STUDY MATERIAL

Unit	**:**	I
Semester	**:**	VI
Class	**:**	M.Sc [Software Engineering]- Third Year
Subject	**:**	Computer Graphics

UNIT–I :

Introduction And Hardware - Representative User Of Computer Graphics – Vector Display And Raster Display Architectures – Display Processor – Interactive Input Devices – Output Primitives – Software Portability And Graphics Standards – Conceptual Frame Work For Interactive Graphics.

What is computer Graphics?

The term of Graphics comes from Greek "graphikos" which means 'something written' e.g. autograph. So, Graphics are visual images or designs on some surface, such as a wall, canvas, screen, paper, or stone to inform, illustrate, or entertain

Computer graphics is an art of drawing pictures, lines, charts, etc. using computers with the help of programming. Computer graphics image is made up of number of pixels.

What is Pixel?

Pixel is the smallest addressable graphical unit represented on the computer screen. Pixels are normally arranged in a regular **2D grid**, and are often represented using **dots** or **squares.** Each pixel is a sample of an original image, where more samples typically provide a more accurate representation of the original. The **intensity** of each pixel is variable; in color systems, each pixel has typically three or four components such as red, green, and blue, or cyan, magenta, yellow, and black.

In simple words, using a computer as a rendering tool for the generation (from models) and manipulation of images is called computer graphics.

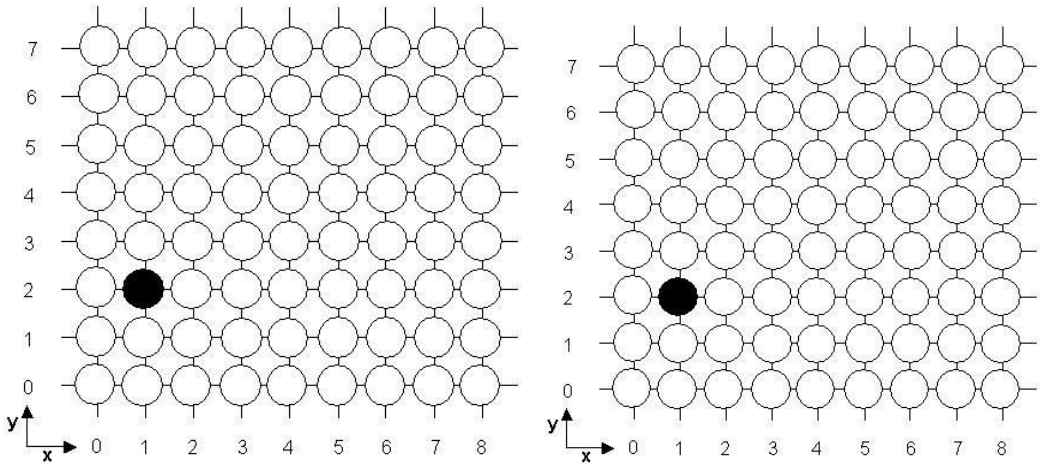
- Models - objects constructed from geometric primitives (points, lines, polygons) specified by their vertices.
- rendering-creating images from models
- Mathematics + Computer Science + Art = computer graphics
- CG is the sub part of "Compute Science" which studies about manipulating visual content, Synthesizing digitally

Introduction

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

Frame Buffer

- Each screen pixel corresponds to a particular entry in a 2D array residing in memory. This memory is called a frame buffer or a bit map.
- The number of **rows** in the frame buffer equals to the number of **raster lines** on the display screen.
- The number of **columns** in this array equals to the number of **pixels** on each raster line.
- The term pixel is also used to describe the row and the column location in the frame buffer array that corresponds to the screen location. A 512x512 display screen requires 262144 pixel memory locations.
- Whenever we wish to display a pixel on the screen, a specific value is placed into the corresponding memory location in the frame buffer array.
- Each screen pixel's location and corresponding memory's location in the frame buffer is accessed by nonnegative integer coordinate pair (x, y).
- The **x** value refers to the **column**, the **y** value to the **row** position.
- The origin of this coordinate system is positioned at the bottom-left corner of the screen or it is positioned at the upper-left corner of the screen.



Interactive Computer Graphics:

Interactive Computer Graphics involves a two way communication between computer and user. Here the observer is given some control over the image by providing him with an input device for example the video game controller of the ping pong game. This helps him to signal his request to the computer.

The computer on receiving signals from the input device can modify the displayed picture appropriately. To the user it appears that the picture is changing instantaneously in response to his commands. He can give a series of commands, each one generating a graphical response from the computer. In this way he maintains a conversation, or dialogue, with the computer.

Interactive computer graphics affects our lives in a number of indirect ways. For example, it helps to train the pilots of our airplanes. We can create a flight simulator which may help the pilots to get trained not in a real aircraft but on the grounds at the control of the flight simulator. The flight simulator is a mock up of an aircraft flight deck, containing all the usual controls and surrounded by screens on which we have the projected computer generated views of the terrain visible on take off and landing.

Flight simulators have many advantages over the real aircrafts for training purposes, including fuel savings, safety, and the ability to familiarize the trainee with a large number of the world's airports.

Non Interactive Computer Graphics: In non-interactive computer graphics otherwise known as passive computer graphics. It is the computer graphics in which user does not have any kind of control over the image. Image is merely the product of static stored program and will work according to the instructions given in the program linearly. The image is totally under the control of program instructions not under the user. Example: screen savers. Applications of 3D Graphics are TV Shows, Video Games, movie we

watch often. The Programs used to make 3D Graphics are Maya, Blender, 3D Studio Max, Bryce and more.

Representative User of Computer Graphics

Advantages of computer graphics

- Computer graphics is one of the most effective and commonly used ways of communication with computer.
- It provides tools for producing picture of "real-world" as well as synthetic objects such as mathematical surfaces in 4D and of data that have no inherent geometry such as survey result.
- It has ability to show moving pictures thus possible to produce animations with computer graphics.
- With the use of computer graphics we can control the animation by adjusting the speed, portion of picture in view the amount of detail shown and so on.
- It provides tools called motion dynamics. In which user can move objects as well as observes as per requirement for example walk throw made by builder to show flat interior and surrounding.
- It provides facility called update dynamics. With this we can change the shape color and other properties of object.
- Now in recent development of digital signal processing and audio synthesis chip the interactive graphics can now provide audio feedback along with the graphical feed backs.

Application of computer graphics

Computer graphics are very useful. Today almost every computer can do some graphics, and people have even come to expect to control their computer through icons and pictures rather than just by typing.

Computer-generated imagery is used for movie making, video game and computer program development, scientific modeling, and design for catalogs and other commercial art. Some people even make computer graphics as art.

There are several uses of CG which are very useful in the current scenario. Some of its uses include,

User interface (Graphical User Interfaces): - Visual object which we observe on screen which communicates with user is one of the most useful applications of the computer graphics. CG is effectively used to make Menus, Icons (Graphical Symbols),to make window manager(multiple windows). Plotting of graphics and chart in industry,

business, government and educational organizations drawing like bars, pie-charts, histogram's are very useful for quick and good decision making.

Office automation and desktop publishing: - It is used for creation and dissemination of information. It is used in -house creation and printing of documents which contains text, tables, graphs and other forms of drawn or scanned images or picture.

Computer aided drafting and design/ Computer aided Manufacturing (CADD/CAM): - It uses graphics to design components and system such as automobile bodies , structures of building etc. Most of engineering and Architecture students are concerned with Design. CAD is used to design various structures such as Computers, Aircrafts, Building, in almost all kinds of Industries (where designing is necessary).

Simulation and Animation: - Use of graphics in simulation makes mathematic models and mechanical systems more realistic and easy to study.

Computer Art and Commerce: - There are many tools provided by graphics which allows used to make their picture animated and attracted which are used in advertising. If we are intelligent enough we can rock by making creative arts using these Graphics tools. For making these arts we generally use CAD packages, paint and Paint brush programs and in animation too. Computer Arts Examples include Logo design (for companies, college, Industries, Institutions), Cartoon drawing, Product advertisements and many.

Cartography: - Computer graphics is also used to represent geographic maps, weather maps, oceanographic charts etc.

Education and training: - In the learning process, acquiring knowledge and skills required for better career path CG is needed. Computer graphics can be used to generate models of physical, financial and economic systems. These models can be used as educational aids. We regularly love to have computerized model to understand any topic easily.

Image processing (Medical): - It is used to process image by changing property of the image. Basically CG is used to improve the picture quality, and visualizing effects. Some of its applications include,

i) Tomography

ii) Ultrasonic medical scanners

iii) Picture enhancements

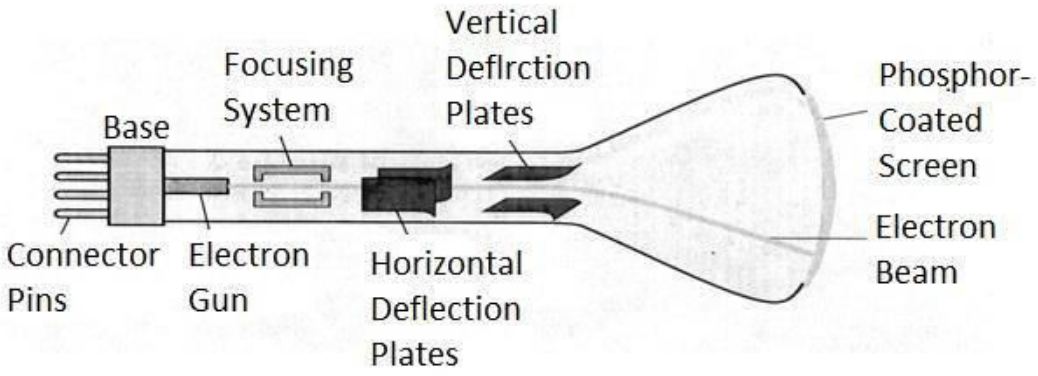
Virtual Reality: Virtual reality (VR) is a technology which allows a user to interact with a computer-simulated environment. The simulated environment can be similar to the real world, for example, simulations for pilot or combat training, or it can differ significantly from reality, as in VR games. It is currently very difficult to create a high-fidelity virtual reality experience, due largely to technical limitations on processing power, image resolution and communication bandwidth. Virtual Reality is often used to describe a wide variety of applications, commonly associated with its immersive, highly visual, 3D environments.

Vector Display and Raster Display Architectures

Display devices

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

Cathode-ray-tube



Cathode ray tube.

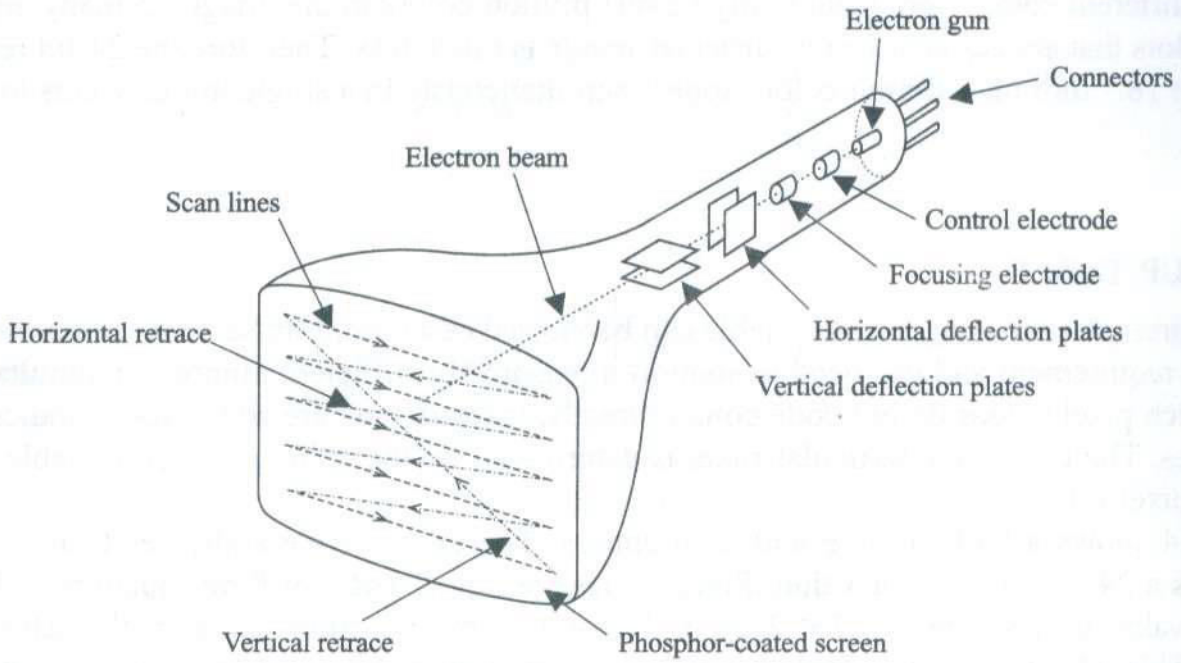
- It is an evacuated glass tube.
- An electron gun at the rear of the tube produce a beam of electrons which is directed towards the screen of the tube by a high voltage typically 15000 to 20000 volts
- Inner side screen is coated with phosphor substance which gives light when it is stroked bye electrons.
- Control grid controls velocity of electrons before they hit the phosphor.

