UNIT 2

OUTPUT PRIMITIVES

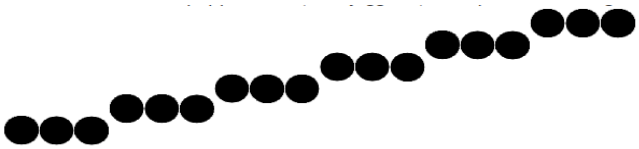
Introduction

In this unit you will learn about the line, circle and ellipse drawing algorithms. Basic shapes like lines, circles and curves play an important role in computer graphics. Such shapes are created as a collection of pixels. Various algorithms have been developed to determine the next pixel position to produce a particular shape. Basic arithmetic operations and some complex operations are performed to determine pixel positions.

Points and Lines

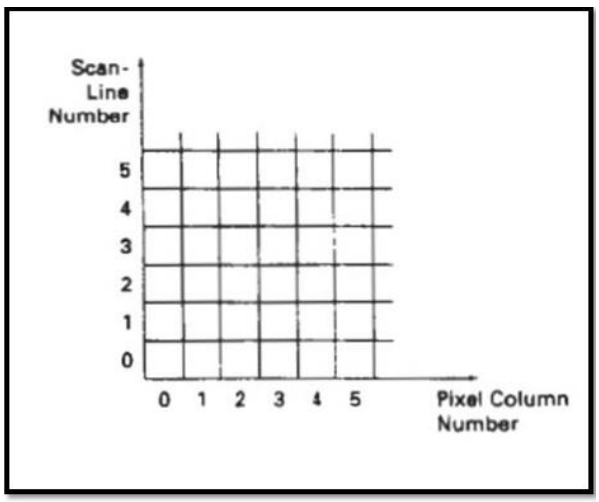
- Point plotting is done by converting a single coordinate position furnished by an application program into appropriate operations for the output device in use.
- Line drawing is done by calculating intermediate positions along the line path between two specified endpoint positions.
- The output device is then directed to fill in those positions between the endpoints with some color.
- For some device such as a pen plotter or random scan display, a straight line can be drawn smoothly from one end point together.
- Digital devices display a straight-line segment by plotting discrete points between the two endpoints.
- Discrete coordinate positions along the line path are calculated from the equation of the line.
- For a raster video display, the line intensity is loaded in frame buffer at the corresponding pixel positions.
- Reading from the frame buffer, the video controller then plots the screen pixels.
- Screen locations are referenced with integer values, so plotted positions may only approximate actual line positions between two specified endpoints.
- For example, line position of (12.36, 23.87) would be converted to pixel position (12,24).
- This rounding of coordinate values to integers causes lines to be displayed

With a stair step appearance (“the jaggies”), as represented in the below figure



- The stair step shape is noticeable in low resolution system, and we can improve their appearance somewhat by displaying them on high resolution system
- More effective techniques for smoothing raster lines are based on adjusting pixel intensities along the line paths
- For raster graphics device−level algorithms discuss here, object positions are specified directly in integer device coordinates.

- Pixel position will be referenced according to scan−line number and column number which is illustrated by following figure

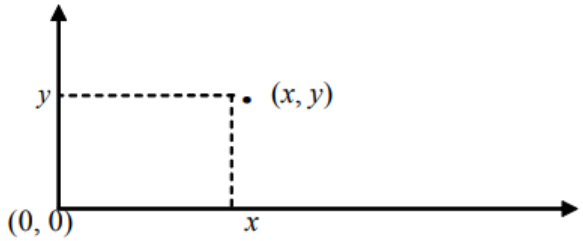


Pixel positions referenced by scan-line number and column number.

- To load the specified color into the frame buffer at a particular position, we will assume we have available low−level procedure of the form **setpixel(x,y)**
- Similarly for retrieve the current frame buffer intensity we assume to have procedure **getpixel(x,y)**

Point

A position in a plane is known as a point and any point can be represented by any ordered pair of numbers (x, y), where x is the horizontal distance from the origin and y is the vertical distance from the origin.



