ICT 4203

Computer Graphics and Animation

Lecture 02

Line Drawing Algorithms

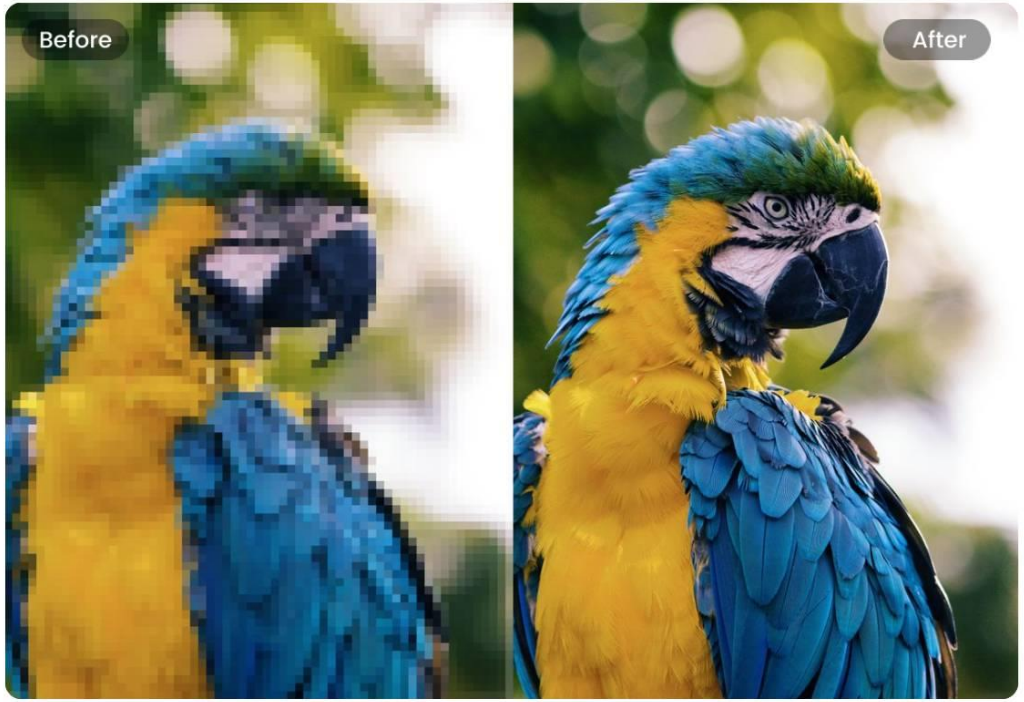
Md. Mahmudur Rahman
Lecturer
Institute of Information Technology

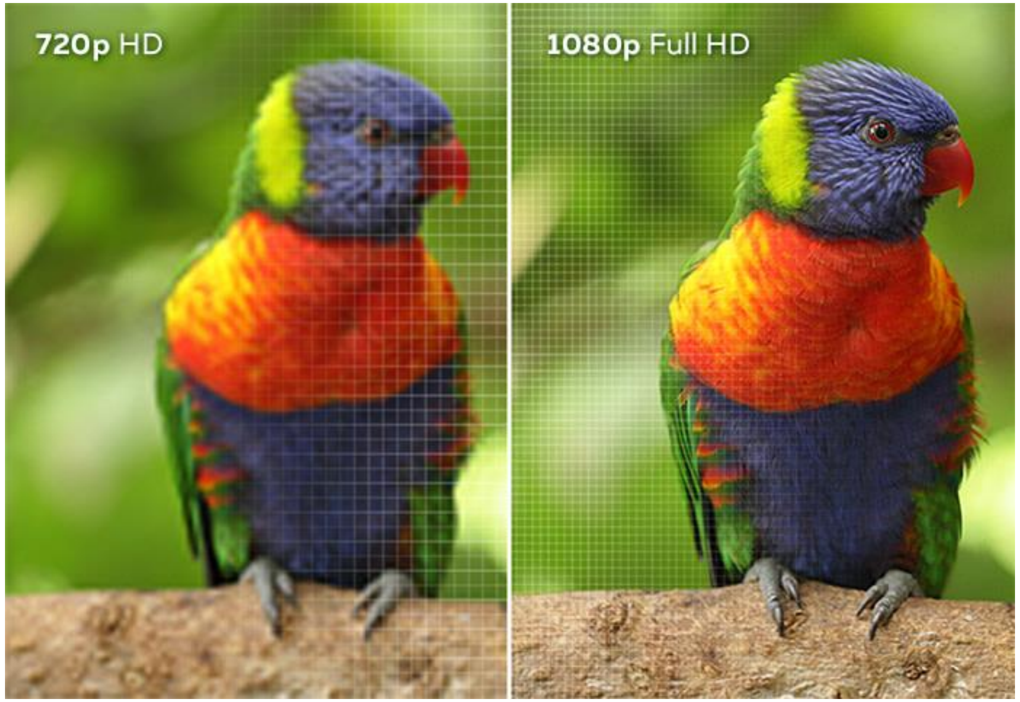
Contents

- Overview
- Scan Conversion
- Line Drawing Algorithms:
  - Direct Use of Line Algorithm
  - DDA

