UNIT- 1

Overview of Computer Graphics

What is computer Graphics?

Computer graphics is an art of drawing pictures, lines, charts, etc. using computers with the help of programming. Computer graphics image is made up of number of pixels. **Pixel** is the smallest addressable graphical unit represented on the computer screen.

Introduction

- Computer is information processing machine. User needs to communicate with computer and the computer graphics is one of the most effective and commonly used ways of communication with the user.
- It displays the information in the form of graphical objects such as pictures, charts, diagram and graphs.
- Graphical objects convey more information in less time and easily understandable formats for example statically graph shown in stock exchange.
- In computer graphics picture or graphics objects are presented as a collection of discrete pixels.
- We can control intensity and color of pixel which decide how picture look like.
- The special procedure determines which pixel will provide the best approximation to the desired picture or graphics object this process is known as **Rasterization.**
- The process of representing continuous picture or graphics object as a collection of discrete pixels is called **Scan Conversion.**

Advantages of computer graphics

- Computer graphics is one of the most effective and commonly used ways of communication with computer.
- It provides tools for producing picture of "real-world" as well as synthetic objects such as mathematical

Surfaces in 4D and of data that have no inherent geometry such as survey result.

- It has ability to show moving pictures thus possible to produce animations with computer graphics.
- With the use of computer graphics we can control the animation by adjusting the speed, portion of picture in view the amount of detail shown and so on.
- It provides tools called motion dynamics. In which user can move objects as well as observes as per requirement for example walk throw made by builder to show flat interior and surrounding.
- It provides facility called update dynamics. With this we can change the shape color and other properties of object.
- Now in recent development of digital signal processing and audio synthesis chip the interactive graphics can now provide audio feedback along with the graphical feed backs.

Application of computer graphics

- User interface: - Visual object which we observe on screen which communicates with user is one of the most useful applications of the computer graphics.
- Plotting of graphics and chart in industry, business, government and educational organizations drawing like bars, pie-charts, histogram's are very useful for quick and good decision making.
- Office automation and desktop publishing: - It is used for creation and dissemination of information. It is used in in-house creation and printing of documents which contains text, tables, graphs and other forms of drawn or scanned images or picture.

- Computer aided drafting and design: - It uses graphics to design components and system such as automobile bodies structures of building etc.

- Simulation and animation: - Use of graphics in simulation makes mathematic models and mechanical systems more realistic and easy to study.
- Art and commerce: - There are many tools provided by graphics which allows used to make their picture animated and attracted which are used in advertising.
- Process control: - Now a day's automation is used which is graphically displayed on the screen.
- Cartography: - Computer graphics is also used to represent geographic maps, weather maps, oceanographic charts etc.
- Education and training: - Computer graphics can be used to generate models of physical, financial and economic systems. These models can be used as educational aids.
- Image processing: - It is used to process image by changing property of the image.

Display devices

- Display devices are also known as output devices.
- Most commonly used output device in a graphics system is a video monitor.

Cathode-ray-tubes

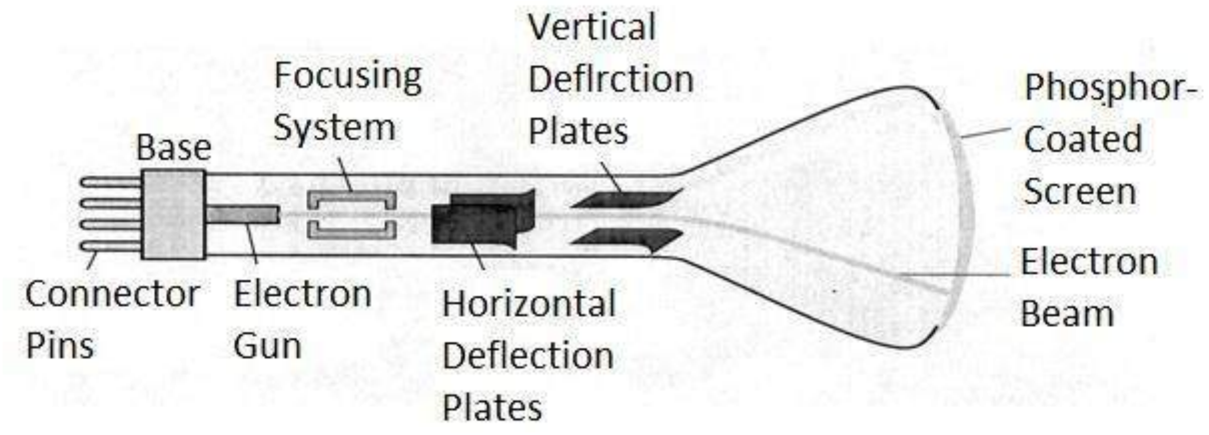


Fig. 1.1: - Cathode ray tube.

- It is an evacuated glass tube.
- An electron gun at the rear of the tube produce a beam of electrons which is directed towards the screen of the tube by a high voltage typically 15000 to 20000 volts
- Inner side screen is coated with phosphor substance which gives light when it is stroked bye electrons.
- Control grid controls velocity of electrons before they hit the phosphor.
- The control grid voltage determines how many electrons are actually in the electron beam. The negative the control voltage is the fewer the electrons that pass through the grid.
- Thus control grid controls Intensity of the spot where beam strikes the screen.
- The focusing system concentrates the electron beam so it converges to small point when hits the phosphor coating.
- Deflection system directs beam which decides the point where beam strikes the screen.
- Deflection system of the CRT consists of two pairs of parallel plates which are vertical and horizontal deflection plates.
- Voltage applied to vertical and horizontal deflection plates is control vertical and horizontal deflection

respectively.

- There are two techniques used for producing images on the CRT screen:

1. Vector scan/Random scansdisplay.
2. Raster scan display.

Vector scan/Random scan display

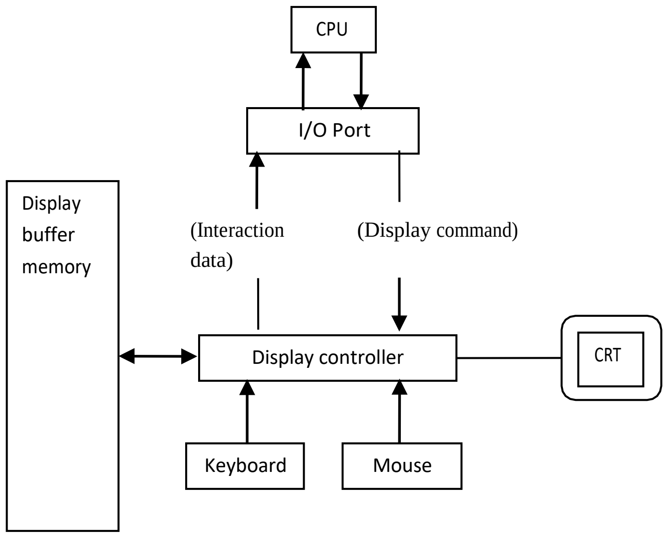


Fig. 1.2: - Architecture of a vector display.

- Vector scan display directly traces out only the desired lines on CRT.
- If we want line between point p1 & p2 then we directly drive the beam deflection circuitry which focus beam directly from point p1 to p2.
- If we do not want to display line from p1 to p2 and just move then we can blank the beam as we move it.
- To move the beam across the CRT, the information about both magnitude and direction is required. This information is generated with the help of vector graphics generator.
- Fig. 1.2 shows architecture of vector display. It consists of display controller, CPU, display buffer memory and CRT.
- Display controller is connected as an I/O peripheral to the CPU.
- Display buffer stores computer produced display list or display program.
- The Program contains point & line plotting commands with end point co-ordinates as well as character plotting commands.
- Display controller interprets command and sends digital and point co-ordinates to a vector generator.
- Vector generator then converts the digital co-ordinate value to analog voltages for beam deflection circuits that displace an electron beam which points on the CRT's screen.
- In this technique beam is deflected from end point to end point hence this techniques is also called random scan.
- We know as beam strikes phosphors coated screen it emits light but that light decays after few milliseconds and therefore it is necessary to repeat through the display list to refresh the screen at least 30 times per second to avoid flicker.
- As display buffer is used to store display list and used to refreshing, it is also called refresh buffer.

Raster scan display

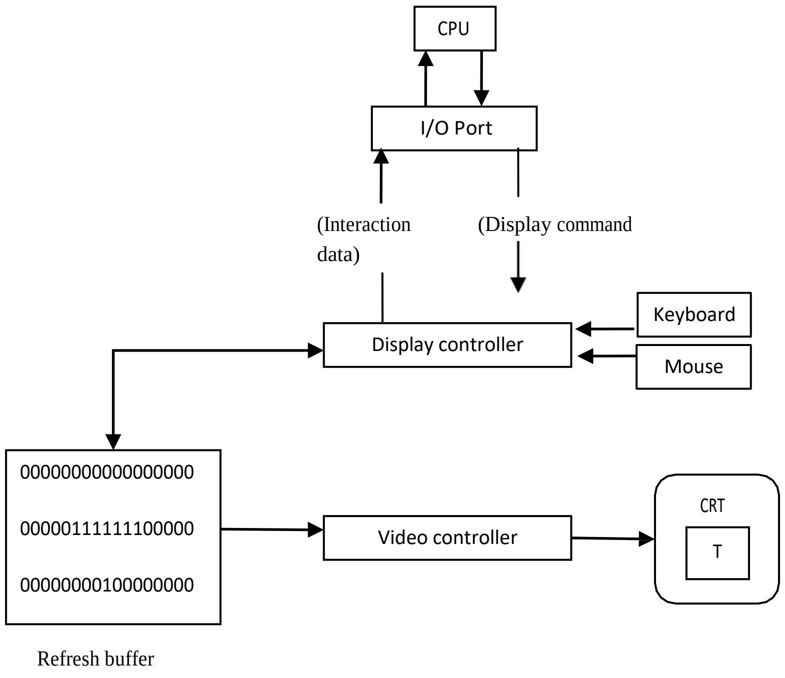


Fig. 1.3: - Architecture of a raster display.

- Fig. 1.3 shows the architecture of Raster display. It consists of display controller, CPU, video controller, refresh buffer, keyboard, mouse and CRT.
- The display image is stored in the form of 1's and 0's in the refresh buffer.
- The video controller reads this refresh buffer and produces the actual image on screen.
- It will scan one line at a time from top to bottom & then back to the top.

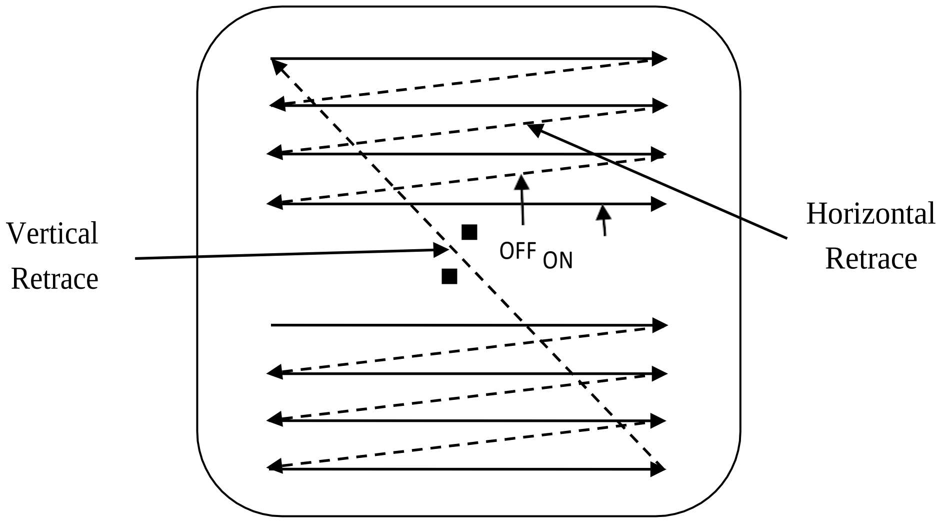


Fig. 1.4: - Raster scan CRT.

- In this method the horizontal and vertical deflection signals are generated to move the beam all over the screen in a pattern shown in fig. 1.4.
- Here beam is swept back & forth from left to the right.
- When beam is moved from left to right it is ON.

