STUDY MATERIAL FOR BCA

COMPUTER GRAPHICS

SEMESTER - VI

ACADEMIC YEAR 2022 - 2023

PREPARED BY

DEPARTMENT OF COMPUTER SCIENCE

UNIT - I

INPUT AND OUTPUT DEVICES

Applications of Computer Graphics:

Computer graphics deals with creation, manipulation and storage of different type of images and objects.

Some of the applications of computer graphics are:

Computer Art:

Using computer graphics we can create fine and commercial art which include animation packages, paint packages. These packages provide facilities for designing object shapes and specifying object motion. Cartoon drawing, paintings, logo design can also be done.

Computer Aided Drawing:

Designing of buildings, automobile, aircraft is done with the help of computer aided drawing, this helps in providing minute details to the drawing and producing more accurate andsharp drawings with better specifications.

Presentation Graphics:

For the preparation of reports or summarizing the financial, statistical, mathematical, scientific, economic data for research reports, managerial reports, moreover creation of bar graphs, pie charts, time chart, can be done using the tools present in computer graphics.

Entertainment:

Computer graphics finds a major part of its utility in the movie industry and game industry. Used for creating motion pictures, music video, television shows, cartoon animation films. In the game industry where focus and interactivity are the key players, computer graphics helps in providing such features in the efficient way.

Education:

Computer generated models are extremely useful for teaching huge number of concepts and fundamentals in an easy to understand and learn manner. Using computer graphics many educational models can be created through which more interest can be generated among the students regarding the subject.

Training:

Specialized system for training like simulators can be used for training the candidates in a way that can be grasped in a short span of time with better understanding. Creation of training modules using computer graphics is simple and very useful.

Graphic operations:

- A general purpose graphics package provides user with Varity of function for creating and manipulating pictures.

- The basic building blocks for pictures are referred to as output primitives. They includes character, string, and geometry entities such as point, straight lines, curved lines, filled areas and shapes defined with arrays of color points.

- Input functions are used for control & process the various input device such as mouse, tablet, etc.

- Control operations are used to controlling and housekeeping tasks such as clearingdisplay screen etc.

- All such inbuilt function which we can use for our purpose are known asgraphics function

Software Standard

- Primary goal of standardize graphics software is portability so that it can be used in anyhardware systems & avoid rewriting of software program for different system

- Some of these standards are discussed below

Graphical Kernel System (GKS)

- This system was adopted as a first graphics software standard by the international standard organization (ISO) and various national standard organizations including ANSI.

- GKS was originally designed as the two dimensional graphics package and then laterextension was developed for three dimensions.

PHIGS (Programmer's Hierarchical Interactive Graphic Standard)

- PHIGS is extension of GKS. Increased capability for object modeling, color specifications, surface rendering, and picture manipulation are provided in PHIGS.

- Extension of PHIGS called "PHIGS+" was developed to provide three dimensional surface shading capabilities not available in PHIGS.

Output primitivesPoints 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 betweentwo specified end point positions.

- The output device is then directed to fill in those positions between the endpoints withsome color.

- For some device such as a pen plotter or random scan display, a straight line can bedrawn smoothly from one end point together.

- Digital devices display a straight line segment by plotting discrete points between thetwo endpoints.

- Discrete coordinate positions along the line path are calculated from the equation ofthe line.

- For a raster video display, the line intensity is loaded in frame buffer at thecorresponding 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).

- Thisroundingofcoordinatevaluestointegerscauseslinestobedisplayed

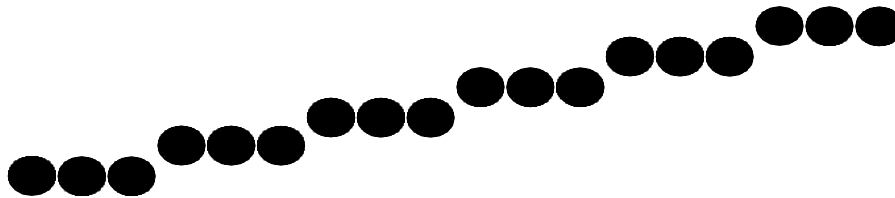- With as t air step appearance ("the jaggies"), as represented in fig 2.1.

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. Pixel positions referenced by scan−line number andcolumn 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 **set pixel (x,y).**

- Similarly for retrieve the current frame buffer intensity we assume to have procedure

Get pixel (x,y).

6

5

4

3

2

1   2   3   4   5

0   6
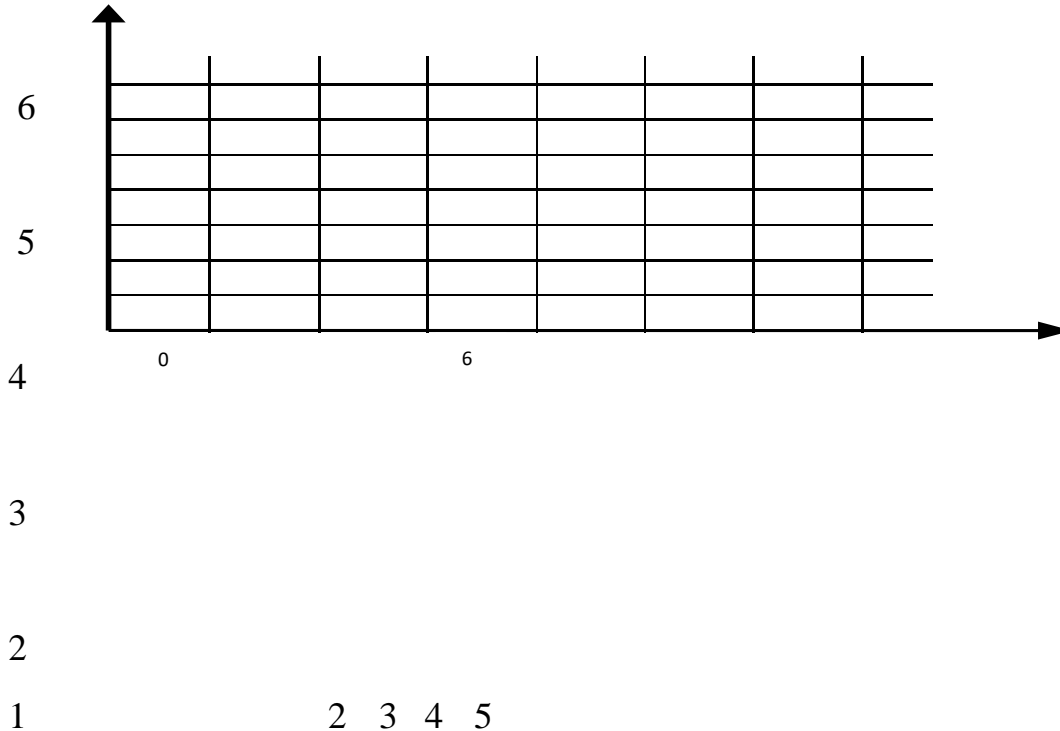
Graphics Packages:

There are mainly two types of graphics packages:

- General programming package

- Special–purpose application package

General programming package

➢ A general programming package provides an extensive set of graphics function that can be used in high level programming language such as C or FORTRAN.

➢ It includes basic drawing element shape like line, curves, polygon, color of elementtransformation etc.

➢ Example: − GL (Graphics Library).

Special-purpose application package

Special−purpose application package are customize for particular application which implement required facility and provides interface so that user need not to worry about how it will work (programming). User can simply use it by interfacing with application.

Example: − CAD, medical and business systems.

GUI: Graphical User Interface

GUI stands for Graphical User Interface. It refers to an interface that allows one to interact with electronic devices like computers and tablets through graphic elements. It uses icons, menus and other graphical representations to display information, as opposed to text-based commands. The graphic elements enable users to give commands to the computer and select functions by using mouse or other input devices.
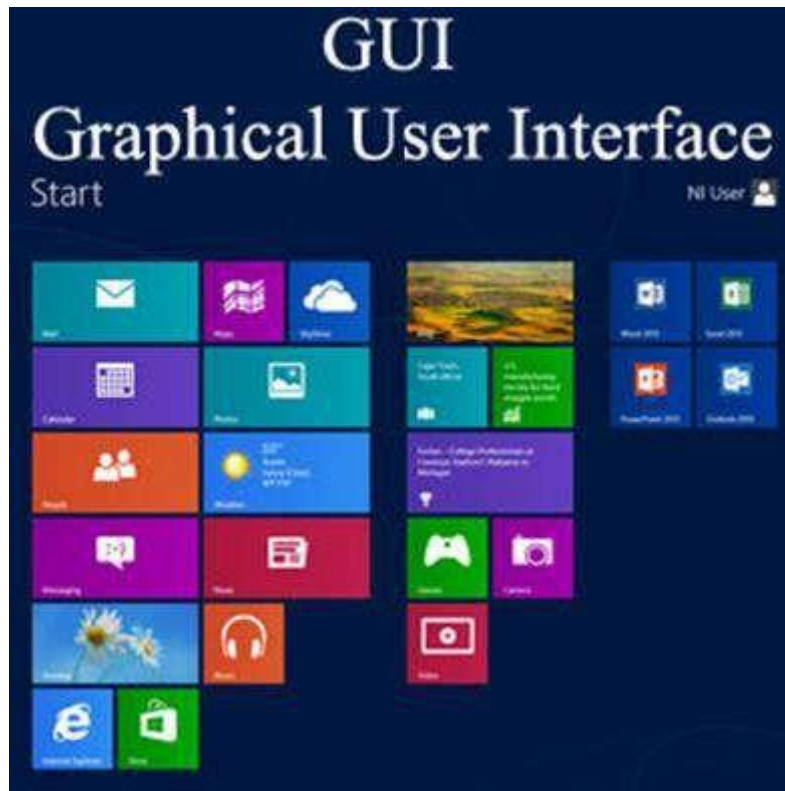
Basic Components of a GUI

➢ **Pointer:** It is a symbol that appears on the display screen. It can be moved to select commands and objects.

> **Pointing device:** It allows you move the pointer and select objects on the screen, e.g. mouse or trackball.

> **Icons:** It refers to small images on the display screen that represent commands, files, windows etc. Using pointer and pointing device, you can execute these commands.

> **Desktop:** It is the display screen that contains the icons.

GUI KEY Benefits

> It allows you to place more information within a program.

> The graphics allow users to use complex programs with greater ease.

> It saves time as you do not need to edit configurations manually.

> You can easily memorize the tasks (point-and-click).

> Helps create user-friendly software with a point-and-click interface.

Input Devices:

The Input Devices are the hardware that is used to transfer transfers input to thecomputer. The data can be in the form of text, graphics, sound, and text. Output device display data from the memory of the computer. Output can be text, numeric data, line,polygon, and other objects.

These Devices include:

1. Keyboard

2. Mouse

3. Trackball

4. Space ball

5. Joystick

6. Light Pen

7. Digitizer

8. Touch Panels

9. Voice Recognition

10. Image Scanner
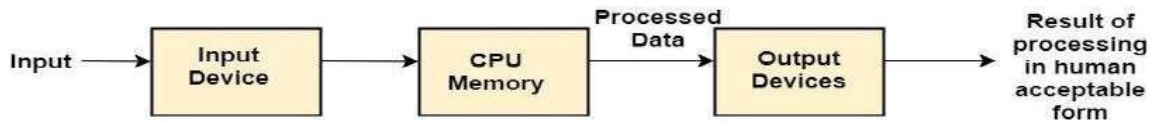
1. Keyboard

The most commonly used input device is a keyboard. The data is entered by pressing the set of keys. All keys are labelled. A keyboard with 101 keys is called a QWERTY keyboard. The keyboard has alphabetic as well as numeric keys. Some special keys are also available.

1. **Numeric Keys:** 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

2. **Alphabetic keys:** a to z (lower case), A to Z (upper case)

3. **Special Control keys:** Ctrl, Shift, Alt

4. **Special Symbol Keys**, " ? @ ~ ? :

5. **Cursor Control Keys:** ↑ → ← ↓

6. **Function Keys:** F1 F2 F3   F9.

7. **Numeric Keyboard:** It is on the right-hand side of the keyboard and used for fast entry ofnumeric data.

Function of Keyboard:

- Alphanumeric Keyboards are used in CAD. (Computer Aided Drafting)

- Keyboards are available with special features line screen co-ordinates entry, Menu selection or graphics functions, etc.

- Special purpose keyboards are available having buttons, dials, and switches. Dials are used to enter scalar values. Dials also enter real numbers. Buttons and switches are used to enter predefined function values.

2. **Mouse:**

A Mouse is a pointing device and used to position the pointer on the screen. It is a small palm size box. There are two or three depression switches on the top. The movement of the mouse along the x-axis helps in the horizontal movement of the cursor and the movement along the y-axis helps in the vertical movement of the cursor on the screen. The mouse cannotbe used to enter text. Therefore, they are used in conjunction with a keyboard.

3. **Trackball**

It is a pointing device. It is similar to a mouse. This is mainly used in notebook or laptop computer, instead of a mouse. This is a ball which is half inserted, and by changing fingers on the ball, the pointer can be moved.

4. **Space ball:**

It is similar to trackball, but it can move in six directions where trackball can move intwo directions only. The movement is recorded by the strain gauge. Strain gauge is applied with pressure. It can be pushed and pulled in various directions. The ball has a diameter around 7.5 cm. The ball is mounted in the base using rollers. One-third of the ball is an inside box, the rest is outside.

Applications:

- It is used for three-dimensional positioning of the object.

- It is used to select various functions in the field of virtual reality.

- It is applicable in CAD applications.

- Animation is also done using space ball.

- It is used in the area of simulation and modeling.

5. **Joystick:**

A Joystick is also a pointing device which is used to change cursor position on a monitor screen. Joystick is a stick having a spherical ball as it's both lower and upper ends as shown in fig.

The lower spherical ball moves in a socket. The joystick can be changed in all four directions. The function of a joystick is similar to that of the mouse. It is mainly used in Computer Aided Designing (CAD) and playing computer games.

6.   Light Pen:

Light Pen (similar to the pen) is a pointing device which is used to select displayed menu item or draw pictures on the monitor screen. It consists of a photocell and an optical system placed in a small tube. When its tip is moved over the monitor screen, and pen button ispressed, its photocell sensing element detects the screen location and sends the corresponding signals to the CPU.
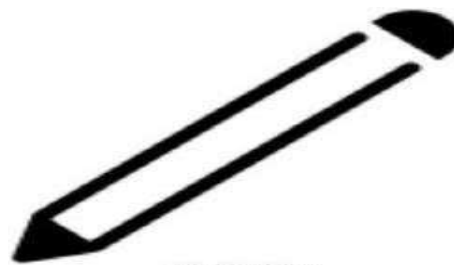
Light Pen

Uses:

- Light Pens can be used as input coordinate positions by providing necessaryarrangements.

- If background color or intensity, a light pen can be used as a locator.

- It is used as a standard pick device with many graphics system.

- It can be used as stroke input devices.

- It can be used as valuators

7.   Digitizers:

Digitizer

The digitizer is   tains a large, smooth board (the appearance is similar to the mechanical drawing board) & an electronic tracking device, which can be changed over the surface to follow

existing lines. The electronic tracking device contains a switch for the user to record the desire x & y coordinate positions. The coordinates can be entered into the computer memory or stored or an off-line storage medium such as magnetic tape.

8. **Touch Panels:**

   - Touch Panels is a type of display screen that has a touch-sensitive transparent panel covering the screen. A touch screen registers input when a finger or other object comes in contact with the screen.

   - When the wave signals are interrupted by some contact with the screen, that located is recorded.

   - Touch screens have long been used in military applications.

9. **Voice Systems (Voice Recognition):**

   - Voice Recognition is one of the newest, most complex input techniques used to interact with the computer. The user inputs data by speaking into a microphone. The simplest form of voice recognition is a one-word command spoken by one person. Each command is isolated with pauses between the words.

   - Voice Recognition is used in some graphics workstations as input devices to accept voice commands. The voice-system input can be used to initiate graphics operations or to enter data. These systems operate by matching an input against a predefined dictionary of words and phrases.

10. **Image Scanner**

   - It is an input device. The data or text is written on paper. The paper is fed to scanner. The paper written information is converted into electronic format; this format is stored in the computer. The input documents can contain text, handwritten material, picture extra.

   - By storing the document in a computer document became safe for longer period of time. The document will be permanently stored for the future. We can change the document when we need. The document can be printed when needed.

- Scanning can be of the black and white or colored picture. On stored picture 2D or 3D rotations, scaling and other operations can be applied.

Output Devices in Computer Graphics:

Printers:

A printer is a peripheral device which is used to represent the graphics or text on paper. The quality is measured by its resolution. The resolution of any printer is measured in dot per inch (dpi).

The printer usually works with the computer and connected via a cable. In present, many digital device support printer features so that we can use Bluetooth, Wi-Fi, and cloud technology to print.

Some types of printers are:

- Impact Printers

- Non-impact Printers

Impact Printers

In impact printers, there is a physical contact established between the print head,ribbon, ink-cartridge, and paper.

