COMPUTER GRAPHICS AND MULTIMEDIA
LECTURE NOTES

Overview of Computer Graphics System

Computer graphics is commonly seen as a computer science branch that deals with the computerized image fusion theory and technology. As simple as a triangle outline, a computer-generated image may represent a scene. The computer has become a powerful tool for producing images quickly and economically.

When a computer is used to create images, the same process is followed as creating images manually. The process's primary computational steps give a boost to several important computer graphics areas.

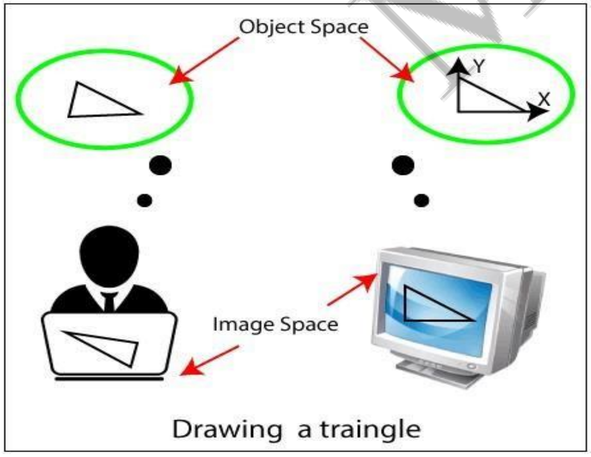
Also on computers, the term computer graphics covers almost everything. Here in the computer graphics program's classroom, we think of computer graphics as drawing images on machines, often known as **rendering**. The images can be photos, sketches, animations, or pictures of items imagined. Or they may be pictures, we cannot see directly, like internal body parts.

We have put a great deal of our time to develop how computer images can replicate real-world scenes. We want objects on computers not only to look more real, but also their colors to be more realistic and how different materials appear. We can call it "real synthesis of the image."

The term computer graphics has been used to define "almost everything on the computer, including text or sound." Generally, the term computer graphics refer to the following things:

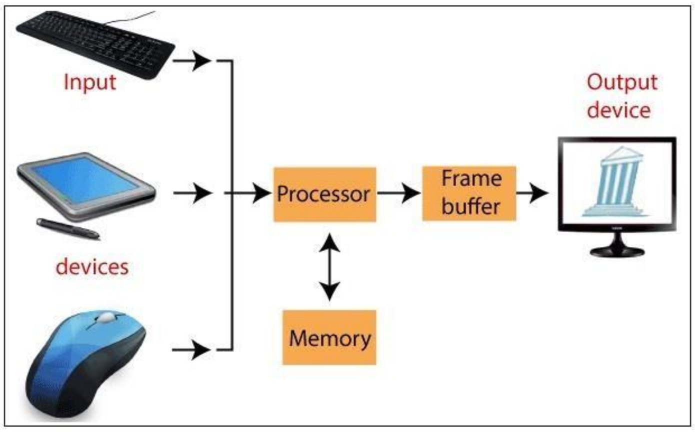
- Computer representation and manipulation of image data.
- Various technologies for creating and manipulating images.
- Computer graphics study is a sub-field of computer science that studies methods for digitally incorporating and manipulating visual content.

The next area of computer graphics that deals with the placement of a triangle is called **transformation**. Here we can use matrices to get the mapping of a triangle in image space. We can also set up the transformation matrix to control the location and orientation of the displayed image. We can also resize the triangle.



Definition of Computer Graphics-Computer graphics can be a series of images which is most often called a video or single image. Computer graphics is the technology that concerns with designs and pictures on computers. That's why, computer graphics are visual representations of data shown on a monitor made on a computer.

"Computer graphics is the use of a computer to define, store, manipulate, interrogate, and represent the pictorial output." An image in computer graphics is made up of a number of pixels.



Types of Computer Graphics

There are two kinds of computer graphics are–

- Interactive Computer Graphics
- Non-Interactive Computer Graphics

Interactive Computer Graphics

In interactive computer graphics, users have some controls over the image, i.e., the user can make any changes to the image produced.

Interactive Computer Graphics involves computer-user two-way communication.

For Example:

- Ping-pong game.
- Drawing on touch screens.
- Display weather forecast or other moving charts/graphs on the screen.
- Animating pictures or graphics in movies.
- Graphics animation in video games.

Working of Interactive Computer Graphics

The modern display of graphics is very simple to build. It is composed of three components:

- Display controller or video controller
- Digital memory or frame buffer
- Television monitor

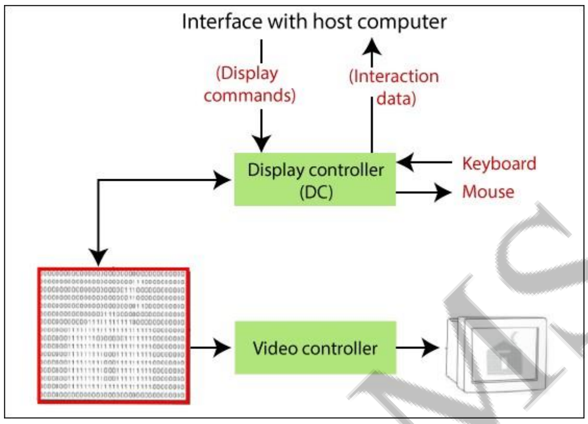
1. Display controller or video controller- It’s a Memory Buffer and TV Monitor interface. Its task is to pass Frame Buffer’s contents to the monitor. The display controller reads each continuous byte of Memory frame buffer data and converts 0’s and 1’s into appropriate video signals.

In today’s term, the display controller is recognized as a display card, and one of our choices can be a VGA(Video Graphics Array) card with a resolution of 640×480. Display Controller is also capable of displaying the image in colors.

2. Digital memory or frame buffer-This is a place where images and pictures are stored as an array (matrix of 0 & 1, 0 represents darkness, and 1 represents image or picture). It is also called a frame buffer.

In today’s term frame buffer is called V-RAM (video RAM), and it helps to store the image in bit form. It helps to increase the speed of graphics.

3. Television monitor- Monitor helps us to view the display, and they make use of CRT(Cathode ray tube) technology.



Advantages

1. Superior Quality.
2. More accurate outcomes or products.
3. Increased Productivity.
4. Lower cost of development.
5. Increases the ability to understand information and interpret patterns significantly.

Non- Interactive Computer Graphics

Non-interactive computer graphics are also known as passive computer graphics. It is a type of computer graphics in which the user has no control over the image. The photo is completely controlled by the instructions of the program, not by the user.

For Example:

- Screen savers.
- Map representation of the data.
- Graphic elements are used in the text, document, and PDF presentation.
- Static images are used in mobile applications and websites.
- Business graphics are used as brochures, business cards, menu of the hotel.

Representation of graphics

We can represent the graphics by following two ways:

1. Raster (Bitmap) Graphics
2. Vector Graphics

1. Raster Graphics: In raster graphics, the image is presented as a rectangular grid of colored squares.

Raster images are also called bitmap images. Bitmap images are stored as the collection of small individual dots called pixels.

Bitmap images require high resolution and anti-aliasing for a smooth appearance.

For example– Paint, Photoshop, etc.

2. Vector Graphics: In vector graphics, the image is represented in the form of continuous geometric objects: line, curve, etc.

Vector images are not based on pixel pattern. They use mathematical formulas to draw line and curves. The lines and curves can be combined to create an image.

For Example– PowerPoint, Corel Draw, etc.

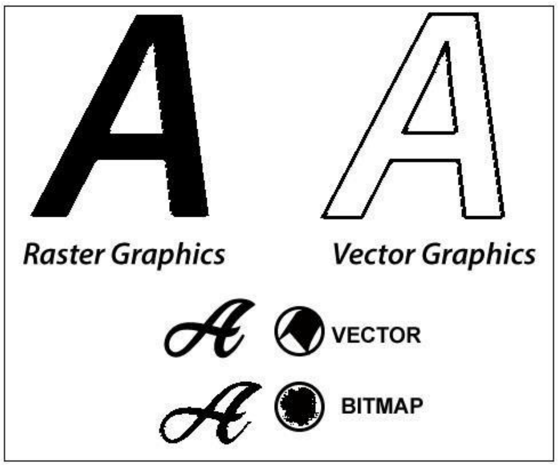


Fig: Raster (Composition of Paths) Vector (Composition of Pixels)

Difference between Raster and Vector Graphics:

Raster Graphics	Vector Graphics
Raster images are the collection of the pixel.	The Vector images are composed of paths.
Scan conversion is required.	Scan Conversion is not required.
Raster Graphics are less costly.	Vector Graphics are more costly compared to raster graphics.
Raster image takes less space to store.	Vector image takes more space.
Raster graphics can draw mathematical curves, polygons, and boundaries.	Vector graphics can only draw continuous and smooth lines.
File Extension: .BMP, .TIF, .JPG etc.	**File Extension:** .SVG, .PDF, .AI etc.

Computer graphics is an art of drawing pictures on computer screens with the help of programming. It involves computations, creation, and manipulation of data. In other words, we can say that computer graphics is a rendering tool for the generation and manipulation of images.

Video Display Devices

Cathode Ray Tube

CRT stands for Cathode ray tube. It is a technology which is used in traditional computer monitor and television.

Cathode ray tube is a particular type of vacuum tube that displays images when an electron beam collides on the radiant surface.

The primary output device in a graphical system is the video monitor.

Component of CRT:

- **Electron Gun:** The electron gun is made up of several elements, mainly a heating filament (heater) and a cathode.

The electron gun is a source of electrons focused on a narrow beam facing the CRT.

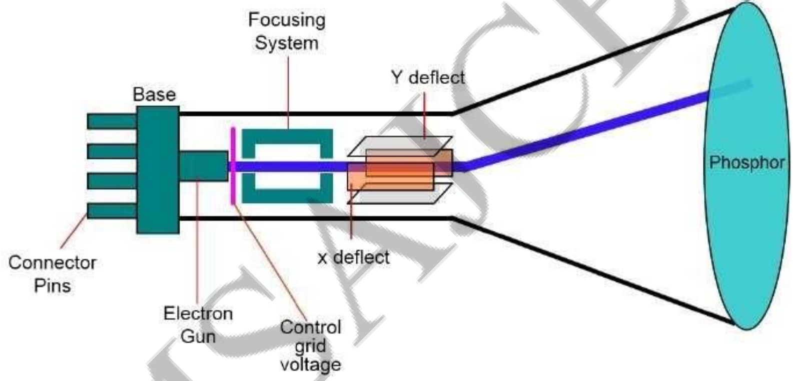
- **Focusing & Accelerating Anodes:** These anodes are used to produce a narrow and sharply focused beam of electrons.

- **Horizontal & Vertical Deflection Plates:** These plates are used to guide the path of the electron the beam. The plates produce an electromagnetic field that bends the electron beam through the area as it travels.
- **Phosphorus-coated Screen:** The phosphorus coated screen is used to produce bright spots when the high-velocity electron beam hits it.

The main element of a video monitor is the **Cathode Ray Tube** CRT**,** shown in the following illustration.

The operation of CRT is very simple −

- The electron gun emits a beam of electrons cathode rays.
- The electron beam passes through focusing and deflection systems that direct it towards specified positions on the phosphor-coated screen.
- When the beam hits the screen, the phosphor emits a small spot of light at each position contacted by the electron beam.
- It redraws the picture by directing the electron beam back over the same screen points quickly.



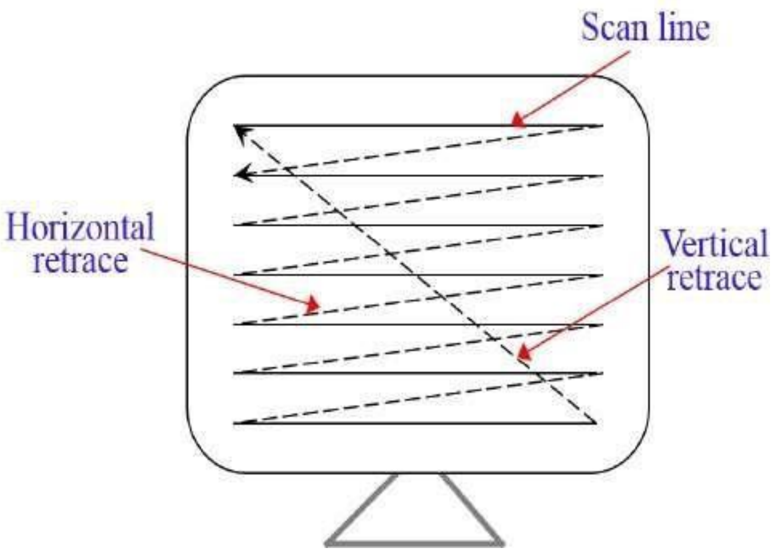
There are two ways Random scan and Raster scan by which we can display an object on the screen.

Raster Scan

In a raster scan system, the electron beam is swept across the screen, one row at a time from top to bottom. As the electron beam moves across each row, the beam intensity is turned on and off to create a pattern of illuminated spots.

Picture definition is stored in memory area called the **Refresh Buffer** or **Frame Buffer**. This memory area holds the set of intensity values for all the screen points. Stored intensity values are then retrieved from the refresh buffer and "painted" on the screen one row scan line at a time as shown in the following illustration.

Each screen point is referred to as a **pixel** picture element or **pel**. At the end of each scan line, the electron beam returns to the left side of the screen to begin displaying the next scan line.

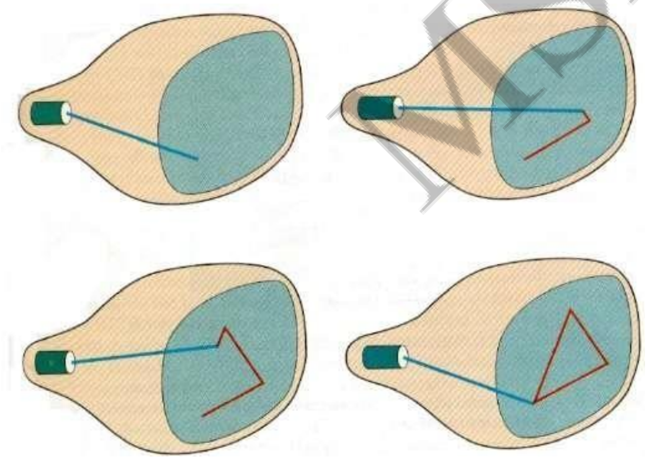


Random Scan /Vector Scan

In this technique, the electron beam is directed only to the part of the screen where the picture is to be drawn rather than scanning from left to right and top to bottom as in raster scan. It is also called **vector display, stroke-writing display,** or **calligraphic display**.

Picture definition is stored as a set of line-drawing commands in an area of memory referred to as the **refresh display file**. To display a specified picture, the system cycles through the set of commands in the display file, drawing each component line in turn. After all the line-drawing commands are processed, the system cycles back to the first line command in the list.

Random-scan displays are designed to draw all the component lines of a picture 30 to 60 times each second.



Application of Computer Graphics

Computer Graphics has numerous applications, some of which are listed below −

- **Computer graphics user interfaces** GUIs − A graphic, mouse-oriented paradigm which allows the user to interact with a computer.
- **Business presentation graphics** − "A picture is worth a thousand words".

- **Cartography** − Drawing maps.
- **Weather Maps** − Real-time mapping, symbolic representations.
- **Satellite Imaging** − Geodesic images.
- **Photo Enhancement** − Sharpening blurred photos.
- **Medical imaging** − MRIs, CAT scans, etc. - Non-invasive internal examination.
- **Engineering drawings** − mechanical, electrical, civil, etc. - Replacing the blueprints of the past.
- **Typography** − The use of character images in publishing - replacing the hard type of the past.
- **Architecture** − Construction plans, exterior sketches - replacing the blueprints and hand drawings of the past.
- **Art** − Computers provide a new medium for artists.
- **Training** − Flight simulators, computer aided instruction, etc.
- **Entertainment** − Movies and games.
- **Simulation and modeling** − Replacing physical modeling and enactments

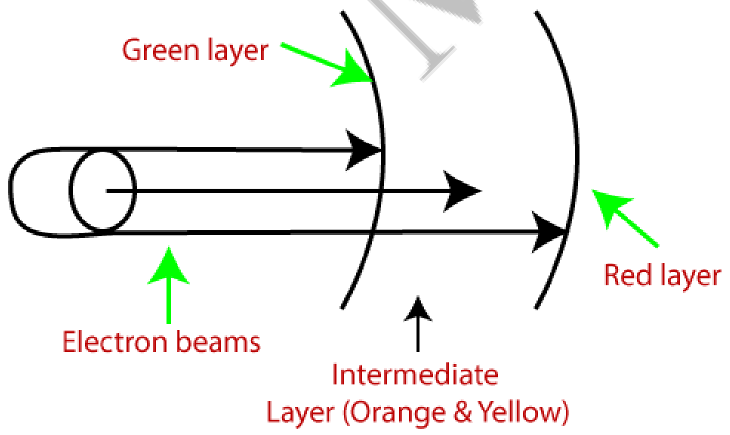
Color CRT Monitor: It is similar to a CRT monitor.

The basic idea behind the color CRT monitor is to combine three basic colors- Red, Green, and Blue. By using these three colors, we can produce millions of different colors.

The two basic color display producing techniques are:

1. **Beam–Penetration Method:** It is used with a **random scan** monitor for displaying pictures. There are two phosphorus layers- Red and Green are coated inside the screen. The color shown depends on how far the electron beam penetrates the phosphorus surface.

A powerful electron beam penetrates the CRT, it passes through the red layer and excites the green layer within. A beam with slow electrons excites only the red layer. A beam with the medium speed of electrons, a mixture of red and green light is emitted to display two more colors- orange and yellow.



Advantages:

1. Better Resolution
2. Half cost
3. Inexpensive

Disadvantages:

1. Only four possible colors
2. Time Consuming

2. Shadow–Mask Method: It is used with a **raster scan** monitor for displaying pictures. It has more range of color than the beam penetration method. It is used in television sets and monitors.

Structure:

1. It has three phosphorus color dots at each position of the pixel.

First Dot: **Red** color
Second Dot: **Green** color
Third Dot: **Blue** color

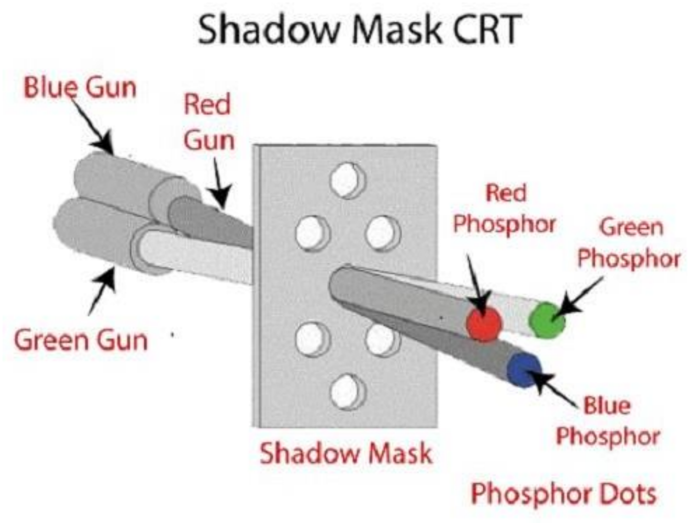
1. It has three different guns. Each for one color.
2. It has a metal screen or plate just before the phosphorus screen, named **"Shadow-Mask."**
3. It also has a shadow grid just behind the phosphorus coated screen with tiny holes in a triangular shape.

Working: A Shadow Mask is a metal plate with tiny holes present inside a color monitor.

A Shadow Mask directs the beam by consuming the electrons so that the beam hits only the desired point and displays a resulting picture.

It has three different guns. These guns direct their beams to shadow mask, which allows them to pass. It is a task of a shadow mask to direct the beam on its particular dot on the screen and produce a picture on the screen.

A Shadow Mask can display a wider range of pictures than beam penetration.



Advantages:

1. Display a wider range picture.
2. Display realistic images.
3. In-line arrangement of RGB color.

Disadvantages:

Difficult to cover all three beams on the same hole.
Poor Resolution.

Liquid crystal display (LCD): The LCD depends upon the light modulating properties of liquid crystals. LCD is used in watches and portable computers. LCD requires an AC power supply instead of DC, so it is difficult to use it in circuits. It generally works on flat panel display technology. LCD consumes less power than LED. The LCD screen uses the liquid crystal to turn pixels on or off. Liquid Crystals are a mixture of solid and liquid. When the current flows inside it, its position changes into the desired color.

For Example: TFT(Thin Film Transistor)

Advantages:

1. Produce a bright image
2. Energy efficient
3. Completely flat screen

Disadvantages:

1. Fixed aspect ratio & Resolution
2. Lower Contrast
3. More Expensive

Light Emitting Diode (LED): LED is a device which emits when current passes through it. It is a semiconductor device. The size of the LED is small, so we can easily make any display unit by arranging a large number of LEDs. LED consumes more power compared to LCD. LED is used on TV, smartphones, motor vehicles, traffic light, etc. LEDs are powerful in structure, so they are capable of withstanding mechanical pressure. LED also works at high temperatures.

Advantages:

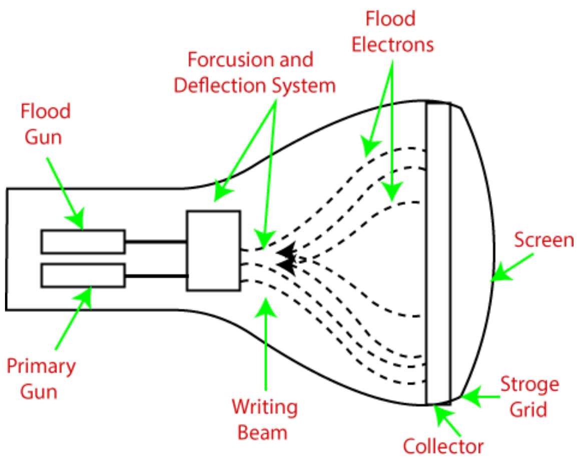
1. The Intensity of light can be controlled.
2. Low operational Voltage.
3. Capable of handling the high temperature.

Disadvantages:

More Power Consuming than LCD.

Direct View Storage Tube (DVST): It is used to store the picture information as a charge distribution behind the phosphor-coated screen.There are two guns used in DVST:

1.Primary Gun: It is used to store the picture information.
2.Flood / Secondary Gun: It is used to display a picture on the screen.



Advantages:

1. Less Time Consuming
2. No Refreshing Required
3. High-Resolution
4. Less Cost

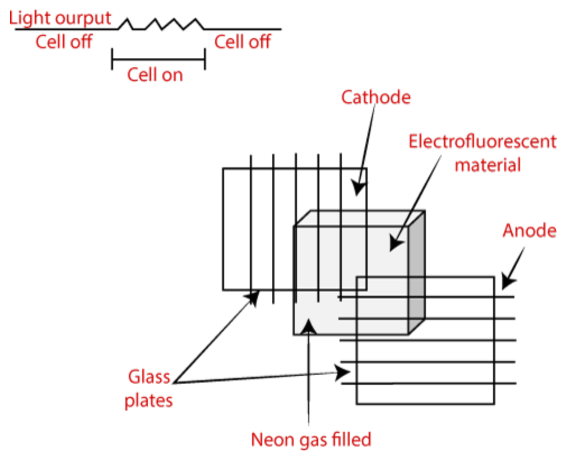
Disadvantages:

1. The specific part of the image cannot be erased.
2. They do not display color.

Plasma Display: It is a type of flat panel display which uses tiny plasma cells. It is also known as **the Gas-Discharge display**.