Representation of a Point at (x, y) Position

Line

Line can be represented by two points, i.e., both the points will be on the line and lines can also be described by an equation. It can also be defined by any point which satisfies the equation on the line. If two points P1 (x1, y1) and P2 (x2, y2) are specifying a line and another third point P (x, y) also satisfies the equation, the slope between points P1 P and P1 P2 will be as follows:

$$\frac{y - y_1}{x - x_1} = \frac{y_2 - y_1}{x_2 - x_1}$$

or $(x - x_1)(y_2 - y_1) = (x_2 - x_1)(y - y_1)$

or $(x_2 - x_1)y = (x - x_1)(y_2 - y_1) + y_1(x_2 - x_1)$

or $y = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x - x_1) + y_1$

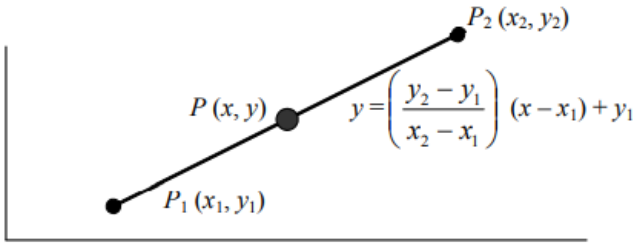


Fig. 2.2 *A Line with Equation*

This is the equation of a line that is passing through the coordinate points (x1 , y1) and (x2 , y2).

Line Segments

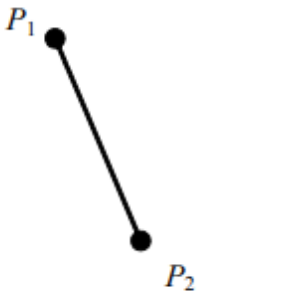
Any line or piece of line having end points is called a line segment. We can find the equation of any line segment using its end points and it can easily be checked whether any point lies on the line segment or not.

A point will be on the line segment if,

(i) It satisfies the equation of the line segment

(ii) Its x-coordinate lies between the x-coordinates of the end points

(iii) Its y-coordinate lies between the coordinates of the end points



A Line Segment

Vector

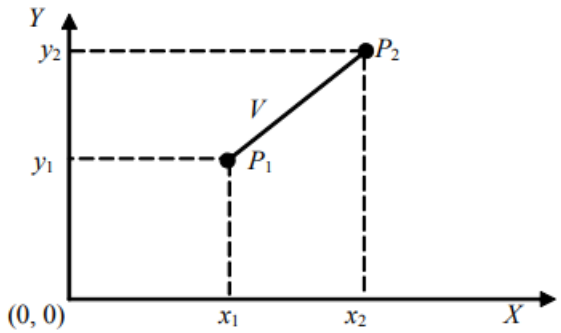
A vector is defined as the difference between two-point positions. Thus, for a two-dimensional vector, as seen in Figure 2.7, we have

$V = P_2 - P_1$

$= (x_2 - x_1, y_2 - y_1)$

$= V_x, V_y$

where the Cartesian components Vx and V y are projections of V onto the x and y axis or you can say Vx is the movement along the x-direction and V y is the movement along the y direction. Given the two point positions, we can obtain vector components in the same way for any co-ordinate frame.



Line Drawing Algorithms

1. DDA algorithm

2. Bresenham's algorithm

1. DDA 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. Digital Differential Analyzer algorithm is the simple line generation algorithm which is explained step by step here. Suppose at step i, the pixels is (xi,yi)

The line of equation for step i

yi=mxi+b.....equation 1

Next value will be

yi+1=mxi+1+b.....equation 2

m = Δy/Δx

yi+1-yi=Δy.....equation 3

yi+1-xi=Δx.....equation 4

yi+1=yi+Δy

Δy=mΔx

yi+1=yi+mΔx

Δx=Δy/m

xi+1=xi+Δx

xi+1=xi+Δy/m

Case1: When |m|<1 then (assume that x1<x2)

x= x1,y=y1 set Δx=1

yi+1=y1+m, x=x+1

Until x = x2

Case2: When |m|>1 then (assume that y1<y2)

x= x1,y=y1 set Δy=1

xi+1= , y=y+1

Until y → y2

Advantage:

1. It is a faster method than method of using direct use of line equation.

2. This method does not use multiplication theorem.

3. It allows us to detect the change in the value of x and y ,so plotting of same point twice is not possible.

4. This method gives overflow indication when a point is repositioned.

5. It is an easy method because each step involves just two additions.

Disadvantage:

1. It involves floating point additions rounding off is done. Accumulations of round off error cause accumulation of error.