- When beam is moved from right to left it is OFF and process of moving beam from right to left after completion of row is known as **Horizontal Retrace.**
- When beam is reach at the bottom of the screen. It is made OFF and rapidly retraced back to the top left to start again and process of moving back to top is known as **Vertical Retrace**.
- The screen image is maintained by repeatedly scanning the same image. This process is known as

Refreshing of Screen.

- In raster scan displays a special area of memory is dedicated to graphics only. This memory is called

Frame Buffer.

- Frame buffer holds set of intensity values for all the screen points.
- That intensity is retrieved from frame buffer and display on screen one row at a time.
- Each screen point referred as pixel or **Pel (Picture Element).**
- Each pixel can be specified by its row and column numbers.
- It can be simply black and white system or color system.
- In simple black and white system each pixel is either ON or OFF, so only one bit per pixel is needed.
- Additional bits are required when color and intensity variations can be displayed up to 24-bits per pixel are included in high quality display systems.
- On a black and white system with one bit per pixel the frame buffer is commonly called a **Bitmap.** And for systems with multiple bits per pixel, the frame buffer is often referred as a **Pixmap.**

Difference between random scan and raster scan

Base of Difference	**Raster Scan System**	**Random Scan System**
Electron Beam	The electron beam is swept across the screen, one row at a time, from top to bottom.	The electron beam is directed only to the parts of screen where a picture is to be drawn.
Resolution	Its resolution is poor because raster system in contrast produces zigzag lines that are plotted as discrete point sets.	Its resolution is good because this system produces smooth lines drawings because CRT beam directly follows the line path.
Picture Definition	Picture definition is stored as a set of intensity values for all screen points, called pixels in a refresh buffer area.	Picture definition is stored as a set of line drawing instructions in a display file.
Realistic Display	The capability of this system to store intensity values for pixel makes it well suited for the realistic display of scenes contain shadow and color pattern.	These systems are designed for line-drawing and can't display realistic shaded scenes.
Draw an Image	Screen points/pixels are used to draw an image.	Mathematical functions are used to draw an image.

Color CRT monitors

- A CRT monitors displays color pictures by using a combination of phosphors that emit different colored light.
- It produces range of colors by combining the light emitted by different phosphors.
- There are two basic techniques for color display:
    1. Beam-penetration technique
    2. Shadow-mask technique

Beam-penetration technique

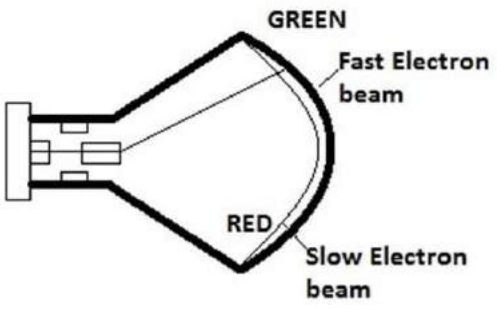


Fig. 1.5: - Beam-penetration CRT

- This technique is used with random scan monitors.
- In this technique inside of CRT coated with two phosphor layers usually red and green. The outer layer of red and inner layer of green phosphor.
- The color depends on how far the electron beam penetrates into the phosphor layer.
- A beam of fast electron penetrates more and excites inner green layer while slow electron excites outer red layer.
- At intermediate beam speed we can produce combination of red and green lights which emit additional two colors orange and yellow.
- The beam acceleration voltage controls the speed of the electrons and hence color of pixel.
- It is a low cost technique to produce color in random scan monitors.
- It can display only four colors.
- Quality of picture is not good compared to other techniques.

Shadow-mask technique

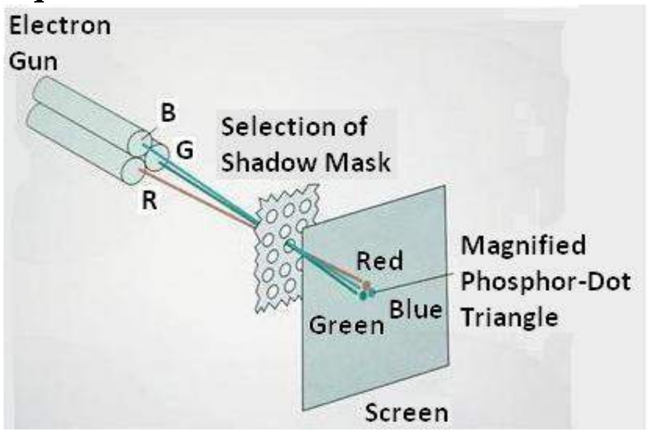


Fig. 1.6: - Shadow-mask CRT.

- It produces wide range of colors as compared to beam-penetration technique.
- This technique is generally used in raster scan displays. Including color TV.
- In this technique CRT has three phosphor color dots at each pixel position. One dot for red, one for green and one for blue light. This is commonly known as **Dot Triangle.**
- Here in CRT there are three electron guns present, one for each color dot. And a shadow mask grid just behind the phosphor coated screen.
- The shadow mask grid consists of series of holes aligned with the phosphor dot pattern.
- Three electron beams are deflected and focused as a group onto the shadow mask and when they pass through a hole they excite a dot triangle.
- In dot triangle three phosphor dots are arranged so that each electron beam can activate only its corresponding color dot when it passes through the shadow mask.
- A dot triangle when activated appears as a small dot on the screen which has color of combination of three small dots in the dot triangle.
- By changing the intensity of the three electron beams we can obtain different colors in the shadow mask CRT.

Direct-view storage tubes (DVST)

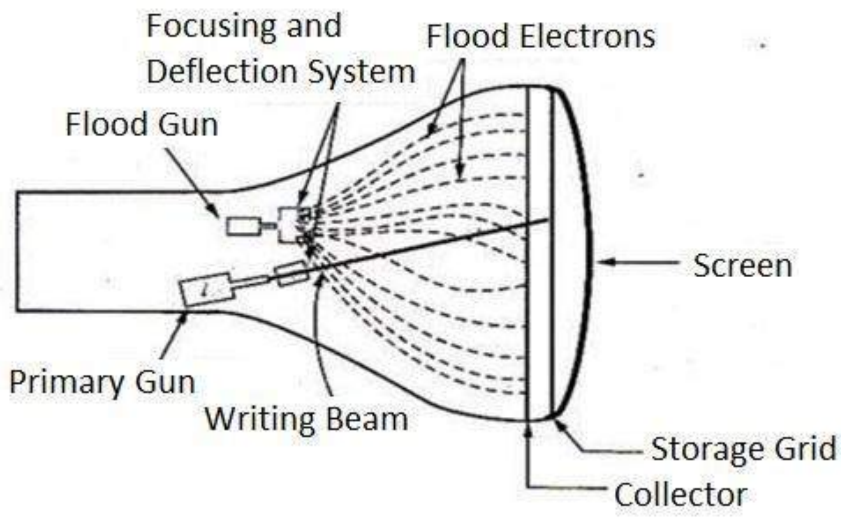


Fig. 1.7: - Direct-view storage tube.

- In raster scan display we do refreshing of the screen to maintain a screen image.
- DVST gives alternative method for maintaining the screen image.
- DVST uses the storage grid which stores the picture information as a charge distribution just behind the phosphor coated screen.
- DVST consists two electron guns a primary gun and a flood gun.
- A primary gun stores the picture pattern and the flood gun maintains the picture display.
- A primary gun emits high speed electrons which strike on the storage grid to draw the picture pattern.
- As electron beam strikes on the storage grid with high speed, it knocks out electrons from the storage grid keeping the net positive charge.
- The knocked out electrons are attracted towards the collector.
- The net positive charge on the storage grid is nothing but the picture pattern.
- The continuous low speed electrons from flood gun pass through the control grid and are attracted to the positive charged area of the storage grid.

- The low speed electrons then penetrate the storage grid and strike the phosphor coating without affecting the positive charge pattern on the storage grid.
- During this process the collector just behind the storage grid smooth out the flow of flood electrons.

Advantage of DVST

- Refreshing of CRT is not required.
- Very complex pictures can be displayed at very high resolution without flicker.
- Flat screen.

Disadvantage of DVST

- They do not display color and are available with single level of line intensity.
- For erasing it is necessary to removal of charge on the storage grid so erasing and redrawing process take several second.
- Erasing selective part of the screen cannot be possible.
- Cannot used for dynamic graphics application as on erasing it produce unpleasant flash over entire screen.
- It has poor contrast as a result of the comparatively low accelerating potential applied to the flood electrons.
- The performance of DVST is somewhat inferior to the refresh CRT.

Flat Panel Display

- The term flat panel display refers to a class of video device that have reduced volume, weight & power requirement compared to a CRT.
- As flat panel display is thinner than CRTs, we can hang them on walls or wear on our wrists.
- Since we can even write on some flat panel displays they will soon be available as pocket notepads.
- We can separate flat panel display in two categories:
  1. **Emissive displays**: - the emissive display or **emitters** are devices that convert electrical energy into light. For Ex. Plasma panel, thin film electroluminescent displays and light emitting diodes.
  2. **Non emissive displays**: - non emissive display or **non emitters** use optical effects to convert sunlight or light from some other source into graphics patterns. For Ex. LCD (Liquid Crystal Display).

Plasma Panels displays

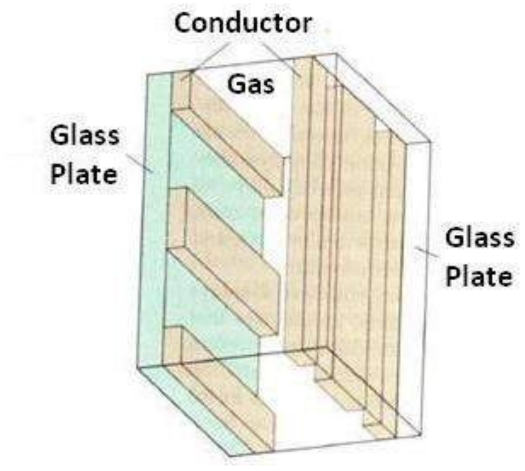


Fig. 1.8: - Basic design of a plasma-panel display device.

- This is also called gas discharge displays.
- It is constructed by filling the region between two glass plates with a mixture of gases that usually includes neon.
- A series of vertical conducting ribbons is placed on one glass panel and a set of horizontal ribbon is built into the other glass panel.
- Firing voltage is applied to a pair of horizontal and vertical conductors cause the gas at the intersection of the two conductors to break down into glowing plasma of electrons and ions.
- Picture definition is stored in a refresh buffer and the firing voltages are applied to refresh the pixel positions, 60 times per second.
- Alternating current methods are used to provide faster application of firing voltages and thus brighter displays.
- Separation between pixels is provided by the electric field of conductor.
- One disadvantage of plasma panels is they were strictly monochromatic device that means shows only one color other than black like black and white.

Thin Film Electroluminescent Displays.

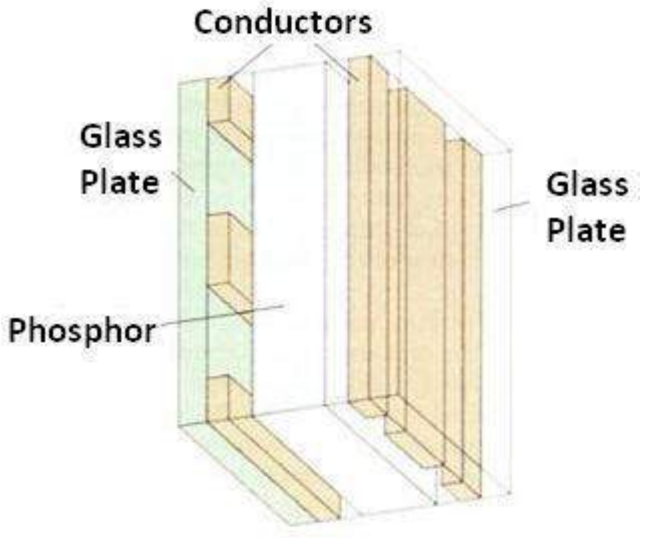


Fig. 1.9: - Basic design of a thin-film electro luminescent display device.

- It is similar to plasma panel display but region between the glass plates is filled with phosphors such as zink sulphide doped with magnesium instead of gas.
- When sufficient voltage is applied the phosphors becomes a conductor in area of intersection of the two electrodes.
- Electrical energy is then absorbed by the manganese atoms which then release the energy as a spot of light similar to the glowing plasma effect in plasma panel.
- It requires more power than plasma panel.
- In this good color and gray scale difficult to achieve.