Components of plasma display:

1. Anode: It is used to deliver a positive voltage. It also has the line wires.
2. Cathode: It is used to provide negative voltage to gas cells. It also has fine wires.
3. Gas Plates: These plates work as capacitors. When we pass the voltage, the cell lights regularly.
4. Fluorescent cells: It contains small pockets of gas liquids when the voltage is passed to this neon gas. It emits light.



Advantages:

1. Wall Mounted
2. Slim
3. Wider angle

Disadvantages:

1. Phosphorus loses luminosity over time.
2. It consumes more electricity than LCD.
3. Large Size

3D Display: It is also called stereoscope display technology. This technology is capable of bringing depth perception to the viewer. It is used for 3D gaming and 3D TVs.

For Example: Fog Display, Holographic Display, Retina Display Etc.

Advantages:

- Impressive Picture Quality
- High Definition
- Motion Communicates

Disadvantage:

- Expensive
- Binocular Fusion

Input Devices

Following are some of the important input devices which are used in a computer −

- Keyboard
- Mouse

- Joy Stick
- Light pen
- Track Ball
- Scanner
- Graphic Tablet
- Microphone
- Magnetic Ink Card Reader(MICR)
- Optical Character Reader(OCR)
- Bar Code Reader
- Optical Mark Reader(OMR)

Keyboard

Keyboard is the most common and very popular input device which helps to input data to the computer. The layout of the keyboard is like that of traditional typewriter, although there are some additional keys provided for performing additional functions.

Keyboards are of two sizes 84 keys or 101/102 keys, but now keyboards with 104 keys or 108 keys are also available for Windows and Internet.

The keys on the keyboard are as follows −

S.No	Keys & Description
1	**Typing Keys** These keys include the letter keys (A-Z) and digit keys (09) which generally give the same layout as that of typewriters.
2	**Numeric Keypad** It is used to enter the numeric data or cursor movement. Generally, it consists of a set of 17 keys that are laid out in the same configuration used by most adding machines and calculators.

3	**Function Keys** The twelve function keys are present on the keyboard which are arranged in a row at the top of the keyboard. Each function key has a unique meaning and is used for some specific purpose.
4	**Control keys** These keys provide cursor and screen control. It includes four directional arrow keys. Control keys also include Home, End, Insert, Delete, Page Up, Page Down, Control(Ctrl), Alternate(Alt), Escape(Esc).
5	**Special Purpose Keys** Keyboard also contains some special purpose keys such as Enter, Shift, Caps Lock, Num Lock, Space bar, Tab, and Print Screen.

Mouse

Mouse is the most popular pointing device. It is a very famous cursor-control device having a small palm size box with a round ball at its base, which senses the movement of the mouse and sends corresponding signals to the CPU when the mouse buttons are pressed.

Generally, it has two buttons called the left and the right button and a wheel is present between the buttons. A mouse can be used to control the position of the cursor on the screen, but it cannot be used to enter text into the computer.

Advantages

- Easy to use
- Not very expensive
- Moves the cursor faster than the arrow keys of the keyboard.

Joystick

Joystick is also a pointing device, which is used to move the cursor position on a monitor screen. It is a stick having a spherical ball at its both lower and upper ends. The lower spherical ball moves in a socket. The joystick can be moved in all four directions.

The function of the joystick is similar to that of a mouse. It is mainly used in Computer Aided Designing (CAD) and playing computer games.

Light Pen

Light pen is a pointing device similar to a pen. It is used to select a 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 the tip of a light pen is moved over the monitor screen and the pen button is pressed, its photocell sensing element detects the screen location and sends the corresponding signal to the CPU.

Track Ball

Track ball is an input device that is mostly used in notebook or laptop computer, instead of a mouse. This is a ball which is half inserted and by moving fingers on the ball, the pointer can be moved.

Since the whole device is not moved, a track ball requires less space than a mouse. A track ball comes in various shapes like a ball, a button, or a square.

Scanner

Scanner is an input device, which works more like a photocopy machine. It is used when some information is available on paper and it is to be transferred to the hard disk of the computer for further manipulation.

Scanner captures images from the source which are then converted into a digital form that can be stored on the disk. These images can be edited before they are printed.

Digitizer

Digitizer is an input device which converts analog information into digital form. Digitizer can convert a signal from the television or camera into a series of numbers that could be stored in a computer. They can be used by the computer to create a picture of whatever the camera had been pointed at.

Digitizer is also known as Tablet or Graphics Tablet as it converts graphics and pictorial data into binary inputs. A graphic tablet as digitizer is used for fine works of drawing and image manipulation applications.

Microphone

Microphone is an input device to input sound that is then stored in a digital form.

The microphone is used for various applications such as adding sound to a multimedia presentation or for mixing music.

Magnetic Ink Card Reader (MICR)

MICR input device is generally used in banks as there are large number of cheques to be processed every day. The bank's code number and cheque number are printed on the cheques with a special type of ink that contains particles of magnetic material that are machine readable.

This reading process is called Magnetic Ink Character Recognition (MICR). The main advantages of MICR is that it is fast and less error prone.

Optical Character Reader (OCR)

OCR is an input device used to read a printed text.

OCR scans the text optically, character by character, converts them into a machine readable code, and stores the text on the system memory.

Bar Code Readers

Bar Code Reader is a device used for reading bar coded data (data in the form of light and dark lines). Bar coded data is generally used in labelling goods, numbering the books, etc. It may be a handheld scanner or may be embedded in a stationary scanner.

Bar Code Reader scans a bar code image, converts it into an alphanumeric value, which is then fed to the computer that the bar code reader is connected to.

Optical Mark Reader (OMR)

OMR is a special type of optical scanner used to recognize the type of mark made by pen or pencil. It is used where one out of a few alternatives is to be selected and marked.

It is specially used for checking the answer sheets of examinations having multiple choice questions.

Output Devices

An output device is a component of hardware or the main physical part of a computer that can be touched and seen. An output device is an electromechanical device.

"The Computer gives instructions and data from input devices and processes it and returns the result called as output."

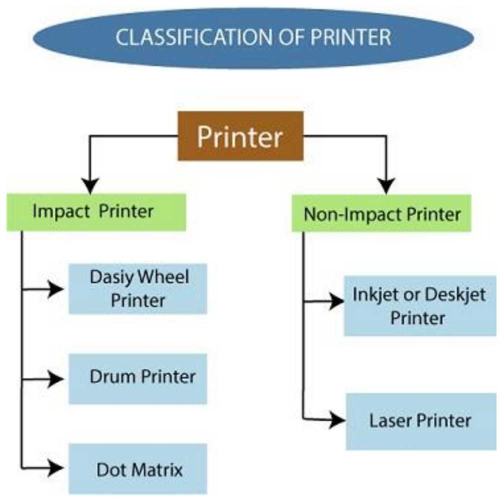
For Example: Printer, Plotter, Monitor, Projector etc.

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.

Types of Printers



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.

Impact printers print the characters by striking them on the ribbon, which is then pressed on the paper.

Characteristics of Impact Printers are the following −

- Very low consumable costs
- Very noisy
- Useful for bulk printing due to low cost
- There is physical contact with the paper to produce an image

These printers are of two types −

- Character printers
- Line printers

Character Printers

Character printers are the printers which print one character at a time.

These are further divided into two types:

- Dot Matrix Printer(DMP)
- Daisy Wheel

Dot Matrix Printer

In the market, one of the most popular printers is Dot Matrix Printer. These printers are popular because of their ease of printing and economical price. Each character printed is in the form of pattern of dots and head consists of a Matrix of Pins of size (5*7, 7*9, 9*7 or 9*9) which come out to form a character which is why it is called Dot Matrix Printer.

Advantages

- Inexpensive
- Widely Used
- Other language characters can be printed

Disadvantages

- Slow Speed
- Poor Quality

Daisy Wheel

Head is lying on a wheel and pins corresponding to characters are like petals of Daisy (flower) which is why it is called Daisy Wheel Printer. These printers are generally used for word-processing in offices that require a few letters to be sent here and there with very nice quality.

Advantages

- More reliable than DMP
- Better quality
- Fonts of character can be easily changed

Disadvantages

- Slower than DMP
- Noisy
- More expensive than DMP

Line Printers

Line printers are the printers which print one line at a time.

These are of two types −

- Drum Printer
- Chain Printer

Drum Printer

This printer is like a drum in shape hence it is called drum printer. The surface of the drum is divided into a number of tracks. Total tracks are equal to the size of the paper, i.e. for a paper width of 132 characters, drum will have 132 tracks. A character set is embossed on the track. Different character sets available in the market are 48 character set, 64 and 96 characters set. One rotation of drum prints one line. Drum printers are fast in speed and can print 300 to 2000 lines per minute.

Advantages

- Very high speed

Disadvantages

- Very expensive
- Characters fonts cannot be changed

Chain Printer

In this printer, a chain of character sets is used, hence it is called Chain Printer. A standard character set may have 48, 64, or 96 characters.

Advantages

- Character fonts can easily be changed.
- Different languages can be used with the same printer.

Disadvantages

- Noisy

Non-impact Printers

Non-impact printers print the characters without using the ribbon. These printers print a complete page at a time, thus they are also called as Page Printers.

These printers are of two types −

- Laser Printers
- Inkjet Printers

Characteristics of Non-impact Printers

- Faster than impact printers
- They are not noisy
- High quality
- Supports many fonts and different character size

Laser Printers

These are non-impact page printers. They use laser lights to produce the dots needed to form the characters to be printed on a page.

Advantages

- Very high speed
- Very high quality output
- Good graphics quality
- Supports many fonts and different character size

Disadvantages

- Expensive
- Cannot be used to produce multiple copies of a document in a single printing

Inkjet Printers

Inkjet printers are non-impact character printers based on a relatively new technology. They print characters by spraying small drops of ink onto paper. Inkjet printers produce high quality output with presentable features.

They make less noise because no hammering is done and these have many styles of printing modes available. Color printing is also possible. Some models of Inkjet printers can produce multiple copies of printing also.

Advantages

- High quality printing
- More reliable

Disadvantages

- Expensive as the cost per page is high
- Slow as compared to laser printer

Monitors

Monitors, commonly called as **Visual Display Unit** (VDU), are the main output device of a computer. It forms images from tiny dots, called pixels that are arranged in a rectangular form. The sharpness of the image depends upon the number of pixels.

There are two kinds of viewing screen used for monitors.

- Cathode-Ray Tube (CRT)
- Flat-Panel Display

Cathode-Ray Tube (CRT) Monitor

The CRT display is made up of small picture elements called pixels. The smaller the pixels, the better the image clarity or resolution. It takes more than one illuminated pixel to form a whole character, such as the letter ‘e’ in the word help.

A finite number of characters can be displayed on a screen at once. The screen can be divided into a series of character boxes - fixed location on the screen where a standard character can be placed. Most screens are capable of displaying 80 characters of data horizontally and 25 lines vertically.

There are some disadvantages of CRT −

- Large in Size
- High power consumption

Flat-Panel Display Monitor

The flat-panel display refers to a class of video devices that have reduced volume, weight and power requirement in comparison to the CRT. You can hang them on walls or wear them on your wrists. Current uses of flat-panel displays include calculators, video games, monitors, laptop computer, and graphics display.

The flat-panel display is divided into two categories −

- **Emissive Displays** − Emissive displays are devices that convert electrical energy into light. For example, plasma panel and LED (Light-Emitting Diodes).
- **Non-Emissive Displays** − Non-emissive displays use optical effects to convert sunlight or light from some other source into graphics patterns. For example, LCD (Liquid-Crystal Device).

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

1. **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: Cars, Ships, Airplanes, Dress design, road and highway blueprints, etc.

Advantages of Flatbed Plotter

1. Larger size paper can be used
2. Drawing Quality is similar to an expert

Disadvantages of Flatbed Plotter

1. Slower than printers
2. More Expensive than printers
3. Do not produce high-Quality text printouts

2. 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).

Advantages of Drum Plotter

1. Draw Larger Size image
2. We can print unlimited length of the image

Disadvantages of Drum Plotter

1. Very costly

Graphics Monitors and Workstations

Graphics Monitor:

i).The name itself giving some idea that, Monitor which is capable of displaying "Graphics". Generally most of the Monitors support Text modes.
ii).So Monitors which are capable of showing and supporting Graphics mode along with the Text modes.

iii).Graphic monitors who can display pictures on its screen, of course it acts like an output device. The monitors which support the following Graphic applications are said to be Graphic Monitors.

The Graphics Applications include,
A).Animation Software's
B).CAD Software's
C).Drawing programs
D).Paint Application programs
E).Presentation Graphics Software's (Excel likewise, creating pie and bar charts)
F).Desktop publishing (MS Office, Share points, Document managements)
So by now, you might have got an Idea of what is Graphic Monitor Actually is??

Workstation:

i).Workstation is also a computer which varies generally with other General Computers

ii).Because these Workstations need good operating power of computer. They must be able to support high power i.e. it must sustain good graphic capabilities.

iii).These kind of computers comes with the following specifications. Unlike normal general computers they consists of,

A).Minimum of 64 Megabytes of RAM

B).Very good Resolution Graphics screen

C).High and Large Screen

D).GUI Graphical User Interface (which helps programs easier to use of the Computer's Graphics)

E).Very good Mass storage Device like Disk Drive

F).Built-in Network Support and many factors

iv) We may also notice that, some of the workstations do not have any disk drives in it. So these kind of disk drives are called as Diskless workstation.

v) Workstations took place of and they lie between Personal computer (General Computers) and minicomputers as far as Computing power is concerned.

vi) These can be used as both types such as stand system (which only consists of one) and Local area network. So basically in this LAN kind, workstations are typically connected together one and other.

Operating Systems for Workstations:

The most generally used common operating systems are,

UNIX:

UNIX is a multitasking operating system. Unix Operating System are to be written in High level programming language, which we can simply say C language. This Unix Operating System is with very flexible and so most commonly used in Workstations.

Windows NT:

Windows NT is also referred as Windows New Technology. It is also a most commonly used Operating system for workstations. This Windows NT highly supports multitasking works and Windows NT has its other versions are such as Windows NT Server and Windows NT Workstation.

Graphics software and standard

- There are mainly two types of graphics software:
    1. General programming package
    2. Special-purpose application package

<u>General programming package</u>

- 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 element transformation etc.
- Example: - GL (Graphics Library).

<u>Special-purpose application package</u>

- Special-purpose application package are customize for particular application which implement required facility and provides interface so that user need not to vory about how it will work (programming). User can simply use it by interfacing with application.
- Example: - CAD, medical and business systems.

Coordinate representations

- Except few all other general packages are designed to be used with Cartesian coordinate specifications.

- If coordinate values for a picture are specified is some other reference frame they must be converted to Cartesian coordinate before giving input to graphics package.
- Special-purpose package may allow use of other coordinates which suits application.
- In general several different Cartesian reference frames are used to construct and display scene.
- We can construct shape of object with separate coordinate system called modeling coordinates or sometimes local coordinates or master coordinates.
- Once individual object shapes have been specified we can place the objects into appropriate positions called world coordinates.
- Finally the World-coordinates description of the scene is transferred to one or more output device reference frame for display. These display coordinates system are referred to as "**Device Coordinates"** or "**Screen Coordinates".**
- Generally a graphic system first converts the world-coordinates position to normalized device coordinates. In the range from 0 to 1 before final conversion to specific device coordinates.
- An initial modeling coordinates position (Xmc,Ymc) in this illustration is transferred to a device coordinates position(Xdc,Ydc) with the sequence (Xmc,Ymc) (Xwc,Ywc) (Xnc,Ync) (Xdc,Ydc).

Graphic Function

- 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 clearing display screen etc.
- All such inbuilt function which we can use for our purpose are known as graphics function

Software Standard

- Primary goal of standardize graphics software is portability so that it can be used in any hardware systems & avoid rewriting of software program for different system
- Some of these standards are discuss below

<u>Graphical Kernel System (GKS)</u>

- 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 later extension 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.

MODULE - II

Output Primitives: Line Drawing Algorithms – Loading the Frame Buffer – Line Function – Circle – Generating Algorithms - Attributes of Output Primitives: Line Attributes – Curve Attributes – Color and Gray scale levels– Area fill Attributes – Character Attributes – Bundled Attributes – Inquiry Functions.

Output Primitives

Points and Lines

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

("the jaggies"), as represented in fig 2.1.

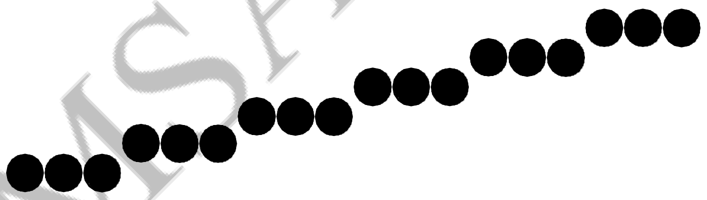


Fig. 2.1: - Stair step effect produced when line is generated as a series of pixel positions.

- The stair step shape is noticeable in low resolution system, and we can improve their appearance somewhat by displaying them on high resolution system.
- More effective techniques for smoothing raster lines are based on adjusting pixel intensities along the line paths.
- For raster graphics device-level algorithms discuss here, object positions are specified directly in integer device coordinates.
- Pixel position will referenced according to scan-line number and column number which is illustrated by following figure.

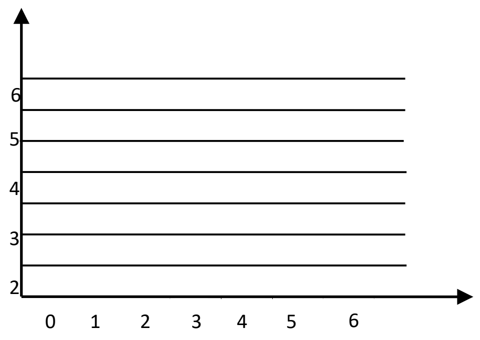


Fig. 2.2: - Pixel positions referenced by scan-line number and column number.

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

Line Drawing Algorithms

The Line drawing algorithm is a graphical algorithm which is used to represent the line segment on discrete graphical media, i.e., printer and pixel-based media.

A line contains two points. The point is an important element of a line.

Properties of a Line Drawing Algorithm

There are the following properties of a good Line Drawing Algorithm.

- **An algorithm should be precise:** Each step of the algorithm must be adequately defined.
- **Finiteness:** An algorithm must contain finiteness. It means the algorithm stops after the execution of all steps.
- **Easy to understand:** An algorithm must help learners to understand the solution in a more natural way.
- **Correctness:** An algorithm must be in the correct manner.
- **Effectiveness:** The steps of an algorithm must be valid and efficient.
- **Uniqueness:** All steps of an algorithm should be clearly and uniquely defined, and the result should be based on the given input.
- **Input:** A good algorithm must accept at least one or more input.
- **Output:** An algorithm must generate at least one output.

Equation of the straight line

We can define a straight line with the help of the following equation.

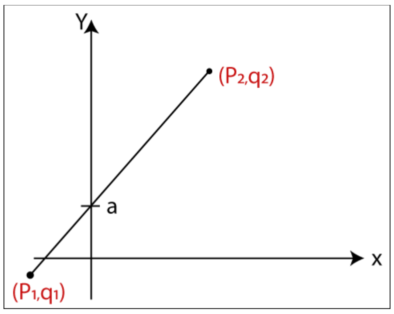
$y= mx + a$

Where,

(x, y) = axis of the line.

m = Slope of the line.

a = Interception point



Let us assume we have two points of the line $(\mathbf{p_1, q_1})$ and $(\mathbf{p_2, q_2})$.
Now, we will put values of the two points in straight line equation, and we get

$\mathbf{y = mx + a}$

$\mathbf{q_2 = mp_2}$ …(1)

$\mathbf{q_1 = mp_1 + a}$ …(2)

We have from equation (1) and (2)

$\mathbf{q_2 - q_1 = mp_2 - mp_1}$

$\mathbf{q_2 - q_1 = m\,(p_2 - p_1)}$

The value of $\mathbf{m = (q_2 - q_1)/\ (p_2 - p_1)}$

$$\mathbf{m = \blacktriangle q\ /\ \blacktriangle p}$$

Algorithms of Line Drawing

There are following algorithms used for drawing a line:

- **DDA (Digital Differential Analyzer) Line Drawing Algorithm**
- **Bresenham's Line Drawing Algorithm**
- **Mid-Point Line Drawing Algorith**

DDA (Digital Differential Analyzer) Line Drawing Algorithm

The Digital Differential Analyzer helps us to interpolate the variables on an interval from one point to another point. We can use the digital Differential Analyzer algorithm to perform rasterization on polygons, lines, and triangles.

Digital Differential Analyzer algorithm is also known as an **incremental method** of scan conversion. In this algorithm, we can perform the calculation in a step by step manner. We use the previous step result in the next step.

As we know the general equation of the straight line is:

$\mathbf{y = mx + c}$

Here, **m** is the slope of $(\mathbf{x_1, y_1})$ and $(\mathbf{x_2, y_2})$.

$\mathbf{m = (y_2 - y_1)/\ (x_2 - x_1)}$

Now, we consider one point $(\mathbf{x_k, y_k})$ and $(\mathbf{x_{k+1}, y_{k+1}})$ as the next point.

Then the slope $\mathbf{m = (y_{k+1} - y_k)/\ (x_{k+1} - x_k)}$

Now, we have to find the slope between the starting point and ending point. There can be following three cases to discuss:

Case 1: If **m < 1**

Then **x** coordinate tends to the Unit interval.

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

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

Case 2: If **m > 1**

Then **y** coordinate tends to the Unit interval.

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

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

Case 3: If **m = 1**

Then **x and y** coordinate tend to the Unit interval.

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

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

We can calculate all intermediate points with the help of above three discussed cases.

Algorithm of Digital Differential Analyzer (DDA) Line Drawing

Step 1: Start.
Step 2: We consider Starting point as **(x_1, y_1),** and endingpoint **(x_2, y_2).**
Step 3: Now, we have to calculate **▲x** and **▲y.**

▲x = x_2-x_1

▲y = y_2-y_1

m = ▲y/▲x

Step 4: Now, we calculate three cases.

If **m < 1**

Then **x** change in Unit Interval

y moves with deviation

$(x_{k+1}, y_{k+1}) = (x_k+1, y_k+1)$

If **m > 1**

Then **x** moves with deviation

y change in Unit Interval

$(x_{k+1}, y_{k+1}) = (x_k+1/m, y_k+1/m)$

If **m = 1**

Then **x** moves in Unit Interval

y moves in Unit Interval

$(x_{k+1}, y_{k+1}) = (x_k+1, y_k+1)$

Step 5: We will repeat step 4 until we find the ending point of the line.
Step 6: Stop.

Example: A line has a starting point (1,7) and ending point (11,17). Apply the Digital Differential Analyzer algorithm to plot a line.

Solution: We have two coordinates,
Starting Point = **(x_1, y_1) = (1,7)**
Ending Point = **(x_2, y_2) = (11,17)**

Step 1: First, we calculate **▲x**, **▲y** and **m.**

▲x = $x_2 - x_1$ = 11-1 = 10

▲y = $y_2 - y_1$ = 17-7 = 10

m = ▲y/▲x = 10/10 = 1

Step 2: Now, we calculate the number of steps.