2. Rounding off operations and floating point operations consumes a lot of time.

3. It is more suitable for generating line using the software. But it is less suited for hardware implementation.

DDA Algorithm:

Step1: Start Algorithm

Step2: Declare x1,y1,x2,y2,dx,dy,x,y as integer variables.

Step3: Enter value of x1,y1,x2,y2.

Step4: Calculate dx = x2-x1

Step5: Calculate dy = y2-y1

Step6: If ABS (dx) > ABS (dy)

Then step = abs (dx) Else

Step7: xinc=dx/step

yinc=dy/step

assign x = x1

assign y = y1

Step8: Set pixel (x, y)

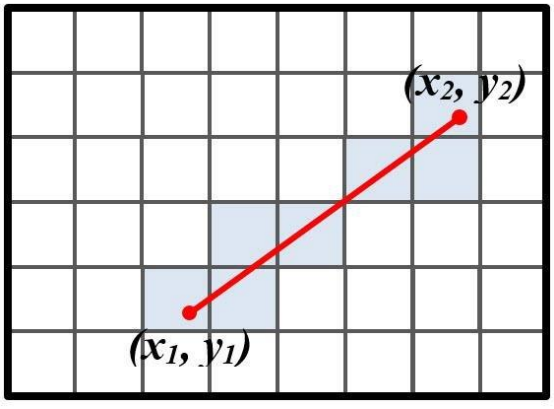
Step9: x = x + xinc

y = y + yinc

Set pixels (Round (x), Round (y))

Step10: Repeat step 9 until x = x2

Step11: End Algorithm



2. Bresenham's Line Algorithm

This algorithm is used for scan converting a line. It was developed by Bresenham. It is an efficient method because it involves only integer addition, subtractions, and multiplication operations. These operations can be performed very rapidly so lines can be generated quickly. In this method, next pixel selected is that one who has the least distance from true line.

Bresenham's Line Algorithm:

Step1: Start Algorithm

Step2: Declare variable x1,x2,y1,y2,d,i1,i2,dx,dy

Step3: Enter value of x1,y1,x2,y2

Where x1,y1are coordinates of starting point

And x2,y2 are coordinates of Ending point

Step4: Calculate dx = x2-x1

Calculate dy = y2-y1

Calculate i1=2*dy

Calculate i2=2*(dy-dx)

Calculate d=i1-dx

Step5: Consider (x, y) as starting point and x_{end} as maximum possible value of x.

If dx < 0

Then x = x2

y = y2

x_{end}=x1

If dx > 0

Then x = x1

y = y1

x_{end}=x2

Step6: Generate point at (x,y)coordinates.

Step7: Check if whole line is generated.

If x > = x_{end}

Stop.

Step8: Calculate co-ordinates of the next pixel

If $d < 0$

Then $d = d + i1$

If $d \geq 0$

Then $d = d + i2$

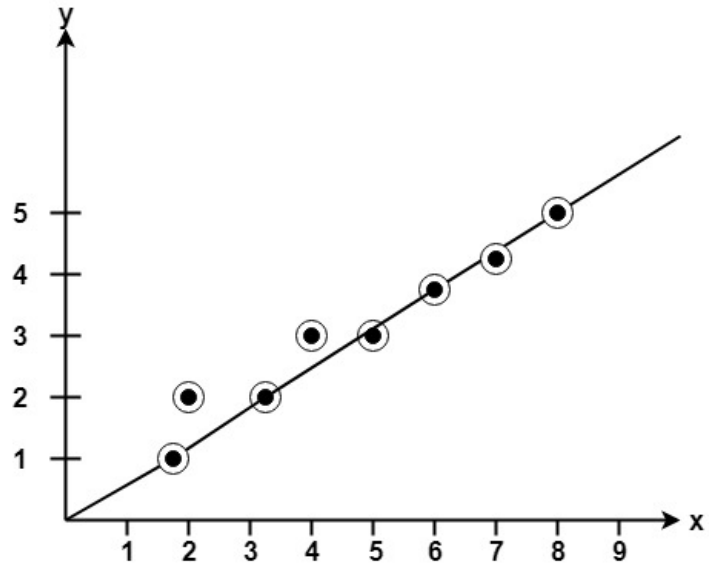
Increment $y = y + 1$

Step9: Increment $x = x + 1$

Step10: Draw a point of latest (x, y) coordinates

Step11: Go to step 7

Step12: End of Algorithm



Advantage:

1. It involves only integer arithmetic, so it is simple.

2. It avoids the generation of duplicate points.

3. It can be implemented using hardware because it does not use multiplication and division.

4. It is faster as compared to DDA (Digital Differential Analyzer) because it does not involve floating point calculations like DDA Algorithm.

Disadvantage:

This algorithm is meant for basic line drawing only Initializing is not a part of Bresenham's line algorithm. So to draw smooth lines, you should want to look into a different algorithm.

Differentiate between DDA & Bresenham's Algorithms

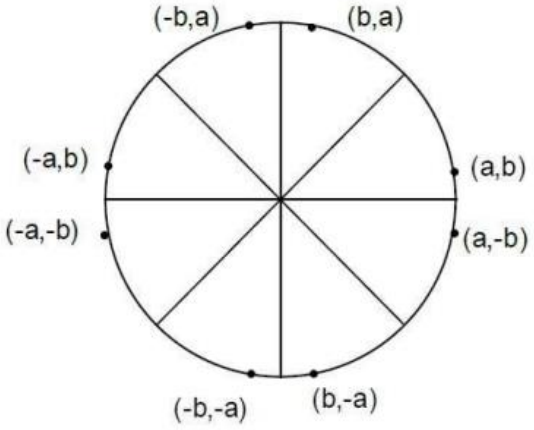
DDA Algorithm	Bresenyhham's Line Algorithm
1. DDA Algorithm use floating point, i.e., RealArithmetic.	1. Bresenham's Line Algorithm use fixed point, i.e., Integer Arithmetic
2. DDA Algorithms uses multiplication &division its operation	2. Bresenham's Line Algorithm uses onlysubtraction and addition its operation
3. DDA Algorithm is slowly than Bresenham'sLine Algorithm in line drawing because ituses real arithmetic (Floating Point operation)	3. Bresenham's Algorithm is faster than DDA Algorithm in line because itinvolves only addition & subtraction in its calculation and uses only integer arithmetic.
4. DDA Algorithm is not accurate and efficient as Bresenham's Line Algorithm.	4. Bresenham's Line Algorithm is more accurate and efficient at DDA Algorithm.
5.DDA Algorithm can draw circle and curves but are not accurate as Bresenham's Line Algorithm	5. Bresenham's Line Algorithm can draw circle and curves with more accurate than DDA Algorithm.