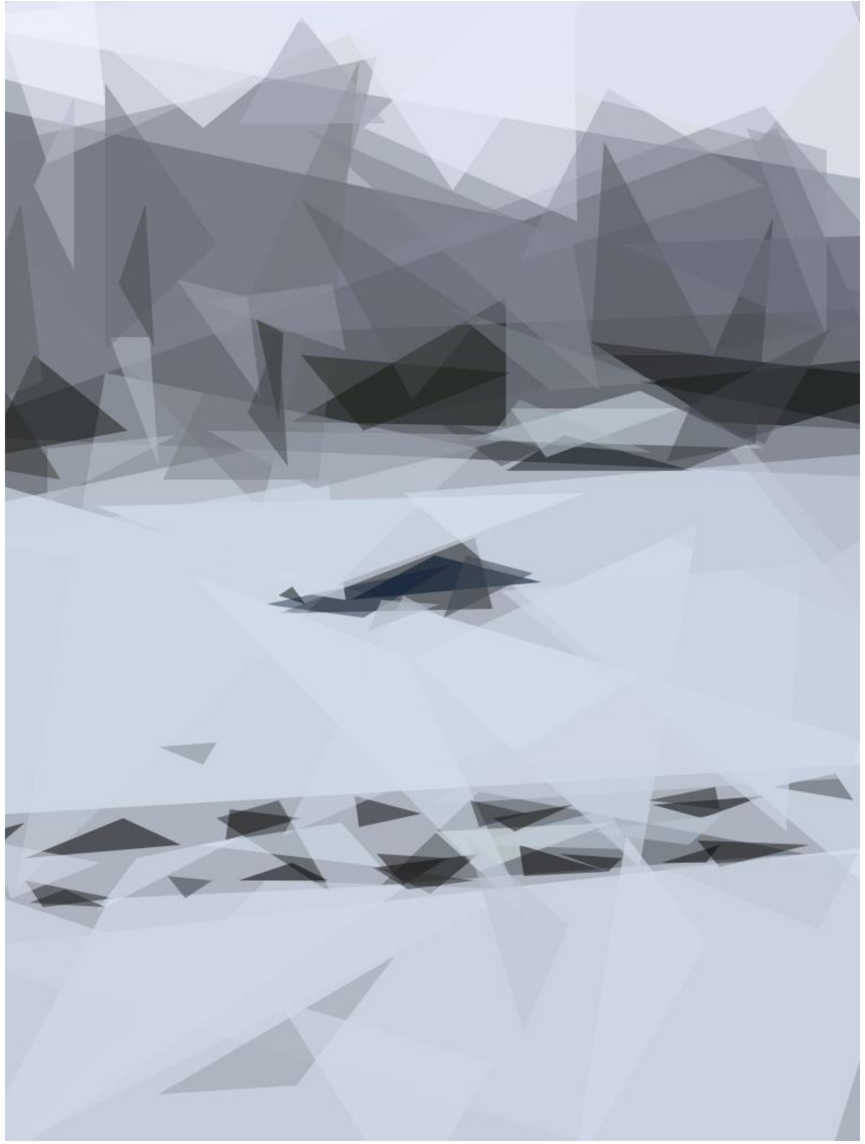
Overview

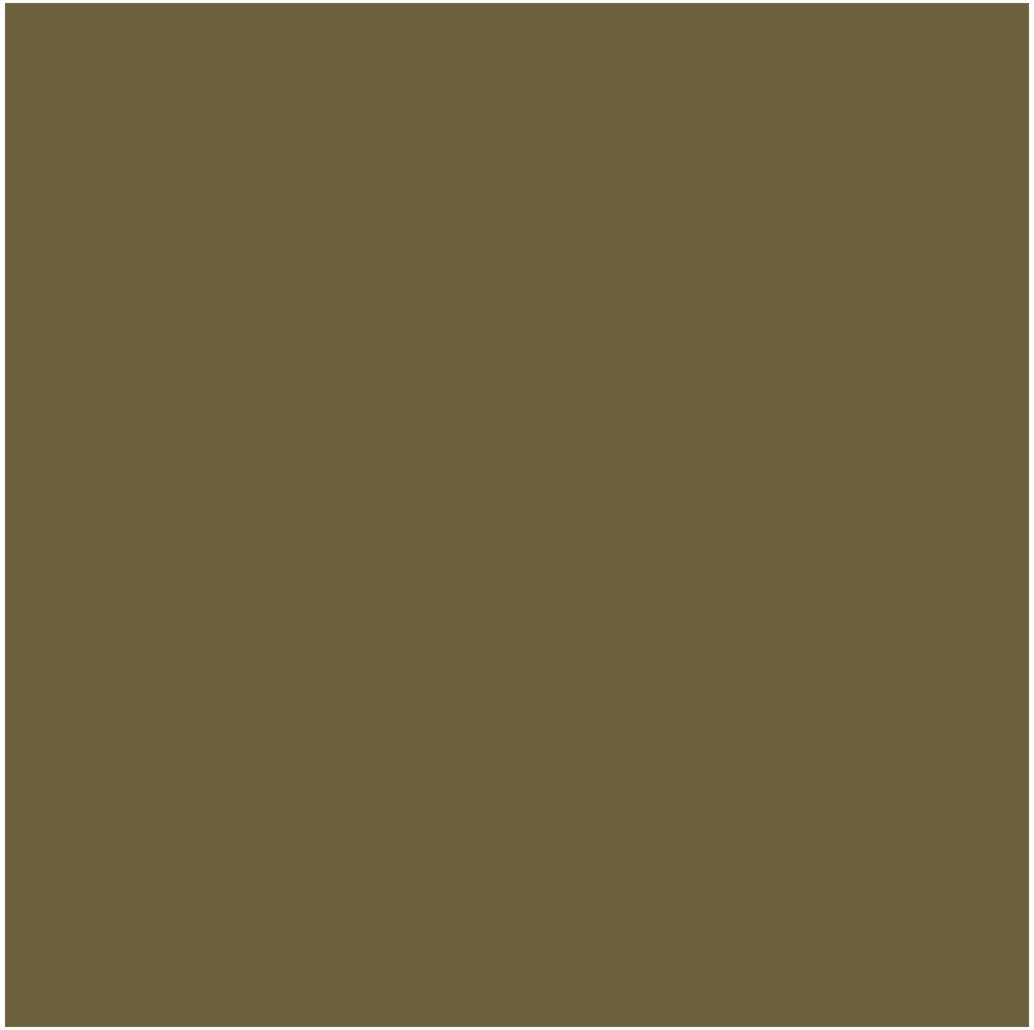
- Many pictures, from 2D drawing to view of 3D objects consists of graphical primitives such as points, lines, circles & filled polygons.
- These picture components are often defined in a continuous space at a higher level of abstraction than individual pixels in the discrete image space.
- For instance, a line is defined by its two endpoints and the line equation, whereas a circle is defined by its radius, center position & the circle equation.
- It is the responsibility of the graphics system or application program to convert each primitive from its geometric definition into a set of pixels that make up the primitive in the image space.