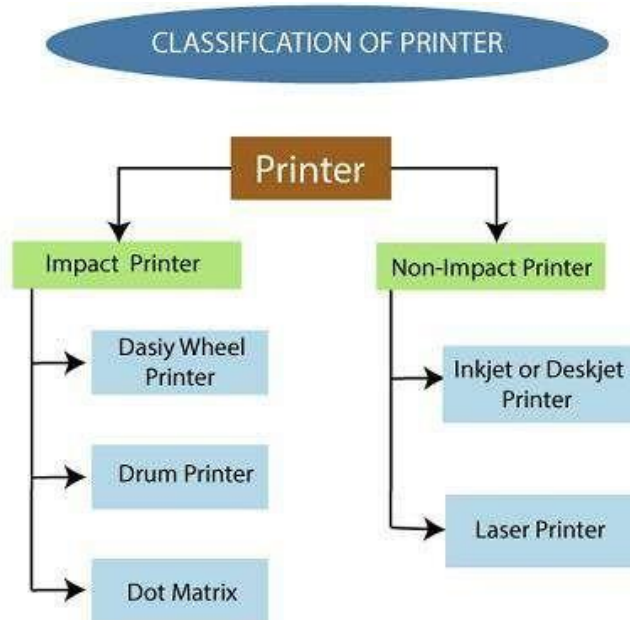The printers hit print head on an ink-filled ribbon than the letter prints on the paper.

Impact printers are works like a typewriter.

These printers have three types:

- Daisy Wheel Printers

- Drum Printers

- Dot Matrix Printer

Types of Printers:

Daisy Wheel Printers:

By these, we can print only one character at a time. The head of this printer looks like a daisy flower, with the printing arms that appear like petals of a flower; that's why it is called "**Daisy printer**. "It can print approx. 90 characters per second.

Daisy Wheel Printer

Daisy Wheel Printer

Daisy wheel printers are used to print the professional quality document. It is alsocalled **"Letter Quality Printer."**

Advantages:

- More reliable

- Better printing Quality

Disadvantages:

- Slow than Dot Matrix

- More Expensive

- Noisy in operation

Drum Printers:

It has a shape like a drum, so it is called **"Drum Printer."** This type of printer contains many characters that are printed on the drum. The surface of the drum is break down into the number of tracks. Total tracks are equal to character132. A drum will have 132 tracks. The number of tracks is divided according to the width of the paper. It can print approx. 150-2500 lines per minute.

**Drum Printer**

Advantages:

- High Speed

- Low Cost

Disadvantages:

- Poor Printing Quality

- Noisy in Operation

Dot Matrix Printer:

It is also known as the **"Impact Matrix Printer**." Dot Matrix Printer can print only one character at a time. The dot matrix printer uses print heads consisting of 9to 24 pins. These pinsare used to produce a pattern of dots on the paper to create a separate character. Dot-matrix printer can print any shapes of character, special character, graphs, and charts.
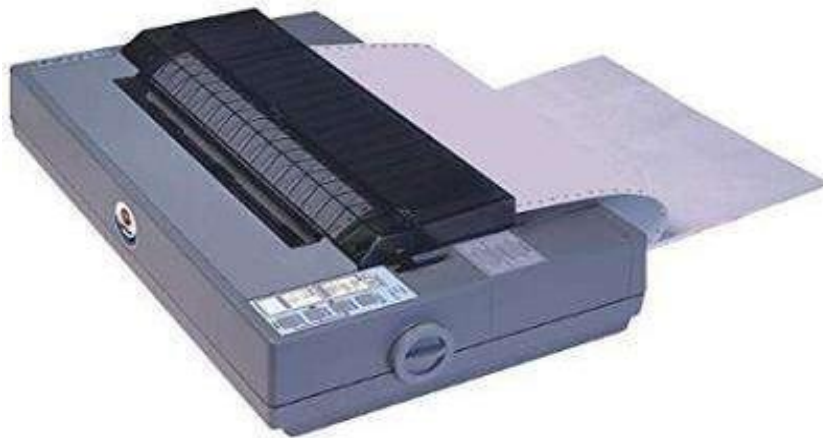
Dot Matrix Printer

Advantages:

- Low Printing Cost

- Large print size

- Long Life

Disadvantages:

- Slow speed

- Low Resolution

Non-impact Printers

In Non-impact printers, there is no physical contact between the print head or paper head. A non-impact printer prints a complete page at a time. The Non-impact printers spray ink on the paper through nozzles to form the letters and patterns. The printers that print the letters without the ribbon and on papers are called Non-impact printer. Non-impact printers are also known as "**Page Printer**."

These printers have two types:

- Inkjet Printer
- Laser Printer

Inkjet Printer:

It is also called "**Deskjet Printer.**" It is a Non-impact printer in which the letters and graphics are printed by spraying a drop of ink on the paper with nozzle head.

A Color inkjet printer has four ink nozzles, sapphire, red, yellow, and black, so it is also called CMYK printer. We can produce any color by using these four colors. The prints and graphics of this printer are very clear. These printers are generally used for home purposes.

Inkjet Printer

Advantages:

- High-Quality Printout
- Low noise

- High Resolution

Disadvantages:

- Less Durability of the print head

- Not suitable for high volume printing

- Cartridges replacement is expensive

Laser Printer:

It is also called **"Page Printer"** because a laser printer process and store the whole page before printing it. The laser printer is used to produce high-quality images and text. Mostly it is used with personal computers. The laser printers are mostly preferred to print a large amount of content on paper.

Laser printer

Advantages:

- High Resolution

- High printing Speed

- Low printing Cost

Disadvantages:

- Costly than an inkjet printer

- Larger and heavier than an inkjet printer

Plotters:

A plotter is a special type of output device. It is used to print large graphs, large designs on a large paper. **For Example:** Construction maps, engineering drawings, architectural plans, and business charts, etc.It was invented by **"Remington rand"** in 1953.It is similar to a printer, but it is used to print vector graphics.

Types of Plotter:

Two types of plotters are-
→ Drum Plotter
→ Flatbed plotter
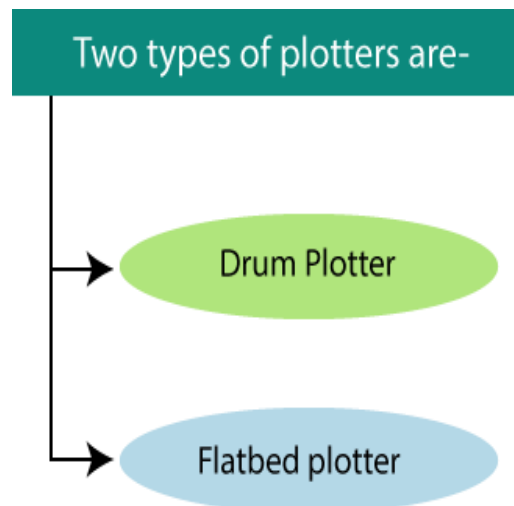
Flatbed Plotter:

In a flatbed plotter, the paper is kept in a stationary position on a table or a tray. A flatbed plotter has more than one pen and a holder. The pen rotates on the paper upside-down and right-left by the using of a motor. Every pen has a different color ink, which is used to draw the multicolor design. We can quickly draw the following designs by using a flatbed printer.

For Example: Cars, Ships, Airplanes, Dress design, road and highway blue prints, etc.

A Flatbed Plotter

Advantages of Flatbed Plotter

- Larger size paper can be used

- Drawing Quality is similar to an expert

Disadvantages of Flatbed Plotter

- Slower than printers

- More Expensive than printers

- Do not produce high-Quality text printouts

Drum Plotter:

It is also called **"Roller plotter."** There is a drum in this plotter. We can apply the paper on the drum. When the plotter works, these drums moves back and forth, and the image is drawn. Drum plotter has more than one pen and penholders. The pens easily moves right to left and left to right. The movement of pens and drums are controlled by graph plotting program. It is used in industry to produce large drawings (up to A0).

A Drum Plotter

Advantages of Drum Plotter:

- Draw Larger Size image

- We can print unlimited length of the image

Disadvantages of Drum Plotter:

1. Very costly

Visual Display Devices:

The primary output device in a graphics system is a video monitor. Although many technologies exist, but the operation of most video monitors is based on the standard Cathode Ray Tube (CRT) design.

Cathode Ray Tubes (CRT):

A cathode ray tube (CRT) is a specialized vacuum tube in which images are produced when an electron beam strikes a phosphorescent surface. It modulates, accelerates, and deflects electron beam(s) onto the screen to create the images. Most desktop computer displays make use of CRT for image displaying purposes.
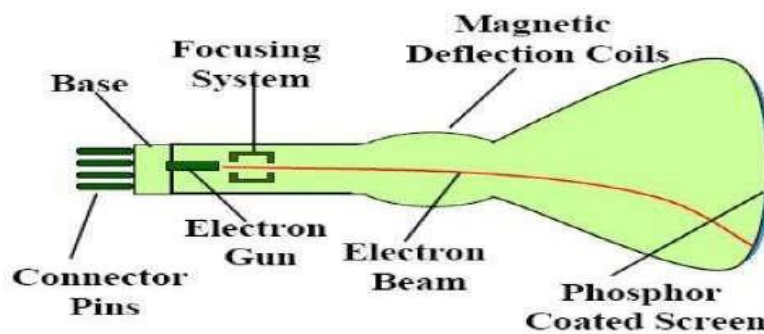
Basic design of a magnetic deflection CRT

Construction of a CRT:

- The primary components are the **heated metal** cathode and a **control grid**.

- The heat is supplied to the cathode (by passing current through the filament).This way the electrons get heated up and start getting ejected out of the cathodefilament.

- This stream of negatively charged electrons is accelerated towards the phosphor screenby supplying a high positive voltage.

- This acceleration is generally produced by means of an accelerating anode.

- Next component is the **Focusing System**, which is used to force the electron beam toconverge to small spot on the screen.

- If there will not be any focusing system, the electrons will be scattered because of their own repulsions and hence we won't get a sharp image of the object.

- This focusing can be either by means of electrostatic fields or magnetic fields.

Types of Deflection:

1. Electrostatic Deflection:

The electron beam (cathode rays) passes through a highly positively charged metal cylinder that forms an electrostatic lens. This electrostatic lens focuses the cathode rays to the center of the screen in the same way like an optical lens focuses the beam of light. Two pairs of parallel plates are mounted inside the CRT tube.
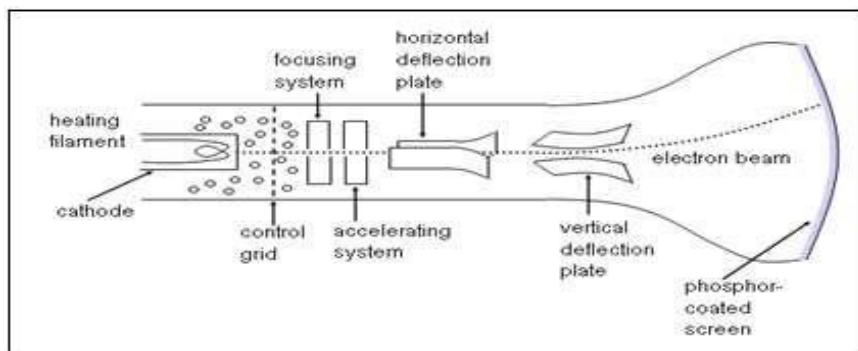
2. Magnetic Deflection:

Here, two pairs of coils are used. One pair is mounted on the top and bottom of the CRT tube, and the other pair on the two opposite sides. The magnetic field produced by both these pairs is such that a force is generated on the electron beam in a direction which is perpendicular to both the direction of magnetic field, and to the direction of flow of the beam. One pair is mounted horizontally and the other vertically.

- Different kinds of phosphors are used in a CRT. The difference is based upon the time for how long the phosphor continues to emit light after the CRT beam has been removed. This property is referred to as **Persistence**.

- The number of points displayed on a CRT is referred to as **resolutions** (eg. 1024x768).
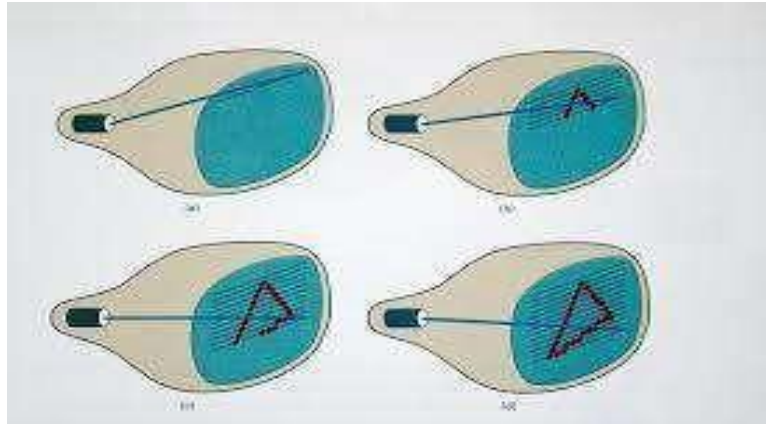
Raster-Scan

- The electron beam is swept across the screen one row at a time from top to bottom. As it moves across each row, the beam intensity is turned on and off to create a pattern of illuminated spots. This scanning process is called refreshing.

- Each complete scanning of a screen is normally called a frame. There fleshing rate, called the frame rate, is normally 60 to 80 frames per second, or described as 60 Hz to 80 Hz.

- Picture definition is stored in a memory area called the frame buffer.

- This frame buffer stores the intensity values for all the screen points. Each screen point is called a pixel (picture element or pel).

- On black and white systems, the frame buffer storing the values of the pixels is called a **bitmap**. Each entry in the bitmap is a 1-bit datawhich determine the on (1) and off (0) of the intensity of the pixel.

On color systems, the frame buffer storing the values of the pixels is called a **pixmap**(Though nowadays many graphics libraries name it as bitmap too). Each entry in the pix map occupies a number of bits to represent the color of the pixel. For a true color display, the number of bits for each entry is 24 (8 bits per red/green/blue channel, each channel 28 =256 levels of intensity value, ie. 256 voltage settings for each of the red/green/blue electron guns).

Random-Scan (Vector Display) or stroke-writing or calligraphic displays:

The CRT's electron beam is directed only to the parts of the screen where a picture is to be drawn. The picture definition is stored as a set of line-drawing commands in a refresh display file or a refresh buffer in memory.

Random-scan generally have higher resolution than raster systems and can produce smooth line drawings, however it cannot display realistic shaded scenes.

Color CRT Monitors:

The CRT Monitor display by using a combination of phosphors. The phosphors are different colors. There are two popular approaches for producing color displays with a CRT are:

- Beam Penetration Method

- Shadow-Mask Method
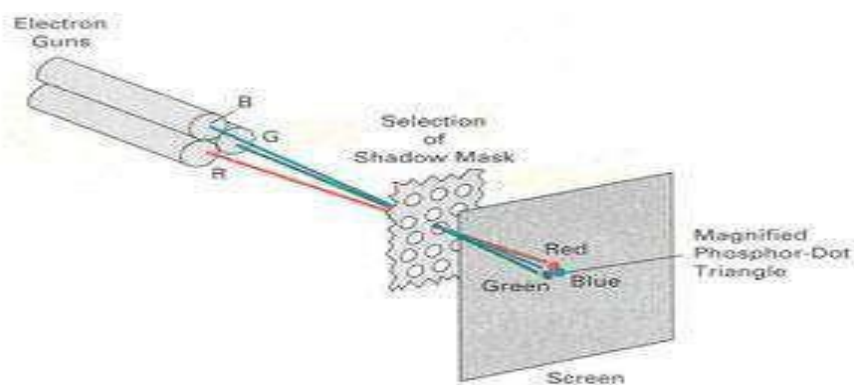
1. Beam Penetration Method:

The Beam-Penetration method has been used with random-scan monitors. In this method, the CRT screen is coated with two layers of phosphor, red and green and the displayed color depends on how far the electron beam penetrates the phosphor layers. This method produces four colors only, red, green, orange and yellow. A beam of slow electrons excites the outer red layer only; hence screen shows red color only. A beam of high-speed electrons excites the inner green layer. Thus screen shows a green color.

Advantages:

* Inexpensive

Disadvantages:

* Only four colors are possible

* Quality of pictures is not as good as with another method.

2. Shadow-Mask Method:

* Shadow Mask Method is commonly used in Raster-Scan System because they produce a much wider range of colors than the beam-penetration method.

* It is used in the majority of color TV sets and monitors.

Construction:

A shadow mask CRT has 3 phosphor color dots at each pixel position.

* One phosphor dot emits: red light

* Another emits: green light

* Third emits: blue light

This type of CRT has 3 electron guns, one for each color dot and a shadow mask grid just behind the phosphor coated screen. Shadow mask grid is pierced with small round holes in a triangular pattern.

Working:

- The deflection system of the CRT operates on all 3 electron beams simultaneously; the 3 electron beams are deflected and focused as a group onto the shadow mask, which contains a sequence of holes aligned with the phosphor- dot patterns.

- When the three beams pass through a hole in the shadow mask, they activate a dotted triangle, which occurs as a small color spot on the screen.

