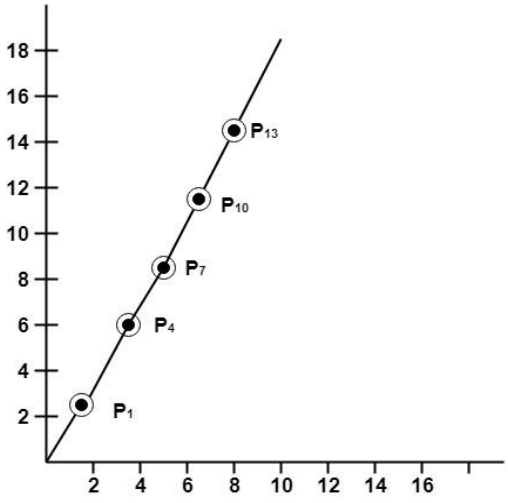


Figure 2.3: Pixels of line

Exercise 2.1 (Problem Solving)

Initial Pixel: 3,4

Destination Pixel: 8,8

Find the pixels and plot the line based on the pixels using DDA Algorithm

2.5 Bresenham's Midpoint Line Incremental Algorithm

Bresenham's Midpoint line algorithm is an incremental line plotting algorithm i.e. at each step we make incremental calculations based on preceding step to find next y value, in order to form a close approximation to a straight line between two points. It chooses the pixels closest to the line with accuracy, consistency and straightness. It is very simple and requires only integer data and simple arithmetic. It avoids division and multiplication and thus avoid truncate errors.

BASISOF ALGORITHM Given the previous pixel P, there are two candidates for the next pixel closest to the line, E and NE . If the M is above the line, choose NE. If M is below the line, choose E.

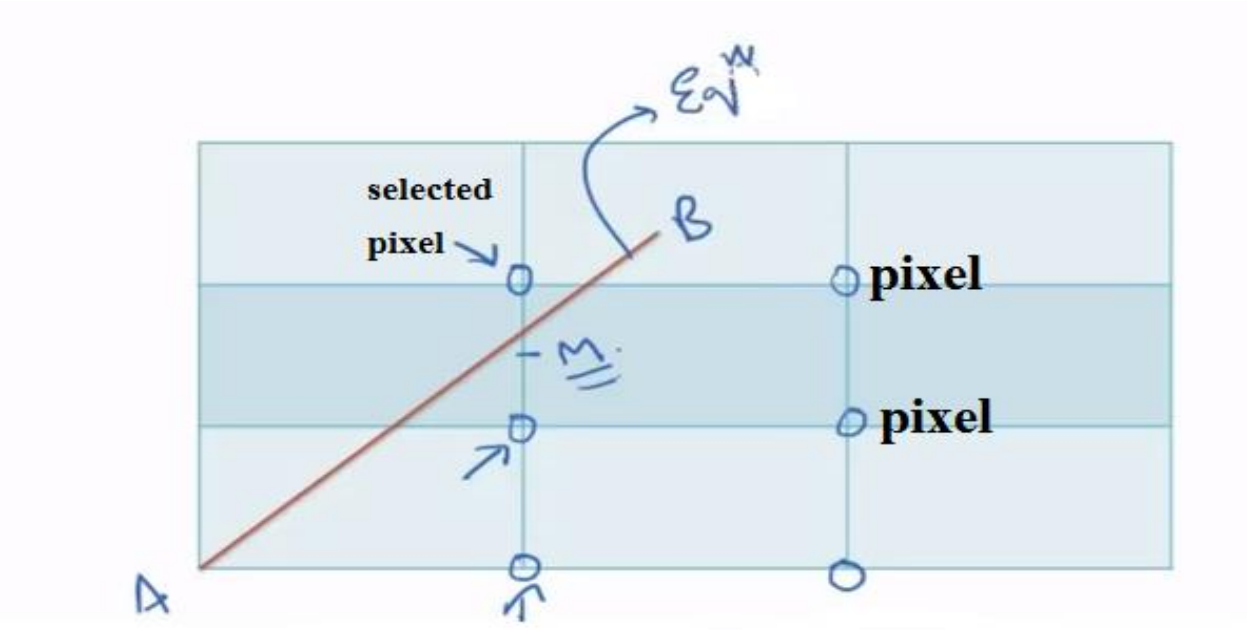
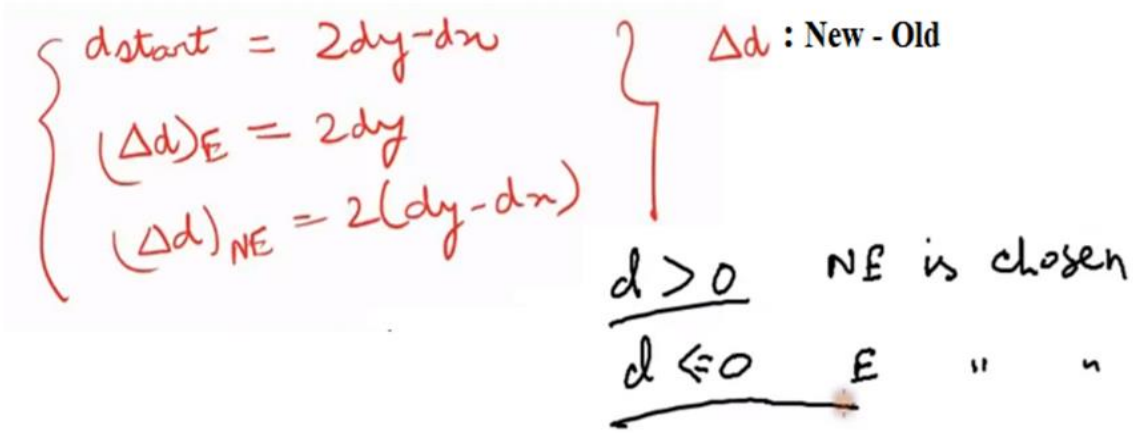