Before
After


720p HD
1080p Full HD

This is the output using 100 triangles





Scan Conversion

- It is a process of representing graphics objects as a collection of pixels.
- The graphics objects are continuous. The pixels used are discrete. Each pixel can have either an on or off-state.
- The circuitry of the video display device of the computer is capable of converting binary values (0, 1) into a pixel on and pixel off information. 0 is represented by pixel off. 1 is represented using pixel on.
- Using this ability, graphics computers represent pictures having discrete dots.
- Any graphics model can be reproduced with a dense matrix of dots or points.
- Most people think of graphics objects as points, lines, circles, and ellipses.
- For generating graphical objects, many algorithms have been developed.

Continue...

Examples of objects which can be scan converted:

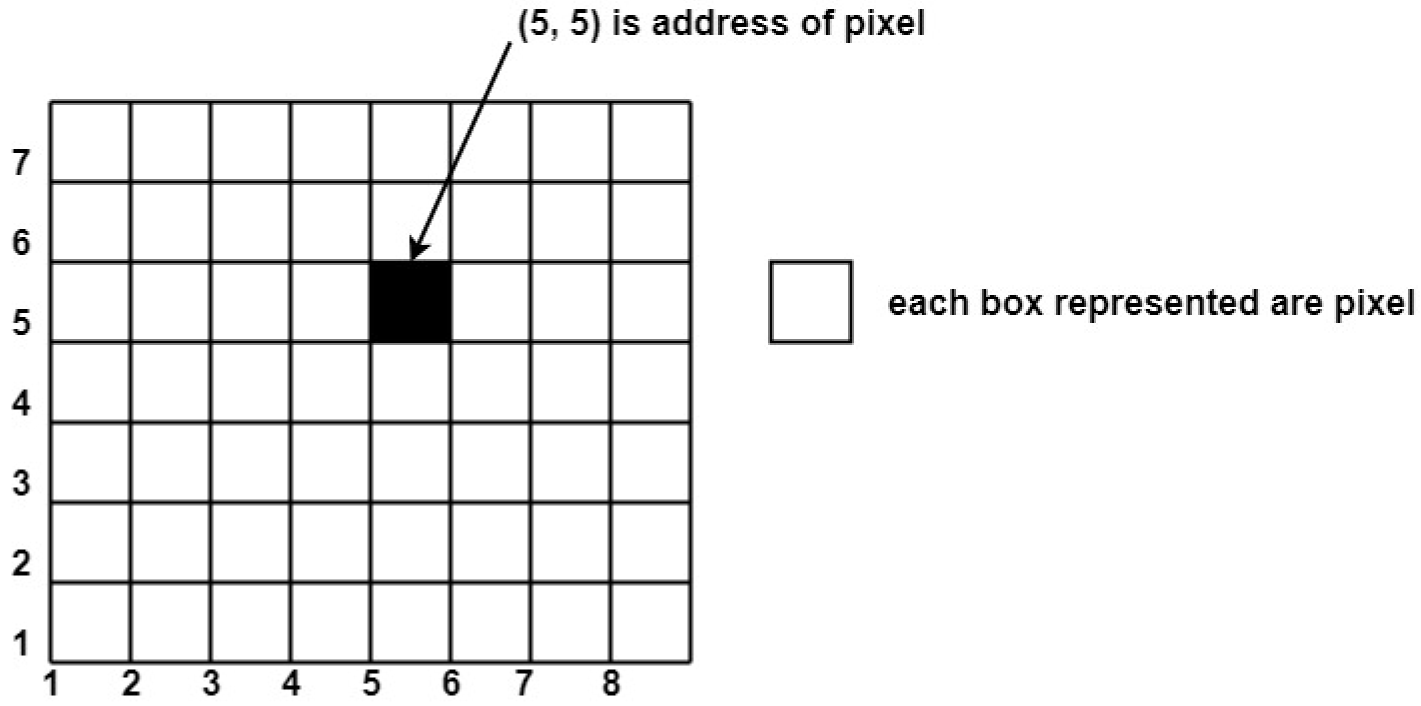
- Point
- Line
- Sector
- Arc
- Ellipse
- Rectangle
- Polygon
- Characters
- Filled Regions

Continue...