Circle Generating Algorithm

1. Bresenham's Algorithm
2. Midpoint Circle Algorithm

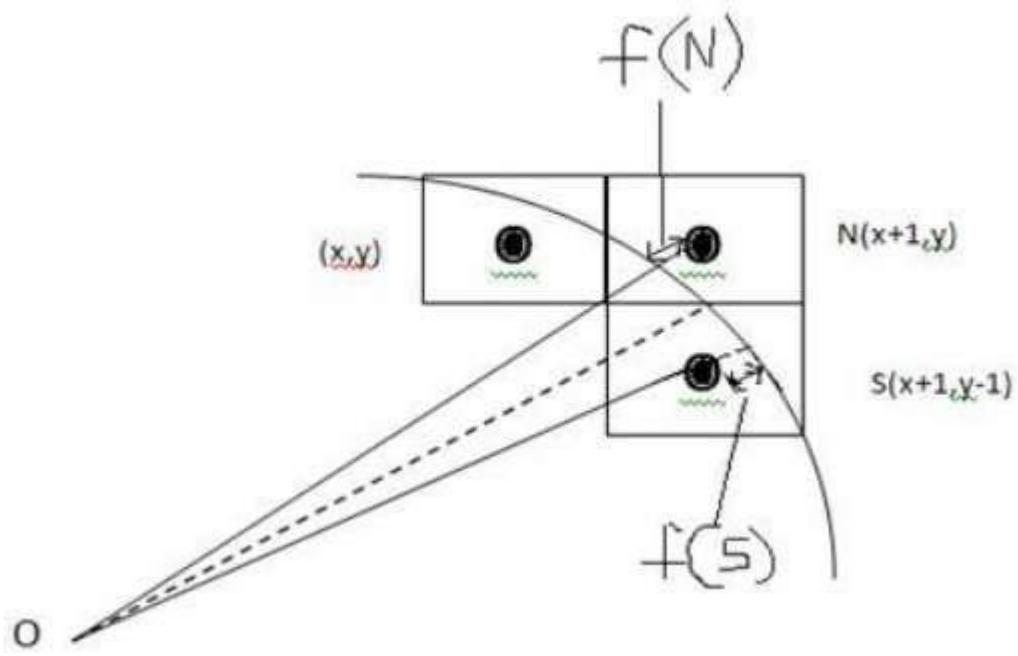
Circle Generation Algorithm Drawing a circle on the screen is a little complex than drawing a line. There are two popular algorithms for generating a circle −Bresenham's Algorithm and Midpoint Circle Algorithm. These algorithms are based on the idea of determining the subsequent points required to draw the circle. Let us discuss the algorithms in detail:

The equation of circle is $X^2+Y^2=r^2$ where r is radius.



1. Bresenham's Algorithm

We cannot display a continuous arc on the raster display. Instead, we have to choose the nearest pixel position to complete the arc. From the following illustration, you can see that we have put the pixel at X,Y location and now need to decide where to put the next pixel − at N X+1,Y or at S X+1,Y−1



This can be decided by the decision parameter d.

If d <= 0, then NX+1, Y is to be chosen as next pixel.

If d > 0, then SX+1, Y−1 is to be chosen as the next pixel.

Algorithm

Step 1 − Get the coordinates of the center of the circle and radius, and store them in x, y, and R respectively. Set P=0 and Q=R.

Step 2 − Set decision parameter D = 3 – 2R.

Step 3 − Repeat through step-8 while P ≤ Q.

Step 4 − Call Draw Circle X, Y, P, Q

Step 5 − Increment the value of P.

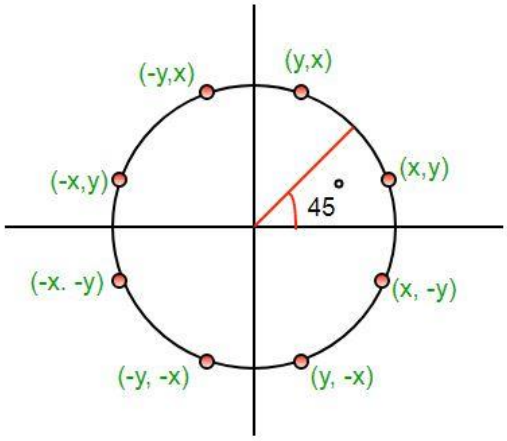
Step 6 − If D < 0 then D = D + 4P + 6.

Step 7 − Else Set R = R - 1, D = D + 4 P−Q + 10.

Step 8 − Call Draw Circle X, Y, P, Q

2. Midpoint Circle Algorithm

The **mid-point** circle drawing algorithm is an algorithm used to determine the points needed for rasterizing a circle. Then use the **mid-point** algorithm to calculate all the perimeter points of the circle in the **first octant** and then print them along with their mirror points in the other octants. This will work because a circle is symmetric about its centre.



f_{circle} = <0 if (x, y) is outside the circle boundary

=0if (x, y) is on the circle boundary

>0 if(x,y)is inside the circle boundary

1.Input radius r and circle center (xc,yc) and obtain the first point on the circumference of the circle centered on the origin as(x0,y0) = (0,r)

2. Calculate the initial value of the decision parameter as P=(5/4)-r

3. At each xk position, starting at k=0, perform the following test.

If Pk<0 the next point along the circle centered on (0,0) is (xk+1,yk) and
Pk+1=Pk+2xk+1+1

Otherwise the next point along the circle is (xk+1,yk-1) and Pk+1=Pk+2xk+1+1-2 yk+1