- The phosphor dots in the triangles are organized so that each electron beam can activate only its corresponding color dot when it passes through the shadow mask.

Advantage:

- Realistic image

- Million different colors to be generated

- Shadow scenes are possible

Disadvantage:

- Relatively expensive compared with the monochrome CRT.

- Relatively poor resolution

- Convergence Problem

Direct View Storage Tubes:

DVST terminals also use the random scan approach to generate the image on the CRT screen. The term "storage tube" refers to the ability of the screen to retain the image which has been projected against it, thus avoiding the need to rewrite the image constantly.

Function of guns:

Two guns are used in DVST:

- **Primary guns:** It is used to store the picture pattern.

- **Flood gun or Secondary gun:** It is used to maintain picture display.

Advantage:

- No refreshing is needed.

- High Resolution

- Cost is very less

Disadvantage:

- It is not possible to erase the selected part of a picture.

- It is not suitable for dynamic graphics applications.

- If a part of picture is to modify, then time is consumed.

Flat Panel Display:

The Flat-Panel display refers to a class of video devices that have reduced volume,weight and power requirement compare to CRT.

Example: Small T.V. monitor, calculator, pocket video games, laptop computers, anadvertisement board in elevator.

Emissive Display:

The emissive displays are devices that convert electrical energy into light. Examples are Plasma Panel, thin film electroluminescent display and LED (Light Emitting Diodes).

Non-Emissive Display:

The Non-Emissive displays use optical effects to convert sunlight or light from someother source into graphics patterns. Examples are LCD (Liquid Crystal Device).

Plasma Panel Display:

Plasma-Panels are also called as Gas-Discharge Display. It consists of an array of smalllights. Lights are fluorescent in nature.

The essential components of the plasma-panel display are:

- **Cathode:** It consists of fine wires. It delivers negative voltage to gas cells. The voltage is released along with the negative axis.

- **Anode:** It also consists of line wires. It delivers positive voltage. The voltage is supplied along positive axis.

- **Fluorescent cells:** It consists of small pockets of gas liquids when the voltage is applied to this liquid (neon gas) it emits light.

- **Glass Plates:** These plates act as capacitors. The voltage will be applied, the cell willglow continuously. The gas will slow when there is a significant voltage difference between horizontal and vertical wires. The voltage level is kept between 90 volts to 120 volts. Plasma level does not require refreshing. Erasing is done by reducing the voltageto 90 volts.

Each cell of plasma has two states, so cell is said to be stable. Displayable point in plasma panel is made by the crossing of the horizontal and vertical grid. The resolution of the plasma panel can be up to 512 * 512 pixels.

Advantage:

- High Resolution

- Large screen size is also possible.

- Less Volume

- Less weight

- Flicker Free Display

Disadvantage:

- Poor Resolution

- Wiring requirement anode and the cathode is complex.

- Its addressing is also complex.

LED (Light Emitting Diode):

In an LED, a matrix of diodes is organized to form the pixel positions in the display and picture definition is stored in a refresh buffer. Data is read from the refresh buffer and converted to voltage levels that are applied to the diodes to produce the light pattern in the display.

LCD (Liquid Crystal Display): Liquid Crystal Displays are the devices that produce a picture by passing polarized light from the surroundings or from an internal light source through liquid-crystal material that transmits the light.

LCD uses the liquid-crystal material between two glass plates; each plate is the right angle to each other between plates liquid is filled. One glass plate consists of rows of conductors arranged in vertical direction. Another glass plate is consisting of a row of conductors arranged in horizontal direction. The pixel position is determined by the intersection of the vertical & horizontal conductor. This position is an active part of the screen.

Liquid crystal display is temperature dependent. It is between zeros to seventy degree Celsius. It is flat and requires very little power to operate.

Advantage:

- Low power consumption.

- Small Size

- Low Cost

Disadvantage:

- LCDs are temperature-dependent (0-70°C)

- LCDs do not emit light; as a result, the image has very little contrast.

- LCDs have no color capability.

- The resolution is not as good as that of a CRT.

Differences between Raster Scan Display and Random ScanDisplay:

Base of Differences	Raster Scan System	Random Scan System
ElectronBeam	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 bedrawn.
Resolution	Its resolution is poor because raster system in contrast produces zig-zag 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 asset 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 todraw an image.	Mathematical functions are used to drawnanimate.

Graphics Accelerator

A graphics accelerator is a piece of dedicated hardware designed and used to rapidly process visual data. It is a full computer in its own right, as it has its own processor, RAM, buses and even I/O mechanisms which it uses to

connect to the computer system; in modern computers this is the PCI-E port. Graphics accelerator is an older term for what is now commonly called a graphics processing unit (GPU).

Graphics accelerators are widely used in industries such as:

- Computer-aided design (CAD)

- Motion pictures for special effects

- Video games

Graphics accelerators are now present not just in PCs and laptops, but many mobile devices like tablets and smart.

Co-Processor

A co-processor is a chip that works side-by-side with the computer's main processor (the chip called the central processing unit, or CPU). The co-processor handles some of the more specialized tasks, such as doing math calculations or displaying graphics on the screen, thereby taking some of the work load off the main processor so it can go on with the business of directing and keeping order over the whole show. A co-processor is installed to reduce the burden on a computer's CPU and thus free it for more general duties such as transferring data and handling multiple tasks. Hones as well.

UNIT - II

ALGORITHMS

Definition

It is a process of representing graphics objects a collection of pixels. The graphics objects are continuous. The pixels used are discrete. Each pixel can have either 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 computer represent picture having discrete dots

Any model of graphics can be reproduced with a dense matrix of dots or points. Most human beings think graphics objects as points, lines, circles, ellipses. For generating graphical object, many algorithms have been developed.

Advantage of developing algorithms for scan conversion:

1. Algorithms can generate graphics objects at a faster rate.

2. Using algorithms memory can be used efficiently.

3. Algorithms can develop a higher level of graphical objects.

Examples of objects which can be scan converted

1. Point

2. Line

3. Sector

4. Arc

5. Ellipse

6. Rectangle

7. Polygon

8. Characters

9. Filled Regions

The process of converting is also called as rasterization. The algorithms implementation varies from one computer system to another computer system. Some algorithms are implemented using the software. Some are performed using hardware or firmware. Some are performed using various combinations of hardware, firmware, and software.

DDA Line 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 (x_i, y_i)

The line of equation for step i

$y_i = mx_i + b$ .... equation 1 Next value will be

$y_{i+1} = mx_{i+1} + b$ .... equation 2

$m = \dfrac{\Delta y}{\Delta x}$

$y_{i+1} - y_i = \Delta y$ .... equation 3

$y_{i+1} - x_i = \Delta x$ .... equation 4

$y_{i+1} = y_i + \Delta y$

$\Delta y = m\Delta x$

$y_{i+1} = y_i + m\Delta x$

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

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

$x = x1, y = y1$ set $\Delta x = 1$ $y_{i+1} = y1 + m$, $x = x + 1$ Until $x = x2$

Case2: When $|m| > 1$ then (assume that $y1 < y2$) $x = x1, y = y1$ set $\Delta y = 1$

$x_{i+1} = \dfrac{1}{m}$, $y = y + 1$

Until $y \rightarrow y2$

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 twiceis 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 offerror 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 forhardware 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/stepassign 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

Example: If a line is drawn from (2, 3) to (6, 15) with use of DDA. How many points will needed togenerate such line?

Solution: P1 (2,3)   P11 (6,15)x1=2

y1=3 x2= 6 y2=15

dx = 6 - 2 = 4

dy = 15 - 3 = 12

$$m = \frac{dy}{dx} = \frac{12}{4}$$

For calculating next value of x takes $x = x + \frac{1}{m}$

$P_1(2, 3)$	point plotted
$P_2(2\frac{1}{3}, 4)$	point plotted
$P_3(2\frac{2}{3}, 5)$	point not plotted
$P_4(3, 6)$	point plotted
$P_5(3\frac{1}{3}, 7)$	point not plotted
$P_6(3\frac{2}{3}, 8)$	point not plotted
$P_7(4, 9)$	point plotted
$P_8(4\frac{1}{3}, 10)$	point not plotted
$P_9(4\frac{2}{3}, 11)$	point not plotted
$P_{10}(5, 12)$	point plotted
$P_{11}(5\frac{1}{3}, 13)$	point not plotted
$P_{12}(5\frac{2}{3}, 14)$	point not plotted
$P_{13}(6, 15)$	point plotted
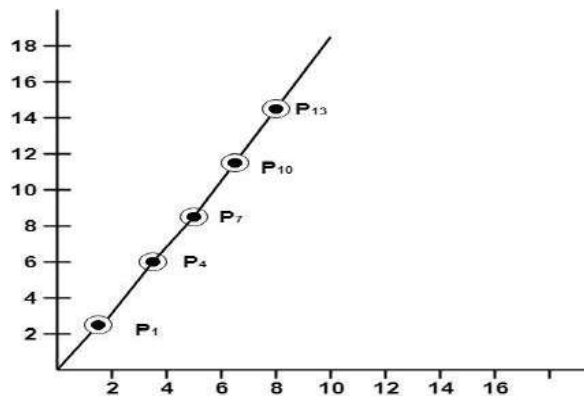
Bresenham's Line Algorithm

This algorithm is used for scan converting a line. It was developed by Brenham. 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.

\The method works as follows:

Assume a pixel P1'(x1',y1'),then select subsequent pixels as we work our way to the night, one pixel position at a time in the horizontal direction toward P2'(x2',y2').
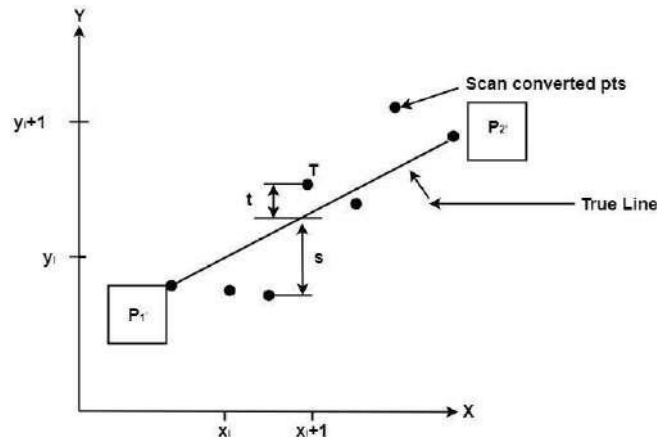
Once a pixel in choose at any step.

Fig: Scan Converting a line.

The next pixel is

1. Either the one to its right (lower-bound for the line)

2. One top its right and up (upper-bound for the line). The line is best approximated by those pixels that fall the least distance from the path between P1',P2'.To chooses the next one between the bottom pixel S and top pixel T.

3. If S is chosen

We have $x_{i+1} = x_i + 1$ and $y_{i+1} = y_i$ If T is chosen

We have $x_{i+1} = x_i + 1$ and $y_{i+1} = y_i + 1$

The actual y coordinates of the line at $x = x_i + 1$ is $y = mx_i + 1 + b$

$$y = m(x_i + 1) + b$$

The distance from S to the actual line in y direction $s = y - y_i$

The distance from T to the actual line in y direction $t = (y_i + 1) - y$

Now consider the difference between these 2 distance valuess - t

When (s-t) <0 $\Rightarrow$ s < t The closest pixel is S When (s-t) ≥0 $\Rightarrow$ s < t The closest pixel is T This difference is

s-t = (y-yi)-[(yi+1)-y]

= 2y - 2yi

$$s - t = 2m(x_i + 1) + 2b - 2y_i - 1$$

[Putting the value of (1)]

-1

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 xendas maximum possible value of x.

If dx < 0

Then x = x2 y = y2 xend=x1

If dx > 0 Then x = x1

y = y1

xend=x2

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

Step7: Check if whole line is generated.

If x > = xend Stop.

Step8: Calculate co-ordinates of the next pixelIf d < 0

Thend = d + i1

If d ≥ 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

Example: Starting and Ending position of the line are (1, 1) and (8, 5). Find intermediate points.
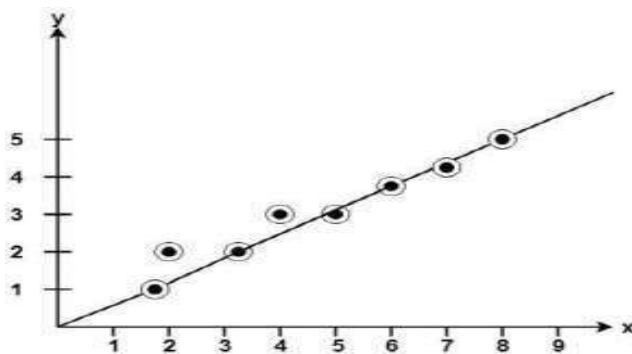
Solution: x1=1

y1=1x2=8y2=5

dx= x2-x1=8-1=7dy=y2-y1=5-1=4

I1=2* Δy=2*4=8 I2=2*(Δy-Δx)=2*(4-7)=-6d = I1-Δx=8-7=1

x	y	d=d+I1 or I2
1	1	d+I2=1+(-6)=-5
2	2	d+I1=-5+8=3
3	2	d+I2=3+(-6)=-3
4	3	d+I1=-3+8=5
5	3	d+I2=5+(-6)=-1
6	4	d+I1=-1+8=7
7	4	d+I2=7+(-6)=1
8	5	

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 anddivision.

4. It is faster as compared to DDA (Digital Differential Analyzer)

because it does not involve floating point calculations like DDA Algorithm.

Disadvantage:

1. 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.

DDA Algorithm	Bresenyhham's Line Algorithm
➢ DDA Algorithm use floating point, i.e.,RealArithmetic.	➢ Bresenham's Line Algorithm use fixed point, i.e., Integer Arithmetic
➢ DDA Algorithms uses multiplication &division its operation	➢ Bresenham's Line Algorithm usesonlysubtraction and addition its operation
➢ DDA Algorithm is slowly than Bresenham'sLine Algorithm in line drawing because it uses real arithmetic (Floating Point operation)	➢ Bresenham's Algorithm is faster than DDA Algorithm in line because itinvolves only addition & subtraction in its calculation and uses only integer arithmetic.
➢ DDA Algorithm is not accurate and efficient as Bresenham's Line Algorithm.	➢ 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	➢ Bresenham's Line Algorithm can drawcircle and curves with more accurate than DDA Algorithm.

Polynomial Method:

The first method defines a circle with the second-order polynomial equation as shown in fig:

$y^2 = r^2 - x^2$
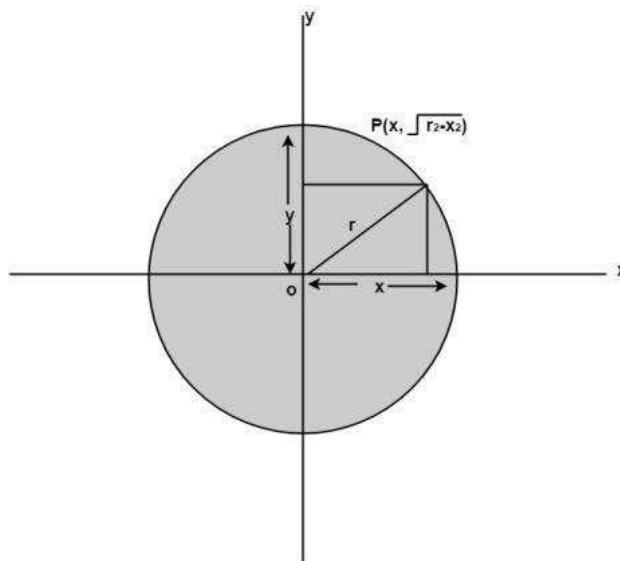
Where x = the x coordinatey = the y coordinate

r = the circle radius

With the method, each x coordinate in the sector, from 90° to 45°, is found by stepping x from 0

$$\frac{r}{\sqrt{2}}$$

$$\sqrt{r^2 - x^2}$$

to   & each y coordinate is found by evaluating   for each step of x.

Algorithm:

Step1: Set the initial variablesr = circle radius

(h, k) = coordinates of circle centerx=o

I = step size X end

Step2: Test to determine whether the entire circle has been scan-converted. If x >xend then stop.

Step3: Compute y = $\sqrt{r^2 - x^2}$

Step4: Plot the eight points found by symmetry concerning the center (h, k) at the current (x, y) coordinates.

Plot (x + h, y +k)   Plot (-x + h, -y + k)
Plot (y + h, x + k)   Plot (-y + h, -x + k)
Plot (-y + h, x + k)   Plot (y + h, -x + k)
Plot (-x + h, y + k) Plot (x + h, -y + k)

Step5: Increment x = x + i

Step6: Go to step (ii).