The term pixel is a short form of the picture element.

- It is also called a point or dot.
- It is the smallest picture unit accepted by display devices.
- A picture is constructed from hundreds of such pixels.
- Pixels are generated using commands.
- Lines, circle, arcs, characters; curves are drawn with closely spaced pixels.
- To display the digit or letter, matrix of pixels is used.
- The closer the dots or pixels are, the better will be the quality of picture.
- Picture will not appear jagged and unclear if pixels are closely spaced.
- So the quality of the picture is directly proportional to the density of pixels on the screen.
- Pixels are also defined as the smallest addressable unit or element of the screen.
- Each pixel can be assigned an address as shown in fig:

Continue...


(5, 5) is address of pixel
7
6
5
4
3
2
1
1 2 3 4 5 6 7 8
each box represented are pixel

Continue...

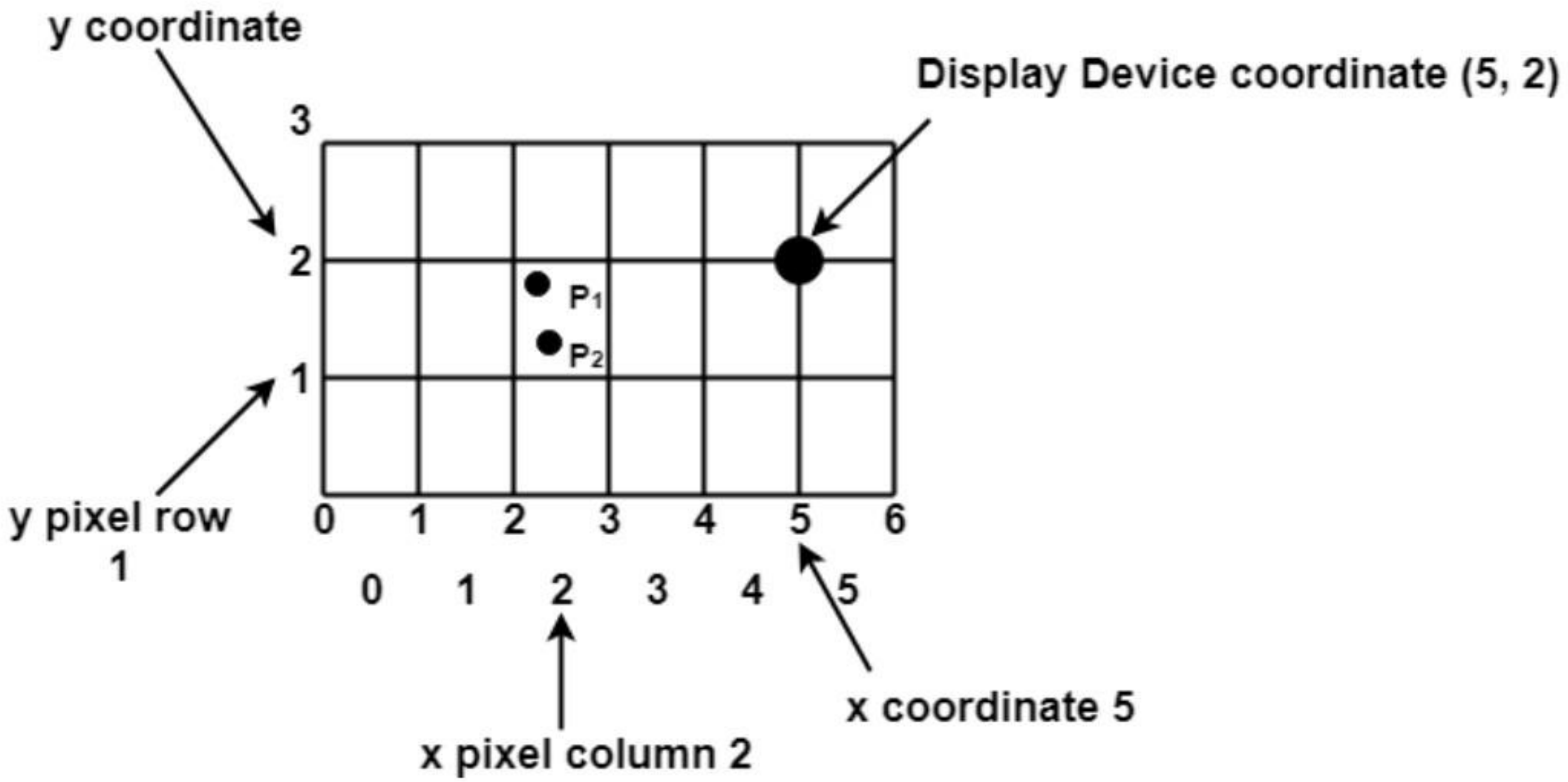
- Different graphics objects can be generated by setting the different intensity of pixels and different colors of pixels.
- Each pixel has some co-ordinate value. The coordinate is represented using row and column.
- P (5, 5) used to represent a pixel in the 5th row and the 5th column.
- Each pixel has some intensity value which is represented in memory of computer called a **frame buffer** or refresh buffer.
- This memory is a storage area for storing pixels values using which pictures are displayed.
- It is also called as digital memory.
- Inside the buffer, image is stored as a pattern of binary digits either 0 or 1.
- So there is an array of 0 or 1 used to represent the picture.
- In black and white monitors, black pixels are represented using 1's and white pixels are represented using 0's.
- In case of systems having one bit per pixel frame buffer is called a bitmap.
- In systems with multiple bits per pixel it is called a pixmap.