Figure: 2.4 DDA Algorithm Concept



Midpoint Line Advantages

- Incremental Method
- No Round Function
- More accurate position
- Only Arithmetic Function

Exercise 2.2 (Problem Solving):

Initial Pixel: 3,4

Destination Pixel: 8,8

Find the pixels and plot line using Midpoint Line Algorithm

2.6 Write the difference between DDA and Midpoint Line Algorithm

Basis for comparison	**DDA Algorithm**	**Bresenham Algorithm**
Speed	Comparatively less	More
Operations used	Multiplication and division	Additions and subtraction
Arithmetic computation values	Floating point	Integer type
Precision	Low	High

2.7 Bresenham's Midpoint Line Incremental Algorithm

Jack E. Bresenham invented this algorithm in 1962. The objective was to optimize the graphic algorithms for basic objects, in a time when computers were not as powerful as they are today.
This algorithm is called incremental, because the position of the next pixel is calculated on the basis of the last plotted one, instead of just calculating the pixels from a global formula. Such logic is faster for computers to work on and allows plotting circles without trigonometry. The algorithm use only integers and that's where the strength is: floating point calculations slow down the processors. All in all, incremental algorithms are 30% faster than the classical ones, based on floating points and trigonometry.

In computer graphics, the midpoint circle algorithm is an algorithm used to determine the points needed for rasterizing a circle. Bresenham's circle algorithm is derived from the midpoint circle algorithm.

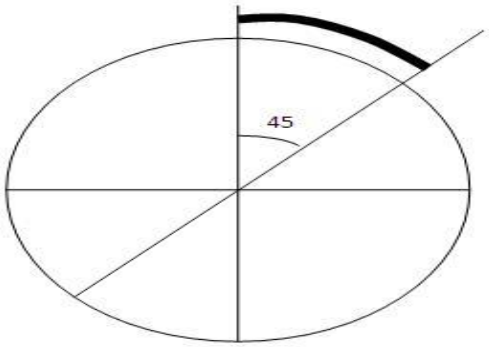


Figure: 2.5 Circle

We know that there are 360 degrees in a circle. First we see that a circle is symmetrical about the x axis, so only the first 180 degrees need to be calculated. Next, we see that it's also symmetrical about the y axis, so now we only need to calculate the first 90 degrees. Finally, we see that the circle is also symmetrical about the 45 degree diagonal axis, so we only need to calculate the first 45 degrees. Bresenham's circle algorithm calculates the locations of the pixels in the first 45 degrees. It assumes that the circle is centered on the origin shifting the original center

coordinates (centerx,centery). So for every pixel (x,y) it calculates, we draw a pixel in each of the 8 octants of the circle.

Drawing the circle

We can reduce our calculation drastically (8th fraction) by making use of the fact that a circle has 8 way symmetry. Thus after calculating a pixel position (x,y) to be plotted, we get 7 other points on the circle corresponding to it. These are:

(x,y); (x,-y); (-x,y); (-x,-y); (y,x); (y,-x); (-y,x); (-y,-x)