Circle Generation Algorithm

Drawing a circle on the screen is a little complex than drawing a line. Thereare two popularalgorithms 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 X2+Y2=r2,X2+Y2=r2, where r is radius.
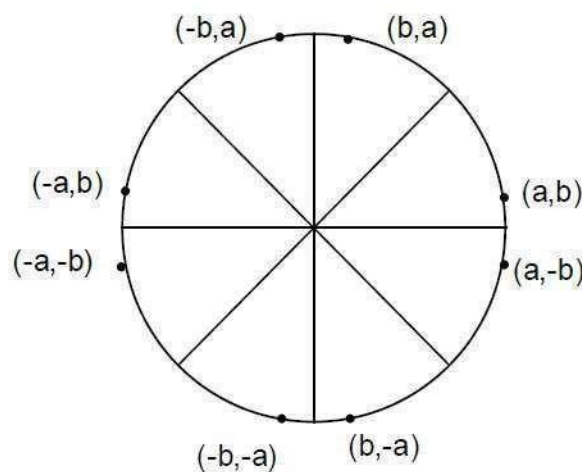
Bresenham's Algorithm

We cannot display a continuous arc on the raster display. Instead, we have to choosethe nearest pixel position to complete the arc.

From the following illustration, you can see that we have put the pixel at X,

YX,Y location

and now need to decide where to put the next pixel − at N X+1,YX+1,Y or at S X+1,Y−1X+1,Y−1.
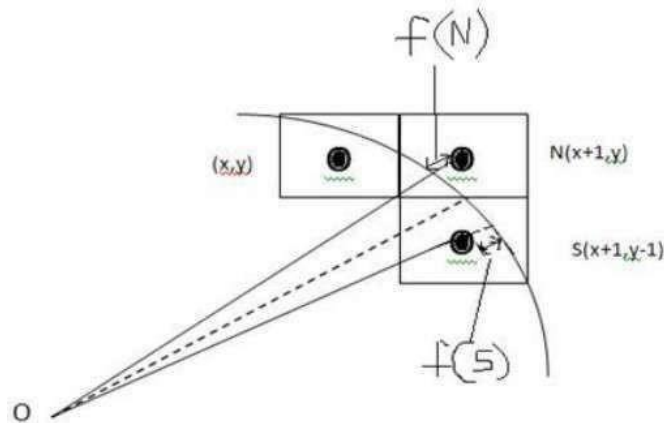
This can be decided by the decision parameter **d**.

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

2. If d > 0, then SX+1,Y−1X+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,yand 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, QX,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 + 4P−QP−Q + 10.

Step 8 − Call Draw Circle X, Y, P, QX, Y, P, Q.

Draw Circle Method(X, Y, P, Q).

Call Putpixel (X + P, Y + Q). Call Putpixel (X - P, Y + Q). Call Putpixel (X + P, Y - Q). Call Putpixel (X - P, Y - Q). Call Putpixel (X + Q, Y + P). Call Putpixel

(X - Q, Y + P).Call Putpixel (X + Q, Y - P).Call Putpixel (X - Q, Y - P).

Flood Fill Algorithm

Sometimes we come across an object where we want to fill the area and its boundary with different colors. We can paint such objects with a specified interior color instead of searching for particular boundary color as in boundary filling algorithm.

Instead of relying on the boundary of the object, it relies on the fill color. In other words, it replaces the interior color of the object with the fill color. When no more pixels of the original interior color exist, the algorithm is completed.

Once again, this algorithm relies on the Four-connect or Eight-connect method of filling in the pixels. But instead of looking for the boundary color, it is looking for all adjacent pixels that are a part of the interior.
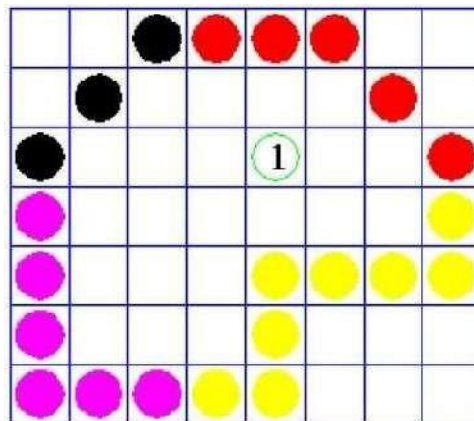
Boundary Fill Algorithm

The boundary fill algorithm works as its name. This algorithm picks a point inside an object and starts to fill until it hits the boundary of the object. The color of the boundary and the color that we fill should be different for this algorithm to work.

In this algorithm, we assume that color of the boundary is same for the entire object. The boundary fill algorithm can be implemented by 4-connected pixels or 8-connected pixels.

Connected Polygon

In this technique 4-connected pixels are used as shown in the figure. We are putting the pixels above, below, to the right, and to the left side of the current pixels and this process will continue until we find a boundary with different color.

Algorithm

Step 1 − Initialize the value of seed point seedx,seedyseedx,seedy, fcolor anddcol.

Step 2 − Define the boundary values of the polygon.

Step 3 − Check if the current seed point is of default color, then repeat the steps 4 and 5 till the boundary pixels reached. If get pixel (x,y) = dcol then repeat step 4 and 5

Step 4 −Change the default color with the fill color at the seed point.SetPixel(seedx, seedy, fcol)

Step 5 − Recursively follow the procedure with four neighbourhood points. FloodFill (seedx – 1, seedy, fcol, dcol)

FloodFill (seedx + 1, seedy, fcol, dcol) FloodFill (seedx, seedy - 1, fcol, dcol) FloodFill (seedx – 1, seedy + 1, fcol, dcol)
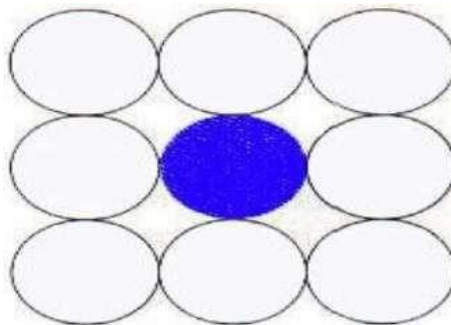
Step 6 − Exit

There is a problem with this technique. Consider the case as shown below where we tried to fill the entire region. Here, the image is filled only partially. In such cases, 4-connected pixels technique cannot be used.
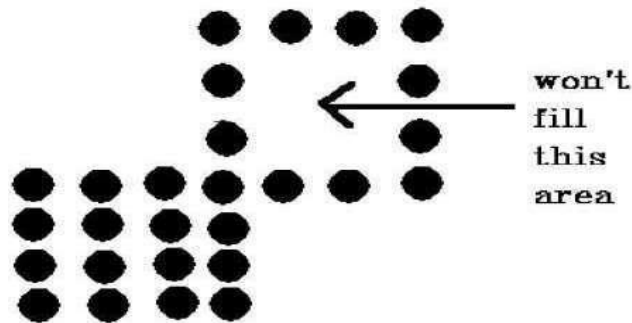
Connected Polygon

In this technique 8-connected pixels are used as shown in the figure. We are putting pixels above, below, right and left side of the current pixels as we were doing in 4-connected technique.

In addition to this, we are also putting pixels in diagonals so that entire area of the current pixel is covered. This process will continue until we find a boundary with different color.

Algorithm

Step 1 − Initialize the value of seed point seedx, seedyseedx,seedy, fcolorand dcol.

Step 2 − Define the boundary values of the polygon.

Step 3 − Check if the current seed point is of default color then repeat the steps 4 and 5 till theboundary pixels reachedIf getpixel(x,y) = dcol then repeat step 4 and 5

Step 4 − Change the default color with the fill color at the seed point.

SetPixel(seedx, seedy, fcol)

Step 5 − Recursively follow the procedure with four neighbourhood points

FloodFill (seedx – 1, seedy, fcol, dcol) FloodFill (seedx + 1, seedy, fcol, dcol) FloodFill (seedx, seedy - 1, fcol, dcol) FloodFill (seedx, seedy + 1, fcol, dcol) FloodFill (seedx – 1, seedy + 1, fcol, dcol) FloodFill (seedx + 1, seedy + 1, fcol, dcol) FloodFill (seedx + 1, seedy - 1, fcol, dcol) FloodFill (seedx – 1, seedy - 1, fcol, dcol)

Step 6 − Exit

The 4-connected pixel technique failed to fill the area as marked in the following figure which won't happen with the 8-connected technique.
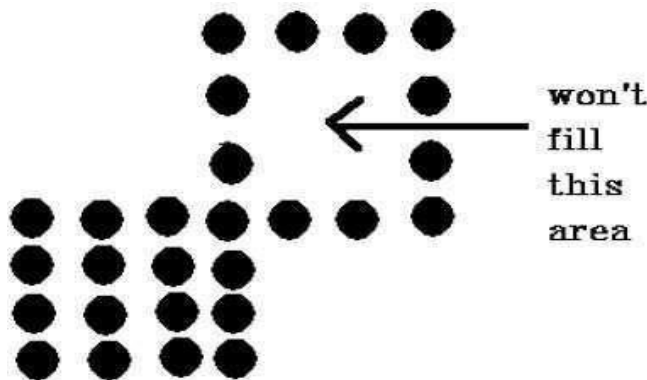
Inside-outside Test

This method is also known as **counting number method**. While filling an object, we often need to identify whether particular point is inside the object or outside it. There are two methods by which we can identify whether particular point is inside an object or outside.

> ➢ Odd-Even Rule

> ➢ Nonzero winding number rule

Odd-Even Rule

In this technique, we count the edge crossing along the line from any point x,y to infinity. If the number of interactions is odd then the point x,y is an interior point. If the number of interactions is even then point x,y is an exterior point. Here is the example to give you the clear idea −
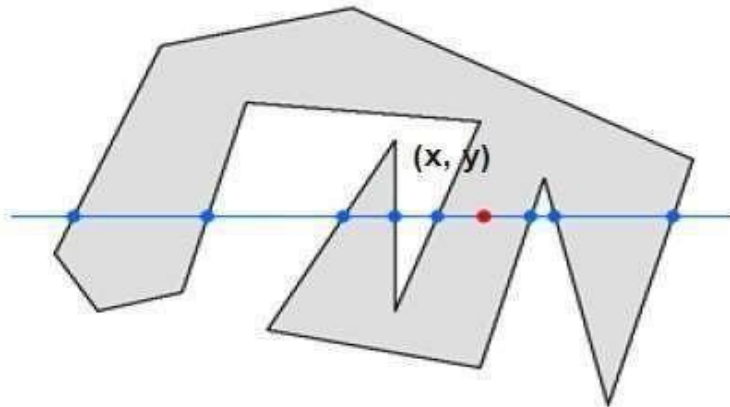
From the above figure, we can see that from the point x,yx,y, the number of interactionspoint on the left side is 5 and on the right side is 3. So the total number of interaction point is 8, which is odd. Hence, the point is considered within the object.

Winding Number Rule

This method is also used with the simple polygons to test the given point is interior or not. It can be simply understood with the help of a pin and a rubber band. Fix up the pin on one of the edge of the polygon and tie-up the rubber banding it and then stretch the rubber band along the edges of the polygon.

When all the edges of the polygon are covered by the rubber band, check out the pin which has been fixed up at the point to be test. If we find at least one wind at the pointconsider it within the polygon, else we can say that the point is not inside the polygon.
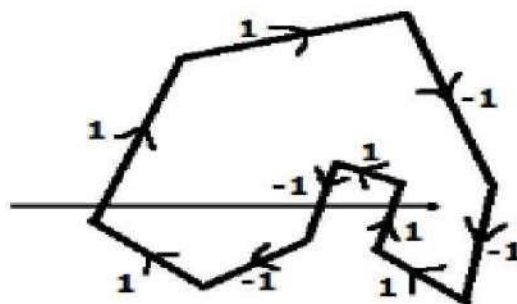
In another alternative method, give directions to all the edges of the polygon. Draw a scan line from the point to be test towards the left most of X direction.

- ➢ Give the value 1 to all the edges which are going to upward direction and all other -1 as direction values.

- ➢ Check the edge direction values from which the scan line is passing and sum up them.

- ➢ If the total sum of this direction value is non-zero, then this point to be tested is an **interior point,** otherwise it is an **exterior point**.

- ➢ In the above figure, we sum up the direction values from which the scan lineis passing then the total is $1 - 1 + 1 = 1$; which is non-zero. So the point is said to be an interior point.

UNIT – III

TRANSFORMATIONS

Two Dimensional Geometric Transformations

Changes in orientations, size and shape are accomplished with geometric transformations that alter the coordinate description of objects.

Basic TransformationTranslation

T(tx,ty)

Translation distances Scale
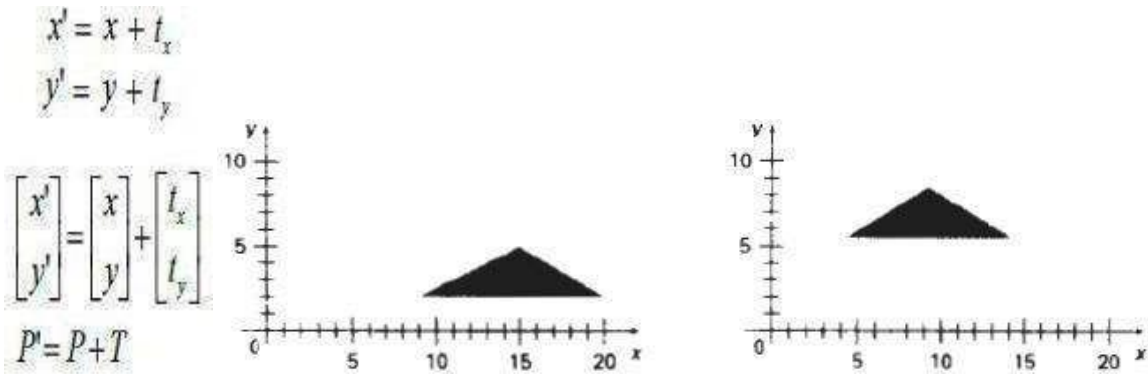
S(sx,sy)

Scale factors

- **Rotatin**

R()

Rotationangle

Translation

     A translation is applied to an object by representing it along a straight line path from one coordinate location to another adding translation distances, tx, ty to original coordinate position (x,y) to move the point to a new position (x',y') tox' = x + tx, y' = y + ty

     The translation distance point (tx,ty) is called translation vector or shift vector. Translation equation can be expressed as single matrix equation by using column vectors to represent the coordinate position and the translation vector as

$$x' = x + t_x$$
$$y' = y + t_y$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$P' = P + T$$

Moving a polygon from one position to another position with the translation vector (-5.5, 3.75)

Rotations:

A two-dimensional rotation is applied to an object by repositioning it along a circular path on xy plane. To generate a rotation, specify a rotation angle θ and the position (xr, yr) of the rotation point (pivot point) about which the object is to be rotated.

Positive values for the rotation angle define counter clock wise rotation about pivot point. Negative value of angle rotates objects in clock wise direction. The transformation can also be described as a rotation about a rotation axis perpendicular to xy plane and passes through pivot point.
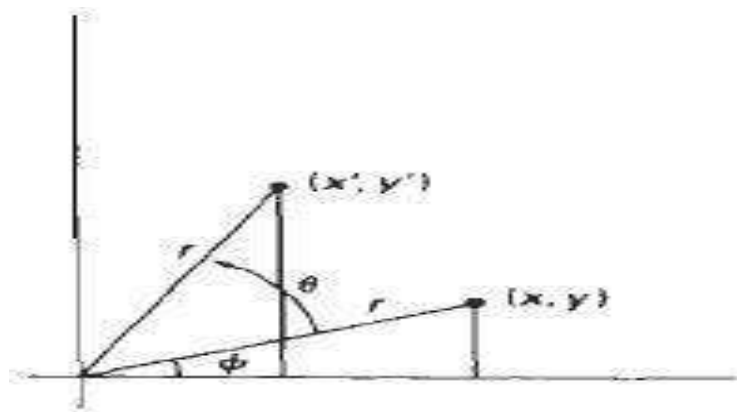
Rotation of a point from position (x, y) to position (x', y') through angle θ relative to coordinate origin

The transformation equations for rotation of a point position P when the pivot point is at coordinate origin. In figure r is constant distance of the point positions Φ is the original angular of the point from horizontal and θ is the rotation angle. The transformed coordinates in terms of angle θ and Φ

x' = rcos(θ+Φ) = rcosθosΦ − rsinθsinΦ y' = rsin(θ+Φ) = rsinθcosΦ + rcosθsinΦ

The original coordinates of the point in polar coordinates x = rcosΦ, y = rsinΦ

The transformation equation for rotating a point at position (x,y) through an angle θ

about origin

x' = xcosθ – ysinθy' = xsinθ + ycosθ

Rotation Equation

Rotation Matrix   R=$cos\emptyset$ P' = R

$-sin\emptyset$

$sin\emptyset$

$cos\emptyset$

Note: Positive values for the rotation angle define counterclockwise rotations about the rotation point and negative values rotate objects in the clockwise.

Scaling

A scaling transformation alters the size of an object. This operation can be carried out for polygons by multiplying the coordinate values (x, y) to each vertex by scaling factor Sx&Sy to produce the transformed coordinates (x', y') scaling factor Sx scales object in x direction while Sy scales in y direction. The transformation equation in matrix form

x'=x.Sx   **y' =y.Sy**

x' x 0 x

$y' = 0$ S · (or)

P' = S. P

Where S is 2 by 2 scaling matrix

(a)   (b)

Turning a square (a) Into a rectangle (b) with scaling factors sx = 2 and sy = 1.Any positive numeric values are valid for scaling factors sx and sy. Values less than 1 reduce the size of the objects and values greater than 1 produce an enlarged object.