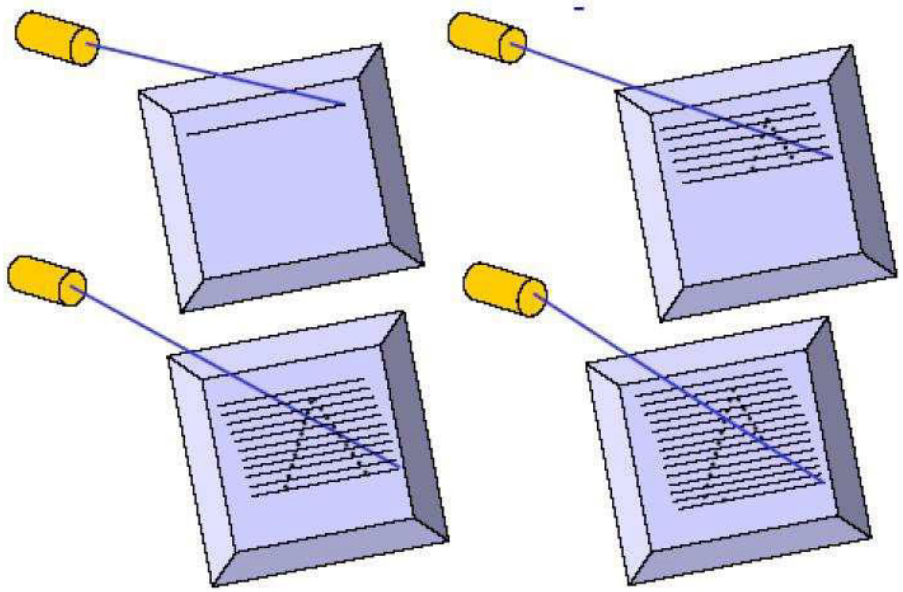
- The control grid voltage determines how many electrons are actually in the electron beam. The negative the control voltage is the fewer the electrons that pass through the grid.
- Thus control grid controls Intensity of the spot where beam strikes the screen.
- The focusing system concentrates the electron beam so it converges to small point when hits the phosphor coating.
- Deflection system directs beam which decides the point where beam strikes the screen.
- Deflection system of the CRT consists of two pairs of parallel plates which are vertical and horizontal deflection plates.
- Voltage applied to vertical and horizontal deflection plates is control vertical and horizontal deflection respectively.
- There are two techniques used for producing images on the CRT screen:
    1. Raster scan display
    2. Vector scan/Random scan display.

Raster scan display

WORKING

- 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.
- The return to the left of the screen, after refreshing each scan line is called **Horizontal retrace**.
- At the end of each frame the electron beam returns to the top left corner of the screen to begin the next frame is called **Vertical retrace**.
- Picture definition is stored in a memory area called the **refresh buffer** or **frame buffer**. • **Refresh buffer** or **frame buffer** is memory area that holds the set of intensity values for all the screen points. Stored intensity values then retrieved from refresh buffer and "**painted**" on the screen one row (**scan line**) at a time.



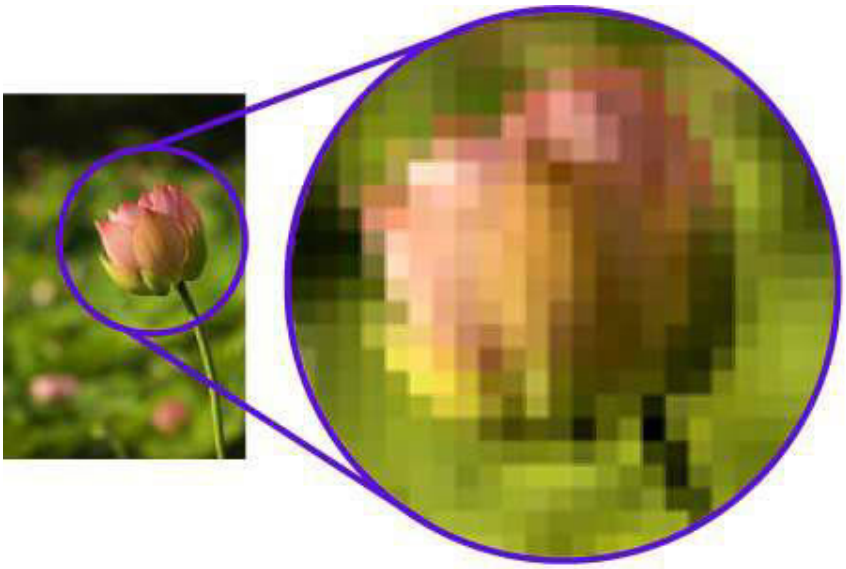


The quality of a raster image is determined by the total number pixels (**resolution**), and the amount of information in each pixel (**color depth**) A black-and-white system: each screen point is either on or off, so only **one bit** per pixel is needed to control the intensity of screen positions. Such type of frame buffer is called Bit map.

DISADVANTAGE

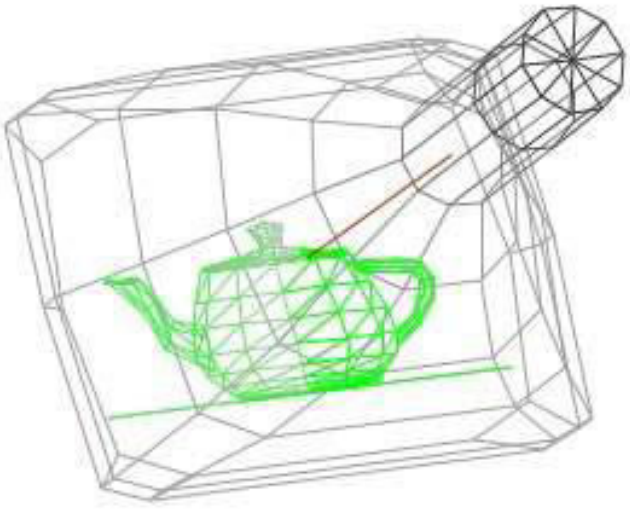
• To increase size of a raster image the pixels defining the image are be increased in either number or size Spreading the pixels over a larger area causes the image to lose detail and clarity.

• Produces jagged lines that are plotted as discrete Points.



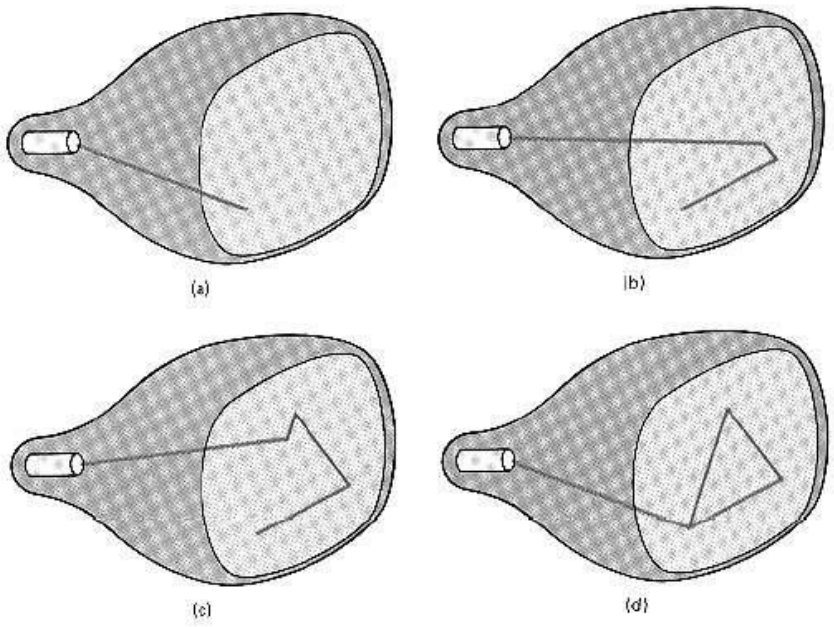
Random scan display

Random scan display is the use of geometrical primitives such as points, lines, curves, and polygons, which are all based upon **mathematical.**



WORKING

- When operated as a random-scan display unit, a CRT has the electron beam directed only to the parts of the screen where a picture is to be drawn.
- Random-scan monitors draw a picture one line at a time and for this reason are also referred to as vector displays (or stroke-writing or calligraphic displays).



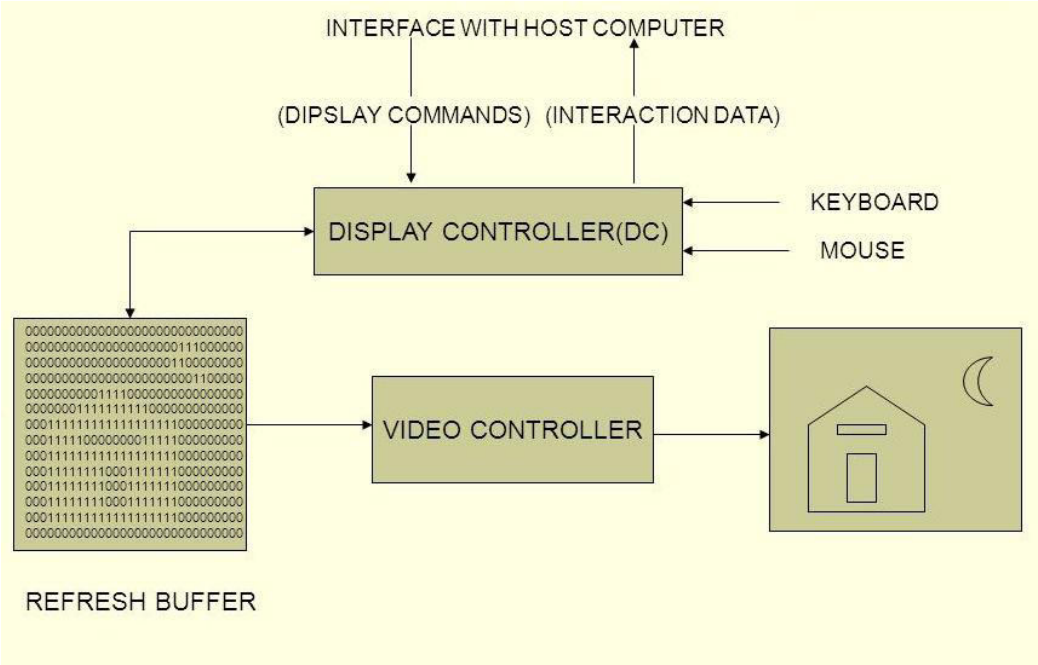
- **Refresh rate** depends on the number of lines to be displayed.
- Picture definition is now stored as a line drawing commands an area of memory referred to as **refresh display file** (**display list**).
- To display a picture, the system cycle through the **set of commands** in the display file, drawing each component line in turn.
- Random scan displays are designed to draw all the component lines of a picture 30 to 60 times each second.
- Vector scan display directly traces out only the desired lines on CRT.
- If we want line between point p1 & p2 then we directly drive the beam deflection circuitry which focus beam directly from point p1 to p2.
- If we do not want to display line from p1 to p2 and just move then we can blank the beam as we move it.
- To move the beam across the CRT, the information about both magnitude and direction is required. This information is generated with the help of vector graphics generator.
- Fig. 1.2 shows architecture of vector display. It consists of display controller, CPU, display buffer memory and CRT.
- Display controller is connected as an I/O peripheral to the CPU.
- Display buffer stores computer produced display list or display program.
- The Program contains point & line plotting commands with end point co-ordinates as well as character plotting commands.

- Display controller interprets command and sends digital and point co-ordinates to a vector generator.
- Vector generator then converts the digital co-ordinate value to analog voltages for beam deflection circuits that displace an electron beam which points on the CRT's screen.
- In this technique beam is deflected from end point to end point hence this techniques is also called random scan.
- We know as beam strikes phosphors coated screen it emits light but that light decays after few milliseconds and therefore it is necessary to repeat through the display list to refresh the screen at least 30 times per second to avoid flicker.
- As display buffer is used to store display list and used to refreshing, it is also called refresh buffer.

Raster scan display Architecture

- Figure shows the architecture of Raster display. It consists of display controller, CPU, video controller, refresh buffer, keyboard, mouse and CRT.
- The display image is stored in the form of 1's and 0's in the refresh buffer.
- The video controller reads this refresh buffer and produces the actual image on screen.
- It will scan one line at a time from top to bottom & then back to the top.
- The screen image is maintained by repeatedly scanning the same image. This process is known as Refreshing of Screen.
- In raster scan displays a special area of memory is dedicated to graphics only. This memory is called Frame Buffer.
- Frame buffer holds set of intensity values for all the screen points.
- That intensity is retrieved from frame buffer and display on screen one row at a time.
- Each screen point referred as pixel or **Pel (Picture Element).**
- Each pixel can be specified by its row and column numbers.
- It can be simply black and white system or color system.
- In simple black and white system each pixel is either ON or OFF, so only one bit per pixel is needed.
- Additional bits are required when color and intensity variations can be displayed up to 24-bits per pixel are included in high quality display systems.