Light Emitting Diode (LED)

- In this display a matrix of multi-color light emitting diode is arranged to form the pixel position in the display. And the picture definition is stored in refresh buffer.
- Similar to scan line refreshing of CRT information is read from the refresh buffer and converted to voltage levels that are applied to the diodes to produce the light pattern on the display.

Liquid Crystal Display (LCD)

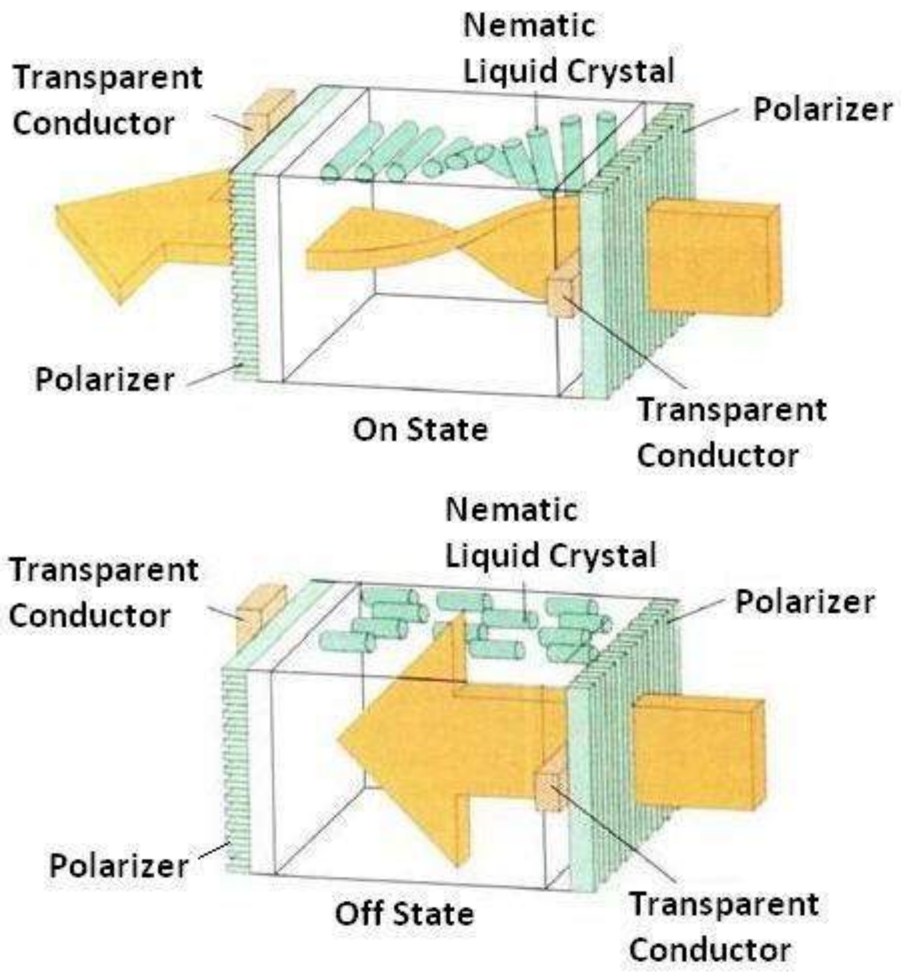


Fig. 1.10: - Light twisting shutter effect used in design of most LCD.

- It is generally used in small system such as calculator and portable laptop.
- This non emissive device produce picture by passing polarized light from the surrounding or from an internal light source through liquid crystal material that can be aligned to either block or transmit the light.
- The liquid crystal refreshes to fact that these compounds have crystalline arrangement of molecules then also flows like liquid.
- It consists of two glass plates each with light polarizer at right angles to each other sandwich the liquid crystal material between the plates.
- Rows of horizontal transparent conductors are built into one glass plate, and column of vertical conductors are put into the other plates.
- The intersection of two conductors defines a pixel position.
- In the ON state polarized light passing through material is twisted so that it will pass through the opposite polarizer.
- In the OFF state it will reflect back towards source.
- We applied a voltage to the two intersecting conductor to align the molecules so that the light is not twisted.
- This type of flat panel device is referred to as a passive matrix LCD.
- In active matrix LCD transistors are used at each (x, y) grid point.

- Transistor cause crystal to change their state quickly and also to control degree to which the state has been changed.
- Transistor can also serve as a memory for the state until it is changed.
- So transistor make cell ON for all time giving brighter display then it would be if it had to be refresh periodically

Advantages of LCD display

- Low cost.
- Low weight.
- Small size
- Low power consumption.

Hardcopy Devices

All the **output devices** can be categorized into two categories

- Hard Copy Devices
- Soft Copy Devices

Hard copy devices are those that give the output in the tangible form. Printers and Plotters are two common hard copy devices.

Soft copy devices give output in the intangible form or the virtual form, e.g. something displayed on a screen. All the computer monitors are covered under this category.

<u>Printers</u>

All the printers irrespective of the technology used can be categorized as

- Impact Printers
- Non Impact Printers

Impact printers are those printers in which there is a direct contact between the printing head and the paper on which the print is produced.

- they work by striking a head or a needle against an inked ribbon which leaves a mark on the paper.
- These printers produce a lot of noise when printing, because of the head striking the paper.
- Examples are ***Dot Matrix, Daisy Wheel*** and ***Line printers***.

In the case of **non-impact printers** the printing head never comes in direct contact with the paper.

- These printers work by spraying ink on the paper.
- Electrostatic or electromagnetic charge is used in these printers.
- Examples are ***Ink-Jet*** and ***Laser*** printers.

<u>Dot-Matrix Printers</u>:

- Dot Matrix is an impact printer.
- These printer forms characters from individual dots.
- These printers have a print head which runs back and forth on a paper.
- The print head has a two-dimensional array of pins called dot matrix. There may be 9 to 24 pins in the dot matrix.
- From this array of pins some pins are drawn out (or driven forward) to form the shape of a character.
- The drawn out pins strike an ink soaked cloth ribbon against a paper. This forms that particular character on the paper.
- Thus dot matrix printers can be used to print different fonts of characters.
- Since mechanical force is used; carbon copies of documents can be taken.
- 40 to 250 characters can be printed per second.

<u>Daisy Wheel Printers</u>:

- This is an impact printer.
- Only preformed fonts of characters can be printed.
- This printer contains a daisy wheel. Daisy wheel is made of plastic or metal. This holds an entire character set as raised characters molded on each "petal".
- A motor rotates the daisy wheel to position the required character between the hammer and the ribbon.
- A small hammer then strikes the petal, which in turn strikes the inked ribbon to leave the character mark on the paper.
- The daisy wheel and hammer are mounted on a sliding carriage similar to that used by dot matrix printers.
- Different fonts cannot be printed using this technology.

Line Printers:

- The line printer is a high speed impact printer in which one line is printed at a time.
- 600-1200 lines can be printed per minute.
- Drum printer is an example of line printers.
- These printers are very expensive.
- These kinds of printers were popular in the early days of computers, but the technology is still in use.

Drum Printers

- In a drum printer, a fixed font character set is engraved onto a number of print wheels.
- There are as many print wheels as the number of columns (letters in a line) the printer could print.
- The print wheels are joined to form a large drum (cylinder),
- This drum spins at high speed and paper and an inked ribbon is moved past the print position.
- As the desired character for each column passes the print position; a hammer strikes the paper from the rear and presses the paper against the ribbon and the drum, causing the desired character to be printed on the paper.

Ink-Jet Printers:

- Inkjet printer is a non impact printer, Core of an inkjet printer is the print head.
- The print head contains an ink cartridge which has a series of nozzles that are used to spray tiny drops of ink on to the paper.
- Ink cartridges come in various combinations, such as separate black and color cartridges, color and black in a single cartridge or even a cartridge for each ink color.
- A motor moves the print head back and forth across the paper.
- Different types of inkjet printers form their droplets of ink in different ways. There are two main inkjet technologies currently used by printer manufacturers
  - **Thermal bubble -** This method is commonly referred to as **bubble jet**. In a thermal inkjet printer, tiny resistors create heat, and this heat vaporizes ink to create a bubble. As the bubble expands, some of the ink is pushed out of a nozzle onto the paper. When the bubble "pops" (collapses), a vacuum is created. This pulls more ink into the print head from the cartridge. A typical bubble jet print head has **300 or 600 tiny nozzles**, and all of them can fire a droplet simultaneously.
  - **Piezoelectric -** This technology uses **piezo crystals**. A crystal is located at the back of the ink reservoir of each nozzle. The crystal receives a tiny electric charge that causes it to vibrate. When the crystal vibrates inward, it forces a tiny amount of ink out of the nozzle. When it vibrates out, it pulls some more ink into the reservoir to replace the ink sprayed out.
- The ink droplets are subjected to an electrostatic field created by a charging electrode as they form. Charged droplets are separated by one or more uncharged “guard droplets” to minimize electrostatic repulsion between neighbouring droplets. The charged droplets pass through an electrostatic field and are directed (deflected) by electrostatic deflection plates to print on the Paper.

Laser Printers:

- A laser printer is a **non impact** printer, which produces a page of text at a time.
- Laser printer uses the principle of **Static Electricity** to print.
- This printer has revolving cylinder called **Drum**.
- Drum is given a **positive charge**.
- A **Laser beam** is used to draw the image to be printed, on the drum with negative charge. This discharges some portion of the charge on the drum. This creates electrostatic image of the print on the drum with no charge, and the background is left positively charged.
- The drum is then exposed to **toner** from which positively charged toner particles mixed with carbon black are released. Since positive charge repels positive charge, the toner particles settles on the discharged areas of the

drum, this is exactly the image to be printed.

- The paper is then pressed against the drum; this transfers the toner particles on to the paper.

- Paper is then passed through a **fuser**, which is a set of heated rollers; this melts the carbon black on the paper to form the desired print.

Plotters:

Another hard copy output device is plotter. Plotter is a printing device which can draw continuous lines. This is useful to print vector graphics rather than raster graphics unlike normal printers. Plotters are widely used in applications like CAD.

- Plotters print by moving one or more pen across the surface of a piece of paper. This means that plotters are restricted to line art, rather than raster graphics as with other printers.
- Pen plotters can draw complex line art, including text, but do so slowly because of the mechanical movement of the pens. They are often incapable of efficiently creating a solid region of color, but can draw an area by drawing a number of close, regular lines.
- Plotters offered the fastest way to efficiently produce very large drawings or color high-resolution vector-based artwork when computer memory was very expensive and processor power was very limited.
- There are a number of different types of plotters:
- A **drum plotter** draws on paper wrapped around a drum which turns to produce one direction of the plot, while the pens move to provide the other direction.
- A **flatbed plotter** draws on paper placed on a flat surface; and an electrostatic plotter draws on negatively charged paper with positively charged toner.
- Pen plotters have essentially become obsolete, and have been replaced by large-format inkjet printers and toner based printers.
- They are most frequently used for CAE (computer-aided engineering) applications, such as CAD (computer-aided design) and CAM (computer-aided manufacturing).

Raster-graphics system **with a display processor**

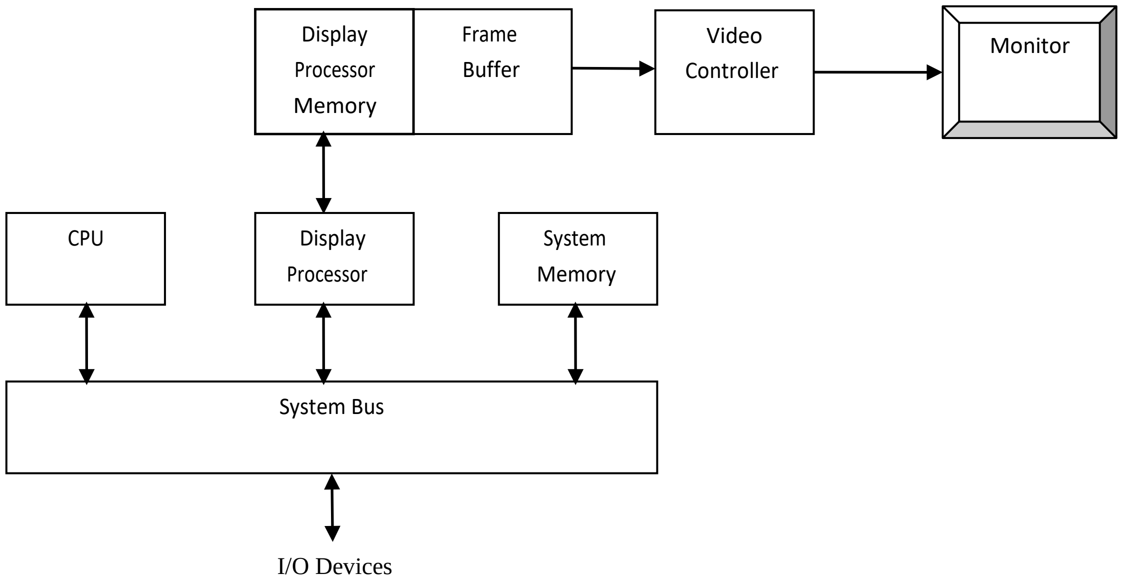


Fig. 1.17: - Architecture of a raster-graphics system with a display processor.