There are two types of Scaling. They are

> ➤ Uniform scaling

> ➤ Non Uniform Scaling

To get uniform scaling it is necessary to assign same value for sx and sy. Unequal values for sx and sy result in a non-uniform scaling.

Matrix Representation and Homogeneous Coordinates

Many graphics applications involve sequences of geometric transformations. An animation, for example, might require an object to be translated and rotated at each increment of the motion. In order to combine sequence of transformations we have to eliminatethematrixaddition.Toachievethiswehaverepresentmatrixas3X3insteadof2X2introd using an additional dummy co-ordinate. Here points are specified by three numbers instead of two. This coordinate system is called as Homogeneous coordinate system and it allows expressing transformation equation as matrix multiplication. Cartesian coordinate position (x, y) is represented as homogeneous coordinate triple(x, y, h)

➢ Represent coordinates as (x, y,h)

➢ Actual coordinates drawn will be (x/h,y/h)

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$P' = T(t_x, t_y) \cdot P$$

For Translation

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$P' = S(s_x, s_y) \cdot P$$

For Scaling

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$P' = R(\theta) \cdot P$$

For Rotation

Composite Transformations

A composite transformation is a sequence of transformations; one followed by the other. We can set up a matrix for any sequence of transformations as a **composite transformation matrix** by calculating the matrix product of the individual transformations

Translation

If two successive translation vectors $(tx1,ty1)$ and $(tx2,ty2)$ are applied to a coordinate position P, the final transformed location P' is calculated as

P' = T(tx2, ty2). {T(tx1, ty1).P}

= {T(tx2, ty2).T(tx1,ty1)}.P

Where P and P' are represented as homogeneous-coordinate column vectors.

$$\begin{bmatrix} 1 & 0 & t_{2x} \\ 0 & 1 & t_{2y} \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & t_{1x} \\ 0 & 1 & t_{1y} \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_{1x} + t_{2x} \\ 0 & 1 & t_{1y} + t_{2y} \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{T}(t_{2x}, t_{2y}) \cdot \mathbf{T}(t_{1x}, t_{1y}) = \mathbf{T}(t_{1x} + t_{2x}, t_{1y} + t_{2y})$$

Which demonstrated the two successive translations are additive.

Rotations

Two successive rotations applied to point P produce the transformed position

P' = R(θ2).{R(θ1).P} = {R(θ2).R(θ1)}.P

By multiplying the two rotation matrices, we can verify that two successive rotation areadditive

R(θ2).R(θ1) = R(θ1 + θ2)

$$\begin{bmatrix} \cos\theta2 & -\sin\theta2 \\ \sin\theta2 & \cos\theta2 \\ 0 & 0 \end{bmatrix}$$

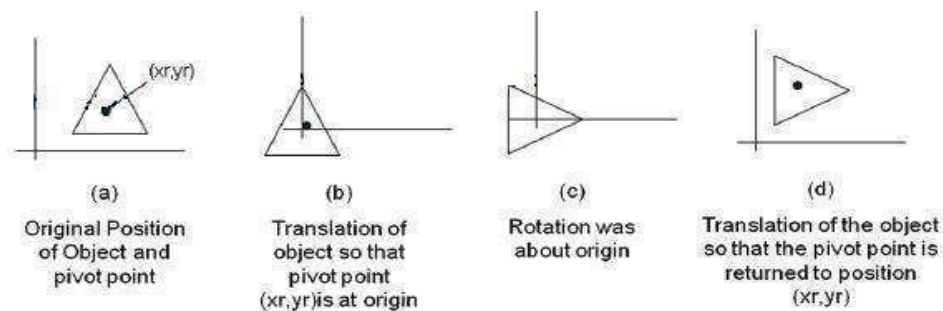So that the final rotated coordinates can be calculated with the composite rotation matrix as P' = R(θ1 + θ2).P

Scaling

Concatenating transformation matrices for two successive scaling operations produces thefollowing composite scaling matrix

$$\begin{bmatrix} sx2 & 0 & 0 \\ 0 & sy2 & 0 \\ 0 & 0 & 1 \end{bmatrix} - \begin{bmatrix} sx1 & 0 & 0 \\ 0 & sy1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

General Pivot-Point Rotation

➢ Translate the object so that pivot-position is moved to the co-ordinate origin

➢ Rotate the object about the co-ordinate origin

(a)
Original Position
of Object and
pivot point

(b)
Translation of
object so that
pivot point
(xr,yr)is at origin

(c)
Rotation was
about origin

(d)
Translation of the object
so that the pivot point is
returned to position
(xr,yr)

Translate the object so that the pivot point is returned to its original position

The composite transformation matrix for this sequence is obtain with
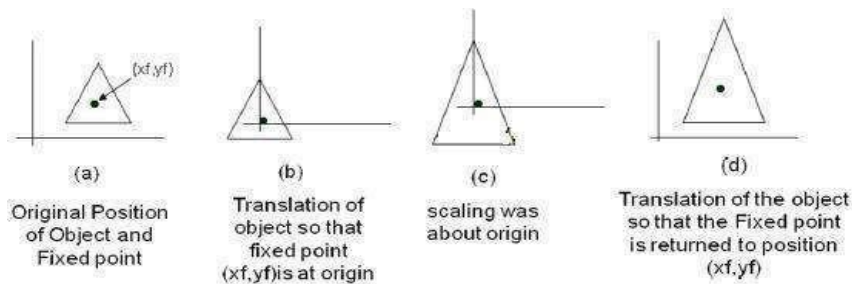
the concatenation Which can also be expressed as $T(xr, yr).R(\theta).T(-xr, -yr) = R(xr, yr, \theta)$

$$
\begin{bmatrix} 1 & 0 & xr \\ 0 & 1 & yr \\ 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} 1 & 0 & -xr \\ 0 & 1 & -yr \\ 0 & 0 & 1 \end{bmatrix}
$$

$$
\begin{bmatrix} \cos\theta & -\sin\theta & xr(1-\cos\theta)+yr\sin\theta \\ \sin\theta & \cos\theta & yr(1-\cos\theta)-xr\sin\theta \\ 0 & 0 & 1 \end{bmatrix}
$$

General fixed point Scaling

➢ Translate object so that the fixed point coincides with the co-ordinate origin

➢ Scale the object with respect to the co-ordinate origin

(a)	(b)	(c)	(d)
Original Position of Object and Fixed point	Translation of object so that fixed point (xf,yf)is at origin	scaling was about origin	Translation of the object so that the Fixed point is returned to position (xf,yf)

Use the inverse translation of step 1 to return the object to its original position

$$
\begin{bmatrix} 1 & 0 & xf \\ 0 & 1 & yf \\ 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix}
\begin{bmatrix} 1 & 0 & -xf \\ 0 & 1 & -yf \\ 0 & 0 & 1 \end{bmatrix}
$$

$$
= \begin{bmatrix} sx & 0 & xf(1-sx) \\ 0 & sy & yf(1-sy) \\ 0 & 0 & 1 \end{bmatrix}
$$

Concatenating the matrices for these three operations produces the required scaling matrix Can also be expressed as T(xf, yf).S(sx, sy).T(-xf, -yf) = S(xf, yf, sx, sy)

**Note:** Transformations can be combined by matrix multiplication

$$
\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \Theta \\ \sin \Theta \\ 0 \end{bmatrix} \right)
$$

Other Transformations

➢ Reflection

➢ Shear

Reflection:

A reflection is a transformation that produces a mirror image of an object. The mirror image for a two-dimensional reflection is generated relative to an axis of reflection by We can choose an axis of reflection in the xy plane or perpendicular to the x y plane or coordinate origin.
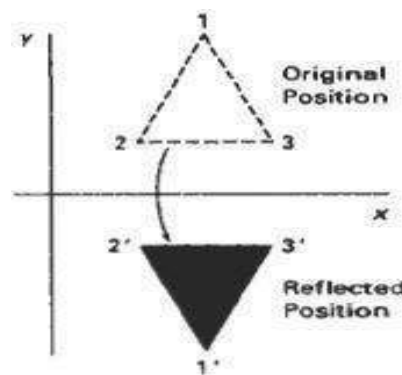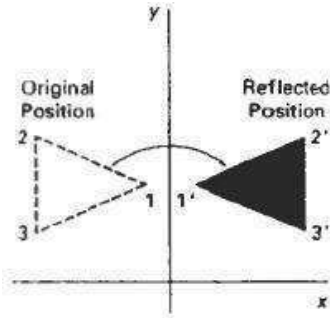

Reflection of an object about the x axis

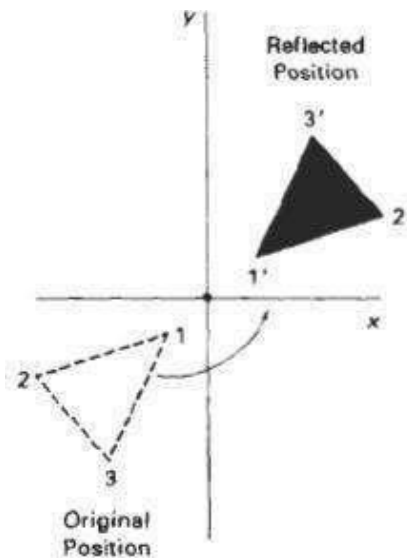Reflection the x axis is accomplished with the transformation matrix

$$
\begin{matrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{matrix}
$$

Reflection of an object about the y axis Reflection they axis is accomplished with the transformation matrix

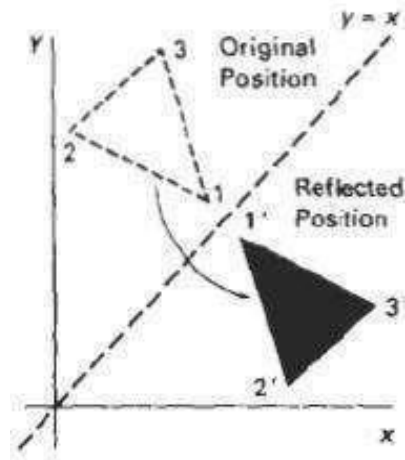$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Reflection of an object about the coordinate origin

Reflection about origin is accomplished with the transformation matrix

$$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Reflection axis as the diagonal line y = x

To obtain transformation matrix for reflection about diagonal y=x the transformation sequence is
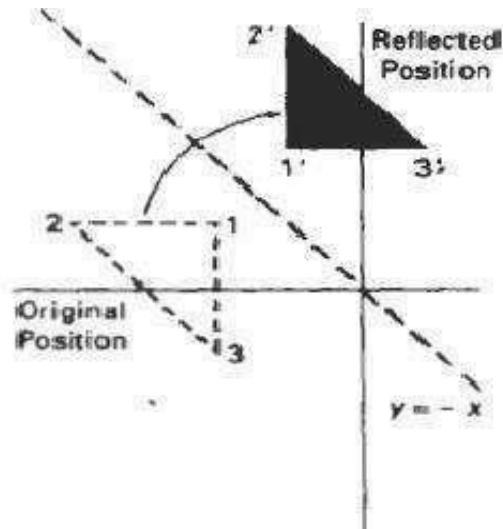
- ➤ Clock wise rotation by45

- ➤ Reflection about x axis

- ➤ Counter clock wise by45

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Reflection about the diagonal line y = x is accomplished with the transformation matrix

Reflection axis as the diagonal line y = -x

To obtain transformation matrix for reflection about diagonal $y = -x$ the transformationsequence is

- ➢ Clock wise rotation by 45

- ➢ Reflection about x axis

- ➢ Counter clock wise by 45

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Reflection about the diagonal line $y = -x$ is accomplished with the transformation matrixShear

A Transformation that slants the shape of an object is called the shear transformation. Two common shearing transformations are used. One shifts x coordinate values and other shifty co- ordinate values. However in both the cases only one co-ordinate (x or y) changes its coordinates and other preserves its values.

(a)　(b)
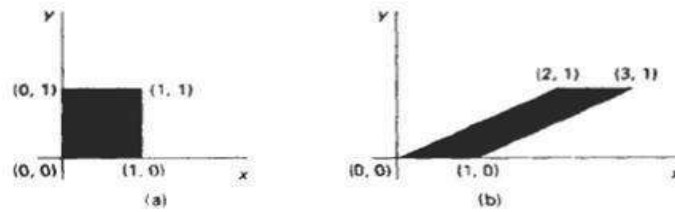
X - Shear

The x shear preserves the y coordinates, but changes the x values which causevertical lines to tilt right or left as shown in figure

The Transformations matrix for x-shear is

$$\begin{bmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

which transforms the coordinates as**x' =x+ shx .y ;y' =y**

Y - Shear

The y shear preserves the x coordinates, but changes the y values which cause horizontal lines which slope up or down The Transformations matrix for y-shear is which transforms the coordinates as**x' = x**

$$\begin{bmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

y' = y + y shx .x

XY - Shear

The transformation matrix for x y-shear

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & sh_x & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Which transforms the coordinates as

x' = x +xshx.yy' = y +yshx

Shearing Relative to other reference line

We can apply x shear and y shear transformations relative to other reference lines. In x shear transformations we can use y reference line and in y shear we can use xreference line.

X - Shear with y reference line

We can generate x-direction shears relative to other reference lines with the transformation matrix

$$\begin{bmatrix} 1 & sh_x & -sh_x \cdot y_{ref} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$
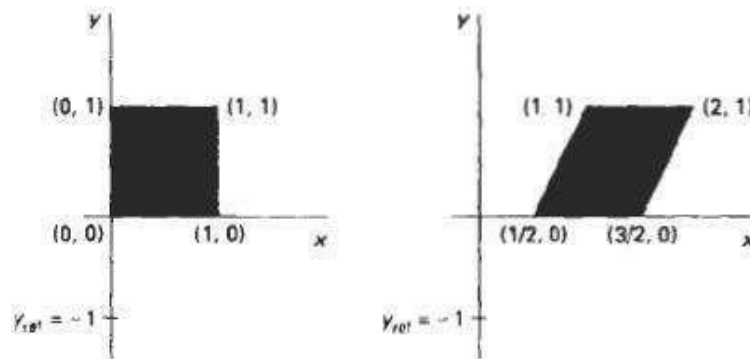
Which transforms the coordinates as

x' = x+xshx (yref y)

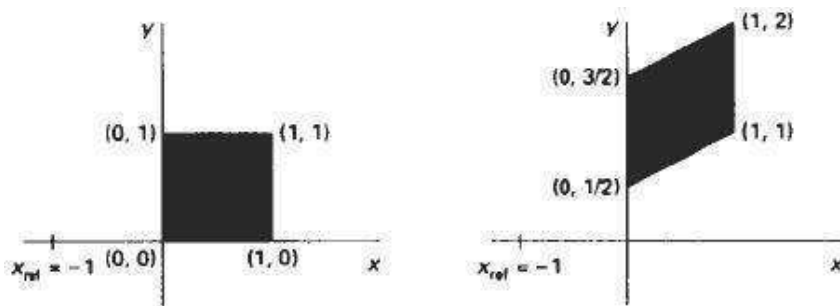**y' = y**

Example   Shx = ½ and Y ref = -1

Y - Shear with x reference line

We can generate y-direction shears relative to other   reference lines   with thetransformation matrix which transforms the coordinates as

$$\begin{bmatrix} 1 & sh_x & -sh_x \cdot y_{ref} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Example x' = x

Y' = shy (x – x ref) + yShy = ½ and x ref = -1

3D Transformation:

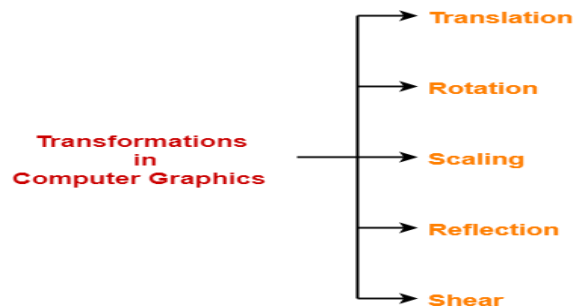In Computer graphics, Transformation is a process of modifying and re-positioning theexisting graphics.

- ➢ 3D Transformations take place in a three dimensional plane.
- ➢ 3D Transformations are important and a bit more complex than 2D Transformations.
- ➢ Transformations are helpful in changing the position, size, orientation,

shape etc. of theobject.

Transformation Techniques:

In Computer Graphics, Various techniques are

- ➢ Translation
- ➢ Rotation
- ➢ Scaling
- ➢ Reflection
- ➢ Shear

In this article, we will discuss about 3D Translation in Computer Graphics.

3D Translation in Computer Graphics:

In Computer graphics, 3D Translation is a process of moving an object from oneposition to another in a three dimensional plane.