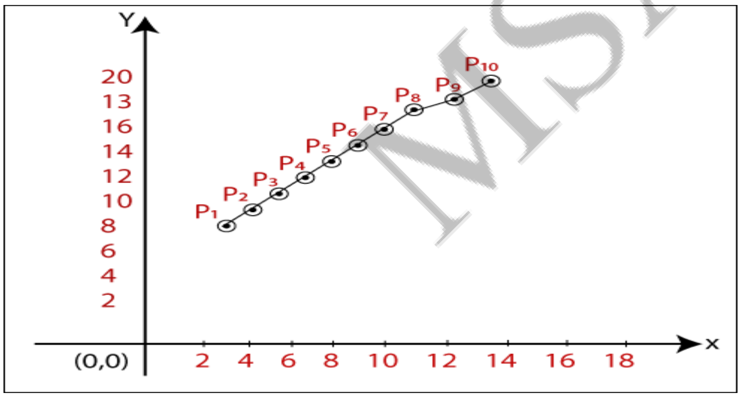
▲x = ▲y = 10

Then, the number of steps = **10**

Step 3: We get **m = 1,** Third case is satisfied.
Now move to next step.

x_k	y_k	x_{k+1}	y_{k+1}	(x_{k+1}, y_{k+1})
1	7	2	8	(2, 8)
		3	9	(3, 9)
		4	10	(4, 10)
		5	11	(5, 11)
		6	12	(6, 12)
		7	13	(7, 13)
		8	14	(8, 14)
		9	15	(9, 15)
		10	16	(10, 16)
		11	17	(11, 17)

Step 4: We will repeat step 3 until we get the endpoints of the line.
Step 5: Stop.



The coordinates of drawn line are-
P_1 = (2, 8)
P_2 = (3, 9)
P_3 = (4, 10)
P_4 = (5, 11)
P_5 = (6, 12)
P_6 = (7, 13)
P_7 = (8, 14)
P_8 = (9, 15)
P_9 = (10, 16)
P_{10} = (11, 17)

Advantages of Digital Differential Analyzer

- It is a simple algorithm to implement.
- It is a faster algorithm than the direct line equation.
- We cannot use the multiplication method in Digital Differential Analyzer.
- Digital Differential Analyzer algorithm tells us about the overflow of the point when the point changes its location.

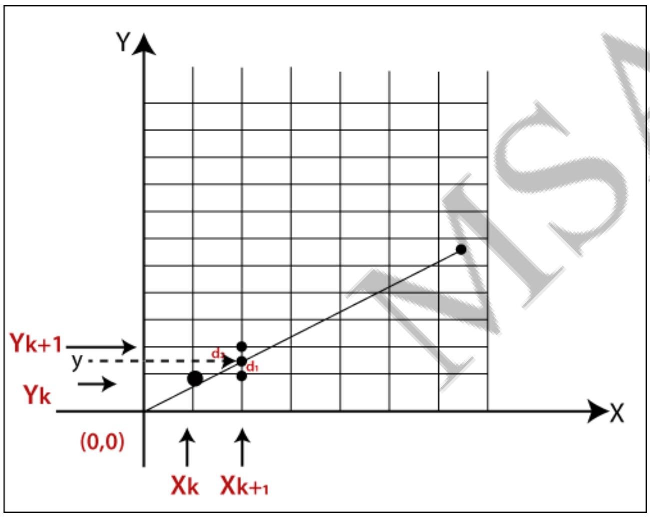
Disadvantages of Digital Differential Analyzer

- The floating-point arithmetic implementation of the Digital Differential Analyzer is time-consuming.
- The method of round-off is also time-consuming.
- Sometimes the point position is not accurate.

Bresenham's Line Drawing Algorithm

This algorithm was introduced by **"Jack Elton Bresenham"** in **1962.** This algorithm helps us to perform scan conversion of a line. It is a powerful, useful, and accurate method. We use incremental integer calculations to draw a line. The integer calculations include addition, subtraction, and multiplication.

In Bresenham's Line Drawing algorithm, we have to calculate the slope (**m**) between the starting point and the ending point.



As shown in the above figure let, we have initial coordinates of a line = (x_k, y_k). The next coordinates of a line = **(x_{k+1}, y_{k+1}).** The intersection point between y_k and y_{k+1} = **y.** Let we assume that the distance between **y** and y_k = d_1. The distance between **y** and y_{k+1} = d_2. Now, we have to decide which point is nearest to the intersection point.

If **m < 1** then **x** = x_k**+1** { Unit Interval}

y = y_k**+1** { Unit Interval}

As we know the equation of a line-

y = mx +b

Now we put the value of x into the line equation, then

y = m(x_k+1) +b ....(1)

The value of $d_1 = y - y_k$

Now we put the value of d_1 in equation (1).

$\mathbf{y = m (x_k+1) + b - y_k}$

Now, we again put the value of **y** in the previous equation then we got,
$\mathbf{d_2 = y_{k+1} - y}$
$\mathbf{\quad = y_k + 1 - m (x_k+1) - b}$

Now, we calculate the difference between $\mathbf{d_1 - d_2}$
If $\mathbf{d_1 < d_2}$
Then $\mathbf{y_{k+1} = y_k}$ **{**we will choose the lower pixel as shown in figure**}**
If $\mathbf{d_1 => d_2}$
Then $\mathbf{y_{k+1} = y_k+1}$ **{**we will choose the upper pixel as shown in figure**}**

Now, we calculate the values of $\mathbf{d_1 - d_2}$
$\mathbf{(d_1 - d_2) = m (x_k+1) + b - y_k - y_k - 1 + m (x_k+1) + b}$

We simplified the above equation and replaced the **m** with **▲y/▲x.**
$\mathbf{(d_1 - d_2) = 2 m (x_{k+1}) - 2y_k + 2b - 1}$

We multiplied **▲x** at both side then we got,
$\mathbf{\blacktriangle x (d_1 - d_2) = \blacktriangle x (2m (x_{k+1}) - 2y_k + 2b - 1)}$

We consider $\mathbf{\blacktriangle x (d_1 - d_2)}$ as a decision parameter $\mathbf{(P_k)}$**,** so
$\mathbf{p_k = \blacktriangle x (d_1 - d_2)}$

After calculation we got,
$\mathbf{P_k = 2\blacktriangle y x_k + 2\blacktriangle y - 2\blacktriangle x y_k + \blacktriangle x (2b-1)}$

Then, the next coordinate of $\mathbf{p_k}$
$\mathbf{p_{k+1} = 2\blacktriangle y x_{k+1} + 2\blacktriangle y - 2\blacktriangle x y_{k+1} + \blacktriangle x (2b-1)}$

Now, the difference between $\mathbf{p_{k+1} - p_k}$ then,
$\mathbf{p_{k+1} - p_k = 2\blacktriangle y (x_{k+1} - x_k) - 2\blacktriangle x (y_{k+1} - y_k)}$
$\mathbf{p_{k+1} = p_k + 2\blacktriangle y (x_{k+1} - x_k) - 2\blacktriangle x (y_{k+1} - y_k)}$ **{Decision parameter coordinate}**

Now, we put the value of $\mathbf{x_{k+1}}$ in above equation then we got,
$\mathbf{p_{k+1} = p_k + 2\blacktriangle y - 2\blacktriangle x (y_{k+1} - y_k)}$ **{New decision parameter when m <1}**

Similarly, if **m >1**, the new decision parameter for next coordinate will be
$\mathbf{p_{k+1} = p_k + 2\blacktriangle y - 2\blacktriangle x (x_{k+1} - x_k)}$ **{New decision parameter when m >1}**
If $\mathbf{p_k >= 0}$ **{For coordinate y}**
Then,
$\mathbf{y_{k+1} = y_k+1}$ **{We will choose the nearest $\mathbf{y_{k+1}}$ pixel}**

The next coordinate will be $\mathbf{(x_k+1, y_k+1)}$
If $\mathbf{p_k < 0}$
Then,
$\mathbf{y_{k+1} = y_k}$ **{We will choose the nearest $\mathbf{y_k}$ pixel}**

The next coordinate will be $\mathbf{(x_k+1, y_k)}$
Similarly,
If $\mathbf{p_k >= 0}$ **{For coordinate x}**

Then,
$x_{k+1} = x_k+1$ **{We will choose the nearest x_{k+1} pixel}**

The next coordinate will be (x_k+1, y_k+1)
If $p_k < 0$

Then,
$x_{k+1} = x_k$ **{We will choose the nearest x_k pixel}**

The next coordinate will be (x_k, y_k+1)

Algorithm of Bresenham's Line Drawing Algorithm

Step 1: Start.
Step 2: Now, we consider Starting point as (x_1, y_1) and ending point (x_2, y_2).
Step 3: Now, we have to calculate $\blacktriangle x$ and $\blacktriangle y$.

$\blacktriangle x = x_2-x_1$
$\blacktriangle y = y_2-y_1$
$m = \blacktriangle y/\blacktriangle x$

Step 4: Now, we will calculate the decision parameter p_k with following formula.

$p_k = 2\blacktriangle y-\blacktriangle x$

Step 5: The initial coordinates of the line are (x_k, y_k), and the next coordinates are (x_{k+1}, y_{k+1}). Now, we are going to calculate two cases for decision parameter p_k

Case 1: If
$p_k < 0$
Then
$p_{k+1} = p_k+2\blacktriangle y$
$x_{k+1} = x_k+1$
$y_{k+1} = y_k$

Case 2: If
$p_k >= 0$
Then
$p_{k+1} = p_k+2\blacktriangle y-2\blacktriangle x$
$x_{k+1} = x_k+1$
$y_{k+1} = y_k+1$

Step 6: We will repeat step 5 until we found the ending point of the line and the total number of iterations = $\blacktriangle x-1$.
Step 7: Stop.

Example: A line has a starting point (9,18) and ending point (14,22). Apply the Bresenham's Line Drawing algorithm to plot a line.

Solution: We have two coordinates,
Starting Point = $(x_1, y_1) = (9,18)$
Ending Point = $(x_2, y_2) = (14,22)$

Step 1: First, we calculate $\blacktriangle x$, $\blacktriangle y$.

$\blacktriangle x = x_2 - x_1 = 14-9 = 5$
$\blacktriangle y = y_2 - y_1 = 22-18 = 4$

Step 2: Now, we are going to calculate the decision parameter (p_k)

$p_k = 2\blacktriangle y-\blacktriangle x$
$= 2 \times 4 - 5 = 3$

The value of $p_k = 3$

Step 3: Now, we will check both the cases.

If p_k >= 0 Then **Case 2** is satisfied.

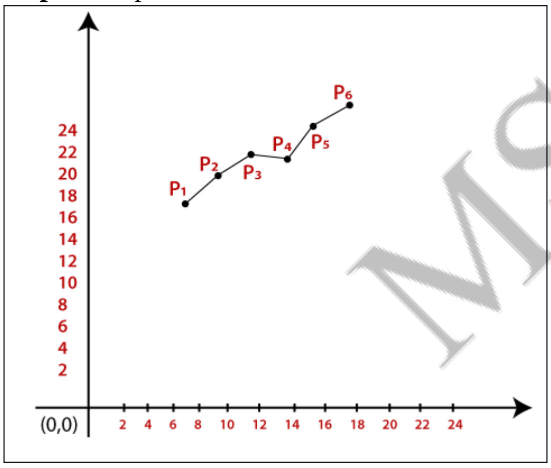
Thus

$$p_{k+1} = p_k + 2\Delta y - 2\Delta x = 3 + (2 \times 4) - (2 \times 5) = 1$$
$$x_{k+1} = x_k + 1 = 9 + 1 = 10$$
$$y_{k+1} = y_k + 1 = 18 + 1 = 19$$

Step 4: Now move to next step. We will calculate the coordinates until we reach the end point of the line.
$\Delta x - 1 = 5 - 1 = 4$

p_k	p_{k+1}	x_{k+1}	y_{k+1}
		9	18
3	1	10	19
1	-1	11	20
-1	7	12	20
7	5	13	21
5	3	14	22

Step 5: Stop.



The Coordinates of drawn lines are-
P_1 = (9, 18)
P_2 = (10, 19)
P_3 = (11, 20)
P_4 = (12, 20)
P_5 = (13, 21)
P_6 = (14, 22)

Advantages of Bresenham's Line Drawing Algorithm

- It is simple to implement because it only contains integers.
- It is quick and incremental

- It is fast to apply but not faster than the Digital Differential Analyzer (DDA) algorithm.
- The pointing accuracy is higher than the DDA algorithm.

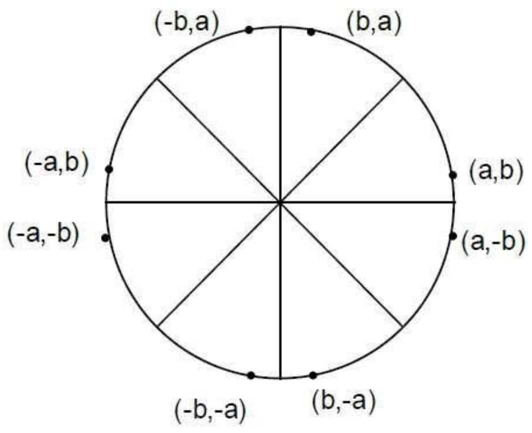
Disadvantages of Bresenham's Line Drawing Algorithm

- The Bresenham's Line drawing algorithm only helps to draw the basic line.
- The resulted draw line is not smooth.

Circle generating algorithm

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

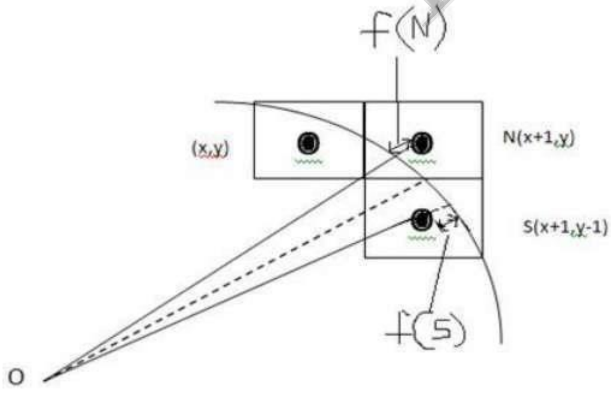
The equation of circle is X2+Y2=r2,X2+Y2=r2, where r is radius.



Bresenham's Algorithm

We cannot display a continuous arc on the raster display. Instead, we have to choose the 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**.

- If d <= 0, then NX+1,YX+1,Y is to be chosen as next pixel.
- 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, y, and R respectively. Set P=0 and Q=R.

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

Step 3 − Repeat through step-8 while $P \leq Q$.

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).

Mid Point Algorithm

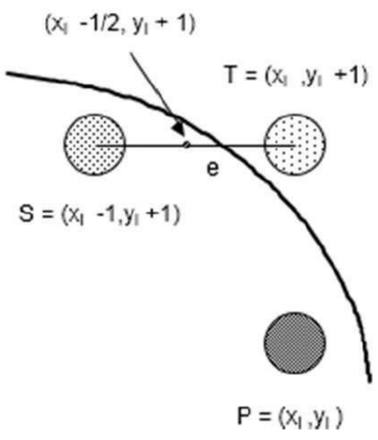
Step 1 − Input radius **r** and circle center (xc,yc)(xc,yc) and obtain the first point on the circumference of the circle centered on the origin as
$(x_0, y_0) = (0, r)$

Step 2 − Calculate the initial value of decision parameter as

P0P0 = 5/4 – r
See the following description for simplification of this equation.

$f(x, y) = x^2 + y^2 - r^2 = 0$

$f(x_i - 1/2 + e, y_i + 1)$
$= (x_i - 1/2 + e)^2 + (y_i + 1)^2 - r^2$
$= (x_i - 1/2)^2 + (y_i + 1)^2 - r^2 + 2(x_i - 1/2)e + e^2$
$= f(x_i - 1/2, y_i + 1) + 2(x_i - 1/2)e + e^2 = 0$



Let $d_i = f(x_i - 1/2, y_i + 1) = -2(x_i - 1/2)e - e^2$
Thus,

If $e < 0$ then $di > 0$ so choose point $S = (x_i - 1, y_i + 1)$.
$d_{i+1} = f(x_i - 1 - 1/2, y_i + 1 + 1) = ((x_i - 1/2) - 1)^2 + ((y_i + 1) + 1)^2 - r^2$
$= d_i - 2(x_i - 1) + 2(y_i + 1) + 1$
$= d_i + 2(y_{i+1} - x_{i+1}) + 1$

If $e >= 0$ then $di <= 0$ so choose point $T = (x_i, y_i + 1)$
$d_{i+1} = f(x_i - 1/2, y_i + 1 + 1)$
$= d_i + 2y_{i+1} + 1$

The initial value of di is
$d_0 = f(r - 1/2, 0 + 1) = (r - 1/2)^2 + 1^2 - r^2$
$= 5/4 - r$ {1-r can be used if r is an integer}

When point $S = (x_i - 1, y_i + 1)$ is chosen then
$d_{i+1} = d_i + -2x_{i+1} + 2y_{i+1} + 1$

When point $T = (x_i, y_i + 1)$ is chosen then
$d_{i+1} = d_i + 2y_{i+1} + 1$

Step 3 − At each XKXK position starting at K=0, perform the following test −

If $P_K < 0$ then next point on circle $(0,0)$ is (X_{K+1}, Y_K) and
$P_{K+1} = P_K + 2X_{K+1} + 1$
Else
$P_{K+1} = P_K + 2X_{K+1} + 1 – 2Y_{K+1}$

Where, $2X_{K+1} = 2X_{K+2}$ and $2Y_{K+1} = 2Y_{K-2}$.

Step 4 − Determine the symmetry points in other seven octants.

Step 5 − Move each calculate pixel position X,YX,Y onto the circular path centered on (XC,YC)(XC,YC) and plot the coordinate values.
$X = X + X_C, \quad Y = Y + Y_C$

Step 6 − Repeat step-3 through 5 until X >= Y.

Attributes of output primitives

Attributes

The features or characteristics of an output primitive are known as ***Attribute***. In other words, any parameter that affects the way a primitive is to be displayed is known as ***Attribute***. Some attributes, such as colour and size, are basic characteristics of primitive. Some attributes control the basic display properties of primitives. For example, lines can be dotted or dashed, thin or thick. Areas can be filled with one colour or with multiple colours pattern. Text can appear from left to right, slanted or vertical.

Line Attributes:

Basic attributes of a straight line are its type, its width, and its colour. In some graphics packages, line can also be displayed using selected pen or brush options.

1. Line Type: The line type attribute includes solid lines, dashed lines, and dotted lines. We modify the line drawing algorithm to generate such lines by setting the length and space. A dashed line could be displayed by generating spaces that is equal to length of solid part. A dotted line can be displayed by generating very short dashes with spacing equal to or greater than the dash size. Similar methods are used to produce other line-type variations.

Raster line-algorithms displays line type attribute by plotting pixels. For various dashed, dotted patterns, the line-drawing algorithms outputs part of pixels followed by spaces. Plotting dashes with a fixed number of pixels results in unequal-length dashes for different line angles. For example, the length of dash diagonally is more than horizontal dash for same number of pixels. For precision drawings, dash length should remain approximately same for any line angle. For this, we can adjust the pixel number according to line slope.

2. Line Width: A line with more width can be displayed as parallel lines on a video monitor. In raster lines, a standard width line is generated with single pixels at each point. Width lines are displayed by plotting additional pixels along next parallel line paths. For lines with slope less than 1, we can display thick lines by plotting a vertical length of pixels at each x position along the line. Similarly, for lines with slope greater than 1, we can plot thick lines with horizontal widths for each point.

The problem with implementing width options using horizontal and vertical pixel widths is that the width of line is depended on the slope. A 45-degree line will be displayed thinner as compared to vertical or horizontal line plotted with same number of pixel widths.

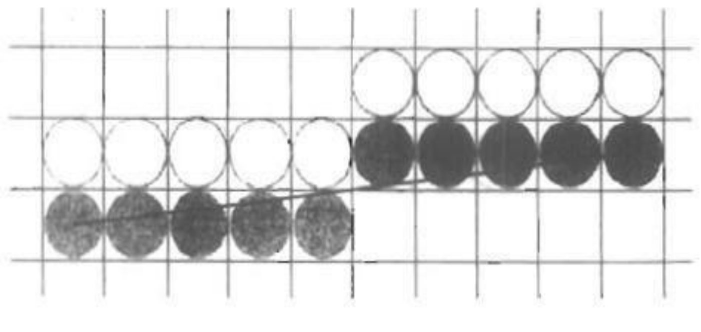


Figure 4-3
Double-wide raster line with slope $|m| < 1$ generated with vertical pixel spans.

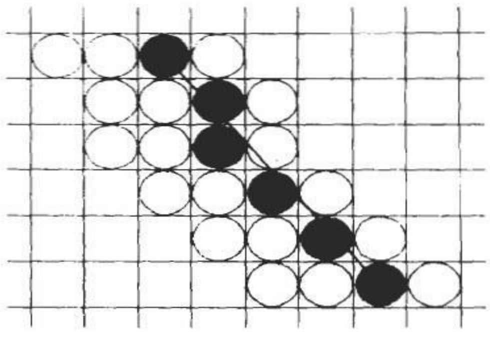
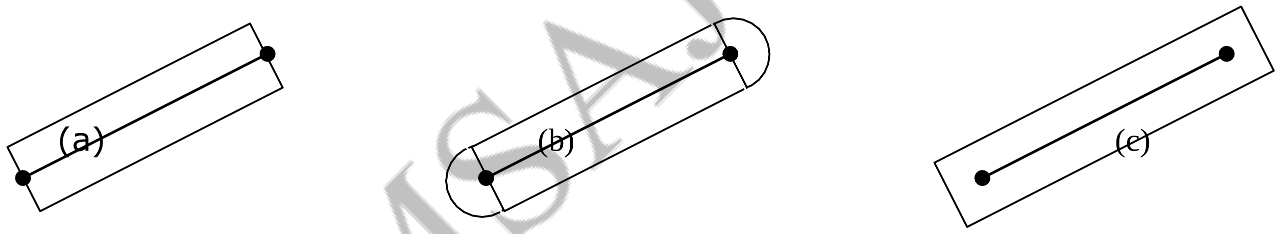


Figure 4-4
Raster line with slope $|m| > 1$ and line-width parameter $lw = 4$ plotted with horizontal pixel spans.

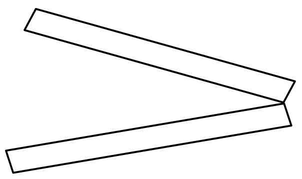
Another problem is that it produces lines whose ends are either horizontal or vertical. We can adjust the shape of the line ends by adding ***Line Caps***.

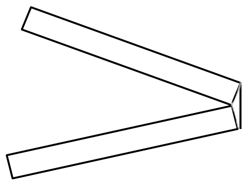
One kind of line cap is the ***Butt Cap.*** It is obtained by adjusting the end positions of lines so that the thick line is displayed with square ends that are perpendicular to the line. Another line cap is the ***Round Cap*** obtained by adding a filled semicircle to each butt cap. The semicircle has diameter equal to thickness of line. The third type of line cap is the ***Projecting Square Cap***. Here, the butt cap is extended to half of line width.