- On a black and white system with one bit per pixel the frame buffer is commonly called a **Bitmap.** And for systems with multiple bits per pixel, the frame buffer is often referred as a **Pixmap.**



Random scan display Architecture

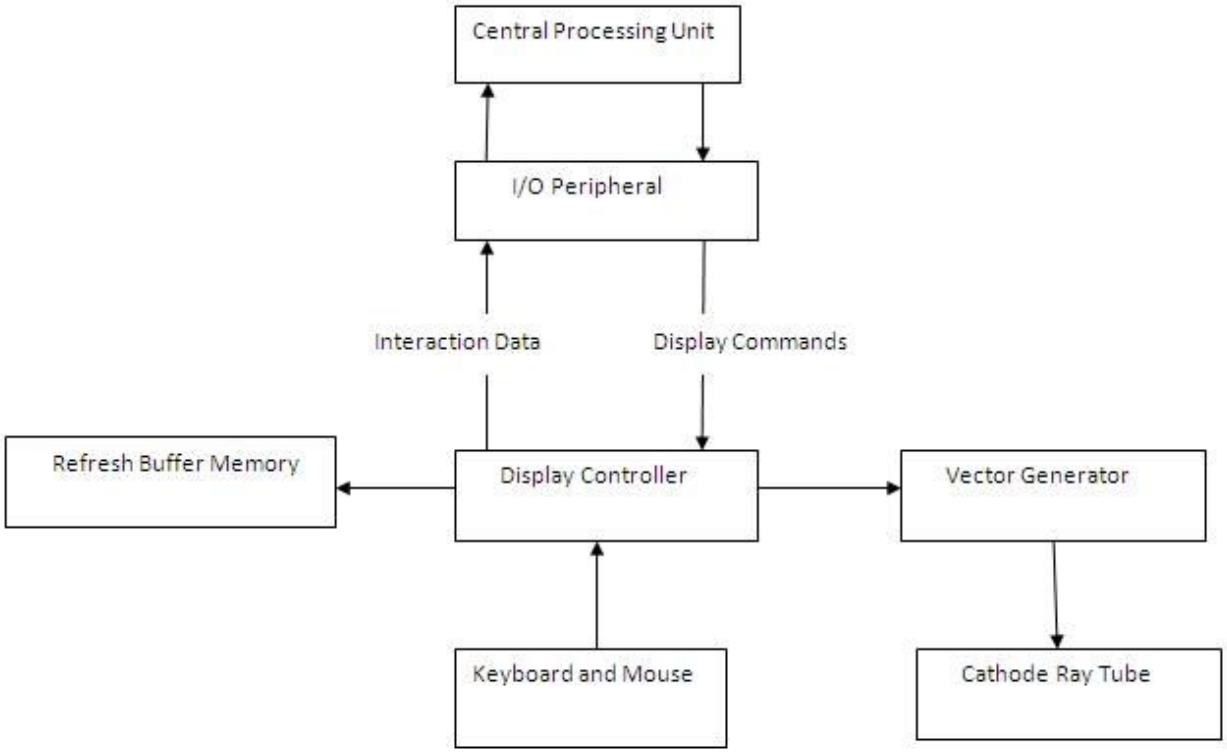
- **Random scan displays** have higher resolution than raster systems.
- Vector displays product smooth line drawing.
- This minimal amount of information translates to a much smaller file size. (file size compared to large raster images)
- On zooming in, and it remains smooth
- The parameters of objects are stored and can be later modified.

Random scan monitors draw a picture one line at a time and for this reason are also referred to as **vector** displays (or **stroke-writing** or **calligraphic** displays).The component lines of a picture can be drawn and refreshed by a random-scan system in any specified order. Refresh rate on a random-scan system depends on the number of lines to be displayed. Picture definition is now stored as a set of line-drawing commands in an area of memory referred to as the refresh display file. Sometimes the **refresh display file** is called the **display list, display program,** or simply the **refresh buffer**. 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 line-drawing commands have been processed, the system cycles back to the first line command in the list. Random-scan displays are designed to draw al the component lines of a picture 30 to 60times each second.

Working of a Vector scan or Random Scan Display architecture

An application program is input and stored in the memory along with a graphics package. Graphics commands in the application program are translated by the graphics package into a display file stored in the system memory. This display file is then accessed by the display processor to refresh the screen. The display processor cycles through each command in the display file program once during every refresh cycle. Sometimes the display processor in a random-scan system is referred to as a display processing unit or a graphics controller.

Graphics patterns are drawn on a random-scan system by directing the electron beam along the component lines of the picture. Lines are defined by the values for their coordinate endpoints, and these input coordinate values are converted to x and y deflection voltages. A scene is then drawn one line at a time by positioning the beam to fill in the line between specified endpoints.



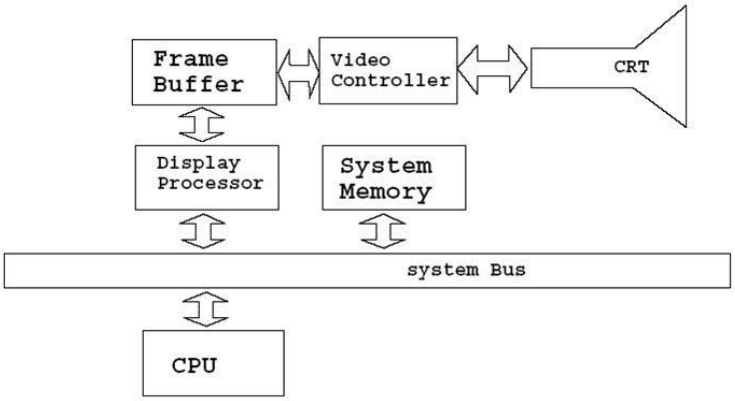
In a Vector scan display, both magnitude and direction of the electron beams are used to create a display. A display controller is used for interpretation commands for the co-ordinates and sends it to a vector generator. Vector generator then converts digital co-ordinates in to analog voltages which are then passed to the Cathode Ray Tube for display. This process is repeated continuously to obtain a display.

Difference between random scan and raster scan

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

Display Processor

- Raster graphics system with a dedicated display processor



Architecture of a raster-graphics system with a display processor.

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

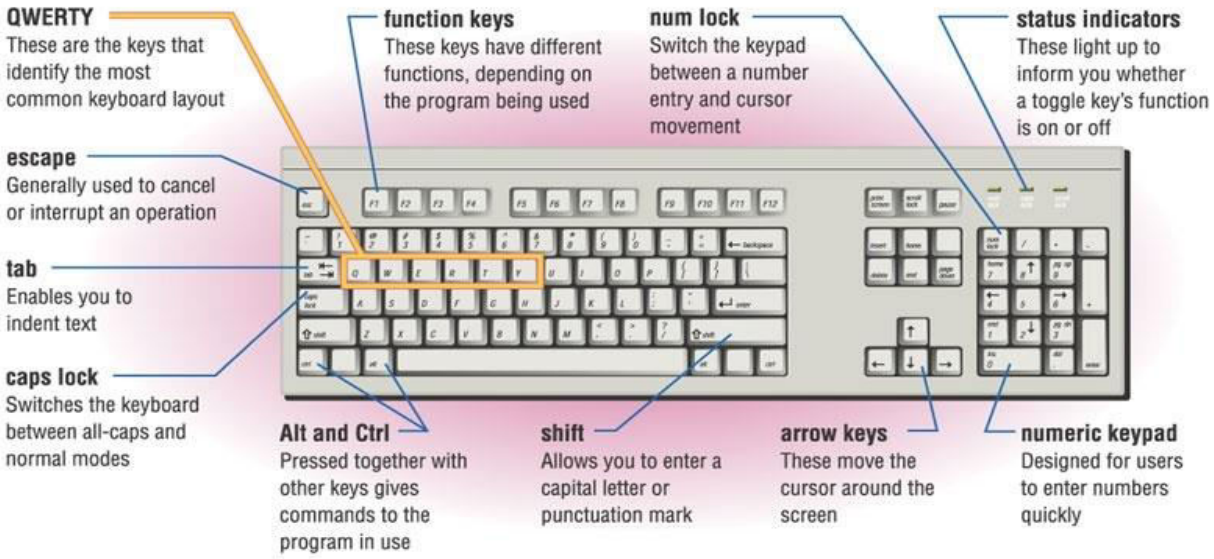
GRAPHICAL INPUT DEVICES

Input devices are things we use to put information into a computer. An input device is any hardware device that sends data to the computer, without any input devices, a computer would only be a display device and not allow users to interact with it, such like a TV. For example, a keyboard is an input device. Input devices other than the keyboard are sometimes called alternate input devices. Mice, trackballs, and light pens are all alternate input devices.

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.

Types of Keyboard

There are following types of Keyboards:

1. ERGONOMIC KEYBOARD

- The artifact of this keyboard is slightly broader and different in shape, when compared with the normal keyboard.
- In this key board certain space will be existing between the two sets of keys and the countered shape of this key board allow the users to place their hands in the natural position to type.
- These key boards are mostly used by the people who often work with the key board as their usage is easier and is less stressful for the wrist. The following figure: Ergonomic keyboard shows how the set of keys are separated with gaps in between.

Wireless Keyboard

- A wireless keyboard, the name itself doles the meaning that this keyboard can be operated without addressing a wired connection to the processor.
- The wireless keyboards are also referred as Cordless keyboards; these keyboards require batteries to provide the electricity which usually delivered through a PS/2 or USB cable. “AA” or “AAA” batteries are most widely used standard batteries for wireless keyboards.
- Apple Macs are known to revolutionize the wireless keyboard by making them thinner than the wired ones. These keyboards usually work at 2.4 GHz frequency

and come with a dongle that connects and makes them communicate with the computer.

Desktop computer keyboards, such as the 101-key US traditional keyboards or the 104-key Windows keyboards, include alphabetic characters, punctuation symbols, numbers and a variety of function keys.

Laptop

The laptop computer keyboard is a small version of the typical QWERTY keyboard. A typical laptop has the same keyboard type as a normal keyboard, except for the fact that most laptop keyboards condense the symbols into fewer buttons to accommodate less space.

Gaming and Multimedia

The gaming keyboards are designed for the convenience of the gamers and these types of keyboards provide the required controls on the keyboards like back lighting.

Thumb -sized keyboard

Smaller external keyboards have been introduced for devices without a built -in keyboard, such as PDAs, and smart phones. Small keyboards are also useful where there is a limited workspace.

Virtual Keyboard

The virtual keyboards are not actually physical keyboards, but they are simulated using a software.

Foldable Keyboard

Foldable keyboards are extremely good for travelling. • Simply roll them up and then unroll them when you need them again.

Keyboard Layouts

QWERTY - Common layout

QWERTZ - Used in Germany, Hungary and Czech Republic

AZERTY - It is used by most French speakers based in Europe

DVORAK - Alternative for QWERTY

Dvorak layout uses less finger motion, increases typing rate, and reduces errors compared to the standard QWERTY

Key Type Example

Alphanumeric	A-Z, 0-9
Punctuation	. , ! " ?
Modifiers	Shift, Space Bar, Enter, Ctrl, Alt
Navigation	Arrows, Home, Page Up
System Command	PrtScn, Esc, F1, Start

Mouse

Mouse Is a pointing device which is used to control the movement of a mouse pointer on the screen to make selections from the screen. A mouse has one to five buttons. The bottom of the mouse is flat and contains a mechanism that detects movement of the mouse.

Advantages

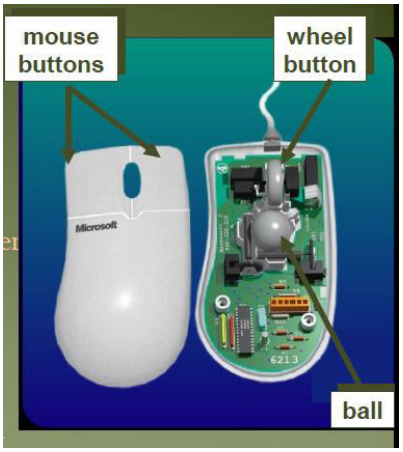
- Easy to use
- Not very expensive
- Moves the cursor faster than the arrow keys of the keyboard.
- Can be installed without any installation software

Types of Mice

- Wheel mouse – Contains a rotating wheel used to scroll vertically within a text document; connects to PS/2 port or USB port
- Cordless mouse – Uses infrared signals to connect to the computer's IrDA port; it must be within sight of the receiving port

How does a mechanical mouse work?