Consider a point object O has to be moved from one position to another in a 3D plane.Let

- ➢ Initial coordinates of the object O = $(X_{old}, Y_{old}, Z_{old})$

- ➢ New coordinates of the object O after translation = $(X_{new}, Y_{new}, Z_{old})$

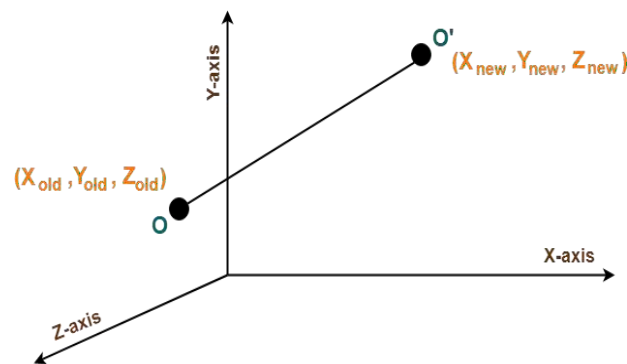- ➢ Translation vector or Shift vector = (T_x, T_y, T_z)

T_x defines the distance the X_{old} coordinate has to be moved.

➤ T_y defines the distance the Y_{old} coordinate has to be moved.

➤ T_z defines the distance the Z_{old} coordinate has to be moved

➤ .

3D Translation in Computer Graphics

This translation is achieved by adding the translation coordinates to the old coordinates of the

Given a Translation vector (T_x, T_y, T_z)-

➤ $X_{new} = X_{old} + T_x$ (This denotes translation towards X axis)

➤ $Y_{new} = Y_{old} + T_y$ (This denotes translation towards Y axis)

➤ $Z_{new} = Z_{old} + T_z$ (This denotes translation towards Z axis)

In Matrix form, the above translation equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

3D Translation Matrix

3D Rotation in Computer Graphics:

In Computer graphics, 3D Rotation is a process of rotating an object with respect to anangle in a three dimensional plane.

Consider a point object O has to be rotated from one angle to another in a 3D plane.

Let-

> Initial coordinates of the object O = $(X_{old}, Y_{old}, Z_{old})$

> Initial angle of the object O with respect to origin = Φ

> Rotation angle = θ

> New co-ordinates of the object O after rotation = $(X_{new}, Y_{new}, Z_{new})$

In 3 dimensions, there are 3 possible types of rotation:

> X-axis Rotation

> Y-axis Rotation

> Z-axis Rotation

For X-Axis Rotation:

This rotation is achieved by using the following rotation equations-

> $X_{new} = X_{old}$

> $Y_{new} = Y_{old} \times \cos\theta - Z_{old} \times \sin\theta$

> $Z_{new} = Y_{old} \times \sin\theta + Z_{old} \times \cos\theta$

In Matrix form, the above rotation equations may be represented as,

For Y-Axis Rotation:

This rotation is achieved by using the following rotation equations-

- $X_{new} = Z_{old} \times \sin\theta + X_{old} \times \cos\theta$

- $Y_{new} = Y_{old}$

- $Z_{new} = Y_{old} \times \cos\theta - X_{old} \times \sin\theta$

In Matrix form, the above rotation equations may be represented as-

$$
\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}
$$

3D Rotation Matrix
(For Y-Axis Rotation)

$$
\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}
$$

3D Rotation Matrix
(For Z-Axis Rotation)

For Z-Axis Rotation:

This rotation is achieved by using the following rotation equations-

- $X_{new} = X_{old} \times \cos\theta - Y_{old} \times \sin\theta$

- $Y_{new} = X_{old} \times \sin\theta + Y_{old} \times \cos\theta$

➢ Znew = Zold

In Matrix form, the above rotation equations may be represented as-

3D Scaling in Computer Graphics:

In computer graphics, scaling is a process of modifying or altering the size of objects.

➢ Scaling may be used to increase or reduce the size of object.

➢ Scaling subjects the coordinate points of the original object to change.

➢ Scaling factor determines whether the object size is to be increased or reduced.

➢ If scaling factor > 1, then the object size is increased.

➢ If scaling factor < 1, then the object size is reduced.

Consider a point object O has to be scaled in a 3D plane. Let:

➢ Initial coordinates of the object O = (Xold, Yold,Zold)

➢ Scaling factor for X-axis = Sx

➢ Scaling factor for Y-axis = Sy

➢ Scaling factor for Z-axis = Sz

➢ New coordinates of the object O after scaling = (Xnew, Ynew, Znew)

This scaling is achieved by using the following scaling equations-

➢ Xnew = Xold x Sx

➢ Ynew = Yold x Sy

➢ Znew = Zold x Sz

In Matrix form, the above scaling equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} S_X & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

3D Scaling Matrix
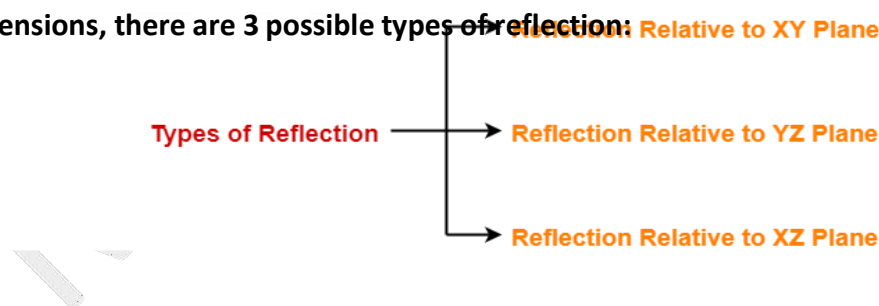
3D Reflection in Computer Graphics-
- Reflection is a kind of rotation where the angle of rotation is 180 degree.
- The reflected object is always formed on the other side of mirror.
- The size of reflected object is same as the size of original object.

Consider a point object O has to be reflected in a 3D plane.
Let,
- Initial coordinates of the object O = $(X_{old}, Y_{old}, Z_{old})$
- New coordinates of the reflected object O after reflection = $(X_{new}, Y_{new}, Z_{new})$

In 3 dimensions, there are 3 possible types of reflection: Relative to XY Plane

Types of Reflection ⟶ Reflection Relative to YZ Plane

⟶ Reflection Relative to XZ Plane

- Reflection relative to XY plane
- Reflection relative to YZ plane
- Reflection relative to XZ plane

Reflection Relative to XY Plane:

This reflection is achieved by using the following reflection equations:

- Xnew = Xold
- Ynew = Yold
- Znew = -Zold

In Matrix form, the above reflection equations may be represented as-

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} X \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

3D Reflection Matrix
(Reflection Relative to XY plane)

Reflection Relative to YZ Plane:

This reflection is achieved by using the following reflection equations-

➢ Xnew = -Xold

➢ Ynew = Yold

➢ Znew = Zold

$$\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} X \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}$$

3D Reflection Matrix
(Reflection Relative to YZ plane)

In Matrix form, the above reflection equations may be represented as-

Reflection Relative to XZ Plane:

This reflection is achieved by using the following reflection equations-

➢ Xnew = Xold

- ➢ $Ynew = -Yold$

- ➢ $Znew = Zold$

In Matrix form, the above reflection equations may be represented as:

$$
\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}
$$

3D Reflection Matrix
(Reflection Relative to XZ plane)

In Computer graphics,
3D Shearing is an ideal technique to change the shape of an existing object in a three dimensional plane

In a three dimensional plane, the object size can be changed along X direction, Y direction as well as Z direction.So, there are three versions of shearing:

Versions of Shearing ⟶ **Shearing in X axis**

⟶ **Shearing in Y axis**
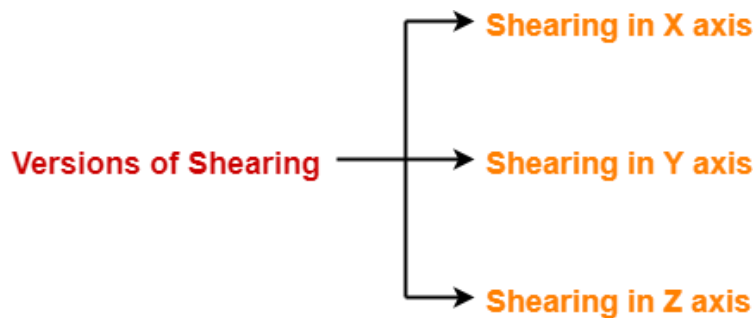
⟶ **Shearing in Z axis**

1. Shearing in X direction
2. Shearing in Y direction
3. Shearing in Z direction

Consider a point object O has to be sheared in a 3D plane. Let:
- ➢ Initial coordinates of the object O = (X_{old}, Y_{old}, Z_{old})
- ➢ Shearing parameter towards X direction = Sh_x
- ➢ Shearing parameter towards Y direction = Sh_y
- ➢ Shearing parameter towards Z direction = Sh_z
- ➢ New coordinates of the object O after shearing = (X_{new}, Y_{new}, Z_{new})

Shearing in X Axis-

3D Shearing in Computer Graphics:

Shearing in X axis is achieved by using the following shearing equations-

- $\rightarrow$ Xnew = Xold
- $\rightarrow$ Ynew = Yold + Shy x Xold
- $\rightarrow$ Znew = Zold + Shz x Xold

In Matrix form, the above shearing equations may be represented as:

$$
\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ Sh_y & 1 & 0 & 0 \\ Sh_z & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}
$$

3D Shearing Matrix
(In X axis)

Shearing in Y Axis-

Shearing in Y axis is achieved by using the following shearing equations:

- $\rightarrow$ Xnew = Xold + Shx x Yold
- $\rightarrow$ Ynew = Yold
- $\rightarrow$ Znew = Zold + Shz x Yold

$$
\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & Sh_x & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & Sh_z & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}
$$

3D Shearing Matrix
(In Y axis)

In Matrix form, the above shearing equations may be represented as-

Shearing in Z Axis-

Shearing in Z axis is achieved by using the following shearing equations-

- ➢ Xnew = Xold + Shx x Zold

- ➢ Ynew = Yold + Shy x Zold

- ➢ Znew = Zold

In Matrix form, the above shearing equations may be represented as-

$$
\begin{bmatrix} X_{new} \\ Y_{new} \\ Z_{new} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & Sh_x & 0 \\ 0 & 1 & Sh_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_{old} \\ Y_{old} \\ Z_{old} \\ 1 \end{bmatrix}
$$

3D Shearing Matrix
(In Z axis)

UNIT – IV

CLIPPING ALGORITHMS

Viewing and clipping

The process of selecting and viewing the picture with different views is called windowing and a process which divides each element of the picture into its visible and invisible portions, allowing the invisible portion to be discarded is called clipping.

Windows and viewports

A world coordinate area selected for display is called a window. An area on a display device to which a window is mapped is called a view port. The window defines what is to be viewed the view port defines where it is to be displayed.

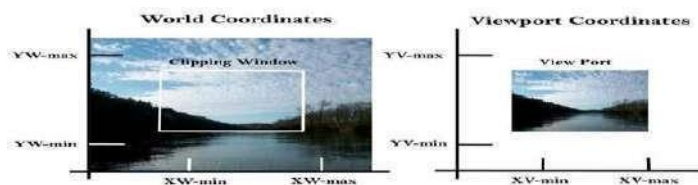A point at position (xw, yw) in window mapped into position (xv,yv) in the associated viewport.

Fig: Window to viewport mapping

The mapping of a part of a world coordinate scene to device coordinate is referred to as viewing transformation. The two dimensional viewing transformation is referred to as window to view port transformation of windowing transformation.

In order to maintain the same relative placement of the point in the viewport as in the window, we require:

$$\frac{xv - xv_{min}}{xv_{max} - xv_{min}} = \frac{xw - xw_{min}}{xw_{max} - xw_{min}} \quad \dots\dots\dots\dots\dots\dots\dots equation\ 1$$

$$\frac{yv - yv_{min}}{yv_{max} - yv_{min}} = \frac{yw - yw_{min}}{yw_{max} - yw_{min}}$$

Solving these impressions for the viewport position (xv, yv), we have

xv=xvmin+(xw-xwmin)sx

yv=yvmin+(yw-ywmin)sy ....equation 2

Where scaling factors are

$$SX = \frac{xv_{max} - xv_{min}}{xw_{max} - xw_{min}}$$
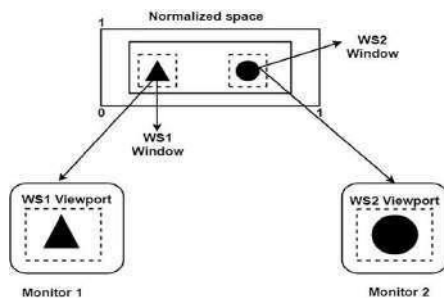
$$sy = \frac{yv_{max} - yv_{min}}{yw_{max} - yw_{min}}$$

Fig:Mapping selected parts of a scene in normalized coordinates to different video monitors with workstation transformation.

Equation (1) and Equation (2) can also be derived with a set of transformation that converts the window or world coordinate area into the viewport or screen coordinate area. This conversation is performed with the following sequence of transformations:

➢ Perform a scaling transformation using a fixed point position (xw min, yw min) that scales the window area to the size of the viewport.

> ➢ Translate the scaled window area to the position of the viewport. Relative proportions of objects are maintained if the scaling factors are the same (sx=sy).

From normalized coordinates, object descriptions are mapped to the various display devices. Any number of output devices can we open in a particular app, and three windows to viewport transformation can be performed for each open output device. This mapping called workstation transformation (It is accomplished by selecting a window area in normalized space and a viewport area in the coordinates of the display device).As in fig, workstation transformation to partition a view so that different parts of normalized space can be displayed on various output devices).

Matrix Representation of the above three steps of Transformation:
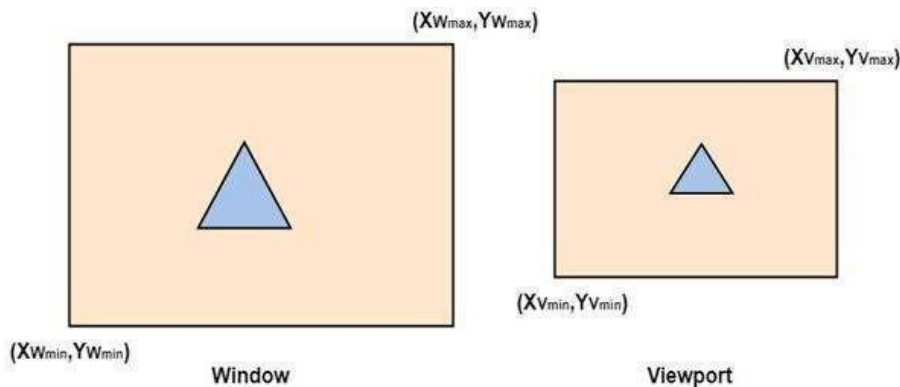
Window   Viewport

Step1: Translate window to origin 1Tx=-Xwmin Ty=-Ywmin

Step2: Scaling of the window to match its size to the viewport Sx=(Xymax-Xvmin)/(Xwmax-Xwmin)

Sy=(Yvmax-Yvmin)/(Ywmax-Ywmin)

Step3: Again translate viewport to its correct position on screen.

Tx=Xvmin Ty=Yvmin

$$T = \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -Xw_{min} & -Yw_{min} & 1 \end{vmatrix}$$

$$S = \begin{vmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{vmatrix}$$

Above three steps can be represented in matrix form:VT=T * S * T1

T = Translate window to the origin S=Scaling of the window to viewport size T1=Translating viewport on screen.

Viewing Transformation= T * S * T1

Advantage of Viewing Transformation:

We can display picture at device or display system according to our need and choice.

Note:

➢ World coordinate system is selected suits according to the application program.

➢ Screen coordinate system is chosen according to the need of design.

➢ Viewing transformation is selected as a bridge between the world and screencoordinate

Line Clipping:

It is performed by using the line clipping algorithm.

The line clipping algorithms are:

➢ Cohen Sutherland Line Clipping Algorithm

➢ Midpoint Subdivision Line Clipping Algorithm

➢ Liang-Barsky Line Clipping Algorithm

Cohen Sutherland Line Clipping Algorithm:

In the algorithm, first of all, it is detected whether line lies inside the screen or it isoutside the screen.