- **Bitmap:** In systems with one bit per pixel, the frame buffer is called a bitmap. In this case, black pixels are typically represented by 1s, while white pixels are represented by 0s. This results in a black-and-white image.
- **Pixmap:** In systems with multiple bits per pixel, the frame buffer is called a pixmap. This allows for a wider range of colors and shades to be represented. The number of bits per pixel directly affects the color depth and the quality of the displayed image.

Scan Converting a Point

- Each pixel on the graphics display does not represent a mathematical point.
- Instead, it means a region which theoretically can contain an infinite number of points.
- Scan-Converting a point involves illuminating the pixel that contains the point.

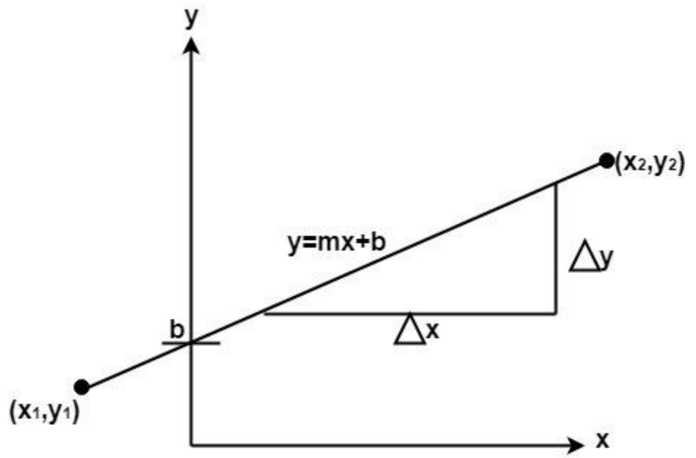
Example: Display coordinates points $P_1\ (2\frac{1}{4}, 1\frac{3}{4})$& $P_2 (2\frac{2}{3}, 1\frac{1}{4})$ as shown in fig would both be represented by pixel (2, 1). In general, a point p (x, y) is represented by the integer part of x & the integer part of y that is pixels [(INT (x), INT (y).



Δx

Scan Converting a Straight Line

- A straight line may be defined by two endpoints & an equation.
- In the following figure, the two endpoints are described by (x_1,y_1) and (x_2,y_2).
- The equation of the line is used to determine the x, y coordinates of all the points
- that lie between these two endpoints.



Using the equation of a straight line,

$y = mx + b$ where $m = \frac{\Delta y}{\Delta x}$ & b = the y intercept,

we can find values of y by incrementing

x from $x = x_1$, to $x = x_2$.

By scan-converting these calculated x, y values, we represent the line as a sequence of pixels.

Properties of Good Line Drawing Algorithm

- **Line should appear Straight:** We must appropriate the line by choosing addressable points close to it. If we choose well, the line will appear straight, if not, we shall produce crossed lines.

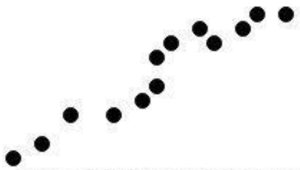


Fig: O/P from a poor line generating algorithm

- **Lines should terminate accurately:** Unless lines are plotted accurately, they may terminate at the wrong place.

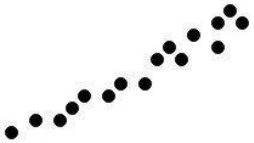


Fig: Uneven line density caused by bunching of dots.

- **Lines should have constant density:** Line density is proportional to the no. of dots displayed divided by the length of the line. To maintain constant density, dots should be equally spaced.
- **Line density should be independent of line length and angle:** This can be done by computing an approximating line-length estimate and to use a line-generation algorithm that keeps line density constant to within the accuracy of this estimate.
- **Line should be drawn rapidly:** This computation should be performed by special-purpose hardware.

Algorithm for line Drawing

- Direct use of line equation
- DDA (Digital Differential Analyzer)
- Bresenham's Algorithm

Direct use of Line Equation

- It is the simplest form of conversion.
- First of all scan P_1 and P_2 points. P_1 has co-ordinates (x_1', y_1') and (x_2', y_2').
- Then, calculate: $m = \frac{y_2' - y_1'}{x_2' - x_1'}$ $b = y_1' - mx_1'$
- If $|m| \leq 1$, for every integer value of x between and excluding x_1' & x_2', calculate the corresponding value of y using the equation: $y = mx + b$, and scan convert (x, y).
- If $|m| > 1$, for every integer value of y between and excluding y_1' & y_2', calculate the corresponding value of x using the equation, and scan convert (x, y).

Drawbacks:

- Though this approach is mathematically sound, it involves floating-point computation (multiplication & addition) in every step. It slows down the processing speed.
- The challenge is to find a way to achieve the same goal as quickly as possible.

Direct use of Line Equation

Example: A line with starting point as (0, 0) and ending point (6, 18) is given. Calculate value of intermediate points and slope of line.