Where 2xk+1=2xk+2 and 2yk+1=2yk-2

4. Determine symmetry points in the other seven octants.

5. Move each calculated pixel position (x,y) onto the circular path centered at (xc,yc) and plot the coordinate values.

x=x+xc

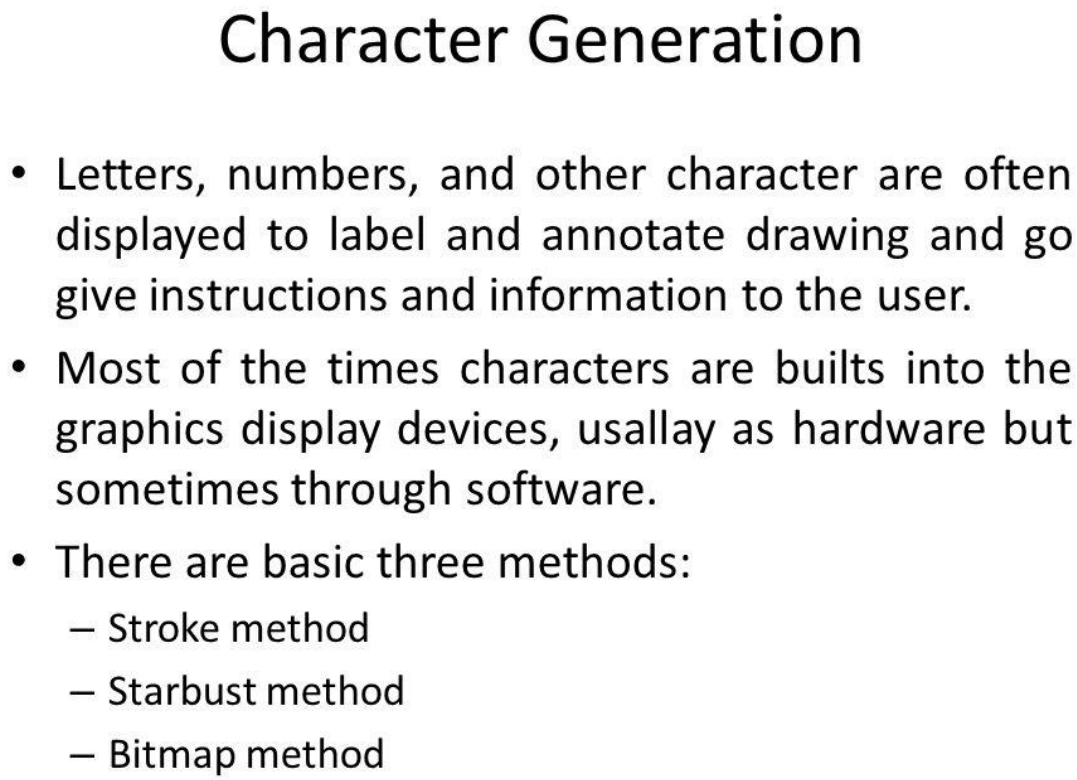
y=y+yc

6. Repeat step 3 through 5 until x>=y

Character Generation

Character Generator

A character generator adds characters or animated text to video in video-editing applications. A character generator can be hardware or software based. Character generators are largely used during the broadcast of live television presentations or events.

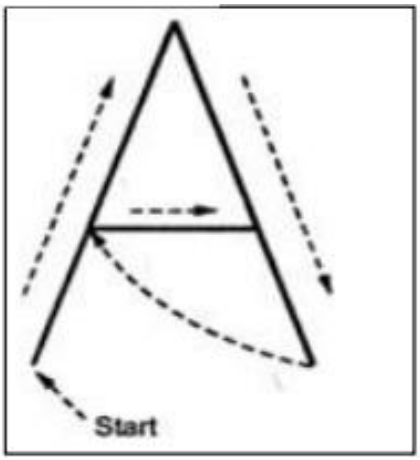


Character generation methods:

1. Stroke
2. Starburst
3. Bitmap

1) STROKE METHOD

Stroke method is based on natural method of text written by human being. In this method graph is drawing in the form of line by line. Line drawing algorithm DDA follows this method for line drawing. This method uses small line segments to generate a character. The small series of line segments are drawn like a stroke of pen to form a character. We can build our own stroke method character generator by calls to the line drawing algorithm. Here it is necessary to decide which line segments are needed for each character and then drawing these segments using line drawing algorithm.