- One way to designing raster system is having separate display coprocessor.
- Purpose of display processor is to free CPU from graphics work.
- Display processors have their own separate memory for fast operation.
- Main work of display processor is digitalizing a picture definition given into a set of pixel intensity values for

store in frame buffer.

- This digitalization process is scan conversion.
- Display processor also performs many other functions such as generating various line styles (dashed, dotted, or solid). Display color areas and performing some transformation for manipulating object.
- It also interfaces with interactive input devices such as mouse.
- For reduce memory requirements in raster scan system methods have been devised for organizing the frame buffer as a line list and encoding the intensity information.
- One way to do this is to store each scan line as a set of integer pair one number indicate number of adjacent pixels on the scan line that are having same intensity and second stores intensity value this technique is called run-length encoding.
- A similar approach is when pixel. Intensity is changes linearly, encoded the raster as a set of rectangular areas (cell encoding).
- Disadvantages of encoding is when run length is small it requires more memory then original frame buffer.
- It also difficult for display controller to process the raster when many sort runs are involved.

Output Primitives

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 end points 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 to other.
- 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 fig 2.1.

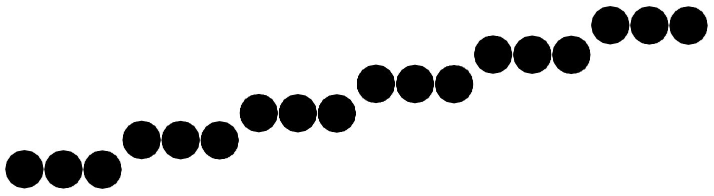


Fig. 2.1: - Stair step effect produced when line is generated as a series of pixel positions.

- 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 referenced according to scan-line number and column number which is illustrated by following figure.

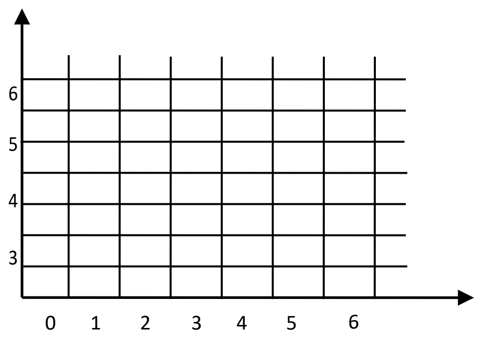


Fig. 2.2: - Pixel positions referenced by scan-line number and column number.

- 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 *setpixe(x, y)*.

or retrieve the current frame buffer intensity we assume to have procedure *getpixel(x, y)*.

Line Drawing Algorithms

- The Cartesian slop-intercept equation for a straight line is "y = mx + b" with 'm' representing slop and 'b' as the intercept.
- The two endpoints of the line are given which are say (x1, y1) and (x2, y2).

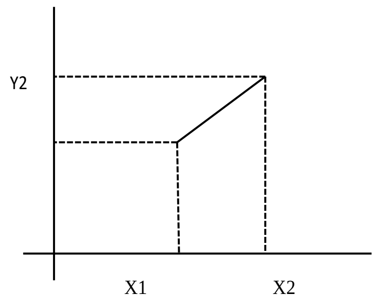


Fig. 2.3: - Line path between endpoint positions.

- We can determine values for the slope m by equation:

m= $(y2 - y1)/(x2 - x1)$

- We can determine values for the intercept b by equation:

$b = y1 - m * x1$

- For the given interval Δx along a line, we can compute the corresponding y interval Δy as:

$\Delta y = m * \Delta x$

- Similarly for Δx:

$\Delta x = \Delta y/m$

- For line with slop $|m| < 1$, Δ can be set proportional to small horizontal deflection voltage and the corresponding vertical deflection voltage is then set proportional to Δy which is calculated from above equation.
- For line with slop $|m| > 1$, Δy can be set proportional to small vertical deflection voltage and the corresponding horizontal deflection voltage is then set proportional to Δx which is calculated from above equation.
- For line with slop $m = 1, \Delta x = \Delta y$ and the horizontal and vertical deflection voltages are equal.

DDA Algorithm

- Digital differential analyzer (DDA) is scan conversion line drawing algorithm based on calculating either

Δy or Δx using above equation.

- We sample the line at unit intervals in one coordinate and find corresponding integer values nearest the line path for the other coordinate.
- Consider first a line with positive slope and slope is less than or equal to 1:

We sample at unit x interval ($\Delta x = 1$) and calculate each successive y value as follow:

$y = m * x + b$

$y_k = m * (x + 1) + b$

In general $y_k = m * (x + k) +$, &

$y_{k+1} = m * (x + k + 1) + b$

Now write this equation in form:

$y_{k+1} - y_k = (m * (x + k + 1) + b) - (m * (x + k) + b)$

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

So that it is computed fast in computer as addition is fast compare to multiplication.

- In above equation k takes integer values starting from 1 and increase by 1 until the final endpoint is reached.
- As m can be any real number between 0 and 1, the calculated y values must be rounded to the nearest integer.
- Consider a case for a line with a positive slope greater than 1:

We change the role of x and y that is sample at unit y intervals ($\Delta y = 1$) and calculate each succeeding x value as:

$x = (y - b)/m$

$x_1 = ((y + 1) - b)/m$

In general $x = ((y + k) - b)/m$, &

$x_{k+1} = ((y + k + 1) - b)/m$

Now write this equation in form:

$x_{k+1} - x_k = (((y + k + 1) - b)/m) - (((y + k) - b)/m)$

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

- Above both equations are based on the assumption that lines are to be processed from left endpoint to the right endpoint.
- If we processed line from right endpoint to left endpoint than: If $\Delta x = -1$ equation become:

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

If $\Delta y = -1$ equation become:

$x_{k+1} = x_k - 1/m$

- Above calculated equations also used to calculate pixel position along a line with negative slope.
- **Procedure for DDA line algorithm.**

```
Void lineDDA (int xa, int ya, int xb, int yb)
{
int dx = xb – xa, dy = yb – ya, steps, k;
float xincrement, yincrement, x = xa, y = ya; if
(abs(dx)>abs(dy))
{
            Steps = abs (dx);
}
else
{
            Steps = abs (dy);
}
xincrement = dx/(float) steps; yincrement =
dy/(float) steps;

setpixel (ROUND (x), ROUND (y));
for(k=0;k<steps;k++)
{
x += xincrement; y += yincrement;
setpixel (ROUND (x), ROUND (y));
}
}
```

Advantages of DDA algorithm

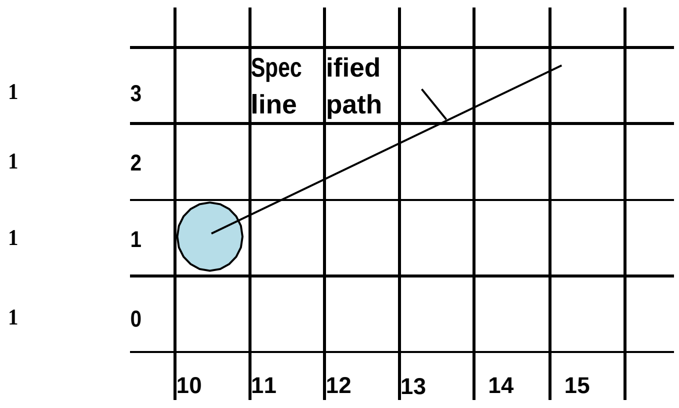
- It is faster algorithm.
- It is simple algorithm.

Disadvantage of DDA algorithm

- Floating point arithmetic is time consuming.
- Poor end point accuracy.

Bresenham’s Line Algorithm

- An accurate and efficient raster line-generating algorithm, developed by Bresenham which scan converts line using only incremental integer calculations that can be modified to display circles and other curves.
- Figure shows section of display screen where straight line segments are to be drawn.



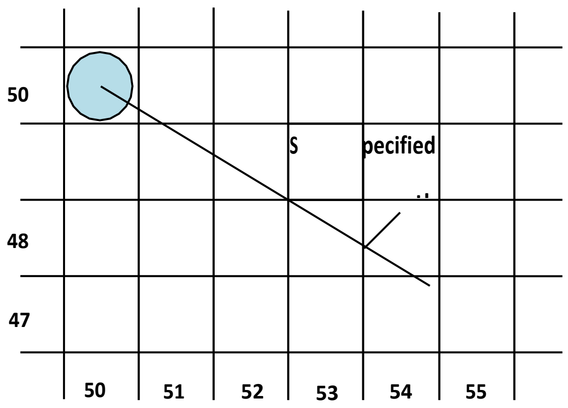


Fig. 2.4: - Section of a display screen where a straight line segment is to be plotted, starting from the pixel at column 10 on scan line 11.

Fig. 2.5: - Section of a display screen where a negative slope line segment is to be plotted, starting from the pixel at column 50 on scan line 50.

- The vertical axes show scan-line positions and the horizontal axes identify pixel column.
- Sampling at unit x intervals in these examples, we need to decide which of two possible pixel position is closer to the line path at each sample step.
- To illustrate bresenham’s approach, we first consider the scan-conversion process for lines with positive slope less than 1.
- Pixel positions along a line path are then determined by sampling at unit x intervals.
- Starting from left endpoint (x_0, y_0) of a given line, we step to each successive column and plot the pixel whose scan-line y values is closest to the line path.
- Assuming we have determined that the pixel at (x_k, y_k) is to be displayed, we next need to decide which pixel to plot in column $x_k + 1$.
- Our choices are the pixels at positions $(x_k + 1, y_k)$ and $(x_k + 1, y_k + 1)$.
- Let’s see mathematical calculation used to decide which pixel position is light up.
- We know that equation of line is:

$y = mx + b$

Now for position $x_k + 1$.

$= m(x_k + 1) + b$

- Now calculate distance bet actual line’s y value and lower pixel as d_1 and distance bet actual line’s y value and upper pixel as d_2.

$d_1 = y - y_k$

$d_1 = m(x_k + 1) + b - y_k$ ....(1)

$d_2 = (y_k + 1) - y$

$d_2 = (y_k + 1) - (x_k + 1) - b$...(2)

- Now calculate $d_1 - d_2$ from equation (1) and (2).