- A rubber or metal ball is on its underside When the ball rolls in a certain direction, electronic circuits in the mouse translate the movement of the mouse into signals the computer understands
- A mouse pad provides better traction and protects the ball from a build up of dust and dirt.



How does an optical mouse work?

- Uses devices that emit & sense light to detect the mouse's movement
- Some use optical sensors; others use laser
- More precise than a mechanical mouse & does not require cleaning but slightly more expensive

Pointing devices – direct [Touch screens]

- Often used for applications with occasional use, for example Bank ATMs, Information Kiosks, etc.
- No extra hardware - used for input and for output
- Can be precise to 1 pixel
- Good for menu choice - not so good for other functions

Disadvantages

- Tiring if at wrong angle (needs to be 30-45% from horizontal)
- Get greasy, jammy

- Finger can obscure screen
- Alternative - use stylus to touch screen, or lightpen

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. A light pen is a small 'pen-shaped' wand, which contains light sensors.
- It is used to choose objects or commands on the screen either by pressing it against the surface of the screen or by pressing a small switch on its side.
- A signal is sent to the computer, which then works out the light pen's exact location on the screen.
- The advantage of a light pen is that it doesn't need a special screen or screen coating.

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.

Advantages of Light Pens

- Allows you to select objects on a display screen - It has great accuracy-drawing directly on the screen so it is more accurate and more precise
- Durability and Accuracy : The light pen and computer respond instantly when you move it or click on one of the buttons located on the pen's sides. The tip cannot damage the computer screen in any way.
- Flexibility: In addition to having the full range of mouse options, including allowing you to drag and drop, you can use the light pen to directly draw or write on the screen. This makes providing your signature relatively simple.
- The amount of work space is reduced when you use a light pen instead of a mouse because you do not need a flat surface to operate a light pen.
- Maintenance and Value: Buying a light pen to use on your CRT computer monitor is cheaper than buying a touch-screen monitor. The same is true for repairs or replacements.

Disadvantages of Light Pens

- Light pens have the advantage of 'drawing' directly onto the screen, but this can become uncomfortable, and they are not as accurate as digitizing tablets.
- Light pens normally require a specially designed monitor to work with.

Trackball and Spaceball

- Trackball is ball that can be rotated with the finger or palm of the hand to produce cursor movement.
- Potentiometer attached to the ball, measure the amount and direction of rotation.
- They are often mounted on keyboard or Z mouse.

- Space ball provide six-degree of freedom i.e. three dimensional.
- In space ball strain gauges measure the amount of pressure applied to the space ball to provide input for spatial positioning and orientation as the ball is pushed or pulled in various directions.
- Space balls are used in 3D positioning and selection operations in virtual reality system, modeling, animation, CAD and other application.

Joystick

- A joy stick consists of small vertical lever mounted on a base that is used to steer the screen cursor around.
- Most joy sticks selects screen positioning according to actual movement of stick (lever).
- Some joy sticks are works on pressure applied on sticks.
- Sometimes joy stick mounted on keyboard or sometimes used alone.
- Movement of the stick defines the movement of the cursor.
- In pressure sensitive stick pressure applied on stick decides movement of the cursor. This pressure is measured using strain gauge.
- This pressure sensitive joy sticks also called as isometric joy sticks and they are non movable sticks.

Data glove

- Data glove is used to grasp virtual objects.
- The glow is constructed with series of sensors that detect hand and figure motions.
- Electromagnetic coupling is used between transmitter and receiver antennas which used to provide position and orientation of the hand.
- Transmitter & receiver Antenna can be structured as a set of three mutually perpendicular coils forming 3D Cartesian coordinates system.
- Input from the glove can be used to position or manipulate object in a virtual scene.

Digitizer

- Digitizer is common device for drawing painting or interactively selecting coordinates position on an object.
- One type of digitizers is graphics tablet which input two dimensional coordinates by activating hand cursor or stylus at selected position on a flat surface.
- Stylus is flat pencil shaped device that is pointed at the position on the tablet.

Image Scanner

- Image Scanner scan drawing, graph, color, & black and white photos or text and can stored for computer processing by passing an optical scanning mechanism over the information to be stored.

- Once we have internal representation of a picture we can apply transformation.
- We can also apply various image processing methods to modify the picture.
- For scanned text we can apply modification operation.

Voice systems

- It is used to accept voice command in some graphics workstations.
- It is used to initiate graphics operations.
- It will match input against predefined directory of words and phrases.
- Dictionary is setup for a particular operator by recording his voice.
- Each word is speak several times and then analyze the word and establishes a frequency pattern for that word along with corresponding function need to be performed.
- When operator speaks command it will match with predefine dictionary and perform desired action.

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.

Touch Panels

- As name suggest Touch Panels allow displaying objects or screen-position to be selected with the touch or finger.
- A typical application is selecting processing option shown in graphical icons.
- Some system such as a plasma panel are designed with touch screen
- Other system can be adapted for touch input by fitting transparent touch sensing mechanism over a screen.

Output Primitives

In computer graphics, a primitive is an image element, such as an arc, a square, or a cone, from which more complicated images, can be constructed. A computer Graphics can be anything like beautiful scenery, images, terrain, trees, or anything else that we can imagine, however all these computer graphics are made up of the

most basic components of Computer Graphics that are called Graphics Output Primitive or simply primitive. The Primitives are the simple geometric functions that are used to generate various Computer Graphics required by the User.

Some most basic Output primitives are point-position(pixel), and a straight line. However different Graphic packages offers different output primitives like a rectangle, conic section, circle, spline curve or may be a surface. Once it is specified what picture is to be displayed, various locations are converted into integer pixel positions within the frame buffer and various functions are used to generate the picture on the two dimensional co ordinate system of output display.

Elements of output primitives

- Points
- Lines
- Arc
- Square
- Cone
- Curve
- Rectangle

Points and Lines

• Point is the fundamental element of picture representation.

• It is the position in the plan defined as either pair or triplets of number depending upon the dimension.

• Two points represent line or edge and 3 or more points a polygon.

• Curved lines are represented by the short straight lines.

Graphics software and standard

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

General programming package

- A general programming package provides an extensive set of graphics function that can be used in high level programming language such as C or FORTRAN.
- It includes basic drawing element shape like line, curves, polygon, color of element transformation etc.
- Example: - GL (Graphics Library).

Special-purpose application package

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

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.

Graphic Function

- A general purpose graphics package provides user with Variety 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 Portability And Graphics Standards

Software Portability

Portability in high-level computer programming is the usability of the same software in different environments. The pre requirement for portability is the generalized abstraction between the application logic and system interfaces. When software with the same functionality is produced for several computing platforms, portability is the key issue for development cost reduction.

Strategies for portability

Software portability may involve:

- Transferring installed program files to another computer of basically the same architecture.
- Reinstalling a program from distribution files on another computer of basically the same architecture.
- Building executable programs for different platforms from source code; this is what is usually understood by "porting".

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

Graphical Kernel System (GKS)

- This system was adopted as a first graphics software standard by the international standard organization (ISO) and various national standard organizations including ANSI.
- GKS was originally designed as the two dimensional graphics package and then later extension was developed for three dimensions.

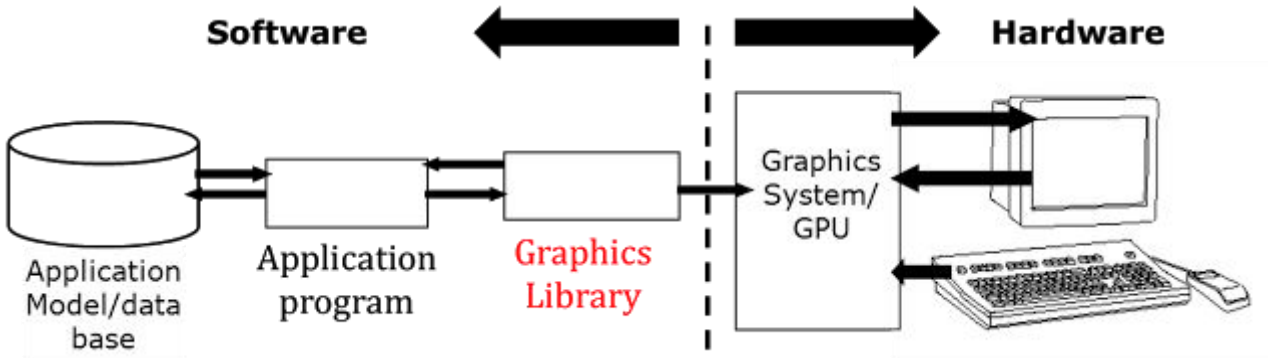
PHIGS (Programmer's Hierarchical Interactive Graphic Standard)

- 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.

Conceptual Framework for Interactive Graphics

- Graphics library/package is an intermediary between application and display hardware (Graphics System)
- Application program maps application objects to views (images) of those objects by calling on graphics library. Application model may contain lots of non-graphical data (e.g., non-geometric object properties)
- User interaction results in modification of model and/or image
- This hardware and software framework is more than 4 decades old but is still useful.



- Conceptual Framework for Interactive Graphics. The Fig. shows the high level conceptual framework for interactive graphics. It consists of input and output devices, graphics system, application program and application model. A computer receives input from input devices, and outputs images to a display device. The input and output devices are called the hardware components of the conceptual framework. There are three software components of conceptual framework. These are:· Application Model· Application Program and· Graphic System

- **Application Model**: The application model captures all the data and objects to be pictured on the screen. It also captures the relationship among the data and objects. These relationships are stored in the database called application database, and referred by the application programs.

- **Application program**: It creates the application model and communicates with it to receive and store the data and information of object's attributes. The application program also handles user input. It produces views by sending series of graphics

output commands to the graphics system .The application program is also responsible for interaction handling. It does this by event handling loops.

- **Graphics System**: It accepts the series of graphics output commands from application program. The output commands contain both a detailed geometric description of what is to be viewed and the attributes describing how the objects should appear. The graphics system is responsible for actually producing the picture from the detailed descriptions and for passing the user's input to the application program for processing.

[END OF FIRST UNIT]

UNIT – II

2D Graphics : Basic Raster Graphic Algorithms For 2D Primitives – Scan Converting Lines – Circles – Ellipes – Filling Rectangle – Character Generation – 2D Transformations – 2D Clipping – Windowing Transformation.

Basic Raster Graphic Algorithms for 2D Primitives

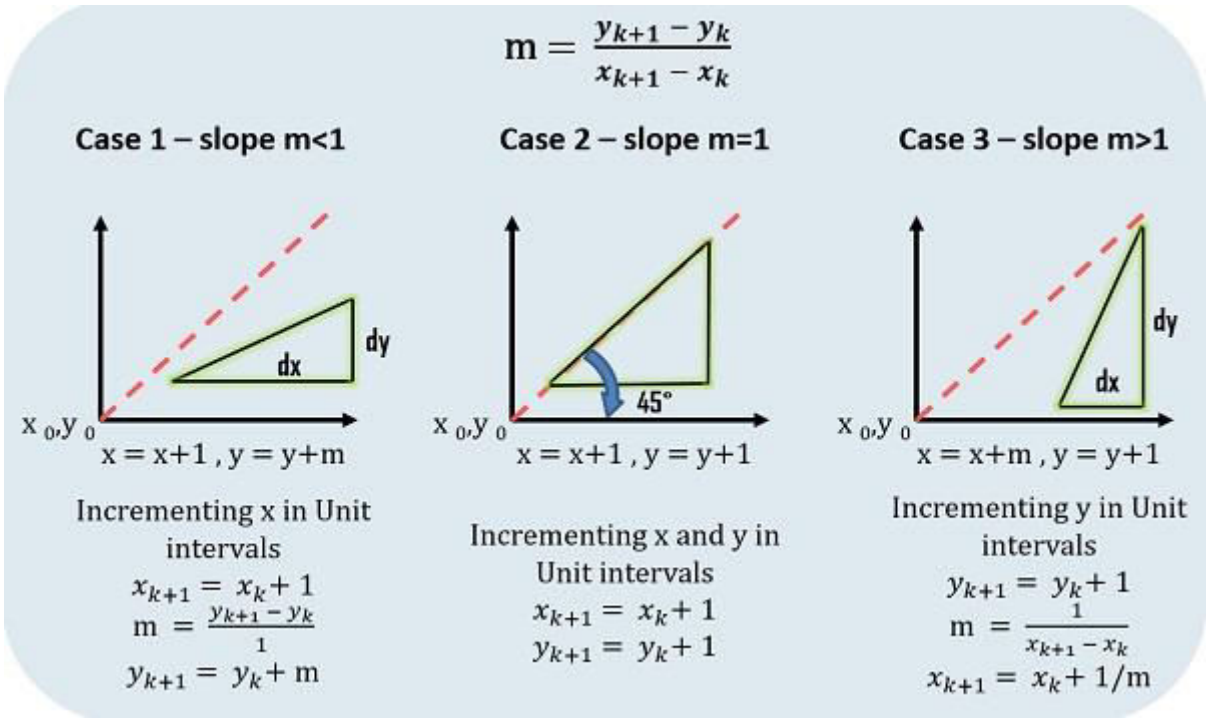
- DDA Line Drawing Algorithm
- Bresenham's Line Drawing Algorithm
- Midpoint Circle Algorithm
- Midpoint Ellipse Algorithm
- Filled Area Primitives

Points on Line Drawing Algorithm

DDA Algorithm

Definition

- A DDA (Digital Differential Analyzer) algorithms is a scan-conversion method for drawing a line which follows an incremental approach. In this algorithm to draw a line the difference in the pixel points is analysed then according to that the line is drawn. The method is said to be incremental because it performs computations at each step and uses the outcome of the previous step.
- Before understanding DDA algorithm, we must understand what is a line and how it is defined? When two points in a plane connected by a line segment and falls under the line equation is known as a line. The line equation mentioned above is y=mx+b where m is the slope (i.e., m = $\Delta y/\Delta x$) and y is the
- intercept of the line (the value of y at the points of the line).
- Depending on the slope three types of situations can arise shown in the below-given diagram



Advantages

1. It is the simplest algorithm
2. It is a is a faster method for calculating pixel positions

Disadvantages of DDA Algorithm

1. Floating point arithmetic in DDA algorithm is still time-consuming
2. The algorithm is orientation dependent. Hence end point accuracy is poor.