2) STARBURST METHOD

In this method a fix pattern of line segments are used to generate characters. Out of these 24 line segments, segments required to display for particular character are highlighted. This method of character generation is called starbust method because of its characteristic appearance. The starbust patterns for characters A and M. the patterns for particular characters are stored in the form of 24 bit code, each bit representing one line segment. The bit is set to one to highlight the line segment; otherwise it is set to zero.

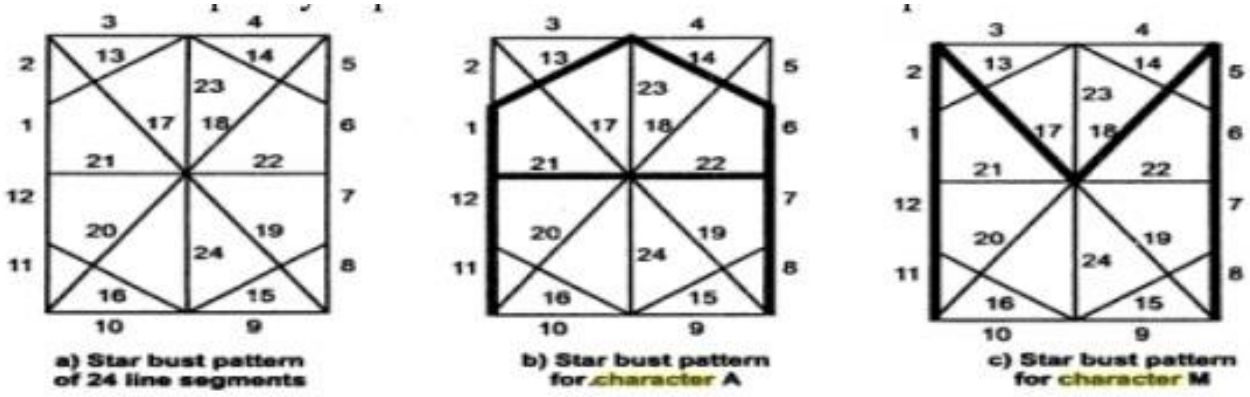
For example, 24-bit code for Character A is 0011 0000 0011 1100 1110 0001 and for character M is 0000 0011 0000 1100 1111 0011.

This method of character generation has some disadvantages. They are

1. The 24-bits are required to represent a character. Hence more memory is required.

2. Requires code conversion software to display character from its 24- bitcode.

3. Character quality is poor. It is worst for curve shaped characters.

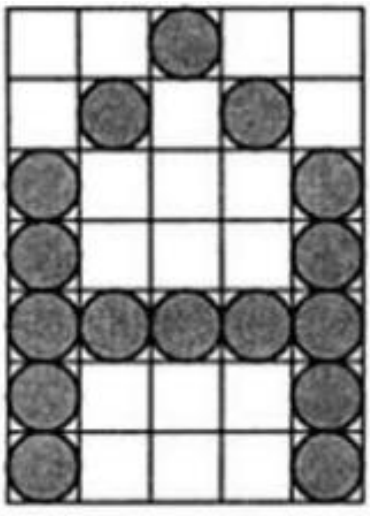


Character A : 0011 0000 0011 1100 11100001

Character M:0000 0011 0000 1100 11110011

3)BITMAP METHOD

Bitmap method is a called dot-matrix method as the name suggests this method use array of bits for generating a character. These dots are the points for array whose size is fixed. In bit matrix method when the dots is stored in the form of array the value 1 in array represent the characters i.e. where the dots appear we represent that position with numerical value 1 and the value where dots are not present is represented by 0 in array. It is also called dot matrix because in this method characters are represented by an array of dots in the matrix form. It is a two dimensional array having columns and rows. A 5x7 array is commonly used to represent characters. However 7x9 and 9x13 arrays are also used. Higher resolution devices such as inkjet printer or laser printer may use character arrays that are over 100x100.



Character A in 5 × 7 dot matrix format