$d_1 - d_2 = (y - y_k) - ((y_k + 1) - y)$

$d_1 - d_2 = \{(x_k + 1) + b - y_k\} - \{(y_k + 1) - m(x_k + 1) - b\}$

$d_1 - d_2 = \{mx_k + m + b - y_k\} - \{y_k + 1 - mx_k - m - b\}$

$d_1 - d_2 = 2(x_k + 1) - 2y_k + 2b - 1$..(3)

- Now substitute $m = \Delta y/\Delta x$ in equation(3)

$d_1 - d_2 = 2\left(\frac{\Delta y}{\Delta x}\right)(x_k + 1) - 2y_k + 2b - 1$ ....(4)

- Now we have decision parameter p_k for k^{th} step in the line algorithm is given by:

$p_k = \Delta(d_1 - d_2)$

$p_k = \Delta(2\Delta y/\Delta x(x_k + 1) - 2y_k + 2b - 1)$

$p_k = 2\Delta y x_k + 2\Delta y - 2\Delta x y_k + 2\Delta x b - \Delta x$

$p_k = 2\Delta y x_k - 2\Delta x y_k + 2\Delta y + 2\Delta x b - \Delta x$ ....(5)

$p_k = 2\Delta y x_k - 2\Delta x y_k + C$ ($Where\ Constant\ C = 2\Delta y + 2\Delta x b - \Delta x$) .... (6)

- The sign of p_k is the same as the sign of $d_1 - d_2$, since $\Delta x > 0$ for our example.
- Parameter c is constant which is independent of pixel position and will eliminate in the recursive calculation for p_k.
- Now if p_k is negative then we plot the lower pixel otherwise we plot the upper pixel.
- So successive decision parameters using incremental integer calculation as:

$p_{k+1} = 2\Delta y x_{k+1} - 2\Delta x y_{k+1} + \mathrm{C}$

- Now Subtract pk from p_{k+1}

$p_{k+1} - p_k = 2\Delta(x_{k+1} - x_k)$ -$2\Delta x(y_{k+1} - y_k)$

$p_{k+1} - p_k = 2\Delta y x_{k+1} - 2\Delta x y_{k+1} + \mathrm{C} - 2\Delta y x_k + 2\Delta x y_k - \mathrm{C}$

But $x_{k+1} = x_k + 1$, so that $(x_{k+1} - x_k) = 1$

$p_{k+1} = p_k + 2\Delta y - 2\Delta(y_{k+1} - y_k)$

- Where the terms $y_{k+1} - y_k$ is either 0 or 1, depends on the sign of parameter p_k.
- This recursive calculation of decision parameters is performed at each integer x position starting at the left coordinate endpoint of the line.
- The first decision parameter p_0 is calculated using equation (5) as first time we need to take constant part into account so:

$p_k = 2\Delta y x_k - 2\Delta x y_k + 2\Delta y + 2\Delta x b - \Delta x$

$p_0 = 2\Delta y x_0 - 2\Delta x y_0 + 2\Delta y + 2\Delta x b - \Delta x$

Now $Substitute\ b = y_0 - mx_0$

$p_0 = 2\Delta y x_0 - 2\Delta x y_0 + 2\Delta y + 2\Delta(y_0 - mx_0) - \Delta x$

$Now\ Substitute\ m = \Delta y/\Delta x$

$p_0 = 2\Delta y x_0 - 2\Delta x y_0 + 2\Delta y + 2\Delta(y_0 - (\Delta y/\Delta x)x_0) - \Delta x$

$p_0 = 2\Delta y x_0 - 2\Delta x y_0 + 2\Delta y + 2\Delta x y_0 - 2\Delta y x_0 - \Delta x$

$p_0 = 2\Delta y - \Delta x$

- Let's see Bresenham's line drawing algorithm for $|m| < 1$
  1. Input the two line endpoints and store the left endpoint in (x_0, y_0).
  2. Load (x_0, y_0) into the frame buffer; that is, plot the first point.
  3. Calculate constants $\Delta x, \Delta y, 2\Delta y$, and $2\Delta y - 2\Delta x$, and obtain the starting value for the decision parameter as

$p_0 = 2\Delta y - \Delta x$

4. At each x_k along the line, starting at $k = 0$, perform the following test: If $p_k < 0$, the next point to plot is $(x_k + 1, y_k)$ and

$p_{k+1} = p_k + 2\Delta y$

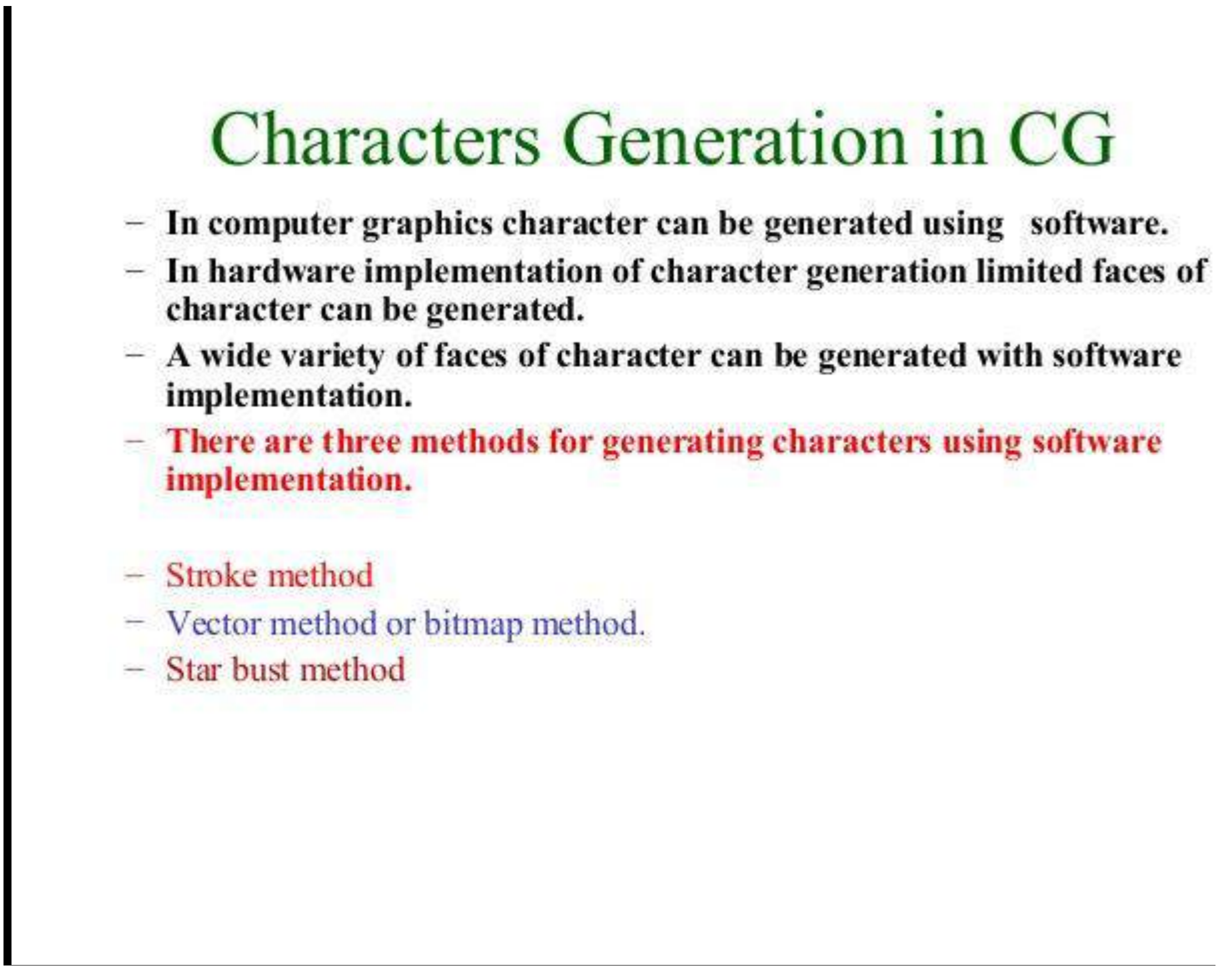
Otherwise, the next point to plot is $(x_k + 1, y_k + 1)$ and

$p_{k+1} = p_k + 2\Delta y - 2\Delta x$

5. Repeat step-4 Δx times.

- Bresenham's algorithm is generalized to lines with arbitrary slope by considering symmetry between the various octants and quadrants of the xy plane.
- For lines with positive slope greater than 1 we interchange the roles of the x and y directions.
- Also we can revise algorithm to draw line from right endpoint to left endpoint, both x and y decrease as we step from right to left.
- When $d_1 - d_2 = 0$ we choose either lower or upper pixel but once we choose lower than for all such case for that line choose lower and if we choose upper the for all such case choose upper.
- For the negative slope the procedure are similar except that now one coordinate decreases as the other increases.

The special case handle separately. Horizontal line ($\Delta y = 0$), vertical line ($\Delta x = 0$) and diagonal line with $|\Delta x| = |\Delta y|$ each can be loaded directly into the frame buffer without processing them through the line plotting algorithm.



Characters Generation in CG

- In computer graphics character can be generated using software.
- In hardware implementation of character generation limited faces of character can be generated.
- A wide variety of faces of character can be generated with software implementation.
- There are three methods for generating characters using software implementation.

- Stroke method
- Vector method or bitmap method.
- Star bust method

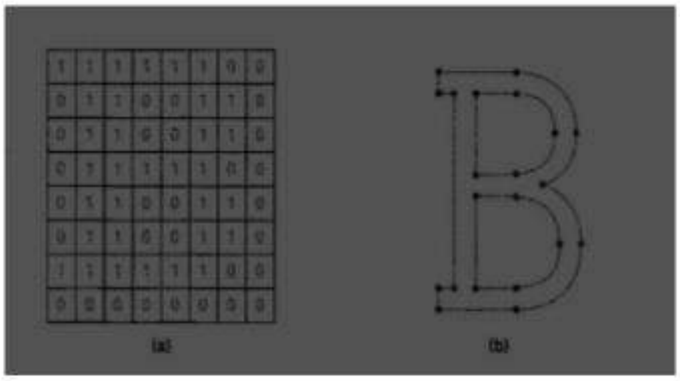
Stoke method

- In this method we use a sequence of line drawing function and arc functions to generate characters.
- We can generate a sequence of character by assigning starting and end point of line or arc.
- By using this method various faces of character can be generated by changing the values (parameters) in line and arc function.

Stoke method

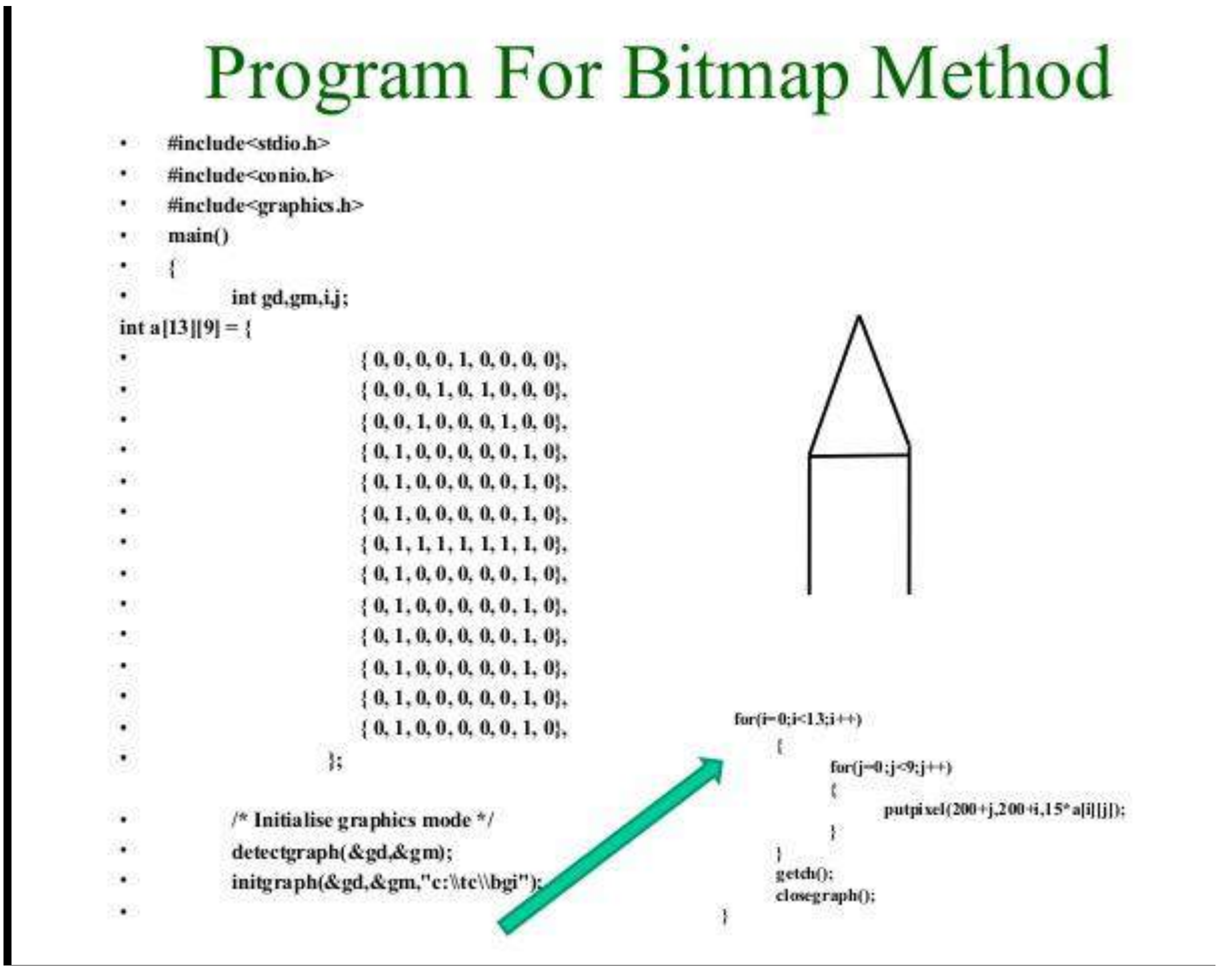
- The main disadvantage of this method is when we draw a diagonal line it produce aliased character.