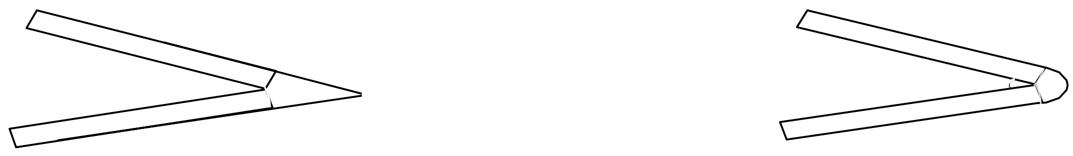
Thick Lines drawn with (a) Butt Cap (b) Round Cap and (c) Projecting Square Cap

Generating thick connected line segments require other considerations. The methods that we have considered for displaying thick lines will not produce smoothly connected line segments. It leaves gaps at the boundaries between lines of different slope. There are three possible methods for smoothly joining two line segments. A ***Miter Join*** is obtained by extending the outer boundaries of each of the two lines until they meet. A ***Round Join*** is produced by covering the connection between the two segments with a circular boundary whose diameter is equal to the line width. A ***Bevel Join*** is generated by displaying the line with butt caps and filling in the triangular gap where the segments meet.





Line without Joins | Bevel Join



Miter Join | Round Join

3. Pen and Brush Options: In some graphic packages, lines can be displayed with pen or brush selection. Options in this category include shape, size and pattern. These shapes are stored in a ***Pixel Mask*** that identifies the pixel positions that are to be set along the line path. Lines generated with pen or brush shaped can be displayed in various widths by changing the size of the mask. Lines can also be displayed with selected patterns.

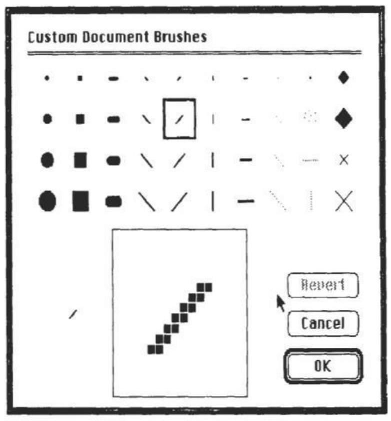


Figure 4-7
Pen and brush shapes for line display.

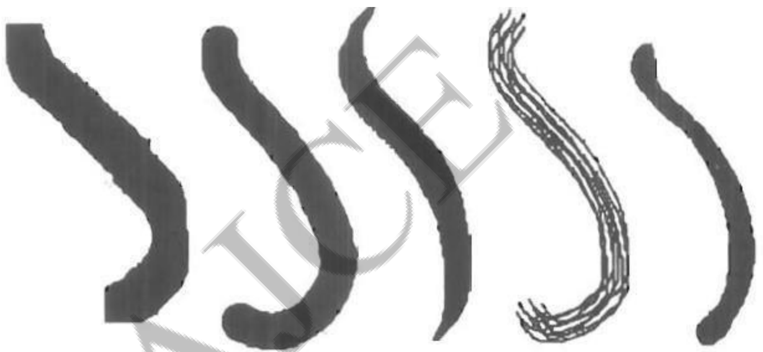


Figure 4-10
Curved lines drawn with a paint program using various shapes and patterns. From left to right, the brush shapes are square, round, diagonal line, dot pattern, and faded airbrush.

4. Line Colour: A system displays a line in the current colour by setting the colour value in the frame buffer at pixel locations. The number of colour choices depends on the number of bits available per pixel in the frame buffer. A line drawn in the background colour is invisible.

COLOR AND GRAYSCALE LEVELS

Various color and intensity-level options can be made available to a user, depending on the capabilities and design objectives of a particular system. General purpose raster-scan systems, for example, usually provide a wide range of colors, while random-scan monitors typically offer only a few color choices, if any. Color options are numerically coded with values ranging from 0 through the positive integers. For CRT monitors, these color codes are then converted to intensity level settings for the electron beams. With color plotters, the codes could control ink-jet deposits or pen selections.

In a color raster system, the number of color choices available depends on the amount of storage provided per pixel in the frame buffer. Also, color-information can be stored in the frame buffer in two ways: We can store color codes directly in the frame buffer, or we can put the color codes in a separate table and use pixel values as an index into this table. With the direct storage scheme, whenever a particular color code is specified in an application program, the corresponding binary value is placed in the frame buffer for each-component pixel in the output primitives to be displayed in that color. A minimum number of colors can be provided in the scheme with 3 bits of storage per pixel, as shown in Table. Each of the three bit positions is used to control the intensity level (either on or off) of the corresponding electron gun in an RGB monitor. The leftmost bit controls the red gun, the middle bit controls the green gun, and the rightmost bit controls the blue gun. Adding more bits per pixel to the frame buffer increases the number of color choices. With 6 bits per pixel, 2 bits can be used for each gun. This allows four different intensity settings for each of the three color guns, and a total of 64 color values are available for each screen pixel. With a resolution of 1024 by 1024, a full-color (24bit per pixel) RGB system needs 3 megabytes of storage for the frame buffer. Color tables are an alternate means for providing extended color capabilities to a user without requiring large frame buffers. Lower-cost personal computer systems, in particular, often use color tables to reduce frame-buffer storage requirements.

COLOR TABLES

Figure 2.1 illustrates a possible scheme for storing color values in a color lookup table (or video lookup table), where frame-buffer values are now used as indices into the color table. In this example, each pixel can reference any one of the 256 table positions, and each entry in the table uses 24 bits to specify an RGB color. For the color code 2081, a combination green-blue color is displayed for pixel location (x, y). Systems employing this particular lookup table would allow a user to select any 256 colors for simultaneous display from a palette of nearly 17 million colors. Compared to a full color system, this scheme reduces the number of simultaneous colors that can be displayed,

but it also reduces the frame buffer storage requirements to 1 megabyte. Some graphics systems provide 9 bits per pixel in the frame buffer, permitting a user to select 512 colors that could be used in each display.

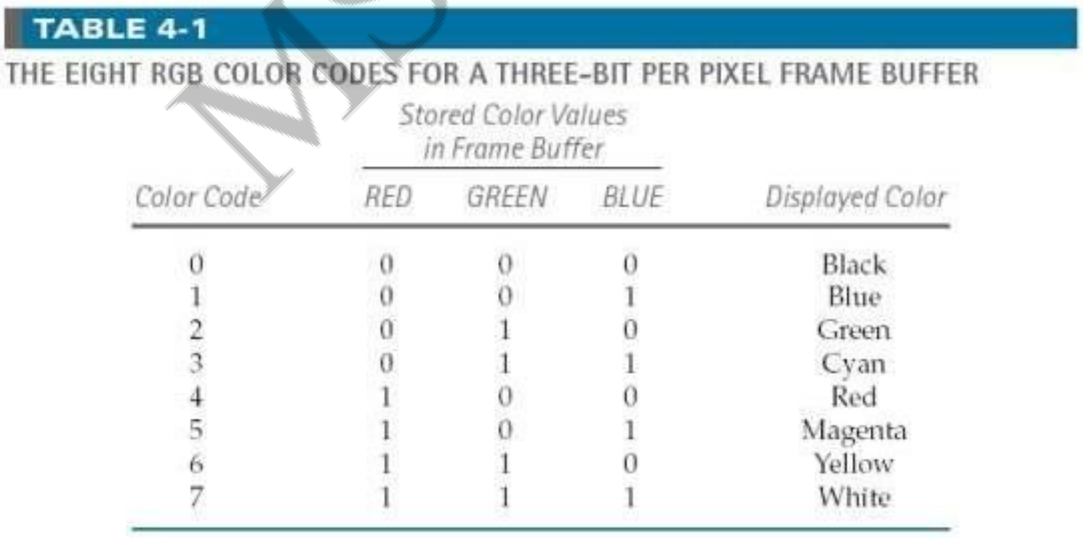
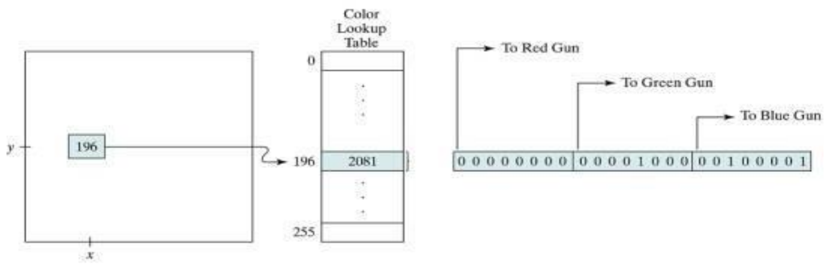


TABLE 4-1

THE EIGHT RGB COLOR CODES FOR A THREE-BIT PER PIXEL FRAME BUFFER

	Stored Color Values in Frame Buffer			
Color Code	RED	GREEN	BLUE	Displayed Color
0	0	0	0	Black
1	0	0	1	Blue
2	0	1	0	Green
3	0	1	1	Cyan
4	1	0	0	Red
5	1	0	1	Magenta
6	1	1	0	Yellow
7	1	1	1	White

(Table 2.1, The eight color codes for a three-bit per pixel frame buffer)



A color lookup table with 24 bits per entry that is accessed from a frame buffer with 8 bits per pixel. A value of 196 stored at pixel position (*x*, *y*) references the location in this table containing the hexadecimal value 0x0821 (a decimal value of 2081). Each 8-bit segment of this entry controls the intensity level of one of the three electron guns in an RGB monitor.

(Fig: 2:1, Color Lookup Table)

There are several advantages in storing color codes in a lookup table. Use of a color table can provide a "reasonable" number of simultaneous colors without requiring large frame buffers. For most applications, 256 or 512 different colors are sufficient for a single picture. Also, table entries can be changed at any time, allowing a user to be able to experiment easily with different color combinations in a design, scene, or graph without changing the attribute settings for the graphics data structure. Similarly, visualization applications can store values for some physical quantity, such as energy, in There are several advantages in storing color codes in a lookup table. Use of a color table can provide a "reasonable" number of simultaneous colors without requiring large frame buffers. For most applications, 256 or 512 different colors are sufficient for a single picture. Also, table entries can be changed at any time, allowing a user to be able to experiment easily with different color combinations in a design, scene, or graph without changing the attribute settings for the graphics data structure. Similarly, visualization applications can store values for some physical quantity, such as energy, in the frame buffer and use a lookup table to try out various color encodings without changing the pixel values. And in visualization and image-processing applications, color tables are a convenient means or setting color thresholds so that all pixel values above or below a specified threshold can be set to the same color. For these reasons, some systems provide both capabilities for color-code storage, so that a user can elect either to use color tables or to store color codes directly in the frame buffer.

GRAYSCALE

With monitors that have no color capability, color functions can be used in an application program to set the shades of gray, or grayscale, for displayed primitives. Numeric values over the range from 0 to 1 can be used to specify grayscale levels, which are then converted to appropriate binary codes for storage in the raster. This allows the intensity settings to be easily adapted to systems with differing grayscale capabilities.

Table lists the specifications for intensity codes for a four-level grayscale system. In this example, any intensity input value near 0.33 would be stored as the binary value 01 in the frame buffer, and pixels with this value would be displayed as dark gray. If additional bits per pixel are available in the frame buffer, the value of 0.33 would be mapped to the nearest level. With 3 bits per pixel, we can accommodate 8 gray levels; while 8 bits per pixel would give us 256 shades of gray. An alternative scheme for storing the intensity information is to convert each intensity code directly to the voltage value that produces this grayscale level on the output device in use.

INTENSITY CODES FOR A FOUR-LEVEL GRAYSCALE SYSTEM

Intensity	Stored Intensity Codes Values In the frame Buffer (Binary Code)		Displayed Gray Scale
0.0	0	(00)	Black
0.33	1	(01)	Dark gray
0.67	2	(1 0)	Light gray
1 .O	3	(11)	White

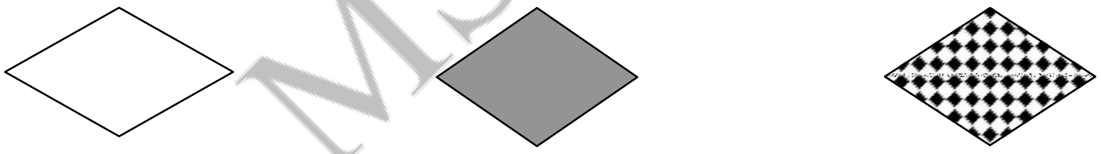
Area-Fill Attributes:

Options for filling a region include a choice between a solid colour and a patterned fill. These options can be applied to polygon regions or regions with curved boundaries.

Areas can be displayed with various fill styles: hollow, solid, pattern and hatch.

Hollow areas are displayed using only the boundary outline, with interior colour the same as the background colour.

A Solid fill is displayed in a single colour including the borders. Fill style can also be with a specific pattern or design. The Hatch fill is used to fill area with hatching pattern.



Hollow Solid Pattern

Polygon fill styles

Diagonal hatch Diagonal Cross-hatch

Polygon fill using hatch patterns

Other fill options include specifications for the edge type, edge width and edge colour. These attributes are same as the line attributes. That is, we can display polygon edges as dotted, dashed, thick or of different colours, etc.

Soft Fill

We may want to fill area again due to 2 reasons:

- It is blurred (unclear) when painted first time, or
- It is repainting of a color area that was originally filled with semitransparent brush, where current color is then mixture of the brush color and the background color "behind" the area.

So that the fill color is combined with the background colors are referred to as Soft-fill.

<u>Character Attributes:</u>

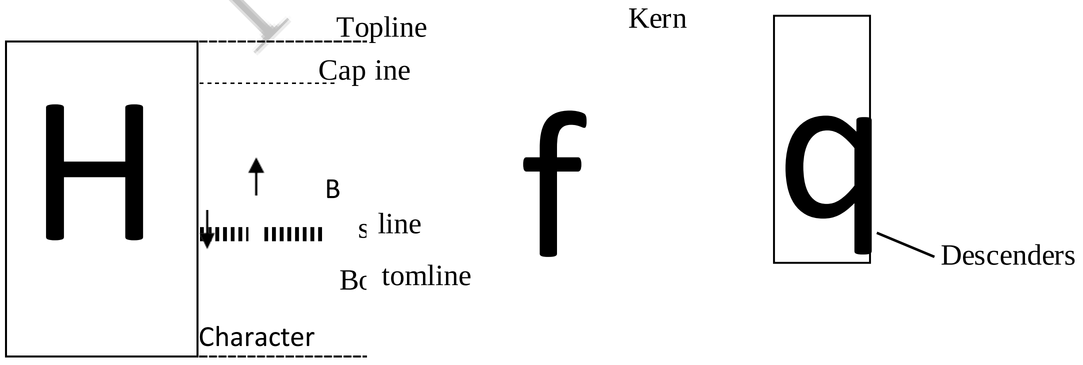
The appearance of displayed characters is controlled by attributes such as font, size, colour and orientation. Attributes can be set both for entire character strings and for individual characters, known as Marker symbols.

1. <u>Text Attributes:</u> There are many text options available, such as font, colour, size, spacing, and orientation.

✦ **Text Style:** The characters in a selected font can also be displayed in various underlining styles (<u>solid</u>, <u>dotted</u>, <u>dashed</u>, <u>double</u>), in **bold**, in *italics*, shadow style, etc. Font options can be made available as predefined sets of grid patterns or as character sets designed with lines and curves.

✦ **Text Colour:** Colour settings for displayed text are stored in the system attribute list and transferred to the frame buffer by character loading functions. When a character string is displayed, the current colour is used to set pixel values in the frame buffer corresponding to the character shapes and position.

✦ **Text Size:** We can adjust text size by changing the overall dimensions, i.e., width and height, of characters or by changing only the width. Character size is specified in ***Points***, where 1 point is 0.013837 inch or approximately 1/72 inch. Point measurements specify the size of the ***Character Body***. Different fonts with the same point specifications can have different character sizes depending upon the design of the font.



Character Body

The distance between ***Topline*** and ***Bottomline*** is same for all characters in a particular size and font, but the width may be different. A smaller body width is assigned to narrow characters such as i, j, l, e t c . compared to broad characters such as W or M. The ***Character Height*** is the distance between the ***Baseline*** and ***Capline*** of characters. Kerned characters, such as f and j, extend beyond the character-width limits. And letters with descenders, such as g, j, p, q, extend below the baseline.

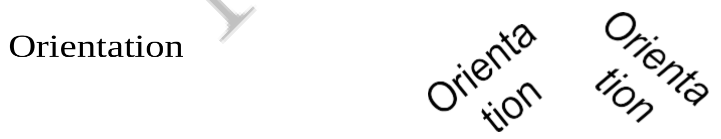
The size can be changed in such a way so that the width and spacing of characters is adjusted to maintain the same text proportions. For example, doubling the height also doubles the character width and the spacing between characters. Also, only the width of the character s can be changes without affecting its height. Similarly, spacing between characters can be increased without changing height or width of individual characters.

The effect of different character-height setting, character-width setting, and character spacing on a text is shown below.

HEIGHT1	WIDTH1	SPACING1
HEIGHT2	**WIDTH2**	**SPACING2**
HEIGHT3	***WIDTH3***	***S P A C I N G 3***

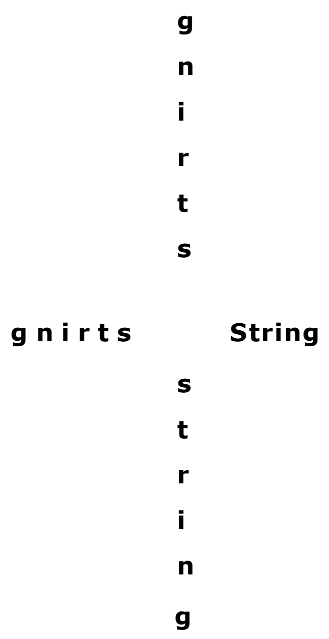
Effect of changing Height, Width and Spacing

- **Text Orientation:** The text can be displayed at various angles, known as orientation. A procedure for orienting text rotates characters so that the sides of character bodies, from baseline to topline at aligned at some angle. Character strings can be arranged vertically or horizontally.



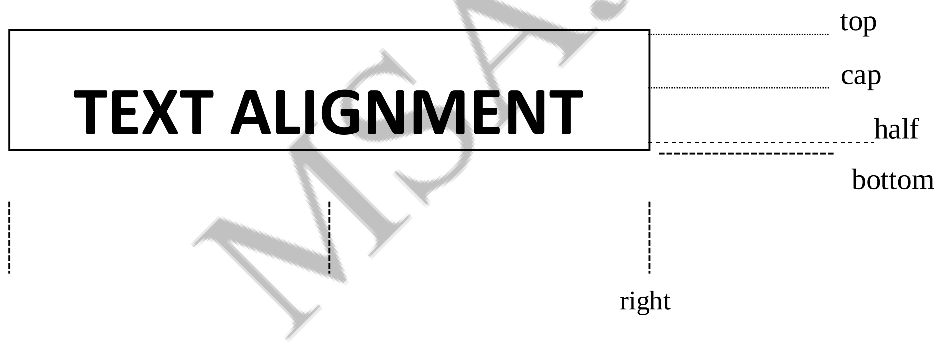
A text orientated by 45 degrees in anticlockwise and clockwise direction

- **Text Path:** In some applications the character strings are arranged vertically or horizontally. This is known as Text Path. Text path can be right, left, up or down.



A text displayed with the four text-path options

- **Text Alignment:** Another attribute for character strings is alignment. This attribute specifies how text is to be positioned with respect to the start coordinates. Vertical alignment can be top, cap, half, base and bottom. Similarly, horizontal alignment can be left, centre and right.



Alignment values for a string

Text Attributes

- In text we are having so many style and design like italic fonts, bold fonts etc.
- For setting the font style in PHIGS package we have one function which is:

setTextFont (tf)

- Where tf is used to specify text font
- It will set specified font as a current character.
- For setting color of character in PHIGS we have function:

setTextColorIndex (tc)

- Where text color parameter tc specifies an allowable color code.
- For setting the size of the text we use function.

setCharacterheight (ch)

- For scaling the character we use function.

setCharacterExpansionFacter (cw)

- Where character width parameter cw is set to a positive real number that scale the character body width.
- Spacing between character is controlled by function

setCharacterSpacing (cs)

- Where character spacing parameter cs can be assigned any real value.
- The orientation for a displayed character string is set according to the direction of the character up vector:

setCharacterUpVector (upvect)

- Parameter upvect in this function is assigned two values that specify the x and y vector components Text is then displayed so that the orientation of characters from baseline to cap line is in the direction of the up vector.
- For setting the path of the character we use function:

setTextPath (tp)

- Where the text path parameter tp can be assigned the value: right, left, up, or down.
- It will set the direction in which we are writing.
- For setting the alignment of the text we use function.

setTextAlignment (h, v)

- Where parameter h and v control horizontal and vertical alignment respectively.
- For specifying precision for text display is given with function.

setTextPrecision (tpr)

- Where text precision parameter tpr is assigned one of the values: string, char, or stroke.
- The highest-quality text is produced when the parameter is set to the value stroke.

Marker Attributes

- A marker symbol display single character in different color and in different sizes.
- For marker attributes implementation by procedure that load the chosen character into the raster at defined position with the specified color and size.
- We select marker type using function.

setMarkerType (mt)

- Where marker type parameter mt is set to an integer code.
- Typical codes for marker type are the integers 1 through 5, specifying, respectively, a dot (.), a vertical cross (+), an asterisk (*), a circle (o), and a diagonal cross (x). Displayed marker types are centered on the marker coordinates.
- We set the marker size with function.

SetMarkerSizeScaleFactor (ms)

- Where parameter marker size ms assigned a positive number according to need for scaling.
- For setting marker color we use function.

setPolymarkerColorIndex (mc)

- Where parameter mc specify the color of the marker symbol

Bundled Attributes

A particular set of attribute values tor a primitive on each output device is then chosen by specifying the appropriate table index. Attributes specified in this manner called bundled attributes

Bundled line Attributes

Entries in the bundle table for line attributes on a specified workstation are set with the function

setPolylineRepresentation (ws, li, lt, lw, lc)

Parameter ws is the workstation identifier and line index parameter li defines the bundle table position. Parameter lt, lw, tc are then bundled and assigned values to set the line type, line width, and line color specifications for designated table index.

Example

setPolylineRepresentation (1, 3, 2, 0.5, 1)

setPolylineRepresentation (4, 3, 1, 1, 7)

A poly line that is assigned a table index value of 3 would be displayed using dashed lines at half thickness in a blue color on work station 1; while on workstation 4, this same index generates solid, standard-sized white lines

Bundle area fill Attributes

Table entries for bundled area-fill attributes are set with

setInteriorRepresentation (ws, fi, fs, pi, fc)

Which defines the attributes list corresponding to fill index fi on workstation ws. Parameter fs, pi and fc are assigned values for the fill style pattern index and fill color.

Bundled Text Attributes

setTextRepresentation (ws, ti, tf, tp, te, ts, tc)

Bundles values for text font, precision expansion factor size an color in a table position for work station ws that is specified by value assigned to text index parameter ti.

Bundled marker Attributes

setPolymarkerRepresentation (ws, mi, mt, ms, mc)

That defines marker type marker scale factor marker color for index mi on workstation ws.

Inquiry Function

Current settings for attributes and other parameters as workstations types and status in the system lists can be retrieved with inquiry functions.

inquirePolylineIndex (lastli)

and

inquireInteriorcColourIndex (lastfc)

Copy the current values for line index and fill color into parameter lastli and lastfc.

MODULE - III

2D Geometric Transformations: Basic Transformation – Matrix Representations – Composite Transformations – Window to View port Co-Ordinate Transformations - Clipping: Point Clipping – Line Clipping – Cohen-Sutherland Line Clipping – Liang Barsky Line Clipping – Polygon Clipping – Sutherland – Hodgeman Polygon Clipping – Curve Clipping – Text Clipping.