Bresenham's Line Drawing Algorithm

Bresenham algorithm also provides an efficient raster scan method for generating lines where incremental integer calculations are used. The developer of the algorithm was Jack Elton Bresenham, and the algorithm was named after him. It generates mathematically precise results with the help of addition, subtraction and multiplication by 2, which can be achieved by a simple arithmetic shift operation.

Advantages

1. Algorithm is fast.
2. Uses only integer calculations

Disadvantages It is meant only for basic line drawing.

Midpoint Circle Drawing Algorithm :-

The midpoint circle drawing algorithm also uses the eight-way symmetry of the circle to generate it. It plots 1/8 part of the circle, i.e. from 90° to 45°, as shown in the figure below. As circle is drawn from 90° to 45°, the x moves in the positive direction and y moves in the negative direction.

To draw a 1/8 part of a circle we take unit steps in the positive x direction and make use of decision parameter to determine which of the two possible y positions is closer to the circle path at each step.

The figure below shows the two possible y Positions (y_k and y_k -1) at sampling position x_k + 1. Therefore, we have to determine whether the pixel at Position (x_k + 1, y_k -1) is closer to the circle. For this purpose decision parameter is used.

Algorithm

1. Input radius r and centre of the circle (x_c, y_c), and obtain the first point on the circumference of a circle centred on the origin as (x_0, y_0) = (0, r)
2. Calculate the initial value of the decision parameter as P_k = 1 – r where r is the radius of the circle.
3. At each x_k position, starting at k= 0, perform the following test:

If (P_k < 0), then the NEXT POINT along the circle centred on (0, 0) is (x_k+1, y_k) Otherwise, the NEXT POINT along the circle is (x_k+1, y_k–1)

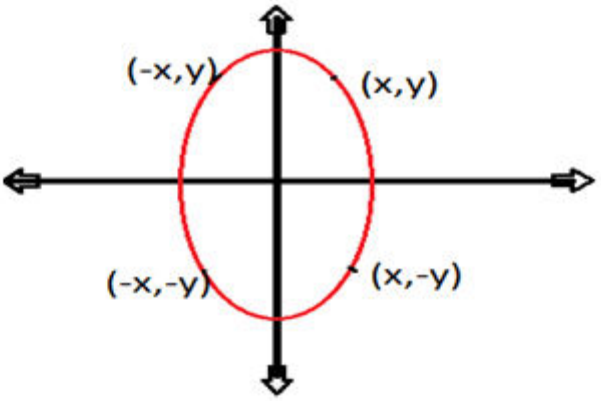
4. Determine the Symmetry Points on the other 7 Octants
5. Move each calculated pixel position (x,y) onto the circular path centered at (x_c,y_c) and plot the coordinate values as x = x + x_c and y = y + y_c
6. Repeat steps 3 to 5 until x >= y

Midpoint ellipse algorithm

The midpoint ellipse method is applied throughout the first quadrant in two parts. Now let us take the start position at **$(0,r_y)$** and step along the ellipse path in clockwise order throughout the first quadrant.

Ellipse function can be defined as:

$$f_{ellipse}(x,y)=r_y^2x^2+r_x^2y^2-r_x^2r_y^2$$



Initial decision parameter

In region **1** the initial value of a decision parameter is obtained by giving starting position = **$(0, r_y)$**.

i.e. $p_{1k}=r_y^2+1/4r_x^2-r_x^2r_y$

When we enter into a region **2** the initial position is taken as the last position selected in region **1** and the initial decision parameter in region 2 is then:

$P_{2k}=r_y^2(x_0+1/2)^2+r_x^2(y_0-1)^2-r_x^2r_y^2$

1. Input r_x, r_y and ellipse center (x_c, y_c). Obtain the first point on an ellipse centered on the origin $(x_0, y_0) = (0, r_y)$.
2. Calculate initial value for decision parameter in region 1 as:

$$p1_0 = r_y^2 - r_x^2 r_y + \tfrac{1}{4} r_x^2$$

3. At each x_k in region 1, starting from k = 0, test $p1_k$:

If $p1_k < 0$, next point (x_k+1, y_k) and

$$p1_{k+1} = p1_k + 2r_y^2 x_{k+1} + r_y^2$$

else, next point (x_{k+1}, y_k-1) and

$$p1_{k+1} = p1_k + 2r_y^2 x_{k+1} - 2r_x^2 y_{k+1} + r_y^2$$

with $2r_y^2x_{k+1} = 2r_y^2x_k + 2r_y^2$, $2r_x^2y_{k+1} = 2r_x^2y_k - 2r_x^2$ and repeat step 1 to 3 until $2r_y^2x \geq 2r_x^2y$.

4. Initial value for decision parameter in region 2:

$$p2_0 = r_y^2\left(x_0 + \tfrac{1}{2}\right)^2 + r_x^2\left(y_0 - 1\right)^2 - r_x^2 r_y^2$$

where (x_0, y_0) is the last position calculate in region 1

5. At each y_k in region 2, starting from k = 0, test $p2_k$:

If $p2_k > 0$, next point is (x_k, y_k-1) and

$$p2_{k+1} = p2_k - 2r_x^2 y_{k+1} + r_x^2$$

else, next point is (x_k+1, y_k-1) and

$$p2_{k+1} = p2_k + 2r_y^2 x_{k+1} - 2r_x^2 y_{k+1} + r_x^2$$

continue until y=0

6. For both region determine symmetry points in the other 3 quadrants
7. Move each calculated pixel position (x,y) onto the elliptical path centered on (x_c,y_c) and plot the coordinate values

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

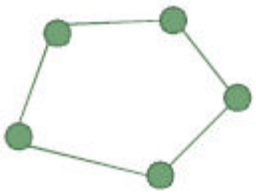
Fill-Area Primitives

Fill (Filled) Area

- An area filled with some solid color or pattern
- To describe the surfaces of solid objects
- Most graphics library routines don't support arbitrary fill shapes.

Polygon

- Polygon is a word derived from two Greek words poly and Gon. Poly means multi and Gon means angle. Thus, the meaning of polygon is multiangled figure. Polygon is a closed figure with many vertices and edges and at each vertex exactly two edges meet and no edge crosses other.
- The line segments which make up a polygon boundary are called edges. The end points of the edges are called vertex of polygon.
- The simple form of possible polygon is triangle having 3 edges and 3 vertex points.
- A plane figure specified by a set of three or more vertices, that are connected in sequence by straight-line segments (edges). Here refer only to those without crossing edges: simple (standard) polygon



Polygon Classifications

Convex Polygon (凸)

All interior angles are less than or equal to 180o degrees

Concave Polygon (凹)

Otherwise

Convex Polygon: This polygon is a polygon in which if you take any two points of polygon then all the points on the line segment joining these two points fall within the polygon itself.

Concave Polygon: This polygon is a polygon in which if the line joining any two points of a polygon does not fall entirely within the polygon.

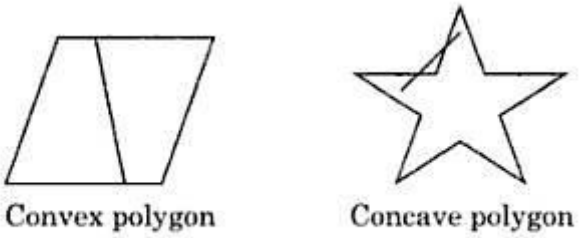
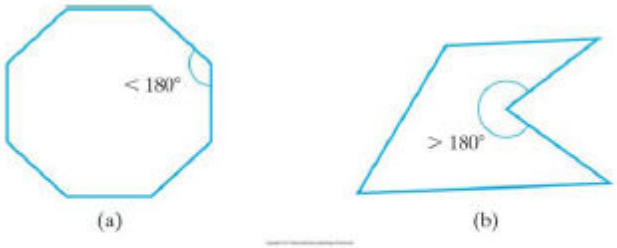


FIGURE 4.3 Classes of polygons.



A convex polygon (a) and a concave polygon (b)

Inside-Outside Tests

Inside-Outside test

Area-filling algorithms and other graphics processes often need to identify interior regions of objects. For simple object, it is a straightforward process.

For complex objects, graphics packages normally use either:

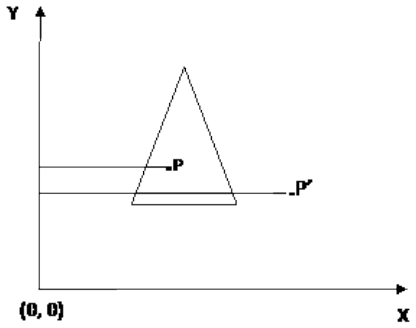
1. Odd-Even rule (Odd-Parity rule or Even-Odd rule)
2. winding number method

Odd-Even rule (Odd-Parity Rule, Even-Odd Rule):

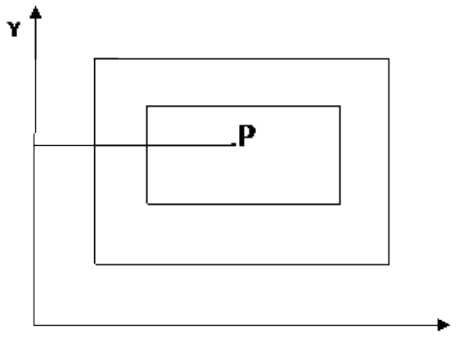
1. Draw a line: (the line path doesn't intersect any line-segment endpoints) From any position P to a distant point outside the coordinate extents of the polygon.
2. Counting the number of edge crossing along the line.
3. If the number of polygon edges crossed by this line is odd then P is an interior point.

Else

P is an exterior point



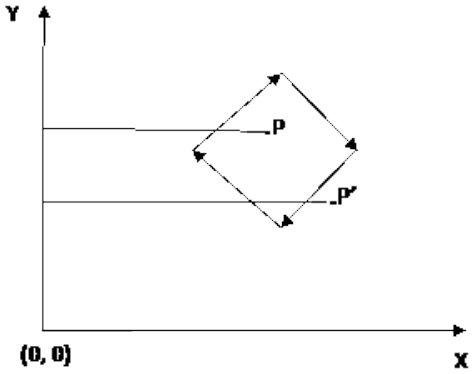
- Let P be a point to be tested. From P draw a line parallel to the X axes as shown in the above figure. Count the no. of intersection points with edges of polygon.
- If the no. of intersection points is odd then the point is inside the polygon. For P there is one intersection point. Therefore P is inside.
- If the no. of intersection points is even then the point is outside the polygon. For P' there is two intersection points. Therefore P' is outside.
    - Circumstances where Even-Odd method fails
    - Overlapping Polygon:



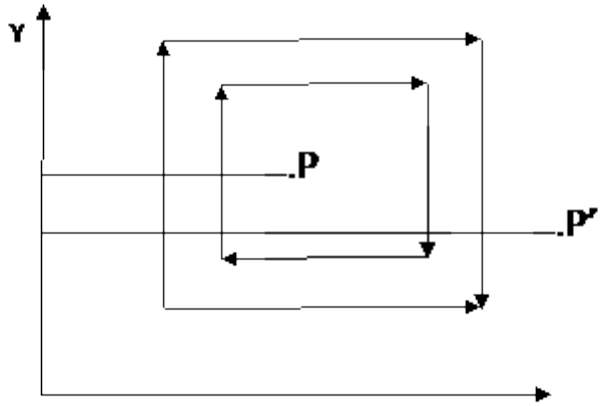
As shown in the figure, for point P numbers of intersection are even. Thus according to Even – Odd method the point P must be outside the polygon. But this point is inside the polygon so in such a condition where a polygon is complex or overlapped then the Even – Odd method will not work. For that the winding no. method is used

Winding Number method:

- In this method, give direction to each and every edge as shown in following figure.