Bitmap Method



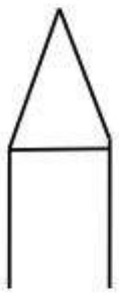
Method

- This method is suitable for producing various character.
- Font size of character can be increased by increasing the size of array.
- This method uses an array of 1’s & 0’s to represent pixels.



Program For Bitmap Method

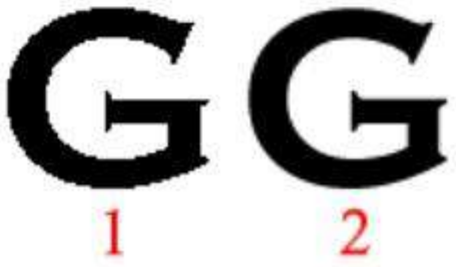
```
•    #include<stdio.h>
•    #include<conio.h>
•    #include<graphics.h>
•    main()
•    {
•         int gd,gm,i,j;
int a[13][9] = {
•                   { 0, 0, 0, 0, 1, 0, 0, 0, 0},
•                   { 0, 0, 0, 1, 0, 1, 0, 0, 0},
•                   { 0, 0, 1, 0, 0, 0, 1, 0, 0},
•                   { 0, 1, 0, 0, 0, 0, 0, 1, 0},
•                   { 0, 1, 0, 0, 0, 0, 0, 1, 0},
•                   { 0, 1, 0, 0, 0, 0, 0, 1, 0},
•                   { 0, 1, 1, 1, 1, 1, 1, 1, 0},
•                   { 0, 1, 0, 0, 0, 0, 0, 1, 0},
•                   { 0, 1, 0, 0, 0, 0, 0, 1, 0},
•                   { 0, 1, 0, 0, 0, 0, 0, 1, 0},
•                   { 0, 1, 0, 0, 0, 0, 0, 1, 0},
•                   { 0, 1, 0, 0, 0, 0, 0, 1, 0},
•                   { 0, 1, 0, 0, 0, 0, 0, 1, 0},
•                 };

•         /* Initialise graphics mode */
•         detectgraph(&gd,&gm);
•         initgraph(&gd,&gm,"c:\\tc\\bgi");
•
```



```
for(i=0;i<13;i++)
    {
        for(j=0;j<9;j++)
        {
            putpixel(200+j,200+i,15*a[i][j]);
        }
    }
    getch();
    closegraph();
}
```

Bitmap Method

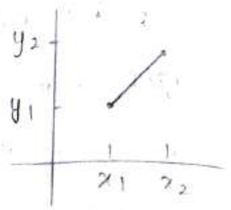
- This method is suitable for producing various character.
- Font size of character can be increased by increasing the size of array.
- The main draw back of this method is this method produce aliased character.



Line Drawing Algo.

The Cartesian slope - intercept eqn for a st. line is

$$y = mx + b \quad — ①$$



line path bet. end pt. (x_1, y_1) (x_2, y_2)

where, m - slope of line

b - y intercept

(x_1, y_1) (x_2, y_2) - 2 endpt. of line segment at positions

We can determine val. for m & b with:

$$m = \frac{y_2 - y_1}{x_2 - x_1} \quad — ②$$

$$b = y_1 - m x_1 \quad — ③$$

Algo. for displaying st. lines ~~&~~ based on line eqn. ① & cal. given in ② & ③

For any given x interval Δx along a line, we can compute the corresponding y interval Δy from ②

$$\Delta y = m \Delta x \quad — ④$$

$\parallel^{ly}$, we can obtain Δx corresponding to a specified Δy as

$$\Delta x = \frac{\Delta y}{m} \quad — ⑤$$

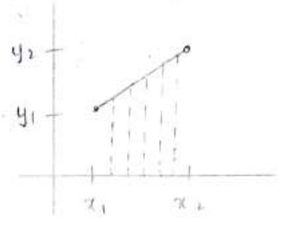
* These eqns. form the basis for determining deflection voltages in analog devices.

* For lines with slope magnitudes $|m| < 1$, Δx can be set $\propto$ to a small horizontal deflection voltage & the corresponding vertical deflection is then set $\propto$ to Δy as calculated from ④.

* For lines whose slopes have magnitudes $|m| > 1$, Δy can be set $\propto$ to a small vertical def. voltage with corresponding horizontal def. voltage set $\propto$ to Δx, calculated from ⑤.

* For lines with $m = 1$, $\Delta x = \Delta y$ & -al & |al def. voltages are equal, In each case, a smooth line with slope m is generated bet. the specified endpts.

* On raster sys., lines r plotted with pixels, & step sizes in -al & |al directions are constrained by pixel separations. That is, we must "sample" a line at discrete positions & determine the nearest pixel to the line at each sampled position. This scan-conversion process for st. lines is illustrated in Fig., for a near -al line with discrete ~~po~~ sample positions along x axis.



St line segment with 5 sampling positions along the x axis bet. x_1 & x_2

Digital differential analyzer Algo. (DDA Algo.)

The DDA is a scan-conversion line algo. based on calculating either Δy / Δx, using ④ / ⑤. We sample the line at unit intervals in 1 coordinate & determine corresponding integer val. nearest the line path for other coordinate.

Consider 1st a line with +ve slope,

If $m \leq 1$, we sample at unit x intervals ($\Delta x = 1$) & compute each successive y val. as:

$$y_{k+1} = y_k + m \quad \text{— ⑥}$$

Subscript k takes integer val. starting from 1, for 1st pt. & increases by 1 until the final endpt. is reached. ∵ m can be any real no. bet. 0 & 1, the calculated y val. must be rounded to nearest int.

For lines with a +ve slope > 1, we reverse the roles of x & y. i.e, we sample at unit y intervals ($\Delta y = 1$) & calculate each succeeding x val. as

$$x_{k+1} = x_k + \frac{1}{m} \quad — (7)$$

Eqn. (6) & (7) are based on assumption that lines r to be processed from left endpt. to right endpt. If this processing is reversed, so that the starting endpt. is at right, then either we have $\Delta x = -1$ &

$$y_{k+1} = y_k - m \quad — (8)$$

or (when $m > 1$) we have $\Delta y = -1$ with

$$x_{k+1} = x_k - \frac{1}{m} \quad — (9)$$

Eqns. (6) thro' (9) can also be used to calculate pixel positions along a line with -ve slope.

If absolute val. of $m < 1$ & start endpt. is at left, we set $\Delta x = 1$ & cal. y val. with eqn. (6).

When start endpt. is at the right (for same slope), we set $\Delta x = -1$ & obtain y positions from eqn. (8).

III^ly^, when absolute val. of a -ve slope is > 1, we use $\Delta y = -1$ & eqn. (9) / we used $\Delta y = 1$ & Eqn. (7).

Algorithm:

```
#define ROUND (a) ((int)(a+0.5))
voidline DDA (int xa, int ya, int xb, int yb)
{
int dx = xb-xa, dy = yb-ya, steps, k;
float xIncrement, yIncrement, x = xa, y=ya;
if (abs(dx) > abs(dy)) steps = abs (dx);
else steps = abs dy ;
xIncrement = dx / (float) steps;
yIncrement = dy / (float) steps;
setpixel (ROUND(x), ROUND(y));
for (k=0 ; k < steps ; k++)
{
x+ = xIncrement ;
y+ = yIncrement ;
setpixel (ROUND(x), ROUND(y));
}
}
```

Algo. Description:

Step 1: Accept I/P as 2 endpt. pixel positions.

Step 2: -al & lal differences bet. the endpt. positions are assigned to parameters dx & dy (cal. dx = xb - xa & dy = yb - ya).

Step 3: The difference with the greater mag. determines the val. of parameter steps.

Step 4: Starting with pixel position (xa, ya), determine the offset needed at each step to generate the next pixel position along line path.

Step 5: Loop the following process for steps no. of times

a) Use a unit of inc. / dec. in the x & y direction.

b) If xa < xb, the val. of inc. in x & y directions are 1 & m.

c) If xa > xb, then the dec. -1 & -m are used.

Ad. of DDA algo.

→ simplest algo.

→ faster method for calculating pixel positions.

Disad.

→ Floating pt. arith. in DDA algo is still time-consuming

→ End pt. accuracy is poor.

Eg: Consider the line from (0, 0) to (4, 6)

1. $xa = 0$, $ya = 0$ &
   $xb = 4$, $yb = 6$

2. $dx = xb - xa$ & $dy = yb - ya$
   $= 4 - 0$ $= 6 - 0$
   $= 6$ $= 6$

3. $x = 0$ & $y = 0$

4. $4 > 6$ (false) so , steps = 6

5. Cal. $x\text{Increment} = \frac{dx}{\text{steps}} = \frac{4}{6} = 0.66$ &

   $y\text{Increment} = \frac{dy}{\text{steps}} = \frac{6}{6} = 1$

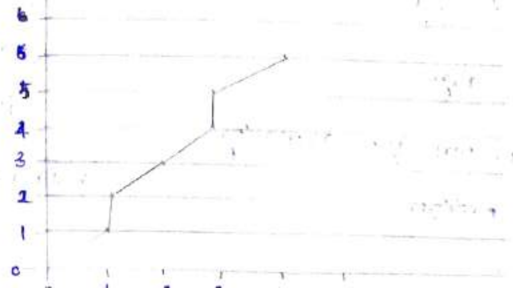
6. Setpixel (x, y) = setpixel (0, 0) (starting pixel position)

7. Iterate the calculation for x Increment & y Increment for steps(6) no. of times.

8. Tabulate each iteration

K	x	y	Plotting pts. (Rounded to Integer)
0	0+0.66 = 0.66	0 + 1 = 1	(1, 1)
1	0.66 + 0.66 = 1.32	1 + 1 = 2	(1, 2)
2	1.32 + 0.66 = 1.98	2 + 1 = 3	(2, 3)
3.	1.98 + 0.66 = 2.64	3 + 1 = 4	(3, 4)
4	2.64 + 0.66 = 3.3	4 + 1 = 5	(3, 5)

Result:



Consider the line from (4, 6) & (0, 0)

1. $xa = 4$, $ya = 6$ &
   $xb = 0$, $yb = 0$

2. $dx = xb - xa$ & $dy = yb - ya$
   $= 0 - 4$ $= 0 - 6$
   $= -4$ $= -6$

3. $x = 4$ & $y = 6$

4. $4 > 6$ (flase) so, steps $= 6$

5. Cal. xIncrement $= \frac{dx}{steps} = \frac{-4}{6} = -0.66$ &

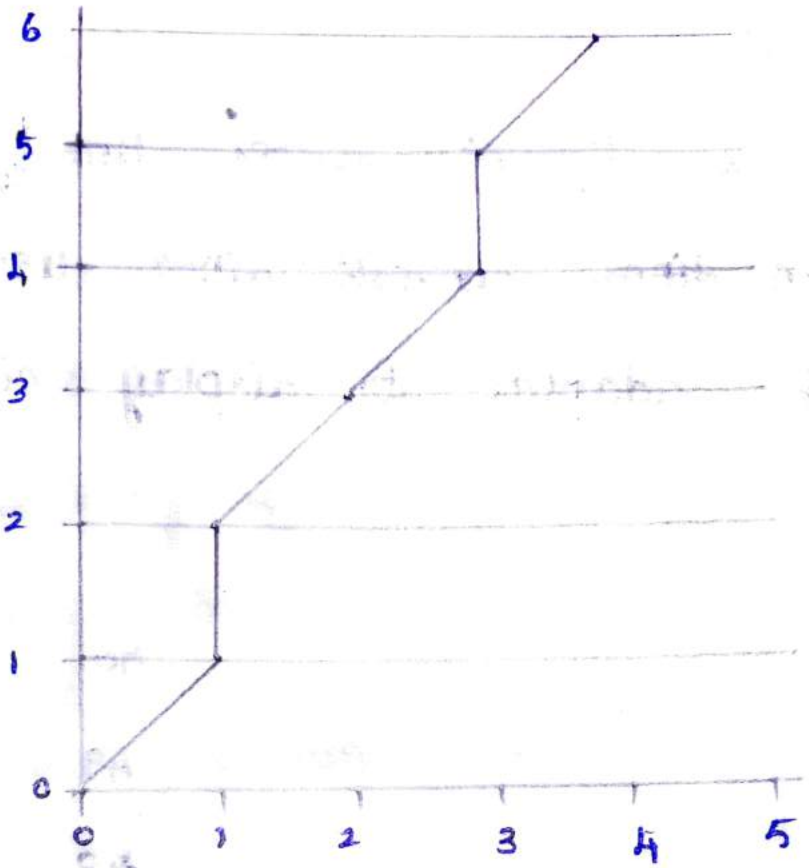
   yIncrement $= \frac{dy}{steps} = \frac{-6}{6} = -1$