2D Transformations

Transformation means changing some graphics into something else by applying rules. We can have various types of transformations such as translation, scaling up or down, rotation, shearing, etc. When a transformation takes place on a 2D plane, it is called 2D transformation.

Transformations play an important role in computer graphics to reposition the graphics on the screen and change their size or orientation.

Homogenous Coordinates

To perform a sequence of transformation such as translation followed by rotation and scaling, we need to follow a sequential process −

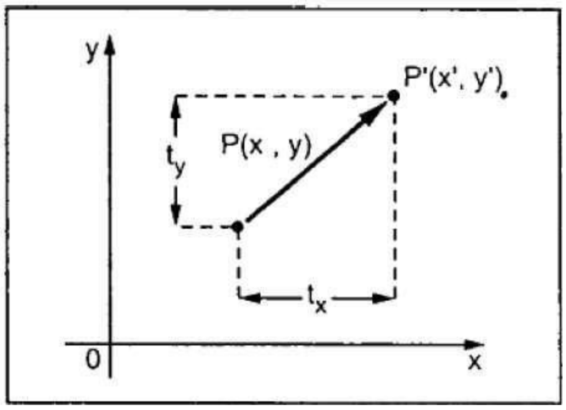
- Translate the coordinates,
- Rotate the translated coordinates, and then
- Scale the rotated coordinates to complete the composite transformation.

To shorten this process, we have to use 3×3 transformation matrix instead of 2×2 transformation matrix. To convert a 2×2 matrix to 3×3 matrix, we have to add an extra dummy coordinate W.

In this way, we can represent the point by 3 numbers instead of 2 numbers, which is called **Homogenous Coordinate** system. In this system, we can represent all the transformation equations in matrix multiplication. Any Cartesian point PX,YX,Y can be converted to homogenous coordinates by P' (X_h, Y_h, h).

Translation

A translation moves an object to a different position on the screen. You can translate a point in 2D by adding translation coordinate (t_x, t_y) to the original coordinate X,YX,Y to get the new coordinate X′,Y′X′,Y′.



From the above figure, you can write that −

X' = X + t_x

Y' = Y + t_y

The pair (t_x, t_y) is called the translation vector or shift vector. The above equations can also be represented using the column vectors.

P=[X][Y]P=[X][Y] p' = [X′][Y′][X′][Y′]T = [tx][ty][tx][ty]

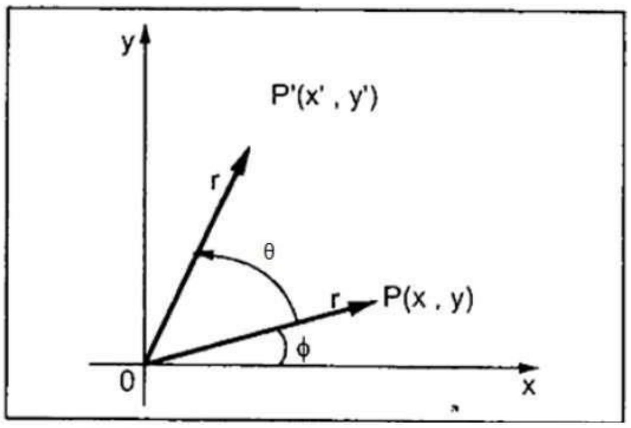
We can write it as −

P' = P + T

Rotation

In rotation, we rotate the object at particular angle θ thetatheta from its origin. From the following figure, we can see that the point PX,YX,Y is located at angle φ from the horizontal X coordinate with distance r from the origin.

Let us suppose you want to rotate it at the angle θ. After rotating it to a new location, you will get a new point P' X′,Y′X′,Y′.



Using standard trigonometric the original coordinate of point PX,YX,Y can be represented as −

$X=r\cos\phi......(1)X=r\cos\phi\ ....\ (1)$

$Y=r\sin\phi......(2)Y=r\sin\phi\ .....\ (2)$

Same way we can represent the point P’ X′,Y′X′,Y′ as −

$x'=r\cos(\phi+\theta)=r\cos\phi\cos\theta-r\sin\phi\sin\theta.......(3)x'=r\cos(\phi+\theta)=r\cos\phi\cos\theta-r\sin\phi\sin\theta\ ......\ (3)$

$y'=r\sin(\phi+\theta)=r\cos\phi\sin\theta+r\sin\phi\cos\theta.......(4)y'=r\sin(\phi+\theta)=r\cos\phi\sin\theta+r\sin\phi\cos\theta.\ .....\ (4)$

Substituting equation 11 & 22 in 33 & 44 respectively, we will get

$x'=x\cos\theta-y\sin\theta x'=x\cos\theta-y\sin\theta$

$y'=x\sin\theta+y\cos\theta y'=x\sin\theta+y\cos\theta$

Representing the above equation in matrix form,

$[X'Y']=[XY][\cos\theta-\sin\theta\sin\theta\cos\theta]OR[X'Y']=[XY][\cos\theta\sin\theta-\sin\theta\cos\theta]OR$

P’ = P . R

Where R is the rotation matrix

$R=[\cos\theta-\sin\theta\sin\theta\cos\theta]R=[\cos\theta\sin\theta-\sin\theta\cos\theta]$

The rotation angle can be positive and negative.

For positive rotation angle, we can use the above rotation matrix. However, for negative angle rotation, the matrix will change as shown below −

$R=[\cos(-\theta)-\sin(-\theta)\sin(-\theta)\cos(-\theta)]R=[\cos(-\theta)\sin(-\theta)-\sin(-\theta)\cos(-\theta)]$

$=[\cos\theta\sin\theta-\sin\theta\cos\theta](\because\cos(-\theta)=\cos\theta and \sin(-\theta)=-\sin\theta)=[\cos\theta-\sin\theta\sin\theta\cos\theta](\because\cos(-\theta)=\cos\theta$ $and \sin(-\theta)=-\sin\theta)$

Scaling

To change the size of an object, scaling transformation is used. In the scaling process, you either expand or compress the dimensions of the object. Scaling can be achieved by multiplying the original coordinates of the object with the scaling factor to get the desired result.

Let us assume that the original coordinates are X,YX,Y, the scaling factors are (S_X, S_Y), and the produced coordinates are X′,Y′X′,Y′. This can be mathematically represented as shown below −

$X' = X \cdot S_X$ and $Y' = Y \cdot S_Y$

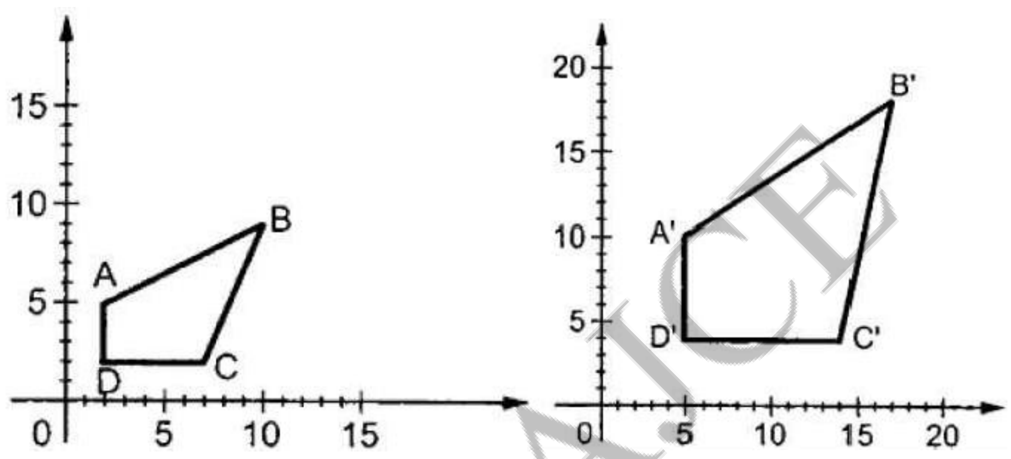
The scaling factor S_X, S_Y scales the object in X and Y direction respectively. The above equations can also be represented in matrix form as below −

(X′Y′)=(XY)[Sx00Sy](X′Y′)=(XY)[Sx00Sy]

OR

P’ = P . S

Where S is the scaling matrix. The scaling process is shown in the following figure.

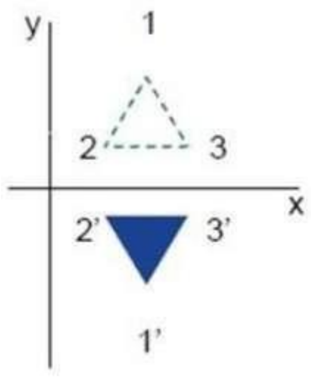


If we provide values less than 1 to the scaling factor S, then we can reduce the size of the object. If we provide values greater than 1, then we can increase the size of the object.

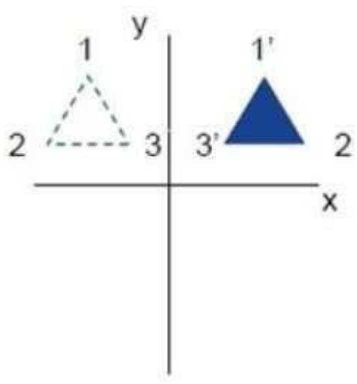
Reflection

Reflection is the mirror image of original object. In other words, we can say that it is a rotation operation with 180°. In reflection transformation, the size of the object does not change.

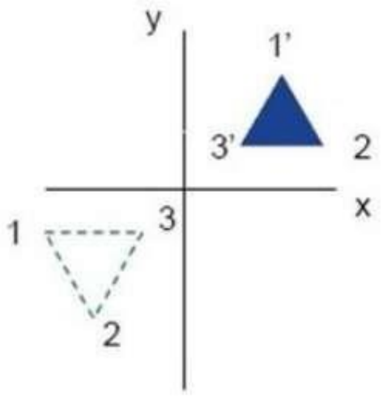
The following figures show reflections with respect to X and Y axes, and about the origin respectively.



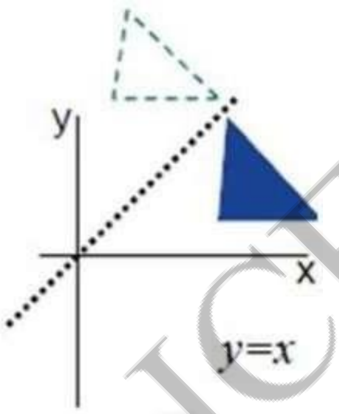
(a)



(b)



(c)



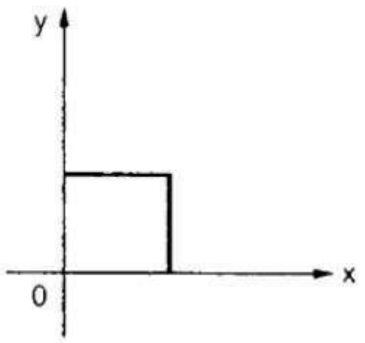
(d)

Shear

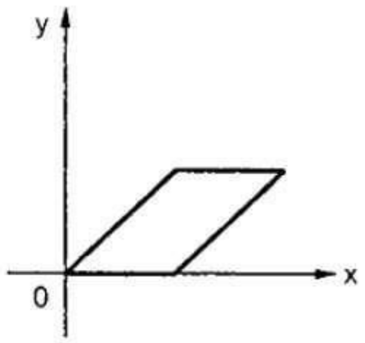
A transformation that slants the shape of an object is called the shear transformation. There are two shear transformations **X-Shear** and **Y-Shear**. One shifts X coordinates values and other shifts Y coordinate values. However; in both the cases only one coordinate changes its coordinates and other preserves its values. Shearing is also termed as **Skewing**.

X- Shear

The X-Shear preserves the Y coordinate and changes are made to X coordinates, which causes the vertical lines to tilt right or left as shown in below figure.



(a) Original object



(b) Object after x shear

The transformation matrix for X-Shear can be represented as −

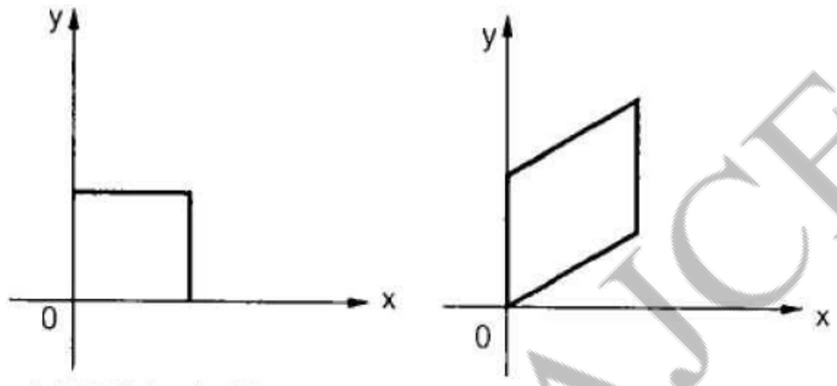
Xsh=⎡⎣⎢⎢100shx10001⎤⎦⎥⎥Xsh=[1shx0010001]

$Y' = Y + Sh_y \cdot X$

$X' = X$

Y-Shear

The Y-Shear preserves the X coordinates and changes the Y coordinates which causes the horizontal lines to transform into lines which slopes up or down as shown in the following figure.



(a) Original object (b) Object after y shear

The Y-Shear can be represented in matrix from as −

Ysh⎡⎣⎢⎢1shy0010001⎤⎦⎥⎥Ysh[100shy10001]

$X' = X + Sh_x \cdot Y$

$Y' = Y$

Matrix representation of transformations.

For translation,
P‘= P+T
x‘ = x + tx
y‘ y ty

This can also be written as ,
x‘ = 1 0 tx . x
y‘ 0 1 ty y
1 0 0 1 1

For rotation,
x‘ = Cosθ -Sinθ . x
y‘ Sinθ Cosθ y

This can be written as,
x‘ = Cos@ -Sin @ 0 . x
y‘ Sin @ Cos @ 0 y
1 0 0 1 1

For scaling,
P‘= S.P

This can be written as,
x‘ = Sx 0 0 . x
y‘ 0 Sy 0 y
1 0 0 1 1

Composite Transformation

If a transformation of the plane T1 is followed by a second plane transformation T2, then the result itself may be represented by a single transformation T which is the composition of T1 and T2 taken in that order. This is written as T = T1·T2.

Composite transformation can be achieved by concatenation of transformation matrices to obtain a combined transformation matrix.

A combined matrix −

[T][X] = [X] [T1] [T2] [T3] [T4] …. [Tn]

Where [Ti] is any combination of

- Translation
- Scaling
- Shearing
- Rotation
- Reflection

The change in the order of transformation would lead to different results, as in general matrix multiplication is not cumulative, that is [A] . [B] ≠ [B] . [A] and the order of multiplication. The basic purpose of composing transformations is to gain efficiency by applying a single composed transformation to a point, rather than applying a series of transformation, one after another.

For example, to rotate an object about an arbitrary point (X_p, Y_p), we have to carry out three steps −

- Translate point (X_p, Y_p) to the origin.
- Rotate it about the origin.
- Finally, translate the center of rotation back where it belonged.

We have learnt matrix representations of transformation. We can set up a matrix for any sequence of transformations as a composite transform matrix by calculating the matrix product of the individual transform.

For e.g. suppose we want to perform rotation of an object about an arbitrary point. This can be performed by applying a sequence of three simple transforms a translation, followed by a rotation followed by another translation. For e.g. suppose we are asked to rotate the triangle through 90^0 .

This can be done by first translate the object to the origin. Rotate it by 90^0 and again translate it by tx = 2, ty=0.

Translate the object to tx=2, ty= 0

Here we have done one translation, one rotation , one translation again. Complex transformation can be described as concatenation of simple ones. Suppose we wish to derive a transformation which will rotate a point through a clockwise angle @ about the point (Rx, Ry). The rotation transformation be applied to rotate points only about the origin. So we must translate points so that (Rx, Ry) becomes the origin.

$$\begin{matrix} x' \\ y' \\ 1 \end{matrix} = \begin{matrix} x \\ y \\ 1 \end{matrix} \; . \; \begin{matrix} 1 & 0 & -Rx \\ 0 & 1 & -Ry \\ 0 & 0 & 1 \end{matrix}$$

then rotation can be applied

$$\begin{matrix} x'' \\ y'' \\ 1 \end{matrix} = \begin{matrix} x' \\ y' \\ 1 \end{matrix} \; \begin{matrix} Cos@ & -Sin@ & 0 \\ Sin@ & Cos@ & 0 \\ 0 & 0 & 1 \end{matrix}$$

Finally, we translate point so that the origin is returned to (Rx,Ry)

$$\begin{matrix} x''' \\ y''' \\ 1 \end{matrix} = \begin{matrix} x'' \\ y'' \\ 1 \end{matrix} \; \begin{matrix} 1 & 0 & Rx \\ 0 & 1 & Ry \\ 0 & 0 & 1 \end{matrix}$$

Window to view port co-ordinate transformation

"The process of selecting and viewing an image with different views, called windowing."

All the objects in the real world have a size. We can measure the size and location of an object by the unit.

For Example-We use the meter unit to measure both size and the location of the object.

When we represent the image of an object on the screen, then we use the screen coordinate system. The screen coordinate system is used to define the location of the object. When we select the screen coordinate system, then the image can be displayed on the screen.

"The Capability to show some part of an object in a window is known as windowing."

"The rectangular area describes in the world coordinate system is called the window."

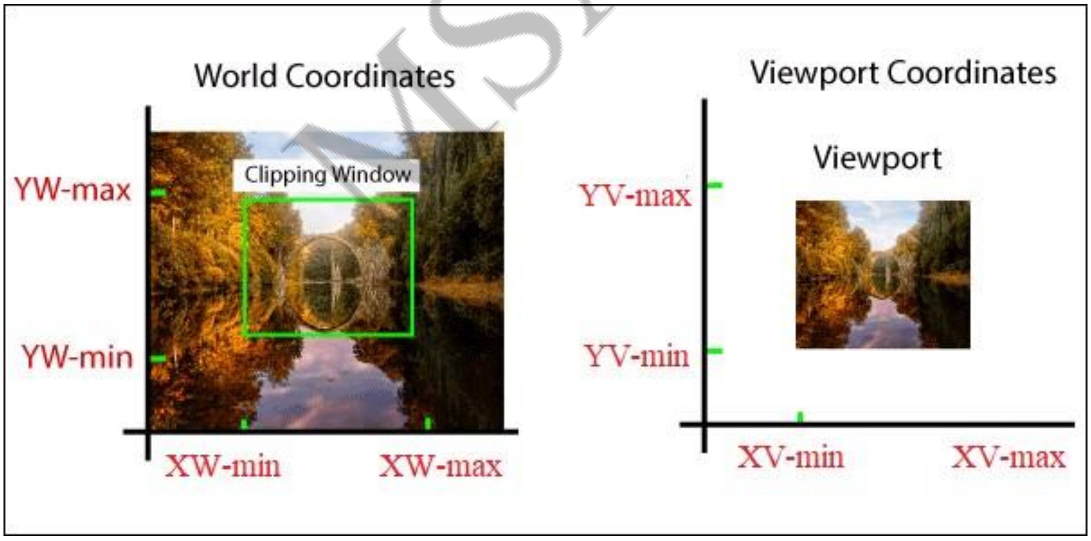
Viewport:

"The viewport can be defined as an area on the screen which is used to display the object." The window is an area space for the object. Viewport surrounds the object. We need the coordinate transformation to display the object on the screen. We can define many viewports on a different area of the screen and also see the same object from a different angle in the viewport.

Window to Viewport transformation:

"Window to viewport transformation is a process of converting two-dimensional or world into a device coordinate."

The object inside the clipping window is mapped to the viewport. The viewport is displayed inside the interface window on the screen. We can use the clipping window to select the part of an object, and the viewport is used to display the selected part of the object on the screen or output device.



Steps for Window to Viewport Transformation: We can follow the following steps for transform window to viewport:

Step 1: Translation of the window towards the Origin– If we shift the window towards the origin, the upper left, and the lower-left corner of the window will be negative (-). The translation factor also should be negative (-).

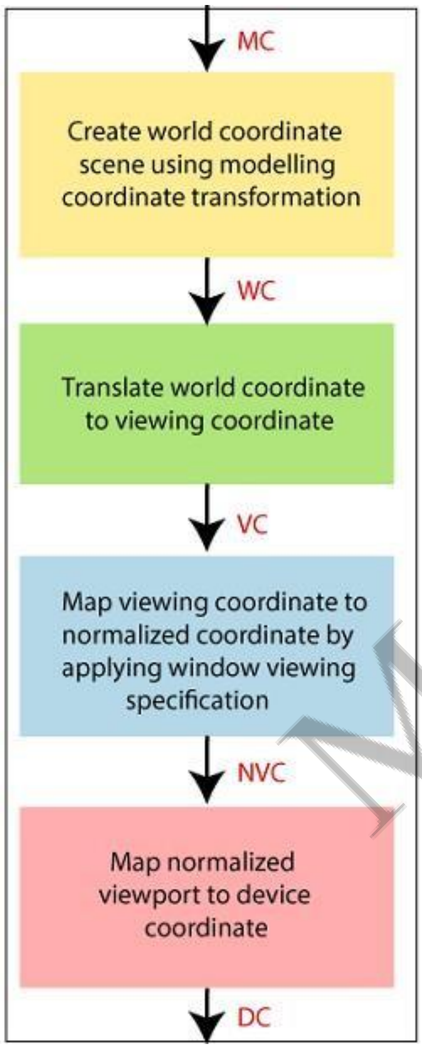
Step 2: Resize the window to viewport size– To Convert the size of the window into the size of the viewport, we will use the following formulas:

$\mathbf{S_x = XV_{max} – XV_{min} / XW_{max} – XW_{min}}$

$\mathbf{S_y = YV_{max} – YV_{min} / YW_{max} – YW_{min}}$

Step 3: Translation of window (Position of window and viewport must be same)– If the lower-left corner of viewport is (0, 0), then the window lower-left corner is already shifted on origin after following step 1.

If the lower-left corner is not equal to (0, 0), then the translation factor should be positive (+).


MC
Create world coordinate scene using modelling coordinate transformation
WC
Translate world coordinate to viewing coordinate
VC
Map viewing coordinate to normalized coordinate by applying window viewing specification
NVC
Map normalized viewport to device coordinate
DC

It may be possible that sometimes the size of viewport is greater or smaller than the window. **In that situation, we have to enlarge or compress the size of the window according to the viewport. We can perform it using some mathematical calculations.**

A point on window = (XW, YW)

Corresponding point on viewport = (XV, YV)

- **To calculate (XV, YV)** –

Normalized point on window = $(XW - XW_{min} / XW_{max} - XW_{min})$

$(YW - YW_{min} / YW_{max} - YW_{min})$

Normalized point on viewport = $(XV - XV_{min} / XV_{max} - XV_{min})$

$(YV - YV_{min} / YV_{max} - YV_{min})$

- **Now, the position of object in viewport and window are same-**

For Coordinate x:

$(XW - XW_{min} / XW_{max} - XW_{min}) = (XV - XV_{min} / XV_{max} - XV_{min})$

For Coordinate y:

$(YW - YW_{min} / YW_{max} - YW_{min}) = (YV - YV_{min} / YV_{max} - YV_{min})$

Now, we get

$XV = XV_{min} + (XW - XW_{min}) S_x$

$YV = YV_{min} + (YW - YW_{min}) S_y$

Here S_x and S_y are the scaling factor for x and y coordinate.