- Check the direction of edges and number of edges intersected by the straight line parallel to X axes. If the direction is upward, count –1 and if the direction is downwards then count is +1.
- No. of winding is not equal to zero than point is inside or if it is equals to 0 then the point is outside the polygon.
- For the figure shown below for point p summation of winding no. is,



For Point P,

Σ winding no. = -1 -1 = -2 , that is not equal to zero so point is inside the polygon.

For Point P',

Σ winding no. = -1 -1 +1 +1 = 0 , that is equal to zero so point is outside the polygon

Polygon Area Filling

- To make the polygon figure to look like a solid object, it is required to be fill in with some color or pattern polygon area filling means changing the color of pixel belonging

to the polygon. This can be done by different methods which are used popularly for polygon filling.

There are three different methods

1. Flood Fill Method
2. Scan Line Fill Method
3. Filling Polygon with pattern

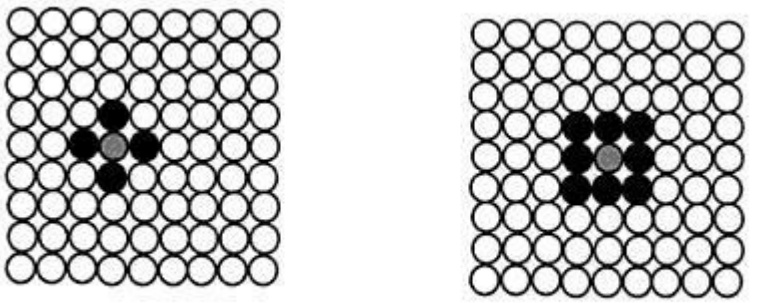
Flood Fill Method:

- It is also known as seed fill method. In this method, a point inside a polygon is needed.
- This point is called as seed point and the process of filling the polygon area is started from this points.
- From the seed point, a point is taken and interior color is replaced by the required color. This process is done till a pixel of boundary color is reached. This process is performed simultaneously in all directions. This can be done in two ways.

–

Four Connected Point Neighborhood Method

– Eight Connected Point Neighborhood Method



(a) Four connected points (b) Eight connected points

FIGURE 4.16 Flood fill.

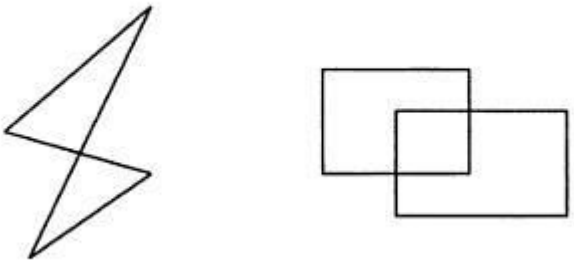


FIGURE 4.17 Flood fill method does not work with the above polygons.

In first method, four surrounded pixels are taken simultaneously and replaced with the required color. In second method, eight surrounded pixels are taken simultaneously and replaced with the required color.

Obviously, recursive function is the best suited concept for the flood fill algorithm.

```
Void fill4 (int x, int y, int fill, int oldcolor) {
    If( getpixel (x, y) == oldcolor)
    {
            Putpixel (x, y, fill);
            fill4 (x+1, y, fill, oldcolor);
            fill4 (x-1, y, fill, oldcolor);
            fill4 (x, y+1, fill, oldcolor);
            fill4 (x, y-1, fill, oldcolor);
}
```

If we do not have a seed point to fill the polygon, we can turn on all the pixels which are inside the polygon but the problem is that, these needs inside test to be perform with each and every pixel on the screen which is very time consuming.

Difference between flood fill and boundary fill algorithm

Boundary Fill Algorithm:

The boundary fill algorithm works as its name. This algorithm picks a point inside an figure and starts to fill until it reaches the boundary of the figure. The color of the boundary and the color that we fill should be different for this algorithm to work. In this algorithm, we assume that color of the boundary is same for the entire figure. The boundary fill algorithm can be implemented by 4-connected pixels or 8-connected pixels.

Flood Fill Algorithm:

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

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

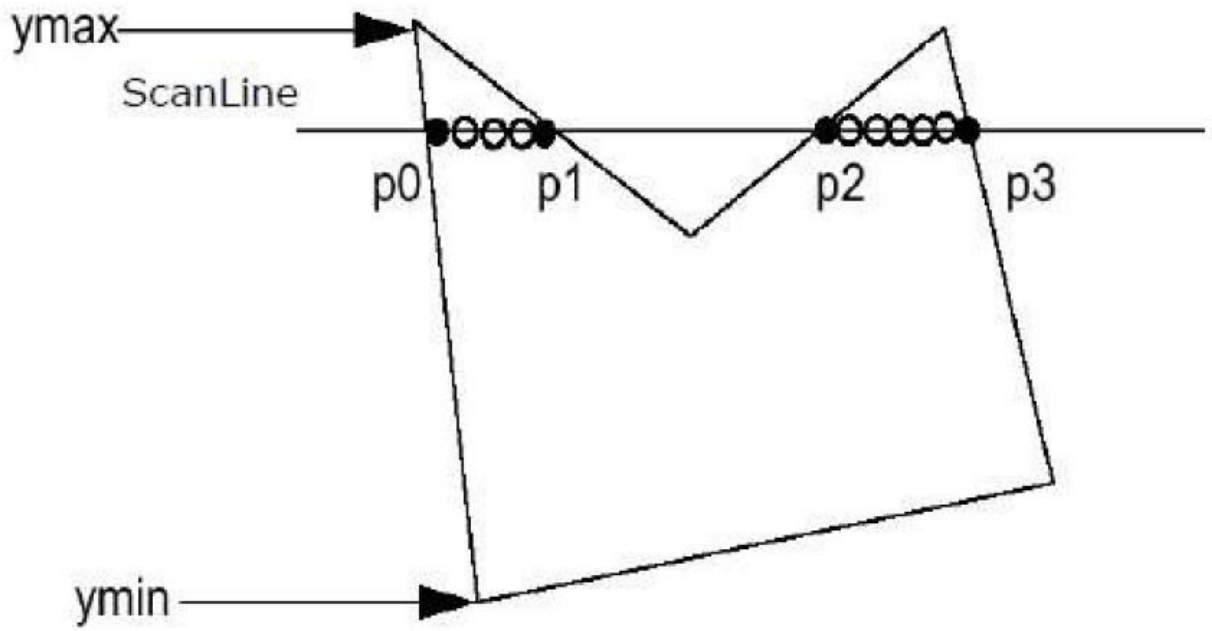
Scan Line Fill Method:

Figures on a computer screen can be drawn using polygons. To fill those figures with color, we need to develop some algorithm. There are two famous algorithms for this purpose: Boundary fill and Scanline fill algorithms.

This method is also called “alternative fill method”. The scan line fill method is a very efficient & cheaper alternative. This is the method in which scan line are used for the filling process. Scan line fill method works very efficiently with self-intersecting polygons as well.

This algorithm works by intersecting scan line with polygon edges and fills the polygon between pairs of intersections. The following steps depict how this algorithm works.

Step 1 − Find out the Ymin and Ymax from the given polygon.



Step 2 − Scan Line intersects with each edge of the polygon from Ymin to Ymax. Name each intersection point of the polygon. As per the figure shown above, they are named as p0, p1, p2, p3.

Step 3 − Sort the intersection point in the increasing order of X coordinate i.e. (p0, p1), (p1, p2), and (p2, p3).

Step 4 − Fill all those pair of coordinates that are inside polygons and ignore the alternate pairs.

Implementation of scanline polygon filling algorithm

Main Components:

To successfully fill in a polygon three main components will be used: **Edge Buckets**, an **Edge Table** and an **Active List**. These components will contain an edge's information, hold all of the edges that compose the figure and maintain the current edges being used to fill in the polygon, respectively.

Edge Buckets

The edge bucket is a structure that holds information about the polygon's edges. The edge bucket looks different pending on the implementation of the algorithm, in my case it looks like this:

yMax	yMin	x	sign	dX	dY	sum

Edge Bucket representation

Here is a breakdown of what each member represent in an edge bucket:

- **yMax:** Maximum Y position of the edge
- **yMin:** Minimum Y position of the edge
- **x:** The current x position along the scan line, initially starting at the same point as the **yMin** of the edge
- **sign:** The sign of the edge's slope (either -1 or 1)
- **dX:** The absolute delta x (difference) between the edge's vertex points
- **dY:** The absolute delta y (difference) between the edge's vertex points
- **sum:** Initiated to zero. Used as the scan lines are being filled to x to the next position

Edge Table (ET)

The ET is a list that contains all of the edges that make up the figure.

Important ET notes:

- When creating edges, the vertices of the edge need to be ordered from left to right
- The edges are maintained in increasing yMin order
- The edges are removed from the ET once the Active List is done processing them
- The algorithm is done filling the polygon once all of the edges are removed from the ET

Active List (AL)

The AL contains the edges that are being processed/used to fill the polygon. Every edge in the AL has a pairing buddy edge, because when filling a scan line, pixels are filled starting from one edge until the buddy edge is encountered.

Important AL notes:

- Edges are pushed into the AL from the Edge Table once an edge's yMin is equal to the current scan line being processed
- Edges will always be in the AL in pairs
- Edges in the AL are maintained in increasing x order.
- The AL will be re-sorted after every pass

Filling Rectangle

A rectangle is filling colour with flood fill function by Four Connected Point Neighborhood Method. The following is the source code to fill a rectangle.

```
#include <graphics.h>
#include <conio.h>
#include<dos.h>
void flood_fill4(int x,int y,int newColor,int oldColor)
{
int c;
c=getpixel(x,y);
if(c==oldColor)
{
setcolor(newColor);
putpixel (x,y,newColor);
delay(10);
flood_fill4(x+1,y,newColor,oldColor);
flood_fill4(x,y+1,newColor,oldColor);
flood_fill4(x-1,y,newColor,oldColor);
flood_fill4(x,y-1,newColor,oldColor);
}
}
void main()
{
int gd=DETECT,gm;
initgraph(&gd,&gm,"C:\\turboc3\\bgi");
rectangle(250,200,300,250);
flood_fill4(251,201,4,0);    // 4(red) - newColor  0(black) - oldColor
                        // x y point should be one greater than rectangle border
getch();
closegraph();
}
```

Character generation

- Letters, numbers, and other character are often displayed to label and annotate drawing and go give instructions and information to the user.

- Most of the times characters are built into the graphics display devices, usually as hardware but sometimes through software.
- There are basic three methods:
  - Stroke method
  - Star bust method
  - Bitmap method

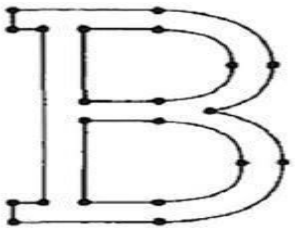
Bitmap Font/ Bitmapped Font: Character Generation

- A simple method for representing the character shapes in a particular typeface is to use rectangular grid patterns.
- The figure below shows the pattern for the particular letter.

1	1	1	1	1	1	0
0	1	1	0	0	1	1
0	1	1	0	0	1	1
0	1	1	1	1	1	0
0	1	1	0	0	1	1
0	1	1	0	0	1	1
1	1	1	1	1	1	0

- When the pattern in the figure copied to the area of the frame buffer, the 1 bits designate which pixel positions to displayed on the monitor.
- Bitmap fonts the simplest to define and display as character grid only need to be mapped to a frame buffer position.
- Bitmap fonts require more space because each variation (size and format) must stored in a font cache.
- It possible to generate different size and other variation from one set but this usually does not produce the good result.

Outline Font: Character Generation



- In this Character Generation method, a character generated using curve section and straight line as combine assembly. The figure shows how it generated.

- To display the character shown in the figure we need to fill interior region of the character.
- This method requires less storage since each variation does not require a distinct font cache.
- We can produce boldface, italic, or different sizes by manipulating the curve definitions for the character outlines.
- But this will take more time to process the outline fonts because they must scan converted into the frame buffer.

Stroke Method: Character Generation

- This method uses small line segments to generate a character.
- The small series of line segments are drawn like a strokes of a pen to form a character as shown in figure.
- We can build our own stroke method.
- By calling a line drawing algorithm.
- Here it is necessary to decide which line segments are needed for each character and
- Then drawing these segments using line drawing algo.
- This method supports scaling of the character.
- It does this by changing the length of the line segments used for character drawing.