Solution: P_1 (0,0) P_2 (6,18)

$x_1=0$
$y_1=0$
$x_2=6$
$y_2=18$

$$M = \frac{\Delta y}{\Delta x} = \frac{y_2 - y_1}{x_2 - x_1} = \frac{18-0}{6-0} = \frac{18}{6} = 3$$

We know the equation of line is

$y = m x + b$
$y = 3x + b$.............equation (i)

Put value of x from initial point in equation (i), i.e., (0, 0)
x =0, y=0

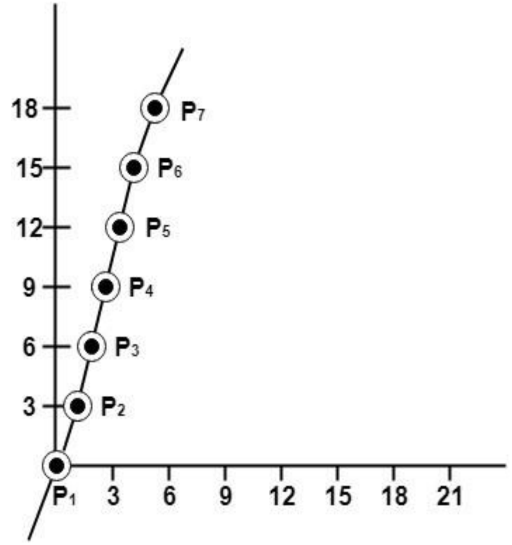
$0 = 3 \times 0 + b$
$0 = b \Rightarrow b=0$

put b = 0 in equation (i)

$y = 3x + 0$
$y = 3x$

Now calculate intermediate points

Let x = 1 ⇒ y = 3 x 1 ⇒ y = 3
Let x = 2 ⇒ y = 3 x 2 ⇒ y = 6
Let x = 3 ⇒ y = 3 x 3 ⇒ y = 9
Let x = 4 ⇒ y = 3 x 4 ⇒ y = 12
Let x = 5 ⇒ y = 3 x 5 ⇒ y = 15
Let x = 6 ⇒ y = 3 x 6 ⇒ y = 18



DDA(Digital Differential Analyzer) 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.
- Suppose at step i, the pixel is (x_i, y_i).
- Next point (x_{i+1}, y_{i+1})

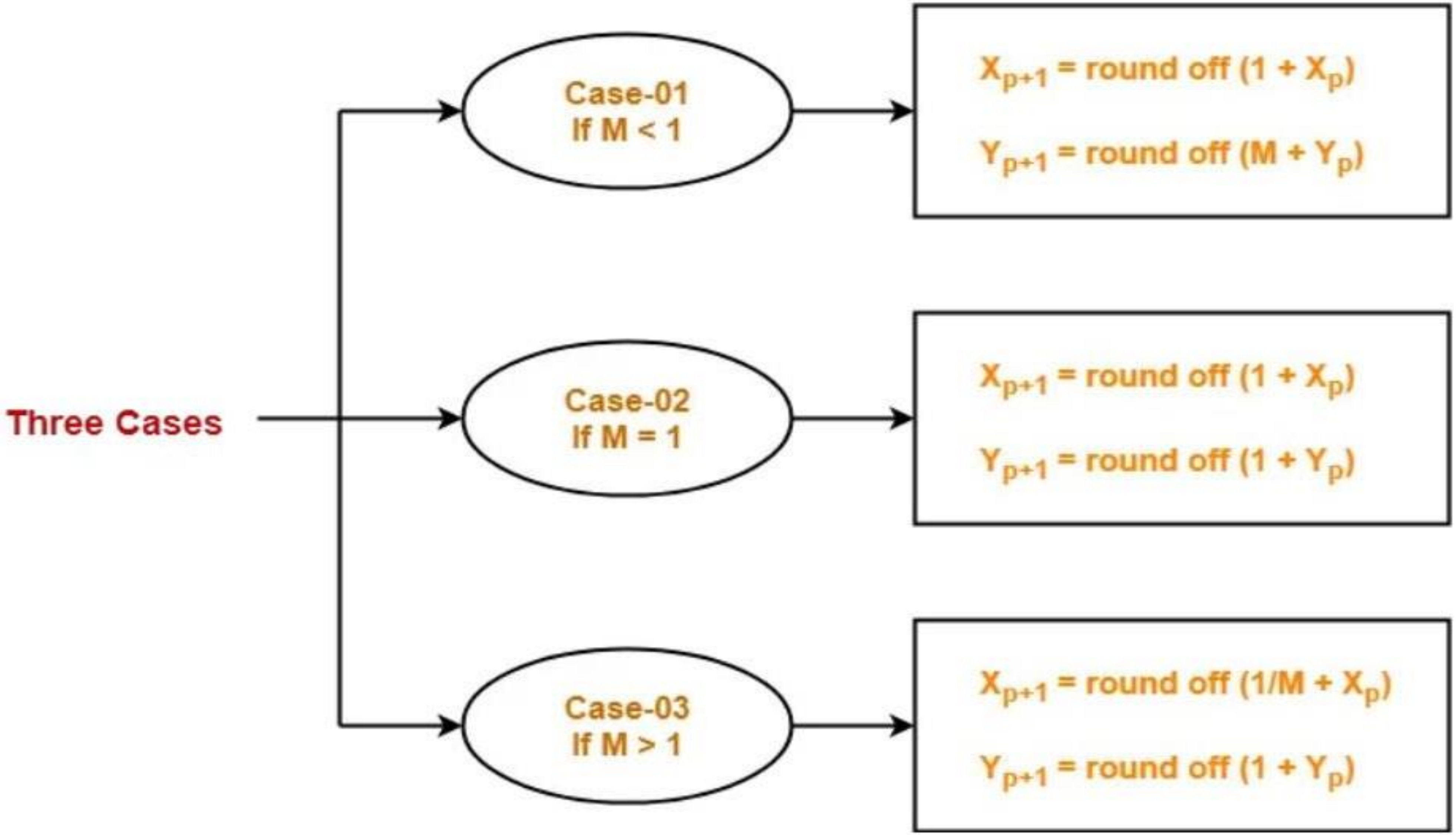
$$\Delta y / \Delta x = m, where \Delta y = y_{i+1} - y_i and \Delta x = x_{i+1} - x_i, We\ have$$