$S_x = XV_{max} - XV_{min} / XW_{max} - XW_{min}$

$S_y = YV_{max} - YV_{min} / YW_{max} - YW_{min}$

Some Important Points:

1. We can select the world coordinate system according to the application program.
2. We can select the screen coordinate system according to the desired design.
3. Viewing transformation is a connection between the world and the screen coordinates.

Advantages:

We can easily represent images on the display screen according to the user's needs.

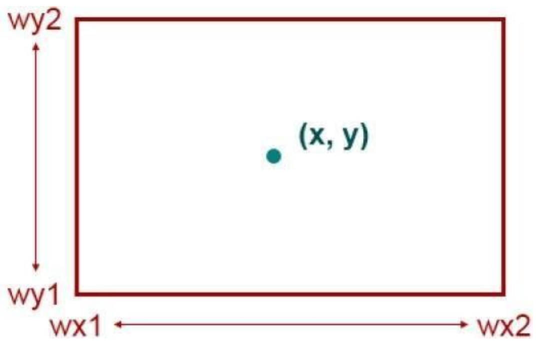
Clipping

The primary use of clipping in computer graphics is to remove objects, lines, or line segments that are outside the viewing pane. The viewing transformation is insensitive to the position of points relative to the viewing volume − especially those points behind the viewer − and it is necessary to remove these points before generating the view.

Point Clipping

Clipping a point from a given window is very easy. Consider the following figure, where the rectangle indicates the window. Point clipping tells us whether the given point X,YX,Y is within the given window or not; and decides whether we will use the minimum and maximum coordinates of the window.

The X-coordinate of the given point is inside the window, if X lies in between Wx1 ≤ X ≤ Wx2. Same way, Y coordinate of the given point is inside the window, if Y lies in between Wy1 ≤ Y ≤ Wy2.

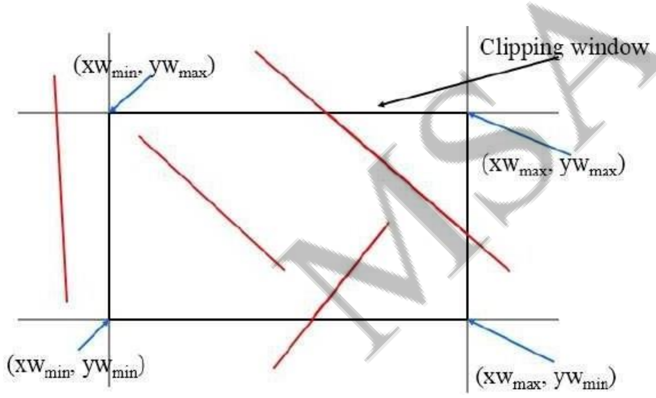


Line Clipping

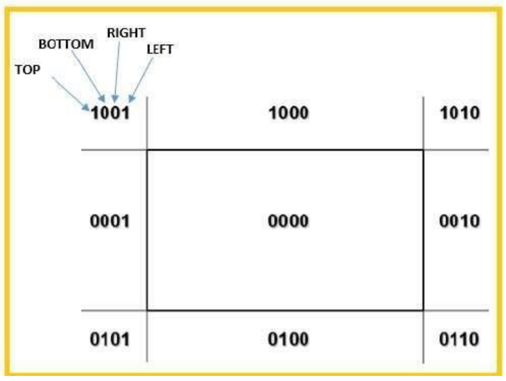
The concept of line clipping is same as point clipping. In line clipping, we will cut the portion of line which is outside of window and keep only the portion that is inside the window.

Cohen-Sutherland Line Clippings

This algorithm uses the clipping window as shown in the following figure. The minimum coordinate for the clipping region is (XWmin,YWmin)(XWmin,YWmin) and the maximum coordinate for the clipping region is (XWmax,YWmax)(XWmax,YWmax).



We will use 4-bits to divide the entire region. These 4 bits represent the Top, Bottom, Right, and Left of the region as shown in the following figure. Here, the **TOP** and **LEFT** bit is set to 1 because it is the **TOP-LEFT** corner.



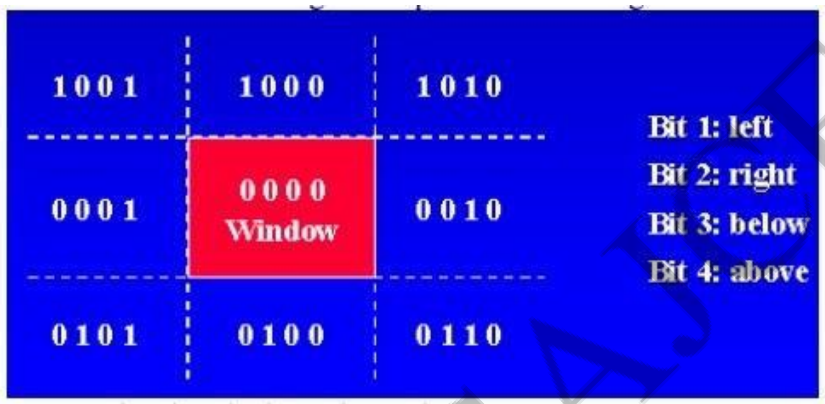
There are 3 possibilities for the line −

- Line can be completely inside the window This line should be accepted.
- Line can be completely outside of the window Thislinewillbecompletelyremovedfromtheregion Thislinewillbecompletelyremovedfromtheregion.
- Line can be partially inside the window Wewillfindintersectionpointanddrawonlythatportionoflinethatisinsideregion Wewillfindintersectionpointanddrawonlythatportionoflinethatisinsideregion.

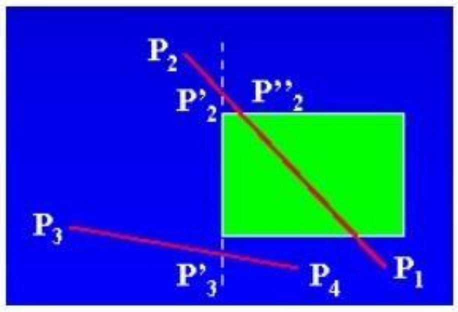
Region code

– A four-digit binary code assigned to every line endpoint in a picture.

– Numbering the bit positions in the region code as 1 through 4 from right to left.



- Bit values in the region code
  - Determined by comparing endpoint coordinates to the clip boundaries
    - A value of 1 in any bit position: The point is in that relative position.
- Determined by the following steps:
  - Calculate differences between endpoint coordinates and clipping boundaries.
  - Use the resultant sign bit of each difference calculation to set the corresponding bit value.
- The possible relationships:
  - Completely contained within the window
    - 0000 for both endpoints.
- Completely outside the window
  - Logical and the region codes of both endpoints, its result is not 0000.
  - Partially



Algorithm

Step 1 − Assign a region code for each endpoints.

Step 2 − If both endpoints have a region code **0000** then accept this line.

Step 3 − Else, perform the logical **AND** operation for both region codes.

Step 3.1 − If the result is not **0000,** then reject the line.

Step 3.2 − Else you need clipping.

Step 3.2.1 − Choose an endpoint of the line that is outside the window.

Step 3.2.2 − Find the intersection point at the window boundary base on region code

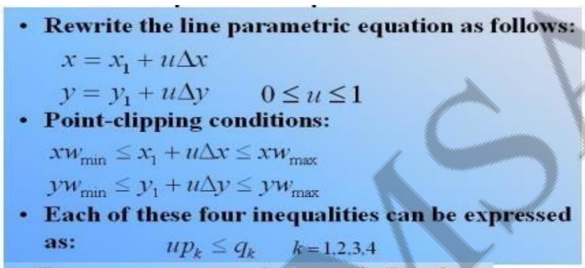
Step 3.2.3 − Replace endpoint with the intersection point and update the region code.

Step 3.2.4 − Repeat step 2 until we find a clipped line either trivially accepted or trivially rejected.

Step 4 − Repeat step 1 for other lines.

Liang-Barsky Line Clipping

Rewrite the line parametric equation as follows:



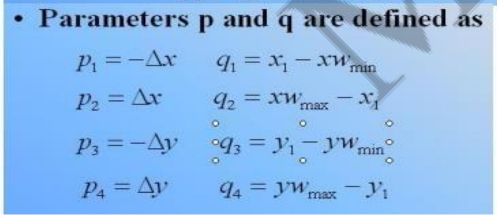
- **Rewrite the line parametric equation as follows:**

$$x = x_1 + u\Delta x$$
$$y = y_1 + u\Delta y \qquad 0 \le u \le 1$$

- **Point-clipping conditions:**

$$xw_{\min} \le x_1 + u\Delta x \le xw_{\max}$$
$$yw_{\min} \le y_1 + u\Delta y \le yw_{\max}$$

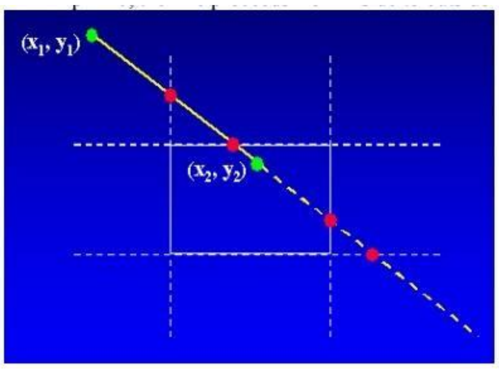
- **Each of these four inequalities can be expressed as:** $up_k \le q_k \quad k = 1,2,3,4$

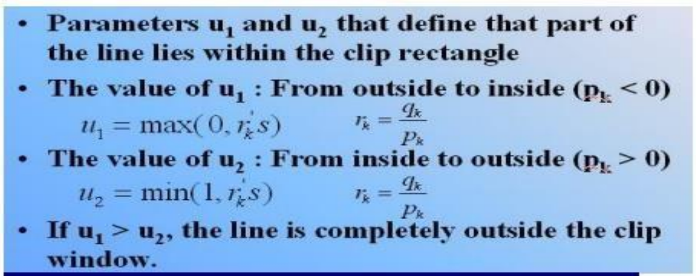


- **Parameters p and q are defined as**

$$p_1 = -\Delta x \qquad q_1 = x_1 - xw_{\min}$$
$$p_2 = \Delta x \qquad q_2 = xw_{\max} - x_1$$
$$p_3 = -\Delta y \qquad q_3 = y_1 - yw_{\min}$$
$$p_4 = \Delta y \qquad q_4 = yw_{\max} - y_1$$

- **pk = 0, parallel to one of the clipping boundary**
- **qk < 0, outside the boundary**
- **qk >= 0, inside the parallel clipping boundary**
- **pk < 0, the line proceeds from outside to the inside**
- **pk > 0, the line proceeds from inside to outside**

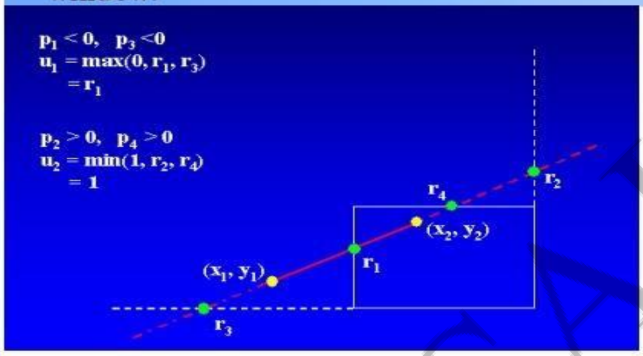




- Parameters u_1 and u_2 that define that part of the line lies within the clip rectangle
- The value of u_1 : From outside to inside ($p_k < 0$)

  $u_1 = \max(0, r_k^{'} s) \qquad r_k = \frac{q_k}{p_k}$
- The value of u_2 : From inside to outside ($p_k > 0$)

  $u_2 = \min(1, r_k^{'} s) \qquad r_k = \frac{q_k}{p_k}$
- If $u_1 > u_2$, the line is completely outside the clip window.

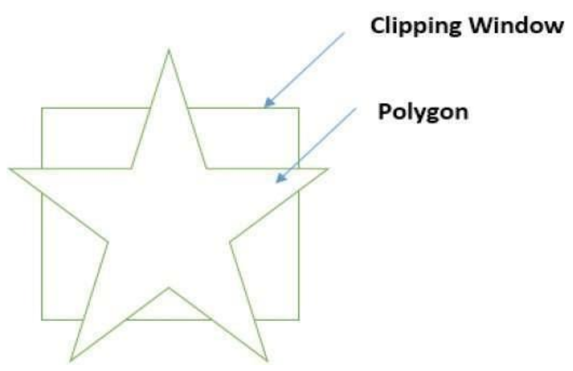


Polygon Clipping

Sutherland Hodgman Algorithm

A polygon can also be clipped by specifying the clipping window. Sutherland Hodgeman polygon clipping algorithm is used for polygon clipping. In this algorithm, all the vertices of the polygon are clipped against each edge of the clipping window.

First the polygon is clipped against the left edge of the polygon window to get new vertices of the polygon. These new vertices are used to clip the polygon against right edge, top edge, bottom edge, of the clipping window as shown in the following figure.



While processing an edge of a polygon with clipping window, an intersection point is found if edge is not completely inside clipping window and the a partial edge from the intersection point to the outside edge is clipped. The following figures show left, right, top and bottom edge clippings −

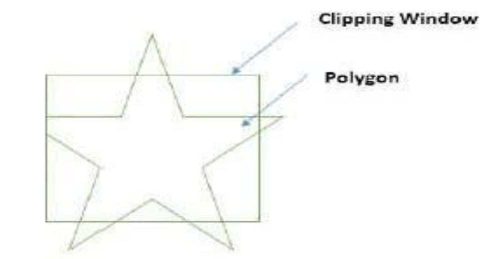


Figure: Clipping Left Edge

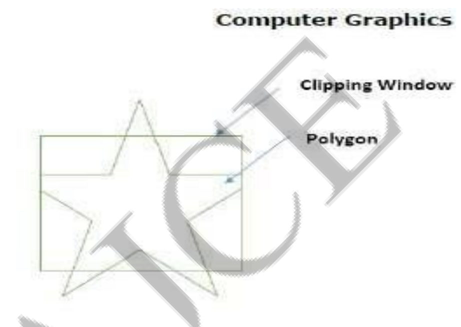


Figure: Clipping Right Edge

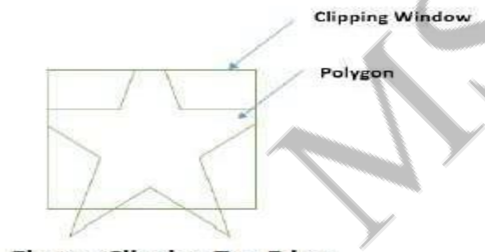


Figure: Clipping Top Edge

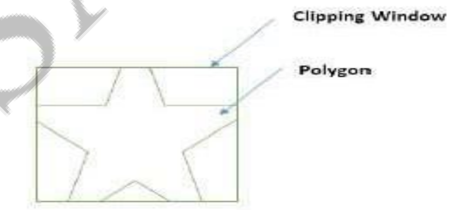


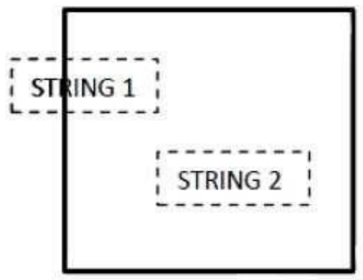
Figure: Clipping Bottom Edge

Text Clipping

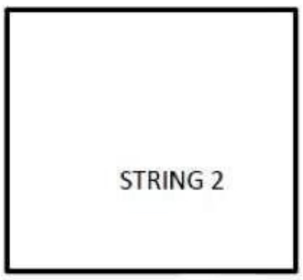
Various techniques are used to provide text clipping in a computer graphics. It depends on the methods used to generate characters and the requirements of a particular application. There are three methods for text clipping which are listed below −

- All or none string clipping
- All or none character clipping
- Text clipping

The following figure shows all or none string clipping −



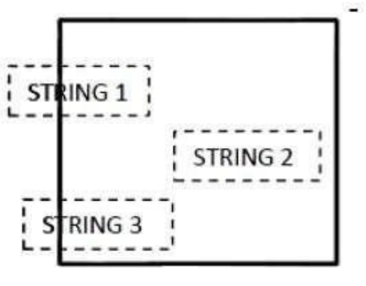
Before Clipping



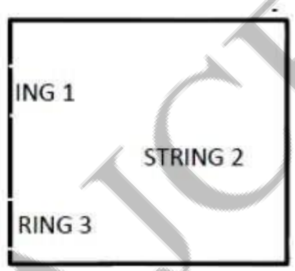
After Clipping

In all or none string clipping method, either we keep the entire string or we reject entire string based on the clipping window. As shown in the above figure, STRING2 is entirely inside the clipping window so we keep it and STRING1 being only partially inside the window, we reject.

The following figure shows all or none character clipping −



Before Clipping

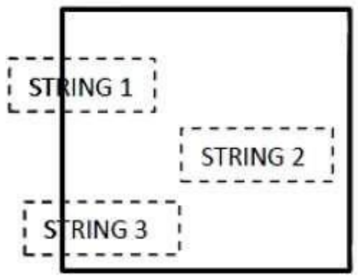


After Clipping

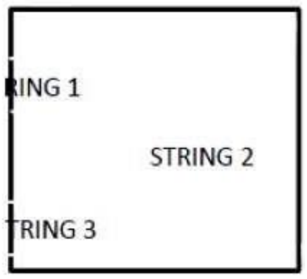
This clipping method is based on characters rather than entire string. In this method if the string is entirely inside the clipping window, then we keep it. If it is partially outside the window, then −

- You reject only the portion of the string being outside
- If the character is on the boundary of the clipping window, then we discard that entire character and keep the rest string.

The following figure shows text clipping −



Before Clipping



After Clipping

This clipping method is based on characters rather than the entire string. In this method if the string is entirely inside the clipping window, then we keep it. If it is partially outside the window, then

- You reject only the portion of string being outside.
- If the character is on the boundary of the clipping window, then we discard only that portion of character that is outside of the clipping window.

Curve Clipping

Curve-clipping procedures will involve nonlinear equations and this requires more processing than for objects with linear boundaries. The bounding rectangle for a circle or other curved object can be used first to test for overlap with a rectangular clip window.

♣If the bounding rectangle for the object is completely inside the window, we save the object.

♣If the rectangle is determined to be completely outside window, we discard the object. In either case, there is no further computation necessary. But if the bounding rectangle test fails, we can look for other computation-saving approaches.

♣For a circle, we can use the coordinate extents of individual quadrants and then octants for preliminary testing before calculating curve-window intersections.

♣For an ellipse, we can test the coordinate extents of individual quadrants Fig: Clipping a filled Circle

MODULE - IV

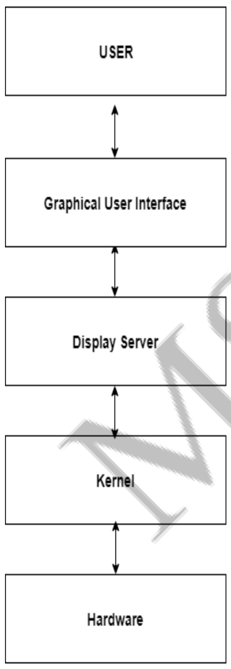
Graphical User Interfaces and Interactive Input Methods: The User Dialogue – Input of Graphical Data – Input Functions – Interactive Picture Construction Techniques – Three Dimensional Concepts: 3D-Display Methods – #Three Dimensional Graphics Packages

Graphical User Interface

GUI is an interface that allows users to interact with different electronic devices using icons and other visual indicators. The graphical user interfaces were created because command line interfaces were quite complicated and it was difficult to learn all the commands in it.

In today's times, graphical user interfaces are used in many devices such as mobiles, MP3 players, gaming devices, smartphones etc.

The below diagram provides the position of the graphical user interface with respect to the computer system:



Elements in Graphical User Interface

Graphical User Interface makes use of visual elements mostly. These elements define the appearance of the GUI. Some of these are described in detail as follows:

Window

This is the element that displays the information on the screen. It is very easy to manipulate a window. It can be opened or closed with the click of an icon. Moreover, it can be moved to any area by dragging it around.In a multitasking environment, multiple windows can be open at the same time, all of them performing different tasks.

There are multiple types of windows in a graphical user interface, such as container window, browser window, text terminal window, child window, message window etc.

Menu

A menu contains a list a choices and it allows users to select one from them. A menu bar is displayed horizontally across the screen such as pull down menu. When any option is clicked in this menu, then the pull down menu appears.

Another type of menu is the context menu that appears only when the user performs a specific action. An example of this is pressing the right mouse button. When this is done, a menu will appear under the cursor.

Icons

Files, programs, web pages etc. can be represented using a small picture in a graphical user interface. This picture is known as an icon. Using an icon is a fast way to open documents, run programs etc. because clicking on them yields instant access.

Controls

Information in an application can be directly read or influences using the graphical control elements. These are also known as widgets. Normally, widgets are used to display lists of similar items, navigate the system using links, tabs etc. and manipulating data using check boxes, radio boxes etc.

Tabs

A tab is associated with a view pane. It usually contains a text label or a graphical icon. Tabs are sometimes related to widgets and multiple tabs allow users to switch between different widgets. Tabs are used in various web browsers such as Internet Explorer, Firefox, Opera, Safari etc. Multiple web pages can be opened in a web browser and users can switch between them using tabs.

In order to be able to interact with the graphical image input methods are required. These can be used to just change the location and orientation of the camera, or to change specific settings of the rendering itself. Different devices are more suitable for changing some settings then others. In this chapter we will specify different types of these devices and discuss their advantages.

Input methods Input methods can be classified using the following categories: – Locator – Stroke – String – Valuator – Choice – Pick

Input methods

Locator

A device that allows the user to specify one coordinate position. Different methods can be used, such as a mouse cursor, where a location is chosen by clicking a button, or a cursor that is moved using different keys on the keyboard. Touch screens can also be used as locators; the user specifies the location by inducing force onto the desired coordinate on the screen.

Stroke

A device that allows the user to specify a set of coordinate positions. The positions can be specified, for example, by dragging the mouse across the screen while a mouse button is kept pressed. On release, a second coordinate can be used to define a rectangular area using the first coordinate in addition.

String

A device that allows the user to specify text input. A text input widget in combination with the keyboard is used to input the text. Also, virtual keyboards displayed on the screen where the characters can be picked using the mouse can be used if keyboards are not available to the application.

Valuator

A device that allows the user to specify a scalar value. Similar to string inputs, numeric values can be specified using the keyboard. Often, up-down-arrows are added to increase or decrease the current value. Rotary devices, such as wheels can also be used for specifying numerical values. Often times, it is useful to limit the range of the numerical value depending on the value.

Choice

A device that allows the user to specify a menu option. Typical choice devices are menus or radio buttons which provide various options the user can choose from. For radio buttons, often only one option can be chosen at a time. Once another option is picked, the previous one gets cleared.

Pick

A device that allows the user to specify a component of a picture. Similar to locator devices, a coordinate is specified using the mouse or other cursor input devices and then back-projected into the scene to determine the selected 3-D object. It is often useful to allow a certain "error tolerance" so that an object is picked even though the user did not exactly onto the object but close enough next to it. Also, highlighting objects within the scene can be used to traverse through a list of objects that fulfill the proximity criterion.