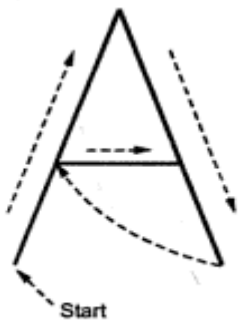
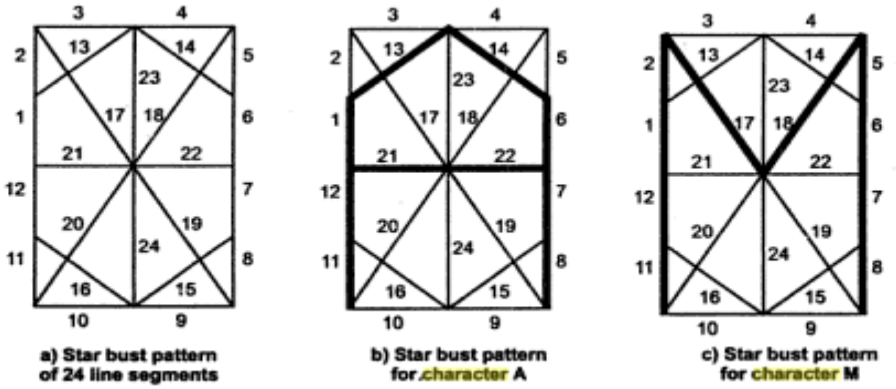


Fig. 2.31 Stroke method

Starburst Method

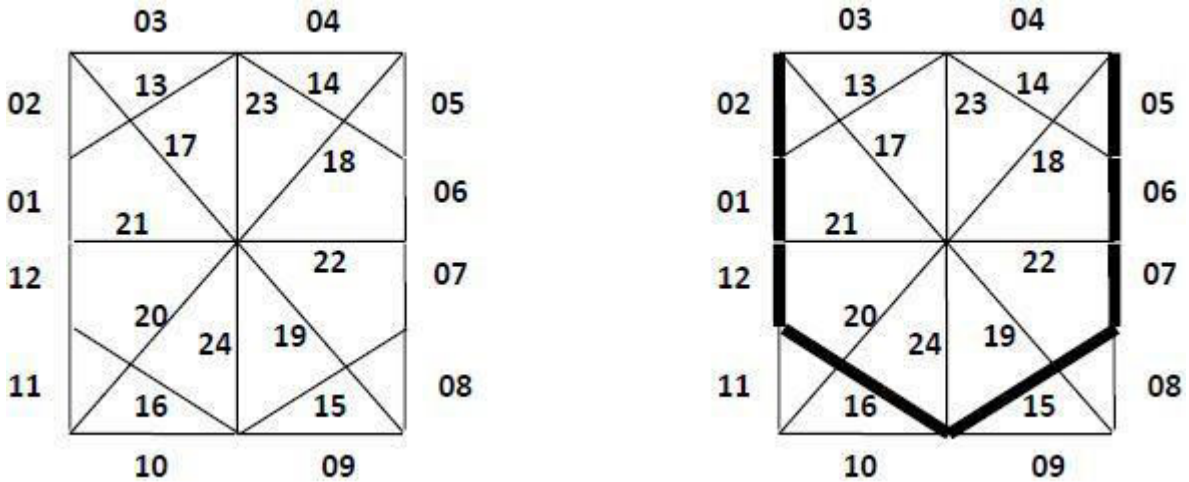
- In this method a fix pattern of line segments are used to generate characters.
- As shown in figure, there are 24 line segments.
- Out of 24 line segments, segments required to display for particular character, are highlighted
- This method is called starbust method because of its characteristic appearance.
-

- Fig shows the starbust patterns for character A and M.
- The patterns for particuler characters are stored in the form of 24 bits code.
- Each bit representing one line segment.
- The bit is set to one to highlight the line segment otherwise it is set to zero.



Character A : 0011 0000 0011 1100 1110 0001
Character M:0000 0011 0000 1100 1111 0011

- This method of character generation is not used now a days because of following disadvantages:
- The 24-bits are required to represent a character. Hence more memory is required.
- Requires code conversion software to display character from its 24 bits code.
- Character quality poor. It doesn't provide curve shapes, so worst for curve shaped character.



- Code for letter V is
- 1 0 0 1 1 1 0 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0

2D Transformations : What is ransformation?

In many cases a complex picture can always be treated as a combination of straight line, circles, ellipse etc., and if we are able to generate these basic figures, we can also generate combinations of them. Once we have drawn these pictures, the need arises to transform these pictures.

We are not essentially modifying the pictures, but a picture in the center of the screen needs to be shifted to the top left hand corner, say, or a picture needs to be increased to twice it's size or a picture is to be turned through 900. In all these cases, it is possible to view the new picture as really a new one and use algorithms to draw them, but a better method is, given their present form, try to get their new counter parts by operating on the existing data. This concept is called **transformation**.

The five basic transformations are

(i) Translation

(ii) Rotation

(iii) Scaling

(iv) Shearing

(v) Reflection

Translation refers to the shifting of a point to some other place, whose distance with regard to the present point is known.

Rotation as the name suggests is to rotate a point about an axis. The axis can be any of the coordinates or simply any other specified line also.

Scaling is the concept of increasing (or decreasing) the size of a picture. (in one or in either directions. When it is done in both directions, the increase or decrease in both directions need not be same) To change the size of the picture, we increase or decrease the distance between the end points of the picture and also change the intermediate points are per requirements.

Reflection: It is also called as mirror or mirror image . It can be either about x axis or y axis. The object is rotated about 180 degree. Reflection can be done about line.

Shearing : It is change of shape of object. It can be in x or y or both directions.

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 −

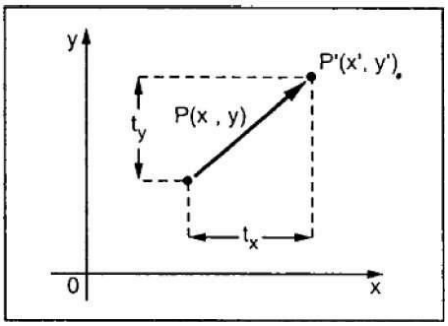
- Translate the coordinates,
- Rotate the translated coordinates, and then
- Scale the rotated coordinates to complete the composite transformation.

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 P(X, Y) can be converted to homogenous coordinates by P' (Xh, Yh, 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 (tx, ty) to the original coordinate (X, Y) to get the new coordinate (X', Y').



The new point: (x', y')

x' = x + tx

y' = y+ ty

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, Y), the scaling factors are (SX, SY), and the produced coordinates are (X', Y'). This can be mathematically represented as shown below

Scale a point P(x,y) by S(sx, sy)

Algebraic:

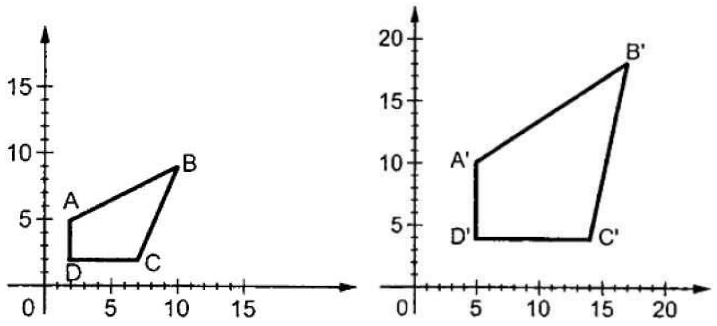
x'= x* sx

y'= y * sy

Matrix:

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

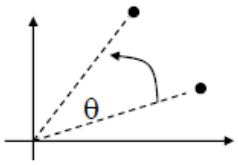
The scaling process is shown in the following figure



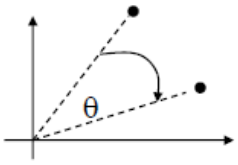
Rotation

In rotation, we rotate the object at particular angle θ (theta) from its origin. From the following figure, we can see that the point P(X, 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').



θ> 0 : Rotate counter clockwise



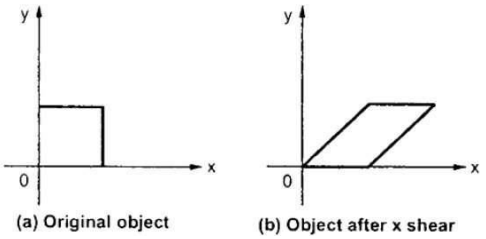
θ< 0 : Rotate clockwise

Shearing

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.

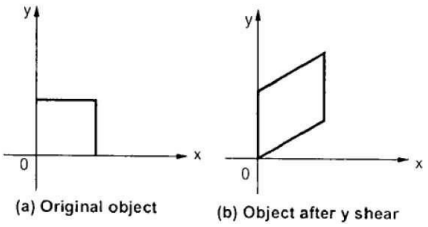
X-Shear

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.



Y-Shear

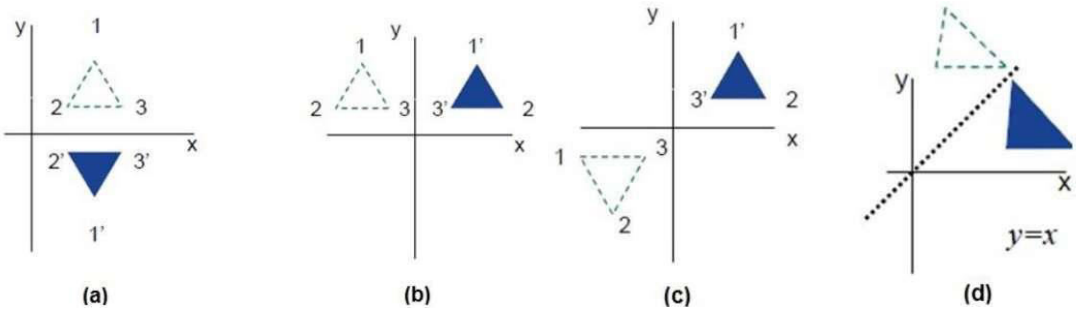
The Y-Shear preserves the X coordinates and changes the Y coordinates which causes the horizontal lines to transform into lines which slopes up or down as shown in the following figure.



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.

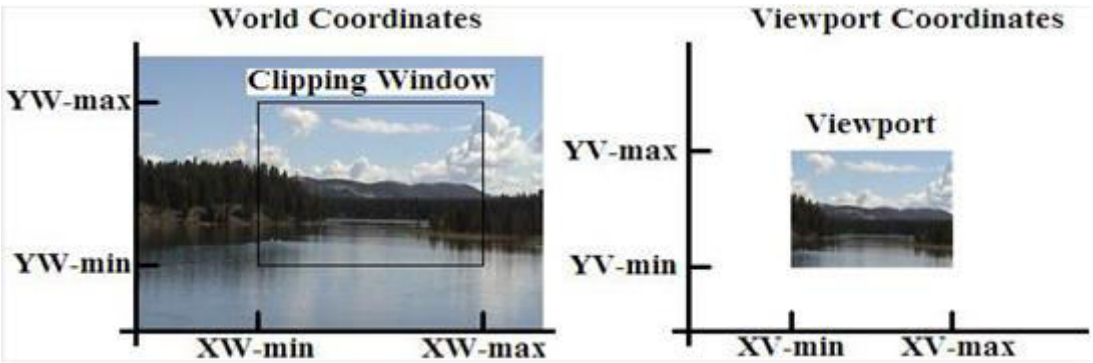
The following figures show reflections with respect to X and Y axes, and about the origin respectively.



Window to viewport transformation

Window-to-Viewport mapping is the process of mapping or transforming a two-dimensional, world coordinate scene to device coordinates. In particular, objects inside the world or clipping window are mapped to the viewport. The viewport is rectangular area on screen where world coordinates are mapped to be displayed. In other words, the clipping window is used to select the part of the scene that is to be displayed. The viewport is used to display selected portion of window on the output device.

Example:



Both Window and Viewport are shown in above figure,

From above figure,

Coordinate Units of Window = (xw, yw)

Coordinate Units of Viewport = (xv, yv)

Where,

xw = X-coordinate in window

yw = Y-coordinate in window

xv = X-coordinate in viewport

yv = Y-coordinate in viewport

Now, let us map (xw,yw) to the (xv,yv).So when mapping takes place then the relative places (positions) in Two areas (window and Viewport) are the same.

In the above figure also showed that,

xwmin = Minimum X-coordinate value in window

xwmax = Maximum X-coordinate value in window

ywmin = Minimum Y-coordinate value in window

ywmax = Maximum Y-coordinate value in window

xvmin = Minimum X-coordinate value in viewport

xvmax = Maximum Y-coordinate value in viewport

yvmin = Minimum X-coordinate value in viewport

yvmax = Maximum Y-coordinate value in viewport

$$\frac{xv - xv_{\min}}{xv_{\max} - xv_{\min}} = \frac{xw - xw_{\min}}{xw_{\max} - xw_{\min}} \qquad \frac{yv - yv_{\min}}{yv_{\max} - yv_{\min}} = \frac{yw - yw_{\min}}{yw_{\max} - yw_{\min}}$$

Using the above formula , we can get viewport coordinates(xv, yv).