All lines come under any one of the following categories:

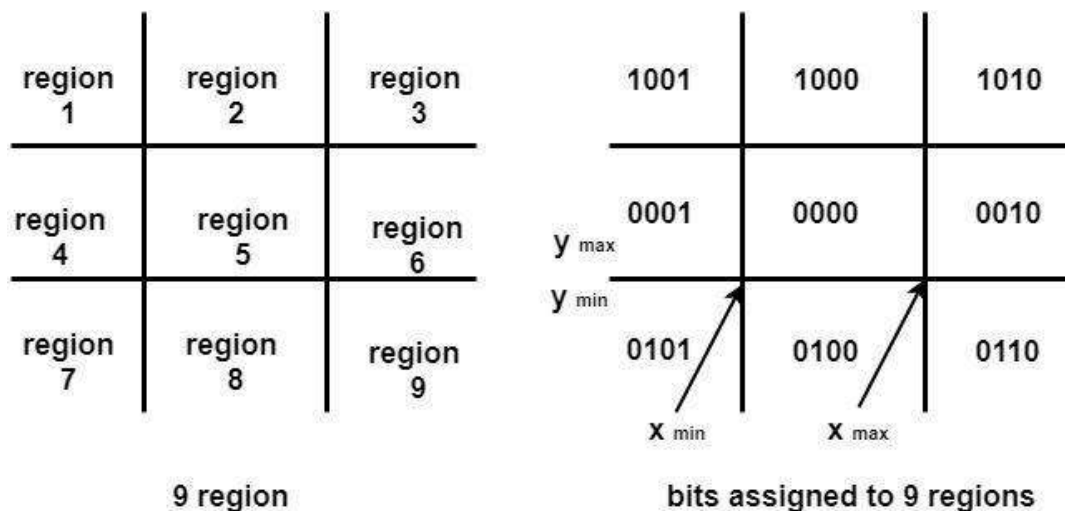➢ Visible

➢ Not Visible

➢ Clipping Case

1. **Visible:**

If a line lies within the window, i.e., both endpoints of the line lies within the window. Aline is visible and will be displayed as it is.

2. **Not Visible:**

If a line lies outside the window it will be invisible and rejected. Such lines will not display. If any one of the following inequalities is satisfied, then the line is considered invisible. Let A $(x1,y2)$ and B $(x2,y2)$ are endpoints of line. x_{min}, x_{max} are coordinates of the window. y_{min}, y_{max} are also coordinates of the window. $x1 > x_{max}$ $x2 > x_{max}$ $y1 > y_{max}$ $y2 > y_{max}$ $x1 < x_{min}$ $x2 < x_{min}$ $y1 < y_{min}$ $y2 < y_{min}$
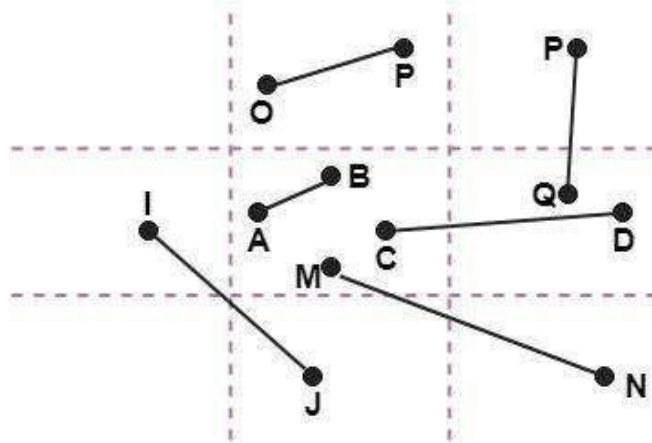
3. **Clipping Case:**

If the line is neither visible case nor invisible case. It is considered to be clipped case. First of all, the category of a line is found based on nine regions given below. All nine regions are assigned codes. Each code is of 4 bits. If both endpoints of the line have end bits zero, then the line is considered to be visible.

9 region   bits assigned to 9 regions

The centre area is having the code, 0000, i.e., region 5 is considered a rectangle window.

Following figure show lines of various types

- ➢ Line AB is the visible case

- ➢ Line OP is an invisible case

- ➢ Line PQ is an invisible line

- ➢ Line IJ are clipping candidates

- ➢ Line MN are clipping candidate

- ➢ Line CD are clipping candidate

Advantage of Cohen Sutherland Line Clipping:

- ➢ It calculates end-points very quickly and rejects and accepts lines quickly.

- ➢ It can clip pictures much large than screen size.

Algorithm of Cohen Sutherland Line Clipping:

Step1: Calculate positions of both endpoints of the line

Step2: Perform OR operation on both of these end-points

Step3: If the OR operation gives 0000 Then line is considered to be visible Else Perform AND operation on both endpoint

If And ≠ 0000

Then the line is invisibleElse

And=0000

Line is considered the clipped case.

Step4: If a line is clipped case, find an intersection with boundaries of the windowm=(y2-y1)(x2-x1)

a) If bit 1 is "1" line intersects with left boundary of rectangle window y3=y1+m(x-X1)

where X = Xwmin

where Xwminis the minimum value of X co-ordinate of window

b) If bit 2 is "1" line intersect with right boundaryy3=y1+m(X-X1)

where X = Xwmax

where X more is maximum value of X co-ordinate of the window

c) If bit 3 is "1" line intersects with bottom boundaryX3=X1+(y-y1)/m

where y = ywmin

ywmin is the minimum value of Y co-ordinate of the window

**d)** If bit 4 is "1" line intersects with the top boundaryX3=X1+(y-y1)/m

where y = ywmax

ywmax is the maximum value of Y co-ordinate of the window

Mid-Point Subdivision Line Clipping Algorithm:

It is used for clipping line. The line is divided in two parts. Mid points of line is obtained by dividing it in two short segments. Again division is done, by finding midpoint. This process is continued until line of visible and invisible category is obtained. Let (x_i, y_i) are midpoint.

$$x_m = \frac{x_1 + x_2}{2} \quad y_m = \frac{y_1 + y_2}{2}$$
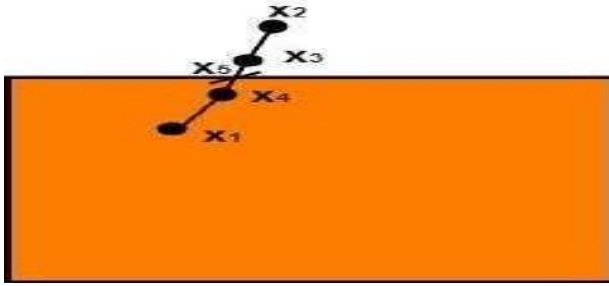
Step1: Find $\frac{x_2 + x_1}{2}$ i.e. $X_3 = \frac{x_2 + x_1}{2}$

Step2: Find $X_4 = \frac{x_3 + x_1}{2}$

Step3: Find $X_5 = \frac{x_4 + x_3}{2}$

x5 lie on point of intersection of boundary of window.

Advantage of midpoint subdivision Line Clipping:

It is suitable for machines in which multiplication and division operation is not possible. Because it can be performed by introducing clipping divides in hardware.

Algorithm of midpoint subdivision Line Clipping:

Step1: Calculate the position of both endpoints of the line

Step2: Perform OR operation on both of these endpoints

Step3: If the OR operation gives 0000 then Line is guaranteed to be visibleelse Perform AND operation on both endpoints.

If AND $\neq$ 0000

Then the line is invisibleelse

AND=6000

Then the line is clipped case.

Step4: For the line to be clipped. Find midpointXm=(x1+x2)/2

$Y_m=(y1+y2)/2$

X_m is midpoint of X coordinate. Y_m is midpoint of Y coordinate.

Step5: Check each midpoint, whether it nearest to the boundary of a window or not.

Step6: If the line is totally visible or totally rejected not found then repeat step 1 to 5.

Step7: Stop algorithm.

Example: Window size is (-3, 1) to (2, 6). A line AB is given having co-ordinates of A (-4, 2) and B (-1, 7). Does this line visible. Find the visible portion of the line using midpoint subdivision?
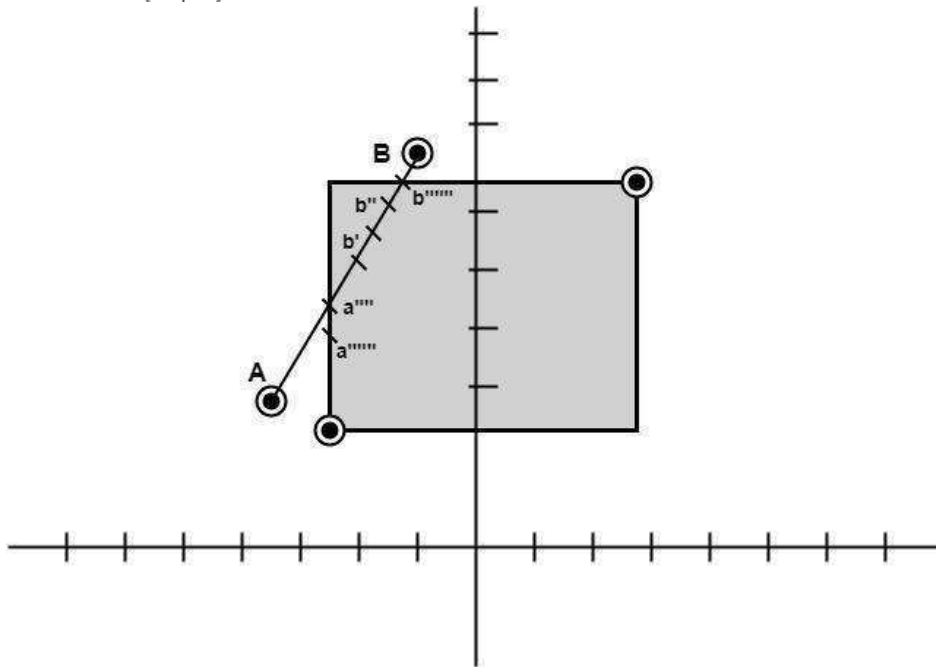
Solution:

Step1: Fix point A (-4, 2)

$$b = \left(\frac{-4+(-1)}{2}, \frac{2+7}{2} \right) = \frac{-5}{2}, \frac{9}{2} = (-2,4)$$

Step2: Find b"=mid of b'and b

$$b" = \left(\frac{-2+(-1)}{2}, \frac{4+7}{2}\right)$$

$$b" = (-1, 5)$$

So (-1, 5) is better than (2, 4)

Find b"&bb"(-1, 5) b (-1, 7)

$$b"" = \left(\frac{-1+(-1)}{2}, \frac{5+7}{2}\right)$$

$$b = (-1,6)$$

So B""to B length of line will be clipped from upper side Now considered left-hand side portion

A and B""are now endpoints

Find mid of A and B""

A (-4, 2) B ""(-1, 6)

$$a' = \left(\frac{-4+(-1)}{2}, \frac{2+6}{2}\right)$$

$$a' = (-2.5, 4)$$

$$a' = (-2, 4)$$

Now good a (-4, 2) and a'(−2,4)

$$a'' = \left(\frac{-4+(-2)}{2}, \frac{2+4}{2}\right)$$

$$a'' = (-3, 3)$$

Now find mid of a" and a

a""""=a (-4, 2) and a"(−3, 3)

$$= \left(\frac{-4+(-3)}{2}, \frac{2+3}{2}\right)$$

$$= \left(\frac{-7}{2}, \frac{5}{2}\right)$$

$$= (-3.5, 2.5)$$

a""""= (-3, 2)
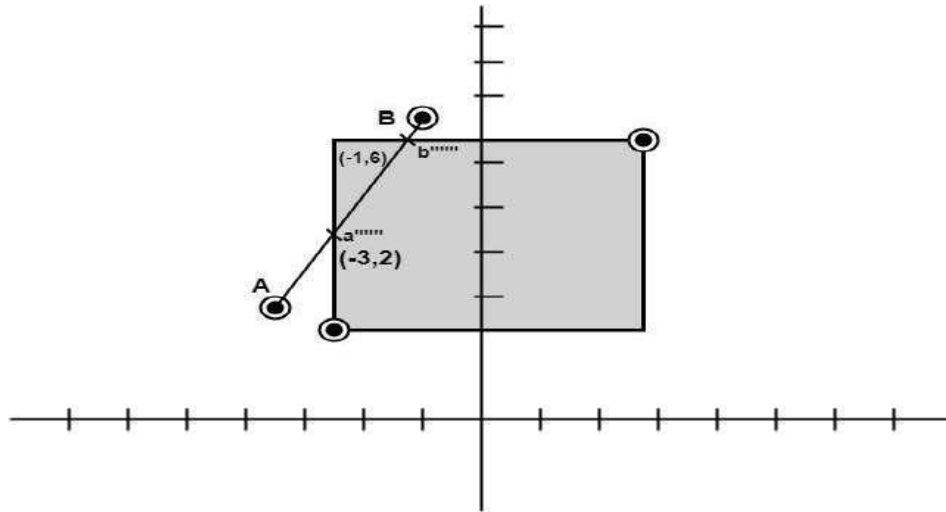
So line from A to a""""will clipped

Line after clipping from both sides will be a""to b""

a""""= (-1, 6)

b""""= (-3, 2)

Sutherland-Hodgeman Polygon Clipping Algorithm:-

A polygon can be clipped by processing its boundary as a whole against each window edge. This is achieved by processing all polygon vertices against each clip rectangle boundary in turn. Beginning with the original set of polygon vertices, we could first clip the polygon against the left rectangle boundary to produce a new sequence of vertices. The new set of vertices could then be successively passed to a right boundary clipper, a top boundary clipper and a bottom boundary clipper, as shown in figure (l). At each step a new set of polygon vertices is generated and passed to the next window boundary clipper. This is the fundamental idea used in the Sutherland - Hodgeman algorithm.

The output of the algorithm is a list of polygon vertices all of which are on the visible side of a clipping plane. Such each edge of the polygon is individually compared with the clipping plane. This is achieved by processing two vertices of each edge of the polygon around the clipping boundary or plane. This results in four possible relationships between the edge and the clipping boundary or Plane. (See Fig. m).

If the first vertex of the edge is outside the window boundary and the second vertex of the edge is inside then the intersection point of the polygon edge with the window boundary and the second vertex are added to the output vertex list (See Fig. m (a)).
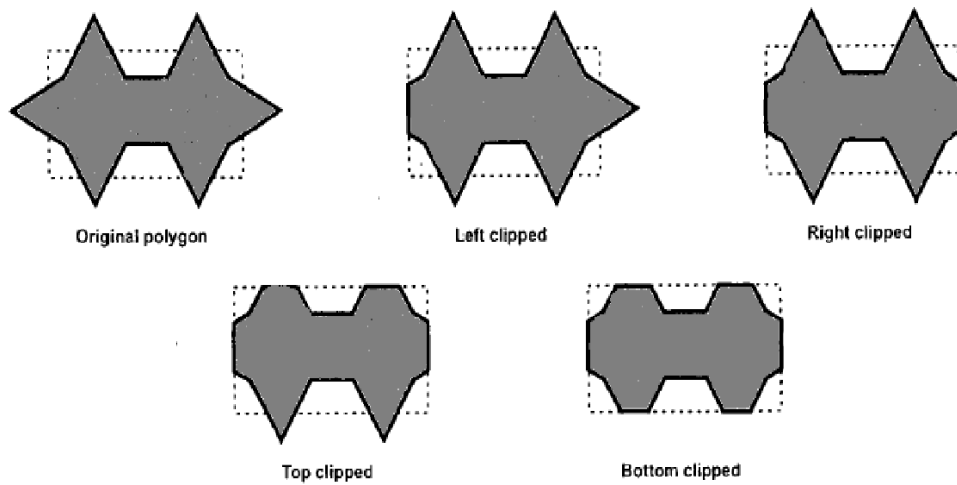

Fig. (l) Clipping a polygon against successive window boundaries

> ➢ If both vertices of the edge are inside the window boundary, only the second vertex is added to the output vertex list. (See Fig. m (b)).

> ➢ If the first vertex of the edge is inside the window boundary and the second vertex of the edge is outside, only the edge intersection with the window boundary is added to the output vertex list. (See Fig. m (c)).

> ➢ If both vertices of the edge are outside the window boundary, nothing is added to the output list. (See Fig. m (d)).

Once all vertices are processed for one clip window boundary, the output list of vertices is clipped against the next window boundary. Going through above four cases we can realize that there are two key processes in this algorithm.