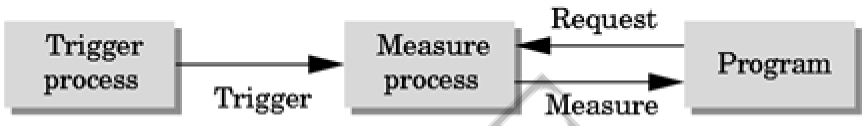
Certain applications do not allow the use of mouse or keyboard. In particular, 3-D environments, where the user roams freely within the scene, mouse or keyboard would unnecessarily bind the user to a certain location. Other input methods are required in these cases, such as a wireless gamepad or a 3-D stylus, that is tracked to identify its 3-D location.

Input Modes

In addition to multiple types of logical input devices, we can obtain the measure of a device in three distinct modes: 1) Request mode, 2) Sample mode, and 3) Event mode. It defined by the relationship between the measure process and the trigger. Normally, the initialization of an input device starts a measure process.

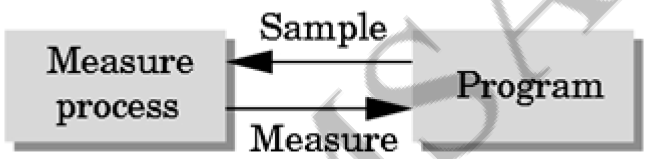
1) Request mode:

In this mode the measure of the device is not returned to the program until the device is triggered. A locator can be moved to different point of the screen. The Windows system continuously follows the location of the pointer, but until the button is depressed, the location will not be returned.



2) Sample mode:

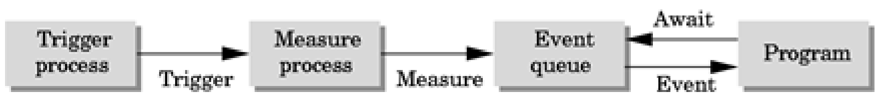
Input is immediate. As soon as the function call in the user program is encountered, the measure is returned, hence no trigger is needed. For both of the above modes, the user must identify which devices is to provide the input. request_locator (device_id, &measure); sample_locator (device_id, &measure); identifier location Think of a flight simulator with many input devices.



3) Event mode:

The previous two modes are not sufficient for handling the variety of possible human-computer interactions that arise in a modern computing environment. The can be done in three steps: 1) Show how event mode can be described as another mode within the measure trigger paradigm. 2) Learn the basics of client-servers when event mode is preferred, and 3) Learn how OpenGL uses GLUT to do this. In an environment with multiple input devices, each with its own trigger and each running a measure process. Each time that a device is triggered, an event is generated. The

device measure, with the identifier for the device, is placed in an event queue. The user program executes the events from the queue. When the queue is empty, it will wait until an event appears there to execute it. Another approach is to associate a function called a callback with a specific type of event. This is the approach we are taking.



Interactive picture construction techniques

Interactive picture- construction methods are commonly used in variety of applications, including design and painting packages. These methods provide user with the capability to position objects, to constrain fig. to predefined orientations or alignments, to sketch fig., and to drag objects around the screen. Grids, gravity fields, and rubber band methods are used to aid in positioning and other picture construction operations. The several techniques used for interactive picture construction that are incorporated into graphics packages are:

(1) Basic positioning methods:- coordinate values supplied by locator input are often used with positioning methods to specify a location for displaying an object or a character string. Coordinate positions are selected interactively with a pointing device, usually by positioning the screen cursor.

(2) Constraints:-A constraint is a rule for altering input coordinates values to produce a specified orientation or alignment of the displayed coordinates. The most common constraint is a horizontal or vertical alignment of straight lines.

(3) Grids:- Another kind of constraint is a grid of rectangular lines displayed in some part of the screen area. When a grid is used, any input coordinate position is rounded to the nearest intersection of two grid lines.

(4) Gravity field:- When it is needed to connect lines at positions between endpoints, the graphics packages convert any input position near a line to a position on the line. The conversion is accomplished by creating a gravity area around the line. Any related position within the gravity field of line is moved to the nearest position on the line. It illustrated with a shaded boundary around the line.

(5) Rubber Band Methods:- Straight lines can be constructed and positioned using rubber band methods which stretch out a line from a starting position as the screen cursor.

(6) Dragging:- This methods move object into position by dragging them with the screen cursor.

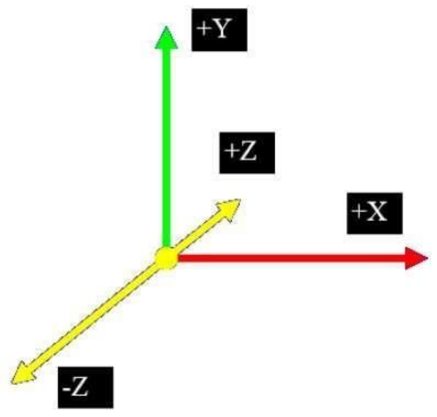
(7) Painting and Drawing:- Cursor drawing options can be provided using standard curve shapes such as circular arcs and splices, or with freehand sketching procedures. **Line** widths, line styles and other attribute options are also commonly found in painting and drawing packages.

THREE DIMENSIONAL DISPLAY METHODS

To obtain display of a three-dimensional scene that has been modeled in world coordinates. We must first set up a coordinate reference for the "camera". This coordinate reference defines the position and orientation for the plane of the camera film which is the plane we want to us to display a view of the objects in the scene. Object descriptions are then transferred to the camera reference coordinates and projected onto the selected display plane. We can then display the objects in wireframe (outline) form, or we **can** apply lighting surface rendering techniques to shade the visible surfaces.

In the 2D system, we use only two coordinates X and Y but in 3D, an extra coordinate Z is added. 3D graphics techniques and their application are fundamental to the entertainment, games, and computer-aided design industries. It is a continuing area of research in scientific visualization.

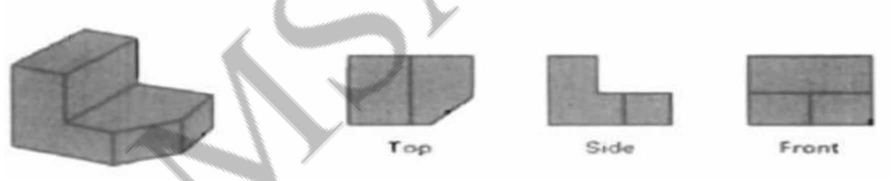
Furthermore, 3D graphics components are now a part of almost every personal computer and, although traditionally intended for graphics-intensive software such as games, they are increasingly being used by other applications.



Parallel Projection

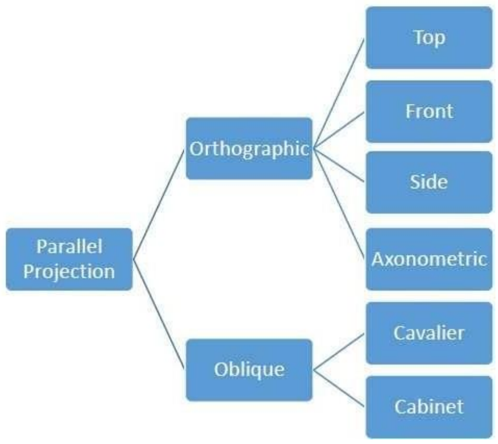
Parallel projection discards z-coordinate and parallel lines from each vertex on the object are extended until they intersect the view plane. In parallel projection, we specify a direction of projection instead of center of projection.

The appearance of the solid object can be reconstructed from the major views



In parallel projection, the distance from the center of projection to project plane is infinite. In this type of projection, we connect the projected vertices by line segments which correspond to connections on the original object.

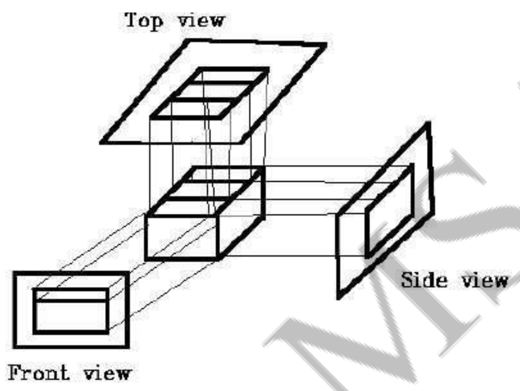
Parallel projections are less realistic, but they are good for exact measurements. In this type of projections, parallel lines remain parallel and angles are not preserved. Various types of parallel projections are shown in the following hierarchy.



Orthographic Projection

In orthographic projection the direction of projection is normal to the projection of the plane. There are three types of orthographic projections −

- Front Projection
- Top Projection
- Side Projection

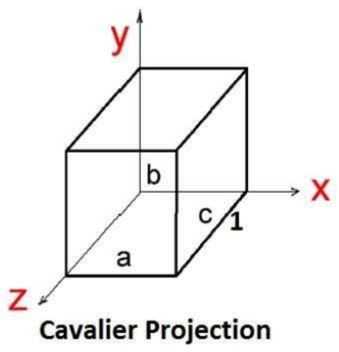


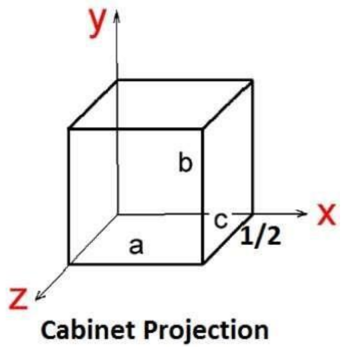
Oblique Projection

In oblique projection, the direction of projection is not normal to the projection of plane. In oblique projection, we can view the object better than orthographic projection.

There are two types of oblique projections − **Cavalier** and **Cabinet**. The Cavalier projection makes 45° angle with the projection plane. The projection of a line perpendicular to the view plane has the same length as the line itself in Cavalier projection. In a cavalier projection, the foreshortening factors for all three principal directions are equal.

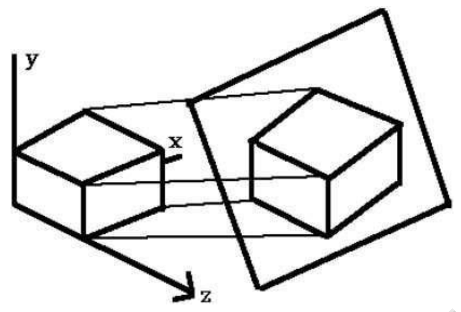
The Cabinet projection makes 63.4° angle with the projection plane. In Cabinet projection, lines perpendicular to the viewing surface are projected at ½ their actual length. Both the projections are shown in the following figure −





Isometric Projections

Orthographic projections that show more than one side of an object are called **axonometric orthographic projections**. The most common axonometric projection is an **isometric projection** where the projection plane intersects each coordinate axis in the model coordinate system at an equal distance. In this projection parallelism of lines are preserved but angles are not preserved. The following figure shows isometric projection −

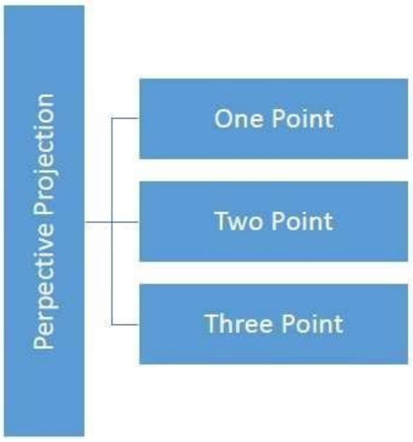


Perspective Projection

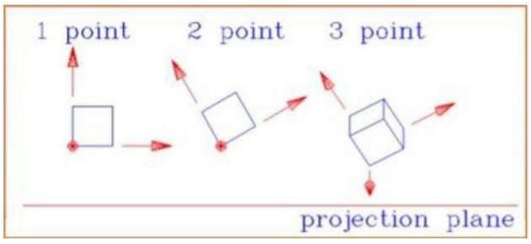
In perspective projection, the distance from the center of projection to project plane is finite and the size of the object varies inversely with distance which looks more realistic.

The distance and angles are not preserved and parallel lines do not remain parallel. Instead, they all converge at a single point called **center of projection** or **projection reference point**. There are 3 types of perspective projections which are shown in the following chart.

- **One point** perspective projection is simple to draw.
- **Two point** perspective projection gives better impression of depth.
- **Three point** perspective projection is most difficult to draw.



The following figure shows all the three types of perspective projection −



DEPTH CUEING

A simple method for indicating depth with wireframe displays is to vary the intensity of objects according to their distance from the viewing position. The viewing position are displayed with the highest intensities, and lines farther away are displayed with decreasing intensities.

Visible Line and Surface Identification

We can also clarify depth relation ships in a wireframe display by identifying visible lines in some way. The simplest method is to highlight the visible lines or to display them in a different color. Another technique, commonly used for engineering drawings, is to display the nonvisible lines as dashed lines. Another approach is to simply remove the nonvisible lines

Surface Rendering

Added realism is attained in displays by setting the surface intensity of objects according to the lighting conditions in the scene and according to assigned surface characteristics. Lighting specifications include the intensity and positions of light sources and the general background illumination required for a scene. Surface properties of objects include degree of transparency and how rough or smooth the surfaces are to be. Procedures can then be applied to generate the correct illumination and shadow regions for the scene.

Exploded and Cutaway View

Exploded and cutaway views of such objects can then be used to show the internal structure and relationship of the object Parts

Three-Dimensional and Stereoscopic View

Three-dimensional views can be obtained by reflecting a raster image from a vibrating flexible mirror. The vibrations of the mirror are synchronized with the display of the scene on the CRT. As the mirror vibrates, the focal length varies so that each point in the scene is projected to a position corresponding to its depth.

Stereoscopic devices present two views of a scene: one for the left eye and the other for the right eye.

Graphics Packages

A graphics package is an application that can be used to create and manipulate images on a computer.

There are two main types of graphics package:

- **painting packages**
- **drawing packages**

Painting packages

- A painting package produces images by changing the colour of pixels on the screen.
- These are coded as a pattern of bits to create a bitmapped graphics file.
- Bitmapped graphics are used for images such as scanned photographs or pictures taken with a digital camera.

Advantages

- The main advantage offered by this type of graphic is that individual pixels can be changed which makes very detailed editing possible.

Disadvantages of painting packages

- Individual parts of an image cannot be resized;
- only the whole picture can be increased or decreased in size.
- Information has to be stored about every pixel in an image which produces files that use large amounts of backing storage space.

Examples of graphics packages that produce bitmapped images include:- MS Paint, PC Paintbrush, Adobe Photoshop and JASC's Paint Shop Pro.

Drawing packages

- A drawing package produces images that are made up from coloured lines and shapes such as circles, squares and rectangles.
- When an image is saved it is stored in a vector graphics file as a series of instructions, which can be used to recreate it.

Main advantages of vector graphics are:

- They use less storage space than bitmap graphics;
- Each part of an image is treated as a separate object, which means that individual parts can be easily modified.

Disadvantages of drawing packages

- They don't look as realistic as bitmap graphics.

Examples of drawing graphics packages include CorelDraw, Micrographix Designer and computer aided design (CAD) packages such as AutoCAD.

MODULE - V

3D Geometric and Modeling Transformations: Translation – Scaling – Rotation – Other Transformations. Visible Surface Detection Methods: Classification of Visible Surface Detection Algorithm –Back face Detection – Depth-Buffer Method – A Buffer Method – Scan-Line Method –Applications of Computer Graphics.

3D Geometric and Modelling Transformations

Three Dimensional Transformations

The geometric transformations play a vital role in generating images of three Dimensional objects with the help of these transformations. The location of objects relative to others can be easily expressed. Sometimes viewpoint changes rapidly, or sometimes objects move in relation to each other. For this number of transformation can be carried out repeatedly.

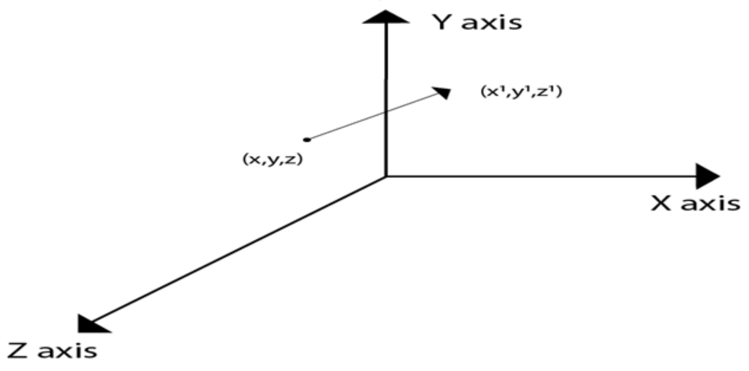
Translation

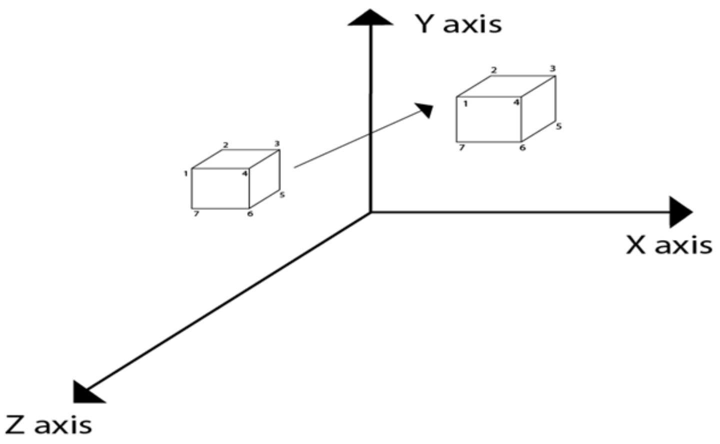
It is the movement of an object from one position to another position. Translation is done using translation vectors. There are three vectors in 3D instead of two. These vectors are in x, y, and z directions. Translation in the x-direction is represented using T_x. The translation is y-direction is represented using T_y. The translation in the z- direction is represented using T_z.

If P is a point having co-ordinates in three directions (x, y, z) is translated, then after translation its coordinates will be $(x^1\ y^1\ z^1)$ after translation. $T_x\ T_y\ T_z$ are translation vectors in x, y, and z directions respectively.

$$x^1 = x + T_x$$
$$y^1 = y + T_y$$
$$z^1 = z + T_z$$

Three-dimensional transformations are performed by transforming each vertex of the object. If an object has five corners, then the translation will be accomplished by translating all five points to new locations. Following figure 1 shows the translation of point figure 2 shows the translation of the cube.





Matrix for translation

$$\left\{\begin{matrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ T_x & T_y & T_z & 1 \end{matrix}\right\} \text{ or } \left\{\begin{matrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{matrix}\right\}$$

Matrix representation of point translation

$$\begin{pmatrix} x^1 \\ y^1 \\ z^1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

Point shown in fig is (x, y, z). It become (x^1,y^1,z^1) after translation. T_x T_y T_z are translation vector.

Example: A point has coordinates in the x, y, z direction i.e., (5, 6, 7). The translation is done in the x-direction by 3 coordinate and y direction. Three coordinates and in the z- direction by two coordinates. Shift the object. Find coordinates of the new position.

Solution: Co-ordinate of the point are (5, 6, 7)
Translation vector in x direction = 3
Translation vector in y direction = 3
Translation vector in z direction = 2

Translation matrix is

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ T_x & T_y & T_z & 1 \end{pmatrix}$$

Multiply co-ordinates of point with translation matrix

$$(x^1y^1z^1) = (5,6,7,1)\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 3 & 3 & 2 & 1\end{pmatrix}$$

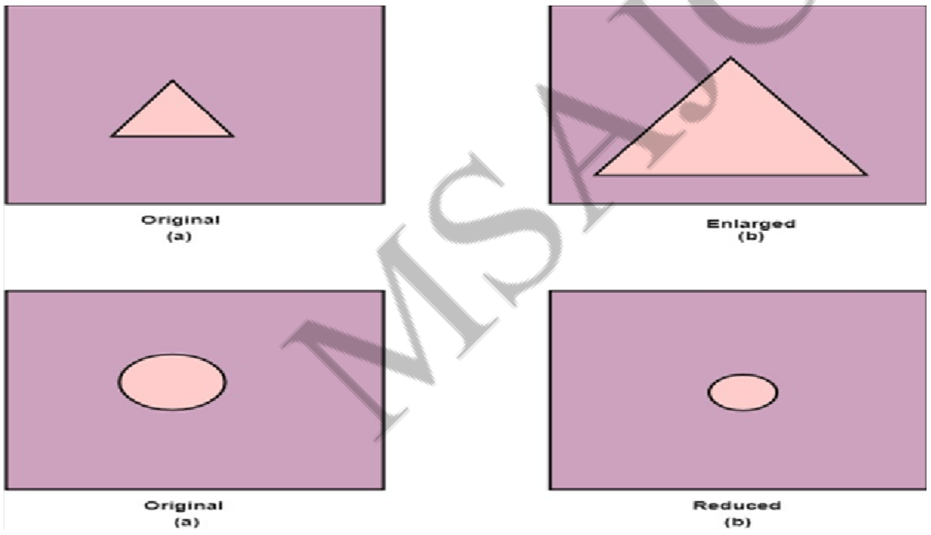
= [5+0+0+30+6+0+30+0+7+20+0+0+1] = [8991]

x becomes x^1=8
y becomes y^1=9
z becomes z^1=9

Scaling

Scaling is used to change the size of an object. The size can be increased or decreased. The scaling three factors are required S_x S_y and S_z.

S_x=Scaling factor in x- direction
S_y=Scaling factor in y-direction
S_z=Scaling factor in z-direction



Matrix for Scaling

$$\left\{\begin{matrix}S_x & 0 & 0 & 0\\ 0 & S_y & 0 & 0\\ 0 & 0 & S_z & 0\\ 0 & 0 & 0 & 1\end{matrix}\right\}$$

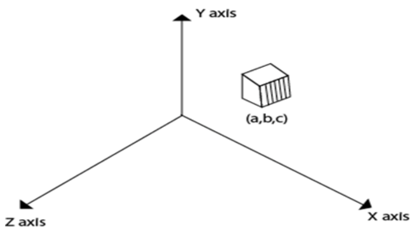
Scaling of the object relative to a fixed point

Following are steps performed when scaling of objects with fixed point (a, b, c). It can be represented as below:

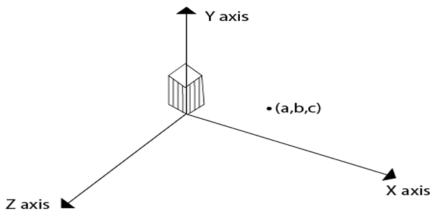
1. Translate fixed point to the origin
2. Scale the object relative to the origin
3. Translate object back to its original position.

Note: If all scaling factors $S_x=S_y=S_z$.Then scaling is called as uniform. If scaling is done with different scaling vectors, it is called a differential scaling.

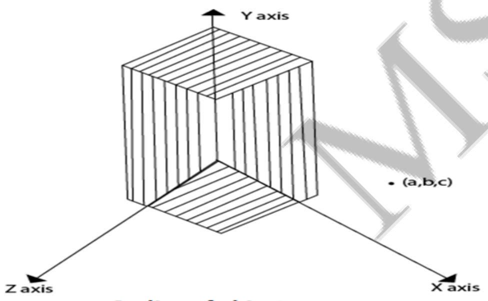
In figure (a) point (a, b, c) is shown, and object whose scaling is to done also shown in steps in fig (b), fig (c) and fig (d).