6. Setpixel (x, y) = setpixel $(4, 6)$: Starting pixel position.

7. Iterat the cal. for xIncrement & yIncrement for steps (6) no. of times. [x = x + xIncrement
   y = y + yIncrement]

8. Tabulat each iteration

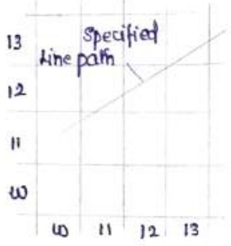
k	x	y	Plotting pts.
0	4 - 0.66 = 3.34	6 - 1 = 5	(3, 5)
1	3.34 - 0.66 = 2.68	5 - 1 = 4	(3, 4)
2	2.68 - 0.66 = 2.02	4 - 1 = 3	(2, 3)
3	2.02 - 0.66 = 1.36	3 - 1 = 2	(1, 2)
4	1.36 - 0.66 = 0.7	2 - 1 = 1	(1, 1)
5	0.7 - 0.66 = 0.04	1 - 1 = 0	(0, 0)



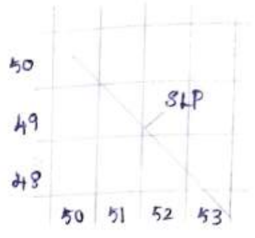
6
5
4
3
2
1
0
0
1
2
3
4
5

03-09-20 Bresenham's Line Algo.

An accurate & efficient raster line-generating algo., developed by B, scan ~~lines~~ converts lines using only ++al int calculations that can be adapted to display circles & other curves.



Section of a display screen where a st. line segment is to be plotted, starting from the pixel at col. 10 on scan line 11



Sec. of a disp. screen where a -ve slope line segment is to be plotted, starting from pixel at col. 50 on scan line 50.

* These fig. illustrate sections of a disp. screen where st. line segments are to be drawn.

* The vertical axes show scan-line positions, & horizontal axes identify pixel col.

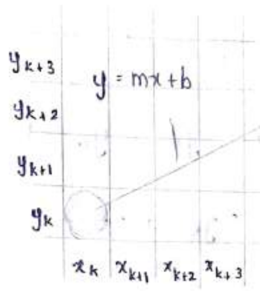
* Sampling at unit x intervals in these eg., we need to decide which of 2 possible pixel positions is closer to the line path at each sample step.

* Starting from left endpt. in Fig., we need to determine at next sample position whether to plot the pixel at position (11, 11) / the one at (11, 12).

* $\|^{ly}$, Fig. shows a -ve slope line path starting from left endpt. at pixel position (50, 50). In this one, do we select next pixel position as (51, 50) / as (51, 49)

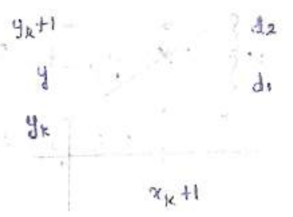
* These questions are answered with Bresenham's line algo. by testing the sign of an integer parameter, whose val. is proportional to difference bet. the separations of 2 pixel positions from actual line path.

* To illustrate B's approach, we 1st consider the scan-conversion process for lines with +ve slope <1. Pixel positions along a line path are then determined by sampling at unit x intervals. Starting from left end pt. (x_0, y_0) of a given line, we step to each successive col. (x position) & plot the pixel whose scan-line y val. is closest to line path.



Section of screen grid showing a pixel in col. x_k on scan line y_k i.e. to be plotted along the path of a line seg. with slope $0 < m < 1$

The above fig. demonstrates kth step in this process. Assuming we have determined that the pixel at (x_k, y_k) is to be displayed, we next need to decide which pixel to plot in col. x_{k+1}. Our choices are the pixels at positions (x_k+1, y_k) & (x_k+1, y_k+1).



Distances bet. pixel positions & the line y coordinate at sampling position x_k+1.

At sampling position x_k+1, we label vertical pixel separations from the mathematical line path as d_1 & d_2. The y coordinate on mathematical line at pixel position x_k+1 is calculated as

$$y = m(x_k+1) + b \quad — ①$$

Then

$$d_1 = y - y_k \qquad \& \qquad d_2 = (y_k + 1) - y$$

$$= m(x_k + 1) + b - y_k \qquad = y_k + 1 - m(x_k+1) - b$$

The difference bet. these 2 separations is

$$d_1 - d_2 = m(x_k+1) + b - y_k - (y_k + 1 - m(x_k+1) - b)$$

$$= m(x_k+1) + b - y_k - y_k - 1 + m(x_k+1) + b$$

$$= 2m(x_k+1) - 2y_k + 2b - 1 \quad — ②$$

A decision parameter p_k for the k th step in the line algo. can be obtained by rearranging Eqn. ② so that it involves only integer calculations. We accomplish this by substituting $m = \Delta y / \Delta x$, where ~~Δy are the~~ Δx & Δy are the -al & lal separations of the end pt. positions.

$$p_k = \Delta x (d_1 - d_2)$$

$$= \Delta x [2m(x_k+1) - 2y_k + 2b - 1]$$

$$= \Delta x . 2m(x_k+1) - 2\Delta x y_k + 2\Delta x b - \Delta x$$

$$= \Delta x . \frac{2\Delta y}{\Delta x}(x_k+1) - 2\Delta x y_k + 2\Delta x b - \Delta x$$

$$= 2\Delta y(x_k+1) - 2\Delta x . y_k + 2\Delta x . b - \Delta x$$

$$= 2\Delta y\, x_k + 2\Delta y - 2\Delta x\, y_k + 2\Delta x . b - \Delta x$$

$$= 2\Delta y . x_k - 2\Delta x . y_k + \underbrace{2\Delta y + 2\Delta x b - \Delta x}$$

$$\boxed{p_k = 2\Delta y . x_k - 2\Delta x . y_k + c} \quad — ③$$

Successive decision parameter

$$P_{k+1} = 2\Delta y . x_{k+1} - 2\Delta x . y_{k+1} + c$$

Subtracting $P_{k+1} - P_k$

17-09-20

$$P_{k+1} - P_k = 2\Delta y . x_{k+1} - 2\Delta x\, y_{k+1} + c - 2\Delta y . x_k + 2\Delta x . y_k - c$$

$$= \underbrace{2\Delta y . x_{k+1} - 2\Delta y . x_k}\ \underbrace{- 2\Delta x . y_{k+1} + 2\Delta x\, y_k}$$

$$P_{k+1} - P_k = 2\Delta y\, (x_{k+1} - x_k) - 2\Delta x\, (y_{k+1} - y_k)$$

$$P_{k+1} = P_k + 2\Delta y\, (x_{k+1} - x_k) - 2\Delta x\, (y_{k+1} - y_k)$$

$(\because x_{k+1} = x_k + 1)$

$$P_{k+1} = P_k + 2\Delta y\, (x_k + 1 - x_k) - 2\Delta x\, (y_{k+1} - y_k)$$

$$\boxed{P_{k+1} = P_k + 2\Delta y - 2\Delta x\, (y_{k+1} - y_k)} \quad — (4)$$

Initial DP

$$\boxed{P_0 = 2\Delta y - \Delta x}$$

B's Line Drawing algo. for $|m| < 1$

1. Input the 2 line endpts. & store the left endpt. in (x_0, y_0).
2. Load (x_0, y_0) into FB, i.e., plot the 1st pt.
3. Calculate const. Δx, Δy, $2\Delta y$ & $2\Delta y - 2\Delta x$ & obtain the starting val. for DP as

$$P_0 = 2\Delta y - \Delta x$$

4. At each x_k along the line, starting at $k=0$, perform the following test:

If $P_k < 0$, the next pt. to plot is (x_k+1, y_k) &

$$P_{k+1} = P_k + 2\Delta y$$

Otherwise, the next pt. to plot is (x_k+1, y_k+1) &

$$P_{k+1} = P_k + 2\Delta y - 2\Delta x$$

5. Repeat step 4 Δx times.

Eg: B Line drawing

End pts. $(20, 10)$ & $(30, 18)$.

Slope $= 0.8$

$\Delta x = x_2 - x_1$ $\quad\quad$ $\Delta y = y_2 - y_1$

$= 30 - 20$ $\quad\quad$ $= 18 - 10$

$= 10$ $\quad\quad$ $= 8$

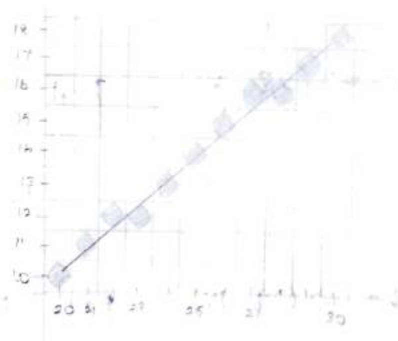
Initial decision parameter

$$P_0 = 2\Delta y - \Delta x$$
$$= 2(8) - 10$$
$$= 6$$

k	$P_k < 0$	p_{k+1}
0	$P_0 < 0 \Rightarrow$ False $\Rightarrow (21, 11)$	$P_{0+1} = 6 + 2(8) - 2(10)$ $= 6 + 16 - 20$ $= 6 - 4$ $P_1 = 2$
1	$P_1 < 0 \Rightarrow$ False $\Rightarrow (22, 12)$	$P_{1+1} = P_1 + 2(8) - 2(10)$ $= 2 + 16 - 20$ $= 2 - 4$ $P_2 = -2$

k	$P_k < 0$	P_{k+1}
2	$P_2 < 0 \Rightarrow$ True $\Rightarrow (23, 12)$	$P_{2+1} = P_2 + 2(8)$ $= -2 + 16$ $P_3 = 14$
3	$P_3 < 0 \Rightarrow$ False $\Rightarrow (24, 13)$	$P_3 + 1 = P_3 + 2(8) - 2(10)$ $= 14 + 16 - 20$ $= 14 - 4$ $P_4 = 10$
4	$P_4 < 0 \Rightarrow$ False $\Rightarrow (25, 14)$	$P_{4+1} = P_4 - 4$ $= 10 - 4$ $P_5 = 6$
5	$P_5 < 0 \Rightarrow$ False $\Rightarrow (26, 15)$	$P_{5+1} = P_5 - 4$ $= 6 - 4$ $P_6 = 2$
6	$P_6 < 0 \Rightarrow$ False $\Rightarrow (27, 16)$	$P_6 + 1 = P_6 - 4$ $= 2 - 4$ $P_7 = -2$
7	$P_7 < 0 \Rightarrow$ True $\Rightarrow (28, 16)$	$P_{7+1} = P_7 + 16$ $= -2 + 16$ $P_8 = 14$
8	$P_8 < 0 \Rightarrow$ False $\Rightarrow (29, 17)$	$P_{8+1} = P_8 ~~+~~ -4$ $= 14 - 4$ $P_9 = 10$
9	$P_9 < 0 \Rightarrow$ False $\Rightarrow (30, 18)$	

k	p_k	(x_{k+1}, y_{k+1})
0	6	(21, 11)
1	2	(22, 12)
2	-2	(23, 12)
3	14	(24, 13)
4	10	(25, 14)
5	6	(26, 15)
6	2	(27, 16)
7	-2	(28, 16)
8	14	(29, 17)
9	10	(30, 18)



Pixel positions along the line path bet. endpts. (20, 10) & (30, 18) plotted with B's line algo.