$$y_{i+1} = y_i + m\Delta x \qquad\qquad y_{i+1} = y_i + m \quad [\Delta x = 1]$$

$$and$$

$$x_{i+1} = x_i + \Delta y / m$$

Continue...

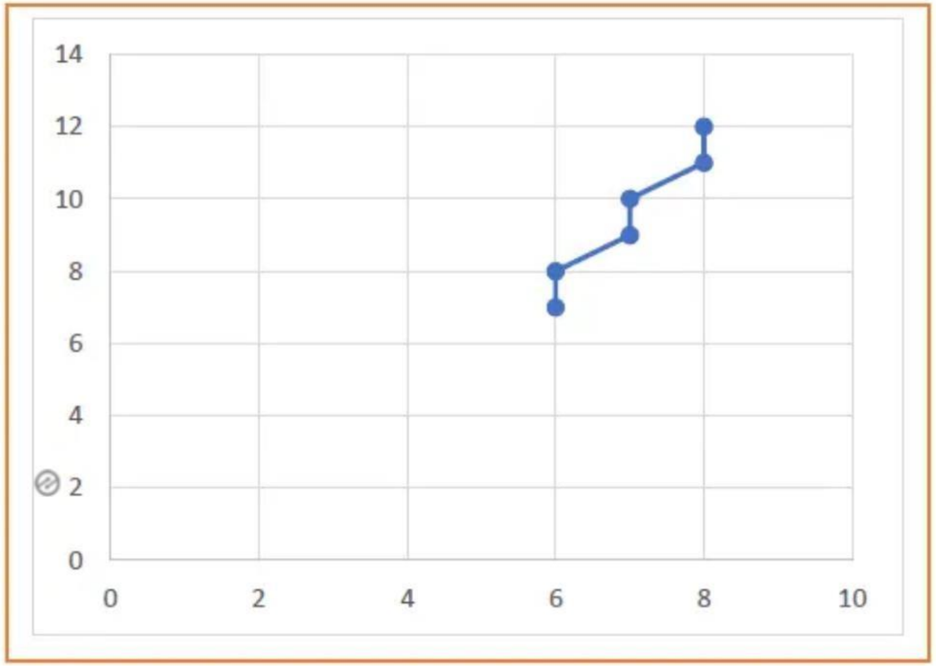

Three Cases
Case-01
If M < 1
X_{p+1} = round off (1 + X_p)
Y_{p+1} = round off (M + Y_p)
Case-02
If M = 1
X_{p+1} = round off (1 + X_p)
Y_{p+1} = round off (1 + Y_p)
Case-03
If M > 1
X_{p+1} = round off (1/M + X_p)
Y_{p+1} = round off (1 + Y_p)

Example -01

- Example: Calculate the points between the starting point (5, 6) and ending point (8, 12).

$$M=\frac{\Delta y}{\Delta x}=\frac{12-6}{8-5}=\frac{6}{3}=2$$

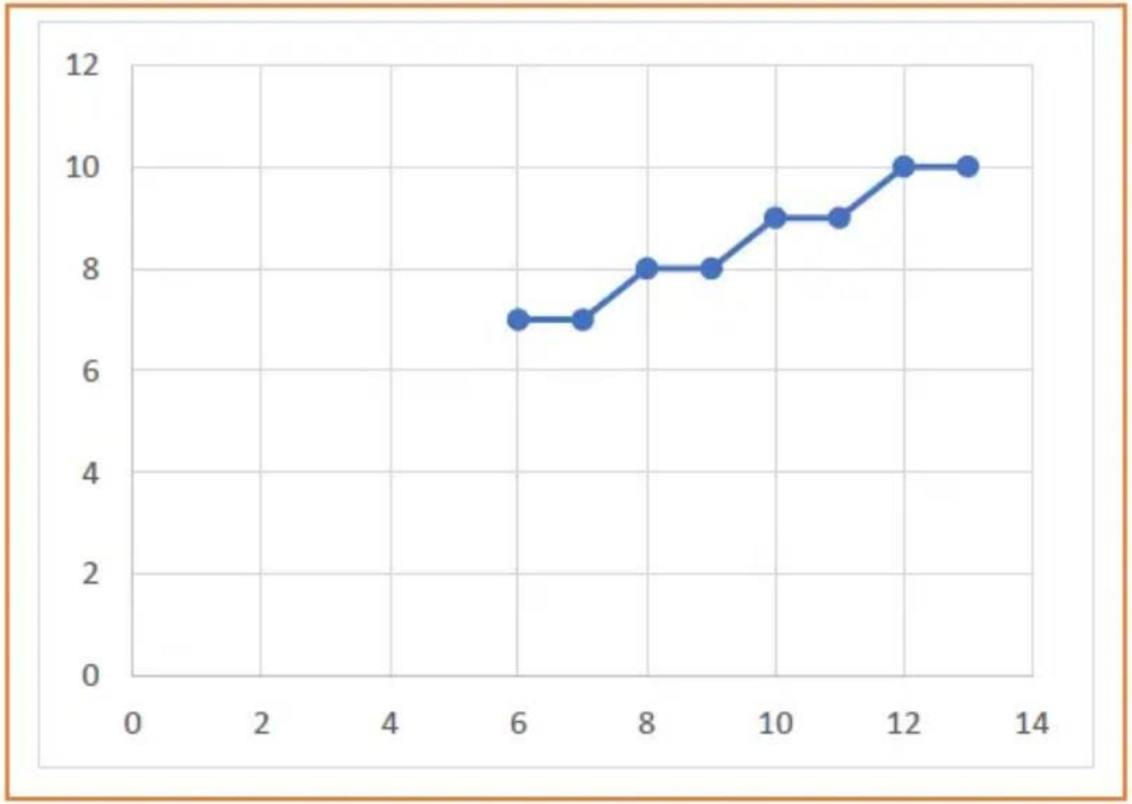
X_p	Y_p	X_{p+1}	Y_{p+1}	Round off (X_{p+1}, Y_{p+1})
5	6	5.5	7	(6, 7)
		6	8	(6, 8)
		6.5	9	(7, 9)
		7	10	(7, 10)
		7.5	11	(8, 11)
		8	12	(8, 12)