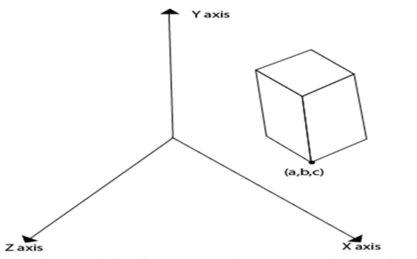
Object & point for scaling
(a)



Object transfer at origin
(b)



Scaling of object
(c)

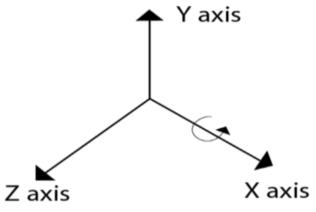


Object shifted to original position after scaling
(d)

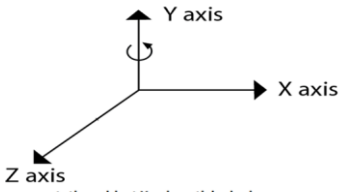
Rotation

It is moving of an object about an angle. Movement can be anticlockwise or clockwise. 3D rotation is complex as compared to the 2D rotation. For 2D we describe the angle of rotation, but for a 3D angle of rotation and axis of rotation are required. The axis can be either x or y or z.

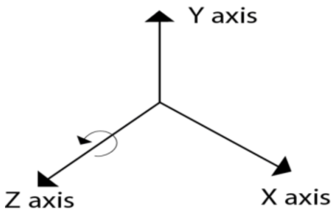
Following figures shows rotation about x, y, z- axis



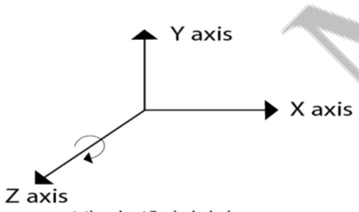
rotation object X axis anticlockwise



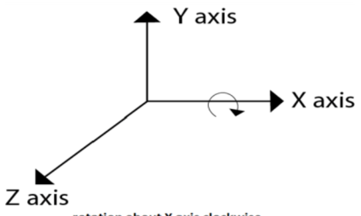
rotation object Y axis anticlockwise



rotation object Z axis anticlockwise

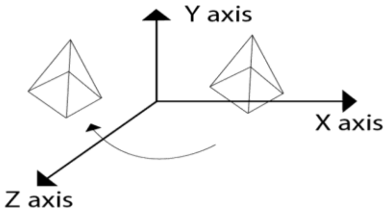


rotation about Z axis clockwise

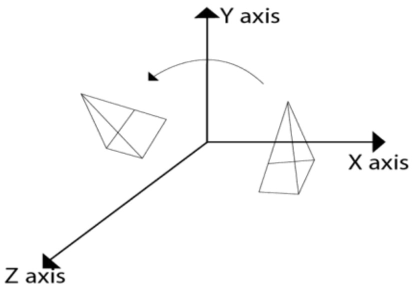


rotation about X axis clockwise

Following figure show rotation of the object about the Y axis



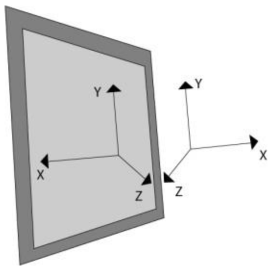
Following figure show rotation of the object about the Z axis



Reflection

It is also called a mirror image of an object. For this reflection axis and reflection of plane is selected. Three-dimensional reflections are similar to two dimensions. Reflection is 180° about the given axis. For reflection, plane is selected (xy,xz or yz). Following matrices show reflection respect to all these three planes.

Reflection relative to XY plane



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

Reflection relative to YZ plane

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

Reflection relative to ZX plane

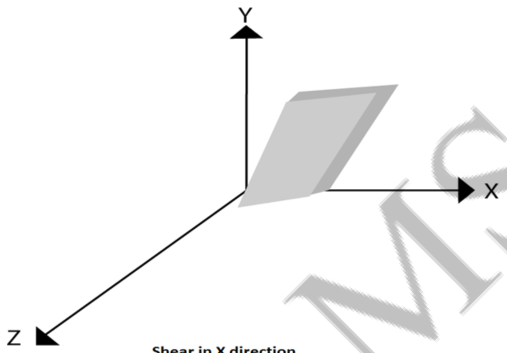
$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Shearing

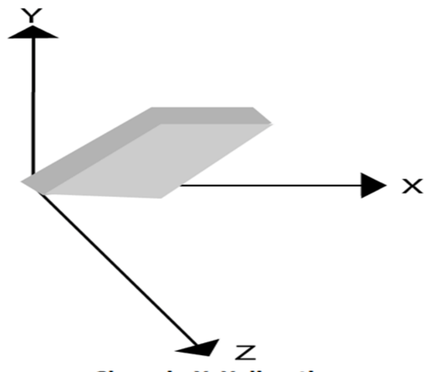
It is change in the shape of the object. It is also called as deformation. Change can be in the x -direction or y -direction or both directions in case of 2D. If shear occurs in both directions, the object will be distorted. But in 3D shear can occur in three directions.

Matrix for shear

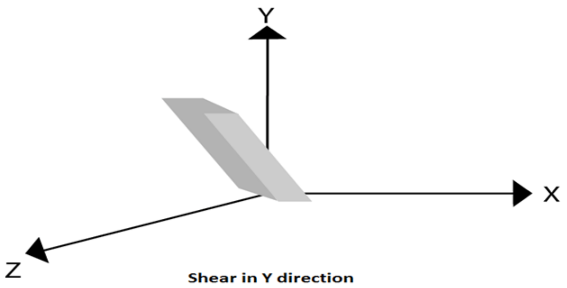
$$\begin{pmatrix} 1 & 0 & a & 0 \\ 0 & 1 & b & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



Shear in X direction
(a)



Shear in X, Y direction
(b)



Shear in Y direction
(c)

Classification of Visible Surface Detection Method

Hidden Surface Removal

1. One of the most challenging problems in computer graphics is the removal of hidden parts from images of solid objects.
2. In real life, the opaque material of these objects obstructs the light rays from hidden parts and prevents us from seeing them.
3. In the computer generation, no such automatic elimination takes place when objects are projected onto the screen coordinate system.
4. Instead, all parts of every object, including many parts that should be invisible are displayed.
5. To remove these parts to create a more realistic image, we must apply a hidden line or hidden surface algorithm to set of objects.
6. The algorithm operates on different kinds of scene models, generate various forms of output or cater to images of different complexities.
7. All use some form of geometric sorting to distinguish visible parts of objects from those that are hidden.
8. Just as alphabetical sorting is used to differentiate words near the beginning of the alphabet 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 to reduce the computing required to generate an image.
11. Different types of coherence are related to different forms of order or regularity in the image.
12. Scan line coherence arises because the display of a scan line in a raster image is usually very similar to the display of the preceding scan line.
13. **Frame coherence** in a sequence of images designed to show motion recognizes that successive frames are very similar.
14. **Object coherence** results from relationships between different objects or between separate parts of the same objects.

15. A hidden surface algorithm is generally designed to exploit one or more of these coherence properties to increase efficiency.
16. Hidden surface algorithm bears a strong resemblance to two-dimensional scan conversions.

Types of hidden surface detection algorithms

1. Object space methods
2. 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.

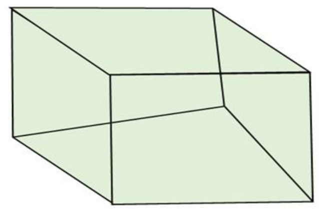
Differentiate between Object space and Image space method

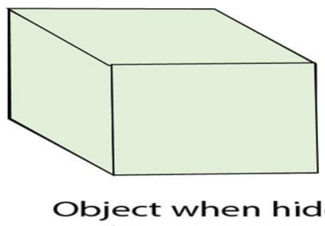
Object Space	**Image Space**
1. Image space is object based. It concentrates on geometrical relation among objects in the scene.	1. It is a pixel-based method. It is concerned with the final image, what is visible within each raster pixel.
2. Here surface visibility is determined.	2. Here line visibility or point visibility is determined.
3. It is performed at the precision with which each object is defined, No resolution is considered.	3. It is performed using the resolution of the display device.

4. Calculations are not based on the resolution of the display so change of object can be easily adjusted.	4. Calculations are resolution base, so the change is difficult to adjust.
5. These were developed for vector graphics system.	5. These are developed for raster devices.
6. Object-based algorithms operate on continuous object data.	6. These operate on object data.
7. Vector display used for object method has large address space.	7. Raster systems used for image space methods have limited address space.
8. Object precision is used for application where speed is required.	8. There are suitable for application where accuracy is required.
9. It requires a lot of calculations if the image is to enlarge.	9. Image can be enlarged without losing accuracy.
10. If the number of objects in the scene increases, computation time also increases.	10. 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 are objected 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:

1. Sorting
2. Coherence
3. Machine

Sorting: All surfaces are sorted in two classes, i.e., visible and invisible. Pixels are colored accordingly. Several sorting algorithms are available i.e.

1. Bubble sort
2. Shell sort
3. Quick sort
4. Tree sort
5. 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

1. Edge coherence
2. Object coherence
3. Face coherence
4. Area coherence
5. Depth coherence
6. Scan line coherence
7. Frame coherence
8. Implied edge coherence

1. Edge coherence: The visibility of edge changes when it crosses another edge or it also penetrates a visible edge.

2. 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.

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

4. Area coherence: It is used to group of pixels cover by same visible face.

5. 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.

6. Scan line coherence: The object is scanned using one scan line then using the second scan line. The intercept of the first line.

7. Frame coherence: It is used for animated objects. It is used when there is little change in image from one frame to another.

8. Implied edge coherence: If a face penetrates in another, line of intersection can be determined from two points of intersection.

Algorithms used for hidden line surface detection

1. Back Face Removal Algorithm
2. Z-Buffer Algorithm
3. Painter Algorithm
4. Scan Line Algorithm
5. Subdivision Algorithm
6. Floating horizon Algorithm

Visible Surface Detection Algorithm

When we view a picture containing non-transparent objects and surfaces, then we cannot see those objects from view which are behind from objects closer to eye. We must remove these hidden surfaces to get a realistic screen image. The identification and removal of these surfaces is called **Hidden-surface problem**.

There are two approaches for removing hidden surface problems − **Object-Space method** and **Image-space method**. The Object-space method is implemented in physical coordinate system and image-space method is implemented in screen coordinate system.

When we want to display a 3D object on a 2D screen, we need to identify those parts of a screen that are visible from a chosen viewing position.

Depth Buffer Z−Buffer Method

This method is developed by Cutmull. It is an image-space approach. The basic idea is to test the Z-depth of each surface to determine the closest visiblevisible surface.

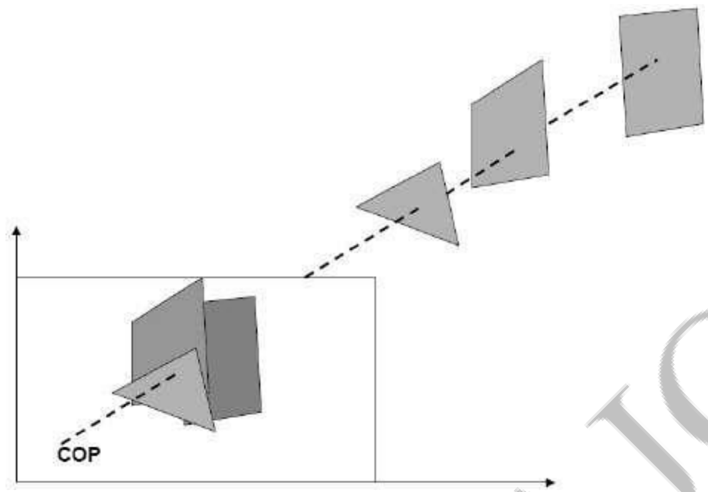
In this method each surface is processed separately one pixel position at a time across the surface. The depth values for a pixel are compared and the closest smallestzsmallestz surface determines the color to be displayed in the frame buffer.

It is applied very efficiently on surfaces of polygon. Surfaces can be processed in any order. To override the closer polygons from the far ones, two buffers named **frame buffer** and **depth buffer,** are used.

Depth buffer is used to store depth values for x,yx,y position, as surfaces are processed 0≤depth≤10≤depth≤1.

The **frame buffer** is used to store the intensity value of color value at each position x,yx,y.

The z-coordinates are usually normalized to the range [0, 1]. The 0 value for z-coordinate indicates back clipping pane and 1 value for z-coordinates indicates front clipping pane.



Algorithm

Step-1 − Set the buffer values −

Depthbuffer x,yx,y = 0

Framebuffer x,yx,y = background color

Step-2 − Process each polygon OneatatimeOneatatime

For each projected x,yx,y pixel position of a polygon, calculate depth z.

If Z > depthbuffer x,yx,y

Compute surface color,

set depthbuffer x,yx,y = z,

framebuffer x,yx,y = surfacecolor x,yx,y

Advantages

- It is easy to implement.
- It reduces the speed problem if implemented in hardware.
- It processes one object at a time.

Disadvantages

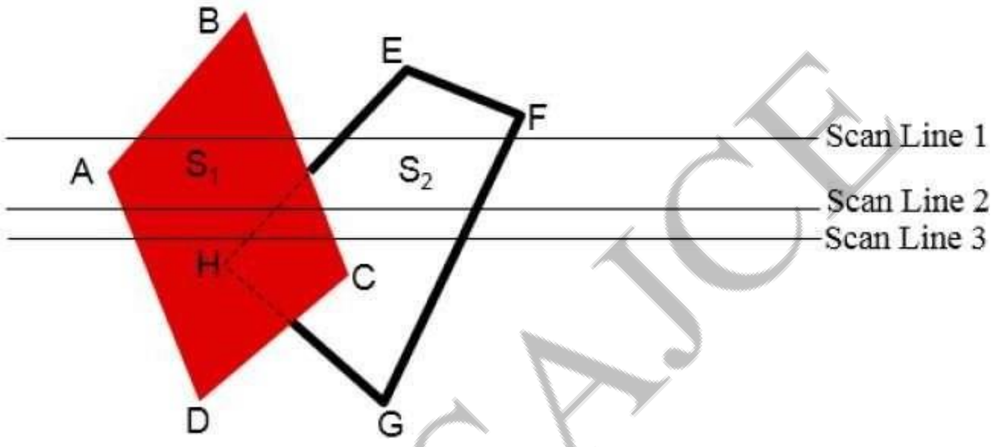
- It requires large memory.
- It is time consuming process.

Scan-Line Method

It is an image-space method to identify visible surface. This method has a depth information for only single scan-line. In order to require one scan-line of depth values, we must group and process all polygons intersecting a given scan-line at the same time before processing the next scan-line. Two important tables, **edge table** and **polygon table,** are maintained for this.

The Edge Table − It contains coordinate endpoints of each line in the scene, the inverse slope of each line, and pointers into the polygon table to connect edges to surfaces.

The Polygon Table − It contains the plane coefficients, surface material properties, other surface data, and may be pointers to the edge table.



To facilitate the search for surfaces crossing a given scan-line, an active list of edges is formed. The active list stores only those edges that cross the scan-line in order of increasing x. Also a flag is set for each surface to indicate whether a position along a scan-line is either inside or outside the surface.

Pixel positions across each scan-line are processed from left to right. At the left intersection with a surface, the surface flag is turned on and at the right, the flag is turned off. You only need to perform depth calculations when multiple surfaces have their flags turned on at a certain scan-line position.

Back-Face Detection

A fast and simple object-space method for identifying the back faces of a polyhedron is based on the "inside-outside" tests. A point x,y,zx,y,z is "inside" a polygon surface with plane parameters A, B, C, and D if When an inside point is along the line of sight to the surface, the polygon must be a back face weareinsidethatfaceandcannotseethefrontofitfromourviewingpositionweareinsidethatface andcannotseethefrontofitfromourviewingposition.

We can simplify this test by considering the normal vector **N** to a polygon surface, which has Cartesian components A,B,CA,B,C.

In general, if V is a vector in the viewing direction from the eye or"camera"or"camera" position, then this polygon is a back face if

V.N > 0

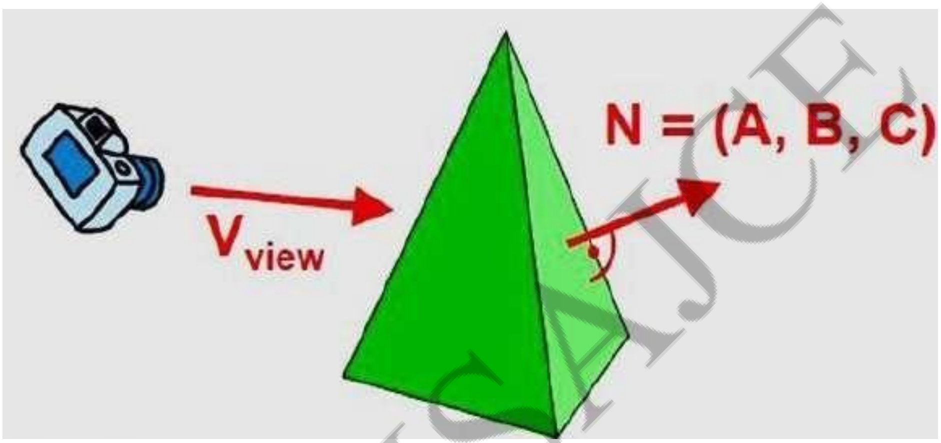
Furthermore, if object descriptions are converted to projection coordinates and your viewing direction is parallel to the viewing z-axis, then −

V = (0, 0, V_z) and V.N = V_ZC

So that we only need to consider the sign of C the component of the normal vector **N**.

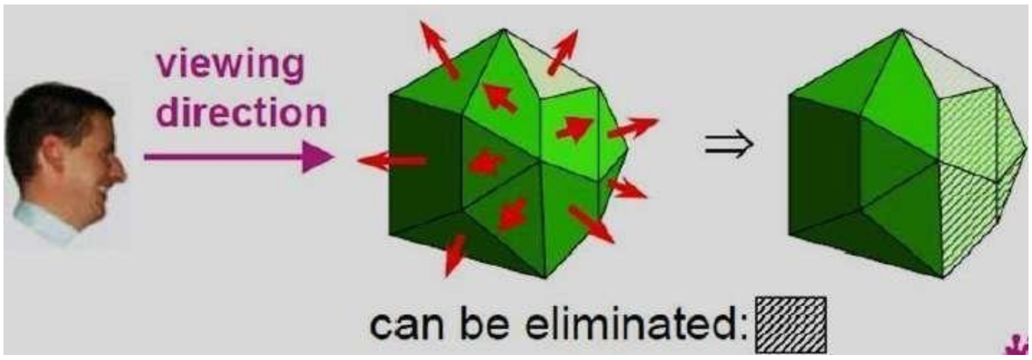
In a right-handed viewing system with viewing direction along the negative ZVZV axis, the polygon is a back face if C < 0. Also, we cannot see any face whose normal has z component C = 0, since your viewing direction is towards that polygon. Thus, in general, we can label any polygon as a back face if its normal vector has a z component value −

C <= 0



Similar methods can be used in packages that employ a left-handed viewing system. In these packages, plane parameters A, B, C and D can be calculated from polygon vertex coordinates specified in a clockwise direction unlikethecounterclockwisedirectionusedinaright−handedsystemunlikethecounterclockwisedirectionusedinaright−handedsystem.

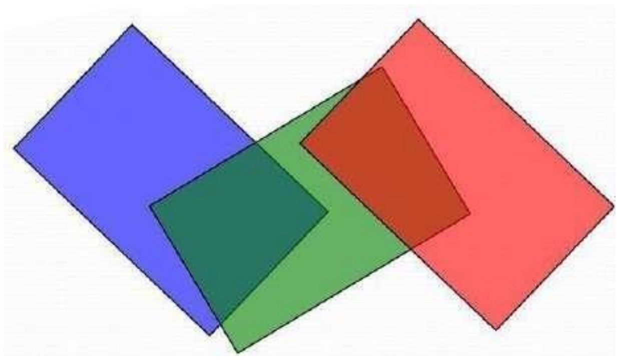
Also, back faces have normal vectors that point away from the viewing position and are identified by C >= 0 when the viewing direction is along the positive ZvZv axis. By examining parameter C for the different planes defining an object, we can immediately identify all the back faces.



A-Buffer Method

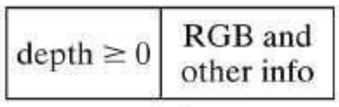
The A-buffer method is an extension of the depth-buffer method. The A-buffer method is a visibility detection method developed at Lucas film Studios for the rendering system Renders Everything You Ever Saw REYESREYES.

The A-buffer expands on the depth buffer method to allow transparencies. The key data structure in the A-buffer is the accumulation buffer.

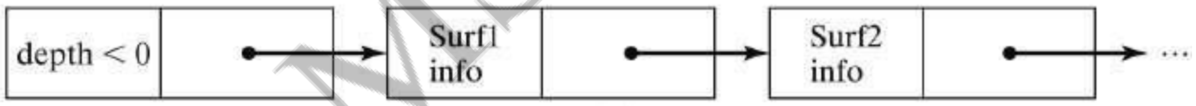


Each position in the A-buffer has two fields −

- **Depth field** − It stores a positive or negative real number
- **Intensity field** − It stores surface-intensity information or a pointer value



(a)



(b)

If depth >= 0, the number stored at that position is the depth of a single surface overlapping the corresponding pixel area. The intensity field then stores the RGB components of the surface color at that point and the percent of pixel coverage.

If depth < 0, it indicates multiple-surface contributions to the pixel intensity. The intensity field then stores a pointer to a linked list of surface data. The surface buffer in the A-buffer includes −

- RGB intensity components
- Opacity Parameter
- Depth
- Percent of area coverage
- Surface identifier

The algorithm proceeds just like the depth buffer algorithm. The depth and opacity values are used to determine the final color of a pixel.

Application of Computer Graphics

1. Education and Training: Computer-generated model of the physical, financial and economic system is often used as educational aids. Model of physical systems, physiological system, population trends or equipment can help trainees to understand the operation of the system.

For some training applications, particular systems are designed. For example Flight Simulator.

Flight Simulator: It helps in giving training to the pilots of airplanes. These pilots spend much of their training not in a real aircraft but on the ground at the controls of a Flight Simulator.

Advantages:

1. Fuel Saving
2. Safety
3. Ability to familiarize the training with a large number of the world's airports.

2. Use in Biology: Molecular biologist can display a picture of molecules and gain insight into their structure with the help of computer graphics.

3. Computer-Generated Maps: Town planners and transportation engineers can use computer-generated maps which display data useful to them in their planning work.

4. Architect: Architect can explore an alternative solution to design problems at an interactive graphics terminal. In this way, they can test many more solutions that would not be possible without the computer.

5. Presentation Graphics: Example of presentation Graphics are bar charts, line graphs, pie charts and other displays showing relationships between multiple parameters. Presentation Graphics is commonly used to summarize

- Financial Reports
- Statistical Reports
- Mathematical Reports
- Scientific Reports
- Economic Data for research reports
- Managerial Reports
- Consumer Information Bulletins
- And other types of reports

6. Computer Art: Computer Graphics are also used in the field of commercial arts. It is used to generate television and advertising commercial.

7. Entertainment: Computer Graphics are now commonly used in making motion pictures, music videos and television shows.

8. Visualization: It is used for visualization of scientists, engineers, medical personnel, business analysts for the study of a large amount of information.

9. Educational Software: Computer Graphics is used in the development of educational software for making computer-aided instruction.

10. Printing Technology: Computer Graphics is used for printing technology and textile design.

Example of Computer Graphics Packages:

1. LOGO
2. COREL DRAW
3. AUTO CAD
4. 3D STUDIO
5. CORE
6. GKS (Graphics Kernel System)
7. PHIGS
8. CAM (Computer Graphics Metafile)
9. CGI (Computer Graphics Interface)