V_1 – Outside
V_2 – Inside
$\therefore$ Save V_1' and V_2

(a)

V_1 – Inside
V_2 – Inside
$\therefore$ Save V_2

(b)

V_1 – Inside
V_2 – Outside
$\therefore$ Save V_1'

(c)

V_1 – Outside
V_2 – Outside
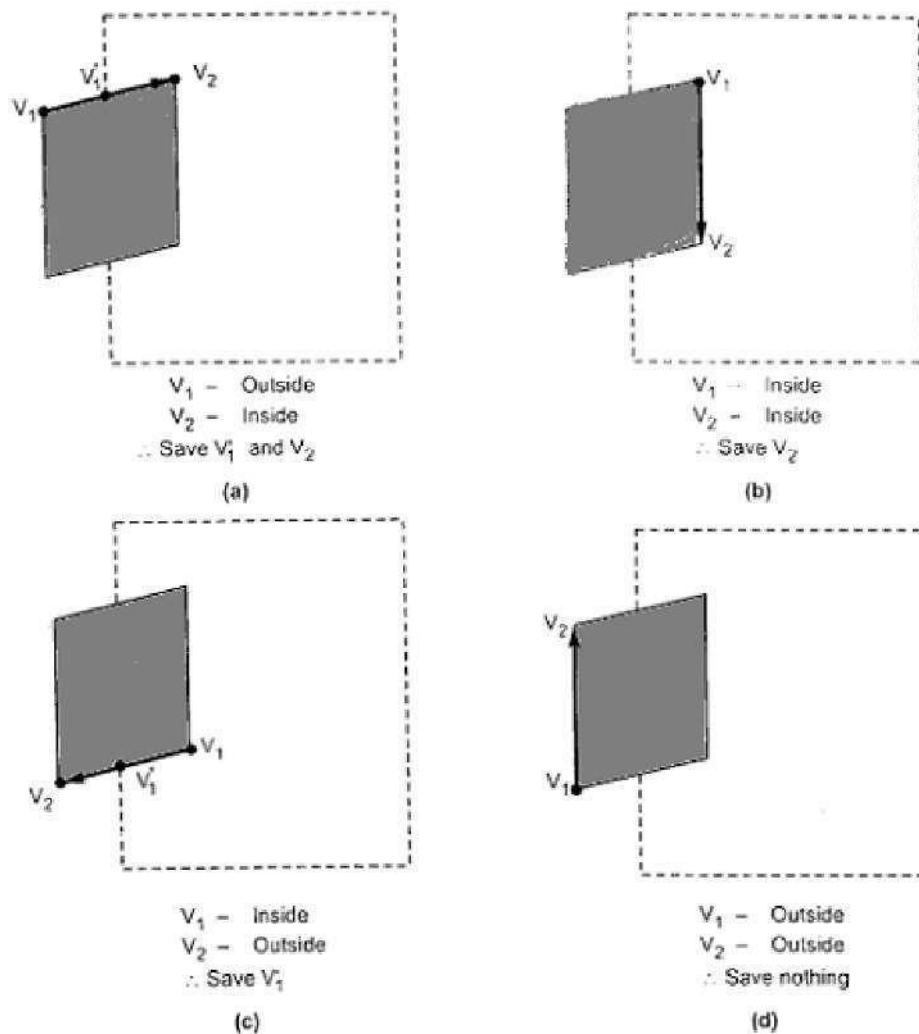$\therefore$ Save nothing

(d)

Fig. (m) Processing of edges of the polygon against the left window boundary

➢ Determining the visibility of a point or vertex (lnside - Outside test) and

➢ Determining the intersection of the polygon edge and the clipping plane.

One way of determining the visibility of a point or vertex is described here. Consider that two points A and B define the window boundary and point under consideration is V, then these three points define a plane. Two vectors which lie in that plane are AB and AV. If this plane is considered in the xy plane, then the vector cross product AV x AB has only a component given by

$$(x_V - x_A) \, (y_B - y_A) - (y_V - y_A) \, (x_B - x_A)$$

The sign of the z component decides the position of Point V with respect to windowboundary.

If z is:

Positive   - Point is on the right side of the window boundary.

Zero   - Point is on the window boundary.

Negative   - Point is on the left side of the window boundary.

Sutherland-Hodgeman Polygon Clipping Algorithm:-

- ➢ Read coordinates of all vertices of the Polygon.

- ➢ Read coordinates of the dipping window

- ➢ Consider the left edge of the window

- ➢ Compare the vertices of each edge of the polygon, individually with the clipping plane.

- ➢ Save the resulting intersections and vertices in the new list of vertices according to four possible relationships between the edge and the clipping boundary.

- ➢ Repeat the steps 4 and 5 for remaining edges or the clipping window. Each time the resultant list of vertices is successively passed to process the next edge of the clipping window.

- ➢ Stop.

Example:

For a polygon and clipping window shown in figure below give the list of vertices after eachboundary clipping.
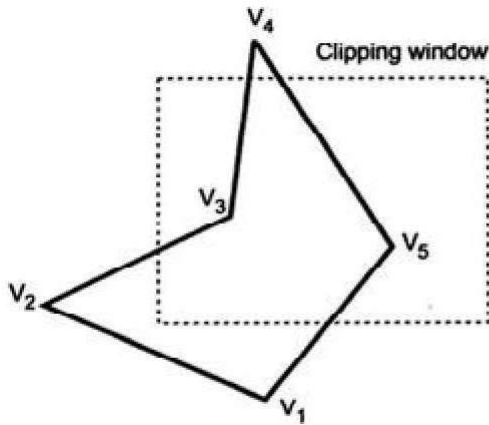
Fig. Before Clipping
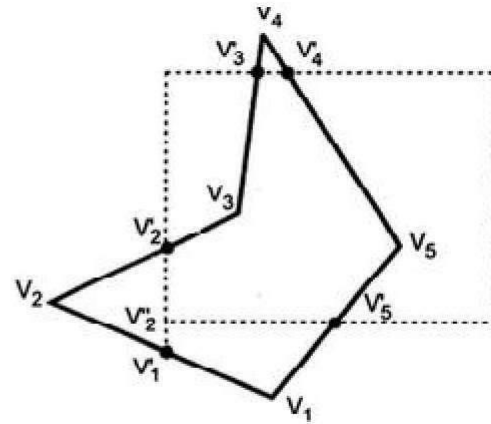
Fig. After Boundary Clipping

Solution:

Original polygon vertices are V1, V2, V3, V4, and V5. After clipping each boundary thenew vertices are as shown in figure above.

After left clipping	:	$V_1, V_1', V_2', V_3, V_4, V_5$
After right clipping	:	$V_1, V_1', V_2', V_3, V_4, V_5$
After top clipping	:	$V_1, V_1', V_2', V_3, V_3', V_4', V_5$
After bottom clipping	:	$V_2'', V_2', V_3, V_3', V_4', V_5, V_5'$

UNIT V

HIDDEN SURFACE ALGORITHMS

Hidden Surface Elimination

1. One of the most challenging problems in computer graphics is the removal of hiddenparts from images of solid objects.

2. In real life, the opaque material of these objects obstructs the light rays from hiddenparts and prevents us from seeing them.

3. In the computer generation, no such automatic elimination takes place when objectsare projected onto the screen coordinate system.

4. Instead, all parts of every object, including many parts that should be invisible aredisplayed.

5. To remove these parts to create a more realistic image, we must apply a hidden line orhidden surface algorithm to set of objects.

6. The algorithm operates on different kinds of scene models, generate various forms ofoutput or cater to images of different complexities.

7. All use some form of geometric sorting to distinguish visible parts of objects from thosethat are hidden.

8. Just as alphabetical sorting is used to differentiate words near the beginning of thealphabet from those near the ends.

9. Geometric sorting locates objects that lie near the observer and are therefore visible.

10. Hidden line and Hidden surface algorithms capitalize on various forms of coherence toreduce the computing required to generate an image.

11. Different types of coherence are related to different forms of order or regularity in theimage.

12. Scan line coherence arises because the display of a scan line in a raster image is usuallyvery similar to the display of the preceding scan line.

13. **Frame coherence** in a sequence of images designed to show motion recognizes thatsuccessive frames are very similar.

14. **Object coherence** results from relationships between different objects or betweenseparate parts of the same objects.

15. A hidden surface algorithm is generally designed to exploit one or more of thesecoherence properties to increase efficiency.

16. Hidden surface algorithm bears a strong resemblance to two-dimensional scanconversions.

Types of hidden surface detection algorithms:

➢ Object space methods

➢ Image space methods

Object space methods:

In this method, various parts of objects are compared. After comparison visible, invisible or hardly visible surface is determined. These methods generally decide visible surface. In the wireframe model, these are used to determine a visible line. So these algorithms are line based instead of surface based. Method proceeds by determination of parts of an object whose view is obstructed by other object and draws these parts in the same color.

Image space methods:

Here positions of various pixels are determined. It is used to locate the visible surface instead of a visible line. Each point is detected for its visibility. If a point is visible, then the pixel is on, otherwise off. So the object close to the viewer that is pierced by a projector through a pixel is determined. That pixel is drawn is appropriate color.

These methods are also called a **Visible Surface Determination**. The implementation of these methods on a computer requires a lot of processing time and processing power of the computer.

The image space method requires more computations. Each object is defined clearly.

Visibility of each object surface is also determined.

**Differentiate between Object space and Image space method:**

Object Space	Image Space
➢ Image space is object based. It concentrates on geometrical relation among objects in the scene.	➢ It is a pixel-based method. It is concerned with the final image, what is visible within each raster pixel.
➢ Here surface visibility is determined.	➢ Here line visibility or point visibility isdetermined.
➢ It is performed at the precision with which each object is defined, No resolution is considered.	➢ It is performed using the resolution of thedisplay device.
➢ Calculations are not based on the resolution of the display so change of object can be easily adjusted.	➢ Calculations are resolution base, so the change is difficult to adjust.
➢ These were developed for vector graphicssystem.	➢ These are developed for raster devices.
➢ Object-based algorithms operate oncontinuous object data.	➢ These operate on object data.
➢ Vector display is used for object method haslarge address space.	➢ Raster systems is used for image space methods have limited address space.
➢ Object precision is used for application where speed is required.	➢ There are suitable for application where accuracy is required.
➢ It requires a lot of calculations if the image isto enlarge.	➢ Image can be enlarged without losingaccuracy.
➢ If the number of objects in the scene increases, computation time also will increase.	➢ In this method, complexity increase with the complexity of visible parts.

Similarity of object and Image space method

In both method sorting is used a depth comparison of individual lines, surfaces areobjected to their distances from the view plane.
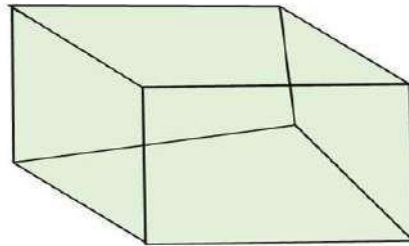
Object with hidden line
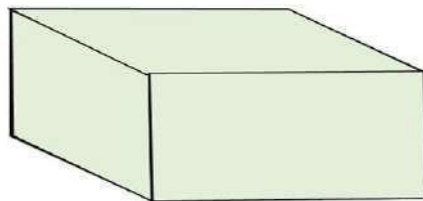
Object when hidden
lines removed

Considerations for selecting or designing hidden surface algorithms: Following three considerations are taken:

- ➢ Sorting

- ➢ Coherence

- ➢ Machine

1. Sorting:

All surfaces are sorted in two classes, i.e., visible and invisible. Pixels are coloredaccordingly.

Several sorting algorithms are available i.e.

- ➢ Bubble sort

- Shell sort

- Quick sort

- Tree sort

- Radix sort

Different sorting algorithms are applied to different hidden surface algorithms. Sorting of objects is done using x and y, z co-ordinates. Mostly z coordinate is used for sorting. The efficiency of sorting algorithm affects the hidden surface removal algorithm. For sorting complex scenes or hundreds of polygons complex sorts are used, i.e., quick sort, tree sort, radix sort.

For simple objects selection, insertion, bubble sort is used.

Coherence

It is used to take advantage of the constant value of the surface of the scene. It is based on how much regularity exists in the scene. When we moved from one polygon of one object to another polygon of same object color and shearing will remain unchanged.

Types of Coherence

- Edge coherence

- Object coherence

- Face coherence

- Area coherence

- Depth coherence

- Scan line coherence

- Frame coherence

- Implied edge coherence

a) **Edge coherence:**

The visibility of edge changes when it crosses another edge or it also penetrates a visible edge.

b) Object coherence:

Each object is considered separate from others. In object, coherence comparison is done using an object instead of edge or vertex. If A object is farther from object B, then there is no need to compare edges and faces.

c) Face coherence:

In this faces or polygons which are generally small compared with the size of the image.

d) Area coherence:

It is used to group of pixels cover by same visible face.

e) Depth coherence:

Location of various polygons has separated a basis of depth. Depth of surface at one point is calculated, the depth of points on rest of the surface can often be determined by a simple difference equation.

f) Scan line coherence:

The object is scanned using one scan line then using the second scan line. The interceptof the first line.

g) Frame coherence:

It is used for animated objects. It is used when there is little change in image from oneframe to another.

h) Implied edge coherence:

If a face penetrates in another, line of intersection can be determined from two pointsof intersection.

Algorithms used for hidden line surface detection

➢ Back Face Removal Algorithm

➢ Z-Buffer Algorithm

i. Back Face Removal Algorithm

It is used to plot only surfaces which will face the camera. The objects on the back side are not visible. This method will remove 50% of polygons from the scene if the parallel projection is used. If the perspective projection is used

then more than 50% of the invisible areawill be removed. The object is nearer to the center of projection, number of polygons from the back will be removed.

It applies to individual objects. It does not consider the interaction between various objects. Many polygons are obscured by front faces, although they are closer to the viewer, so for removing such faces back face removal algorithm is used.

When the projection is taken, any projector ray from the center of projection through viewing screen to object pieces object at two points, one is visible front surfaces, and another isnot visible back surface.

This algorithm acts a preprocessing step for another algorithm. The back face algorithm can be represented geometrically. Each polygon has several vertices. All vertices are numbered in clockwise. The normal M_1 is generated a cross product of any two successive edge vectors. M_1represent vector perpendicular to face and point outward from polyhedron surface

$N_1 = (v_2 - v_1)(v_3 - v_2)$

If $N_1.P \geq 0$ visible

$N_1.P < 0$ invisible

Advantage

 ➢ It is a simple and straight forward method.

 ➢ It reduces the size of databases, because no need of store all surfaces in the database,only the visible surface is stored.

Repeat for all polygons in the scene.

 ➢ Do numbering of all polygons in clockwise direction i.e.v_1 v_2 v_3 v_z

 ➢ Calculate normal vector i.e. N_1 $N_1 = (v_2 - v_1) * (v_3 - v_2)$

 ➢ Consider projector P, it is projection from any vertex Calculate dot product Dot=N.P

 ➢ Test and plot whether the surface is visible or not.

If Dot $\geq$ 0 then surface is visibleelse

Not visible

ii. Z-Buffer Algorithm

It is also called a Depth Buffer Algorithm. Depth buffer algorithm is simplest image space algorithm. For each pixel on the display screen, we keep a record of the depth of an object within the pixel that lies closest to the observer. In addition to depth, we also record the intensity that should be displayed to show the object. Depth buffer is an extension of the frame buffer. Depth buffer algorithm requires 2 arrays, intensity and depth each of which is indexed by pixel coordinates (x, y).

Algorithm

For all pixels on the screen, set depth [x, y] to 1.0 and intensity [x, y] to a background value.

For each polygon in the scene, find all pixels (x, y) that lie within the boundaries of a polygon when projected onto the screen. For each of these pixels:

- ➢ Calculate the depth z of the polygon at (x, y)

- ➢ If z < depth [x, y], this polygon is closer to the observer than others already recorded for this pixel. In this case, set depth [x, y] to z and intensity [x, y] to a value corresponding to polygon's shading. If instead z > depth [x, y], the polygon already recorded at (x, y) lies closer to the observer than does this new polygon, and no action is taken.

- ➢ After all, polygons have been processed; the intensity array will contain the solution.

- ➢ The depth buffer algorithm illustrates several features common to all hidden surface algorithms.
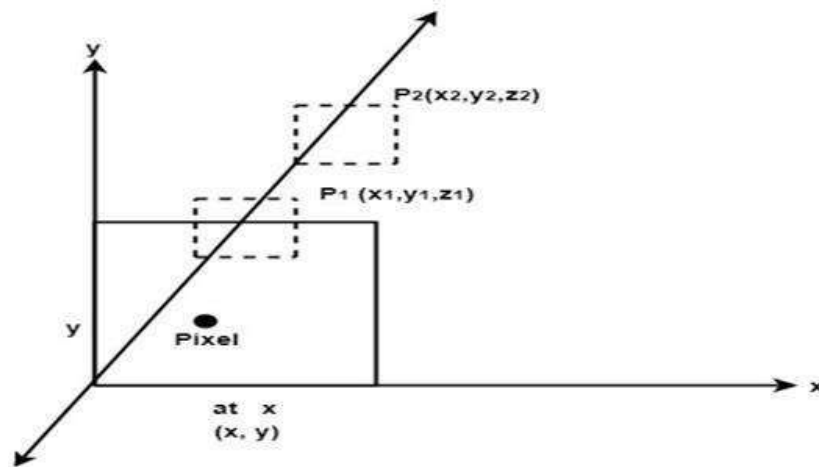
- ➢ First, it requires a representation of all opaque surface in scene polygon in this case.

- ➢ These polygons may be faces of polyhedral recorded in the model of scene or may simply represent thin opaque 'sheets' in the scene.

- ➢ The IInd important feature of the algorithm is its use of a screen coordinate system. Before step 1, all polygons in the scene are transformed into a screen coordinate system using matrix multiplication.

Limitations of Depth Buffer

- ➢ The depth buffer Algorithm is not always practical because of the enormous size ofdepth and intensity arrays.

- ➢ Generating an image with a raster of 500 x 500 pixels requires 2, 50,000 storagelocations for each array.

- ➢ Even though the frame buffer may provide memory for intensity array, the depth arrayremains large.

- ➢ To reduce the amount of storage required, the image can be divided into many smallerimages, and the depth buffer algorithm is applied to each in turn.

- ➢ For example, the original 500 x 500 faster can be divided into 100 rasters each 50 x 50pixels.

- ➢ Processing each small raster requires array of only 2500 elements,

but execution time grows because each polygon is processed many times.

➢ Subdivision of the screen does not always increase execution time instead it can helpreduce the work required to generate the image. This reduction arises because of coherence between small regions of the screen.