Example -02

- Example: Calculate the points between the starting point (5, 6) and ending point (13, 10).

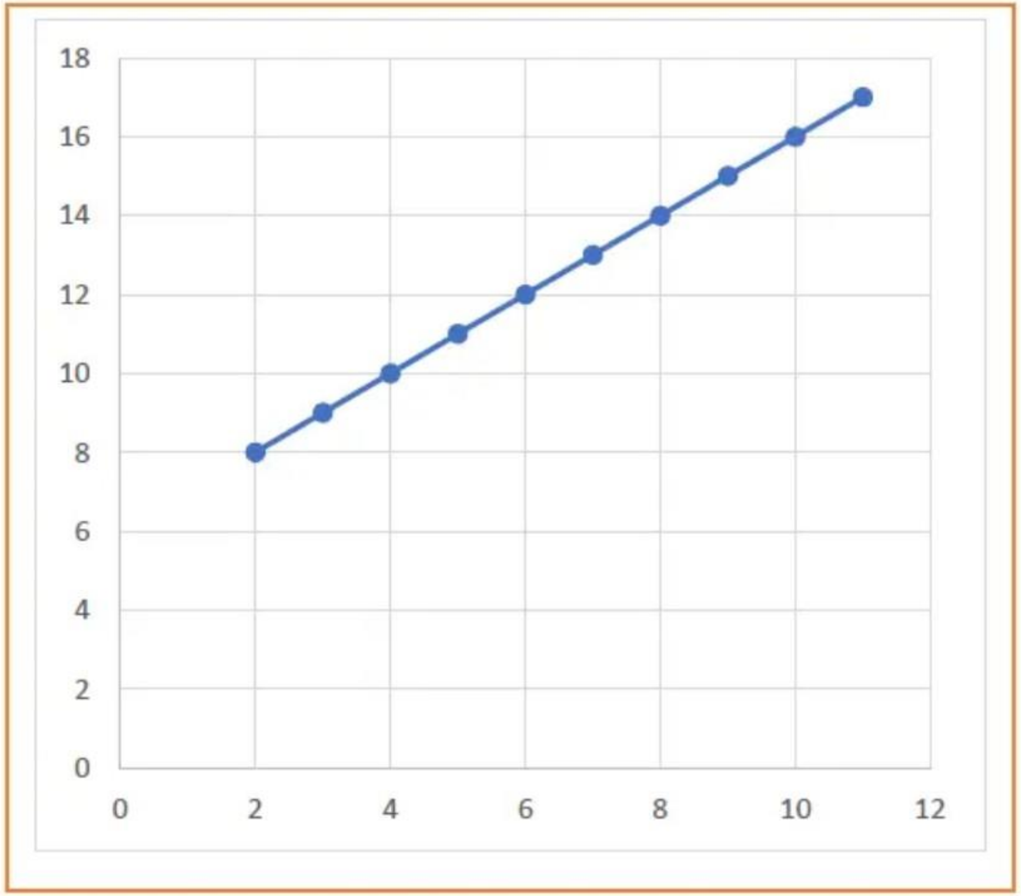
X_p	Y_p	X_{p+1}	Y_{p+1}	Round off (X_{p+1}, Y_{p+1})
5	6	6	6.5	(6, 7)
		7	7	(7, 7)
		8	7.5	(8, 8)
		9	8	(9, 8)
		10	8.5	(10, 9)
		11	9	(11, 9)
		12	9.5	(12, 10)
		13	10	(13, 10)



Example -03

- Example: Calculate the points between the starting point (1, 7) and ending point (11, 17).

X_p	Y_p	X_{p+1}	Y_{p+1}	Round off (X_{p+1}, Y_{p+1})
1	7	2	8	(2, 8)
		3	9	(3, 9)
		4	10	(4, 10)
		5	11	(5, 11)
		6	12	(6, 12)
		7	13	(7, 13)
		8	14	(8, 14)
		9	15	(9, 15)
		10	16	(10, 16)
		11	17	(11, 17)



THANK YOU