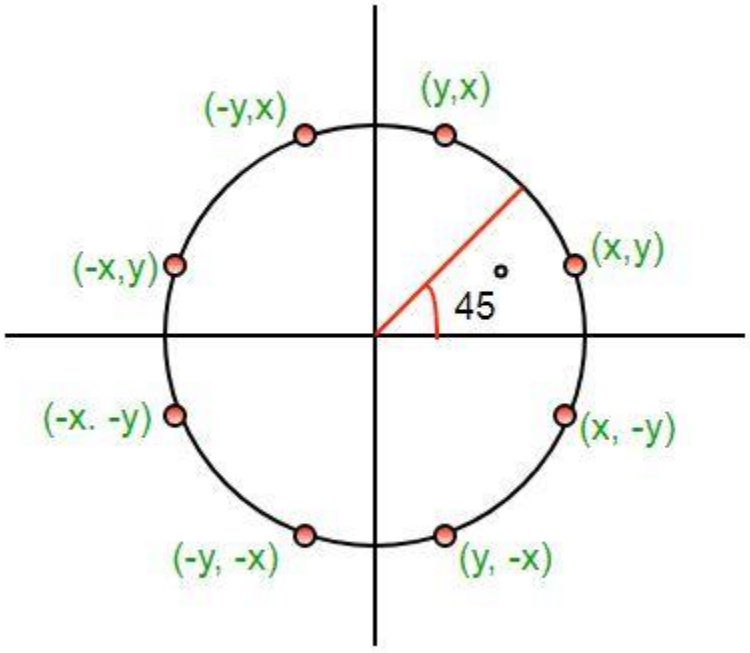
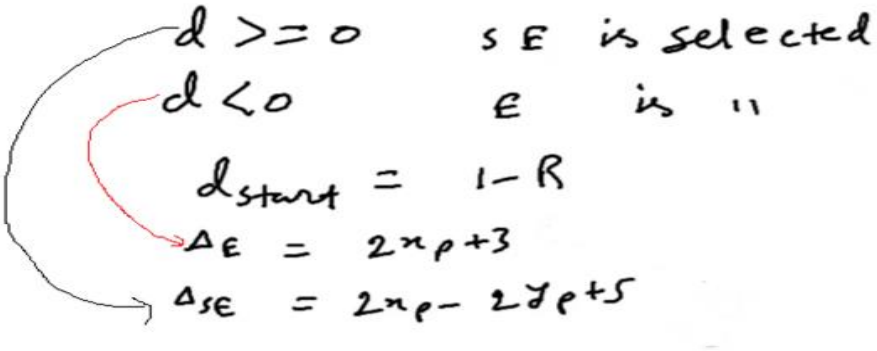


Figure: 2.6 Circle 8 way symmetry

BASISOF ALGORITHM

Given the previous pixel P, there are two candidates for the next pixel closest to the line, E and NE . If the M is above the line, choose SE. If M is below the line, choose E.

Formula:



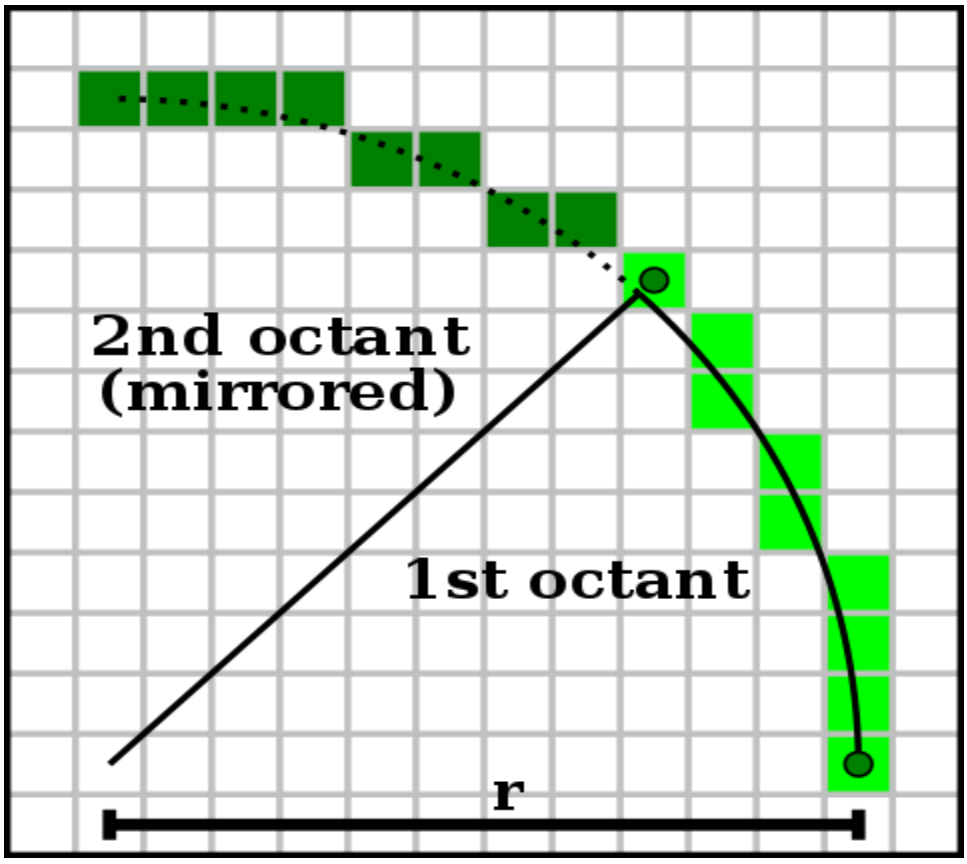


Figure: 2.7 Ploting Circle based on pixels

Exercise 2.3 (Problem Solving)
Center:4,4
Radius:64
Find 2nd, 4th Octant pixels using Midpoint Line Algorithm

References

Chapter 3: Basic Raster Graphics Algorithm for Drawing 2D Primitives. Foley, van Dam, Feiner, Hughes, Computer Graphics: principles and practice, 2nd ed.

https://www.slideshare.net/mohammedarif89/midpoint-circle-algo?qid=33e02b6e-628f-4b43-afcb-ea838ebf72c9&v=&b=&from_search=2

https://slideplayer.com/slide/9120741/

https://en.wikipedia.org/wiki/Midpoint_circle_algorithm

https://studyresearch.in/2018/03/11/bresenhams-circle-algorithm/