An implementation of B line drawing for slopes in range $0 < m < 1$ is given in the following procedure. Endpt. pixel positions for the line are passed to this procedure, & pixels are plotted from left endpt. to right endpt. The call to 'setpixel' loads to preset color val. into the FB at the specified (x, y) pixel position.

```
#include "device.h"
void lineBres (int xa, int ya, int xb, int yb)
{
          Δx                      Δy
   int dx = abs (xa - xb) ,  dy = abs( ya - yb);
         Po - initial decision para
   int p = 2 * dy - dx ;
   int twoDy = 2 * dy , twoDyDx = 2 * (dy - dx);
   int x, y, xEnd ;
// Determine which pt. to use as start, which as end
     if (xa > xb)
     {
        x = xb ;      } starting pt.
        y = yb ;
        xEnd = xa ;   - end pt.
     }
     else
     {
        x = xa ;
        y = ya ;
        xEnd = xb;
     }
     setPixel (x, y) ;
     while (x < xEnd)
     {
        x++ ;
        if (p < 0)
              p += twoDy ;
```

```
    else
    {
      y++;
      ~~p+p~~ p += twoDyDx;
    }
   setPixel (x, y);
  }
}
```

08-10-2020 Midpt. Circle Algo. / B's Circle Algo.

To apply midpt. method, we define a circle definition:

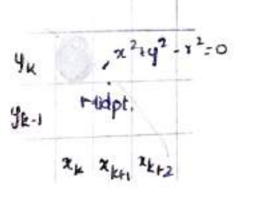
$$f_{circle}(x,y) = x^2 + y^2 - r^2 \qquad — ①$$

* Any pt. (x, y) on the boundary of the circle with radius r satisfies the eqn. $f_{circle}(x, y) = 0$.
* If the pt. is in the interior of circle, the circle fun. is −ve.
* " " " " outside the circle, " " " " +ve.
* To summarize, the relative position of any point (x, y) can be determined by checking the sign of circle function.

$$f_{circle}(x,y) \begin{cases} <0, & \text{if } (x,y) \text{ is inside the circle boundary.} \\ =0, & \text{if } (x,y) \text{ is on the circle boundary.} \\ >0, & \text{if } (x,y) \text{ is outside the circle boundary.} \end{cases} \qquad — ②$$

The circle fun. tests in ② are performed for the midpositions between pixels near the circle path at each sampling step. Thus the circle func. is the decision parameter in the midpt. algo.

Fig. shows the midpt. bet. the 2 candidate pixels at sampling position x_k+1. Assuming we have just plotted the pixel at (x_k, y_k), we next need to determine whether the pixel at position (x_k+1, y_k) or the one at position (x_k+1, y_k-1) is closer to the circle. Our dp is the circle function ① evaluated at the midpt. bet. these 2 pixels:



Midpt. bet. candidate pixels at sampling position x_k+1 along a circular path

$$P_k = f_{circle}\left(x_k+1,\ y_k-\tfrac{1}{2}\right)$$

$$= (x_k+1)^2 + \left(y_k-\tfrac{1}{2}\right)^2 - r^2$$

$$= (x_k)^2 + 2x_k + 1 + (y_k)^2 - 2y_k\left(\tfrac{1}{2}\right) + \left(\tfrac{1}{2}\right)^2 - r^2$$

$$P_k = (x_k)^2 + 2x_k + 1 + (y_k)^2 - y_k + \tfrac{1}{4} - r^2$$

If $P_k < 0$, this midpt. is inside the circle & the pixel on scan line y_k is closer to the circle boundary. Otherwise, the midpt. is outside or on the circle boundary, and we select the pixel on scanline $y_k - 1$.

Successive dp are obtained using incremental calculations. We obtain a recursive expression for the next dp by evaluating the circle func. at sampling position $x_{k+1} + 1 = x_k + 2$

$$P_{k+1} = f_{circle}\left(x_{k+1} + 1,\ y_{k+1} - \tfrac{1}{2}\right)$$

$$= \left[(x_k + 1) + 1\right]^2 + \left(y_{k+1} - \tfrac{1}{2}\right)^2 - r^2$$

$$= (x_k + 1)^2 + 2(x_k + 1) + 1 + (y_{k+1})^2 - 2y_{k+1}\left(\tfrac{1}{2}\right) + \left(\tfrac{1}{2}\right)^2 - r^2$$

$$P_{k+1} = (x_k)^2 + 2x_k + 1 + 2(x_k + 1) + 1 + (y_{k+1})^2 - y_{k+1} + \tfrac{1}{4} - r^2$$

$$P_{k+1} - P_k = (x_k)^2 + 2x_k + 1 + 2(x_k + 1) + 1 + (y_{k+1})^2 - y_{k+1} + \tfrac{1}{4} - r^2$$
$$- (x_k)^2 - 2x_k - 1 - (y_k)^2 + y_k - \tfrac{1}{4} + r^2$$

$$P_{k+1} - P_k = 2(x_k + 1) + 1 + (y_{k+1})^2 - y_{k+1} - (y_k)^2 + y_k$$

$$P_{k+1} - P_k = 2(x_k + 1) + \left[(y_{k+1})^2 - (y_k)^2\right] - (y_{k+1} - y_k) + 1$$

$$P_{k+1} = P_k + 2(x_k + 1) + \left[(y_{k+1})^2 - (y_k)^2\right] - (y_{k+1} - y_k) + 1 \quad - ③$$

where y_{k+1} is either y_k or y_{k-1}, depending on the sign of P_k.

Evaluation of terms $2x_{k+1}$ & $2y_{k+1}$ can also be done ++ly as:

$$2x_{k+1} = 2x_k + 2$$

$$2y_{k+1} = 2y_k - 2$$

The initial dp is obtained by evaluating circle func. at start position $(x_0, y_0) = (0, r)$

$$P_0 = f_{circle}\left(1, r - \frac{1}{2}\right)$$

$$= 1 + \left(r - \frac{1}{2}\right)^2 - r^2$$

$$= 1 + \cancel{r^2} - 2r\left(\frac{1}{2}\right) + \left(\frac{1}{2}\right)^2 - \cancel{r^2}$$

$$= 1 - r + \frac{1}{4}$$

$$P_0 = \frac{5}{4} - r$$

If the radius r is specified as an integer, we can simply round P_0 to

$$P_0 = 1 - r$$

Midpoint Circle Algo.

1. Input radius r & circle center (x_c, y_c) & obtain the first pt. on ~~circen~~ circumference of a circle centered on the origin as

$$(x_0, y_0) = (0, r)$$

2. Cal. the initial val. of dp as

$$P_0 = \frac{5}{4} - r$$

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

If $P_k < 0$, the next pt. along the circle centered on $(0,0)$ is (x_{k+1}, y_k) and

$$P_{k+1} = P_k + 2x_{k+1} + 1$$

Otherwise, the next pt. along the circle is $(x_k + 1, y_k - 1)$ &

$$P_{k+1} = P_k + 2x_{k+1} + 1 - 2y_{k+1}$$

where $2x_{k+1} = 2x_k + 2$ & $2y_{k+1} = 2y_k - 2$.

4. Determine symmetry pt. in the other 7 octant.

5. Move each calculated pixel position (x, y) onto the circular path centered on (x_c, y_c) & plot the coordinate val.

$$x = x + x_c \quad , \quad y = y + y_c$$

6. Repeat steps 3 thro' 5 until $x \geq y$.

Eg:

Given a circle radius $r = 10$, we demonstrate the midpoint circle algo. by determining positions along the circle octant in the 1st quadrant from $x = 0$ to $x = y$. The initial val. of the decision parameter is

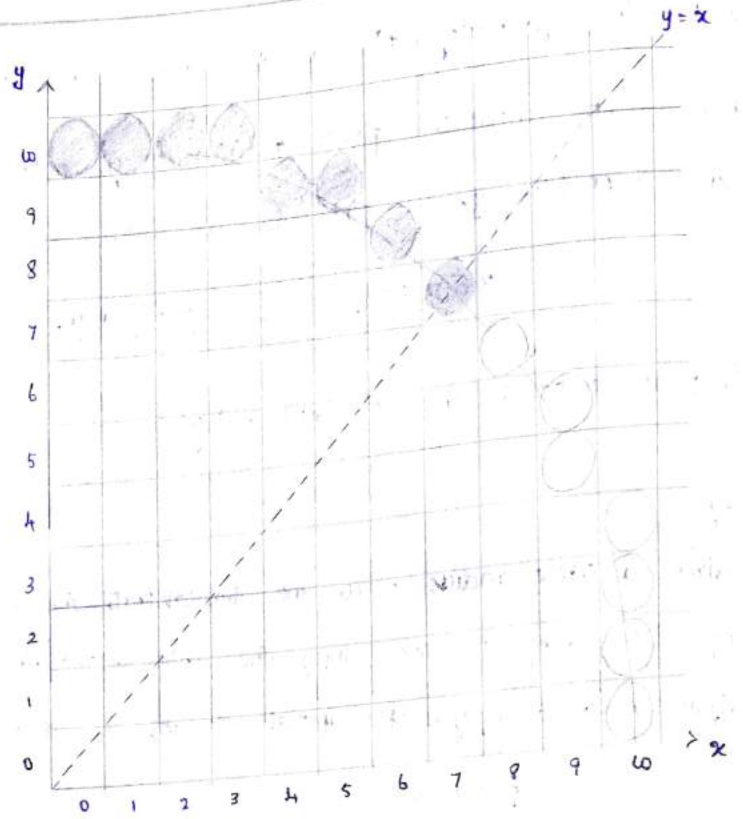
$$P_0 = 1 - r$$
$$= -9$$

$(x_0, y_0) = (0, 10)$

$2x_0 = 0 \quad 2y_0 = 20$

Successive dp val. and positions along the circle path are calculated using the midpt. method as

k	P_k	(x_{k+1}, y_{k+1})	$2x_{k+1}$	$2y_{k+1}$	$P_{k+1} = \begin{cases} P_k < 0 \rightarrow P_k + 2x_{k+1} + 1 \\ P_k > 0 \rightarrow P_k + 2x_{k+1} + 1 - 2y_{k+1} \end{cases}$
0	-9	(1, 10)	2	20	$P_{0+1} = -9 + 2 + 1$ $P_1 = -6$
1	-6	(2, 10)	4	20	$P_{1+1} = -6 + 4 + 1$ $P_2 = -1$
2	-1	(3, 10)	6	20	$P_{2+1} = -1 + 6 + 1$ $P_3 = 6$
3	6	(4, 9)	8	18	$P_{3+1} = 6 + 8 + 1 - 18$ $P_4 = -3$
4	-3	(5, 9)	10	18	$P_{4+1} = -3 + 10 + 1$ $P_5 = 8$
5	8	(6, 8)	12	16	$P_{5+1} = 8 + 12 + 1 - 16$ $P_6 = 5$
6	5	(7, 7)	14	14	



Selected pixel positions (solid circles) along a circle path with radius r = 10 centered on the origin, using the midpt. circle algo. Open circles shows the symmetry positions of 1st quandrant

The following procedure displays a raster cirle on a bilevel monitor using midpoint algorithm. Input to the procedure are the coordinates for the circle center and the radius. Intensities for pixel positions along the circle circumference are loaded into the frame-buffer array with calls to the setPixel routine.

```
#include "device.h"
void circleMidpoint (int xCenter, int yCenter, int radius)
{
   int x = 0;
   int y = radius;
   int p = 1 - radius;
   void circlePlotPoints (int, int, int, int);
/* Plot first set of points */
circlePlotPoints (xCenter , yCenter , x , y);
while (x < y)
{
    x++;
    if (p < 0)
        p += 2 * x + 1 ;
    else
    {
        y--;
        p+= 2 * (x - y) + 1 ;
    }
    circlePlotPoints (xCenter , yCenter , x , y);
}
}
void circlePlotPoints (int xCenter , int yCenter , int x, int y)
{
  setPixel (xCenter + x , yCenter + y);
  setPixel (xCenter - x , yCenter + y);
  setPixel (xCenter + x , yCenter - y);
  setPixel (xCenter - x , yCenter - y);
  setPixel (xCenter + y , yCenter + x);
  setPixel (xCenter - y , yCenter + x);
  setPixel (xCenter + y , yCenter - x);
  setPixel (xCenter - y , yCenter - x);

}
```