2D Clipping

Many graphics application programs give the users the impression of looking through a window at a very large picture. Figure shows the use of this effect in a program for viewing different portions of a large architectural plan at different scales. Viewing an architectural plan through windows of different sizes This makes use of scaling and translation techniques to generate a variety of different views of a single representation of a plan. To display an enlarged portion of a picture, we must not only apply the appropriate scaling and translation but also should identify the visible parts of the picture. This is not straightforward. Certain lines may lie partly inside the visible portion of the picture and partly outside. We cannot display each of these lines in its entirety. Occurrence of wraparound in drawing a partially invisible triangle. The correct way to select visible information for display is to use clipping, a process which divides each element of the picture in to its visible and invisible portions, allowing the invisible portion to be discarded. Clipping can be applied to a variety of different types of picture elements such as pointer, lines, curves, text character and polygons.

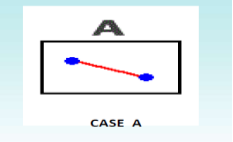
POINT CLIPPING Assuming that the clip window is a rectangle in standard position, we save a point P = (x, y) for display if the following inequalities are satisfied: xwmin≤x≤xwmax & ywmin≤y≤ywmax

Where the edges of the clip window (xwmin, xwmax, ywmin , ywmax) can be either the world-coordinate window boundaries or viewport boundaries. If any one of these four inequalities is not satisfied, the point is clipped (not saved for display). Although point clipping is applied less often than line or polygon clipping, some applications may require a point clipping procedure. For example, point clipping can be applied to scenes involving explosions or sea foam that are modeled with particles (points) distributed in some region of the scene.

Line Clipping: Clipping of lines against a window needs to determine if the line has an intersection with the edges of the window and performs the intersection accordingly. The intersection between the line and an edge of the window is costly since it incorporates floating point operations. Several algorithms are introduced to clip lines against rectangular and non-rectangular windows. Efficient algorithms try to minimize the number of intersections as much as possible. The so called 'Cohen-Sutherland algorithm' is a famous line clipping algorithm.

Cohen-Sutherland Algorithm Given a rectangular clipping window, the algorithm divides the space into 9 regions w.r.t. the window: left, left-top, top, right-top, right, right-bottom, bottom, left-bottom and inside as in Figure.

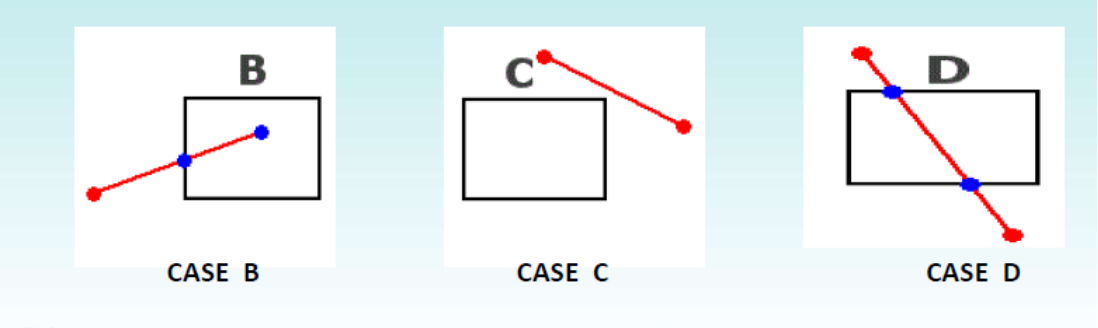
Here are a few cases, where the black rectangle represents the screen, in red are the old endpoints, and in blue the ones after clipping: Case A: Both end-points are inside the screen, so no clipping needed.



Case B: One end-point outside the screen, that one had to be clipped.

Case C: both end points are outside the screen, and no part of the line is visible, don't draw it at all.

Case D: both end points are outside the screen, and a part of the line is visible, clip both endpoints and draw it.



In this algorithm it divides lines & edges into 2 cases.

1) Trivially Accept and 2) Trivially Reject.

There are 3 possibilities for the line –

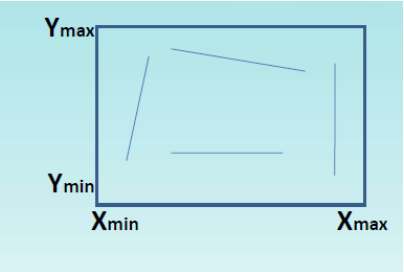
- Line can be completely inside the window (This line should be accepted).
- Line can be completely outside of the window (This line will be completely removed from the region).
- Line can be partially inside the window (We will find intersection point and draw only that portion of line that is inside region).

Conditions of Trivially Accept

Xmin ≤ X ≤ Xmax

Ymin ≤ Y ≤ Ymax

Lines fulfill this conditions then we will mark those lines as trivially accept.



The algorithm divides the 2D space in 9 regions: This is also known as ABRL CODE

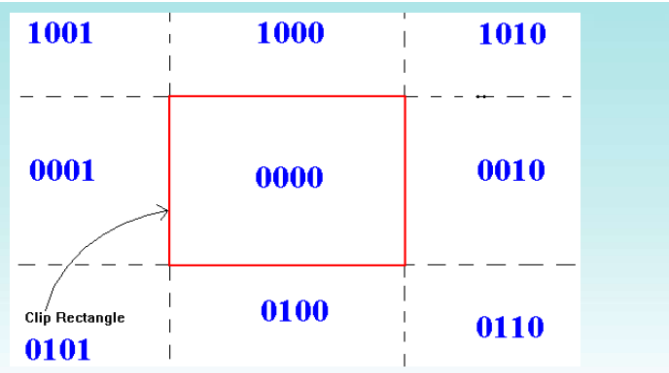
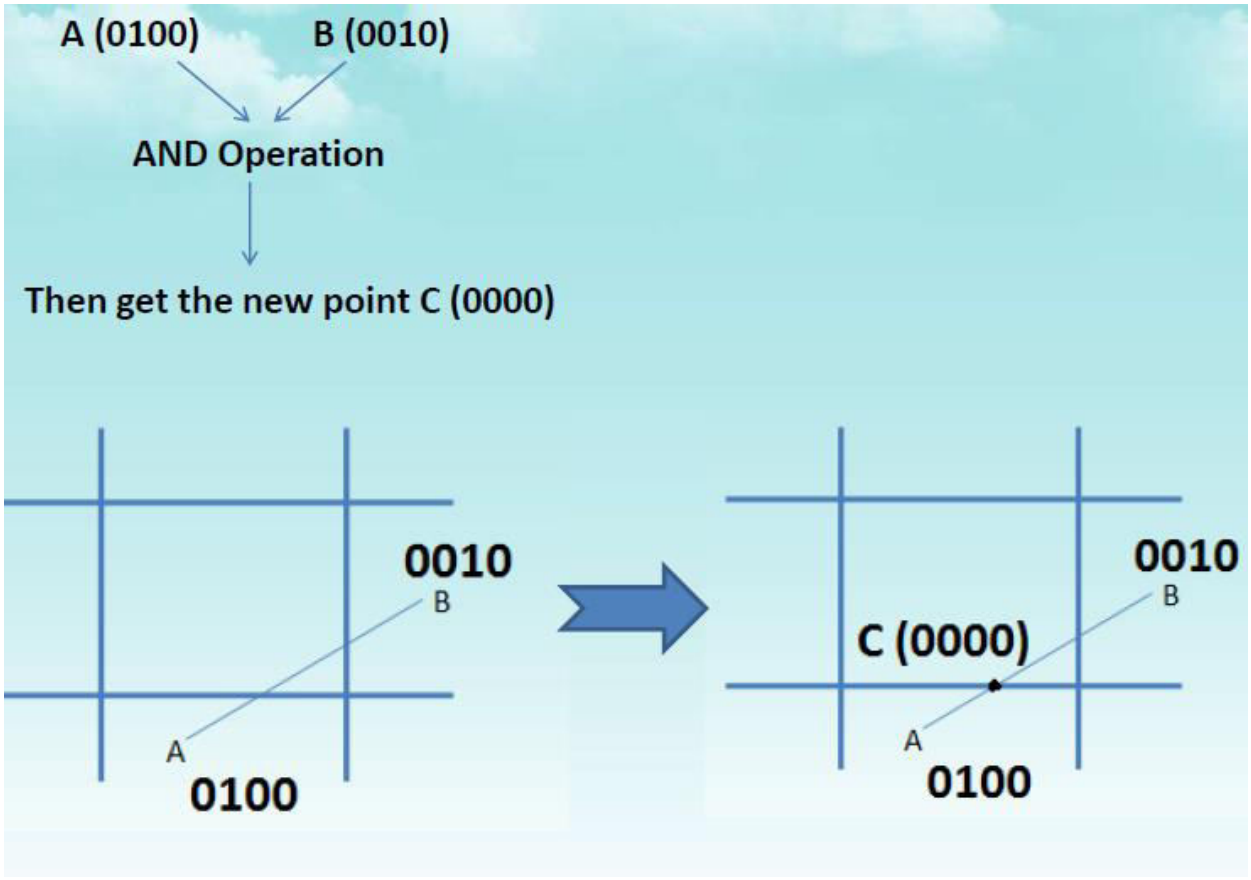
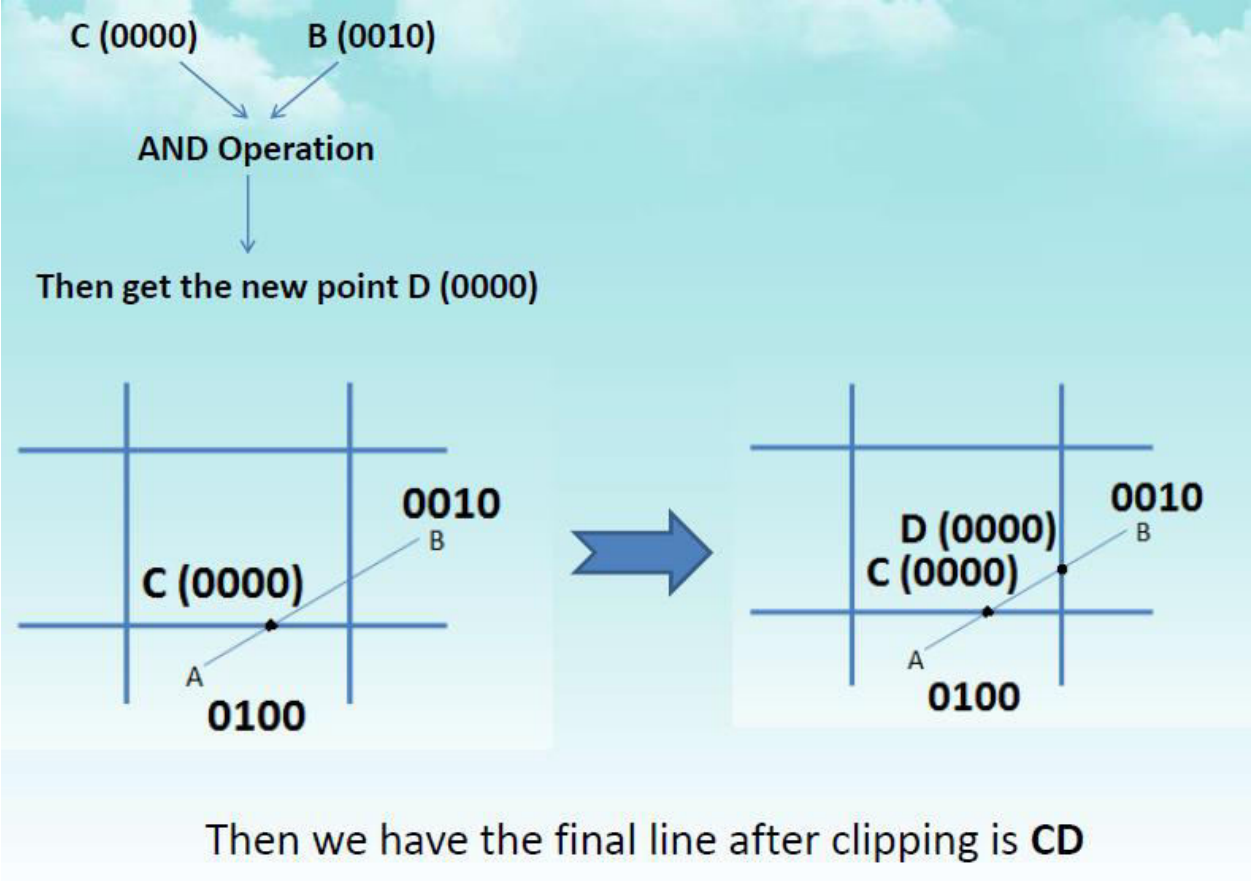


Figure: 2D space in 9 regions

The center region is the screen or Window Position (0000).

- ✓ If the region is above the screen, the first bit is 1.
- ✓ If the region is below the screen, the second bit is 1.
- ✓ If the region is to the right of the screen, the third bit is 1.
- ✓ If the region is to the left of the screen, the fourth bit is 1.





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.

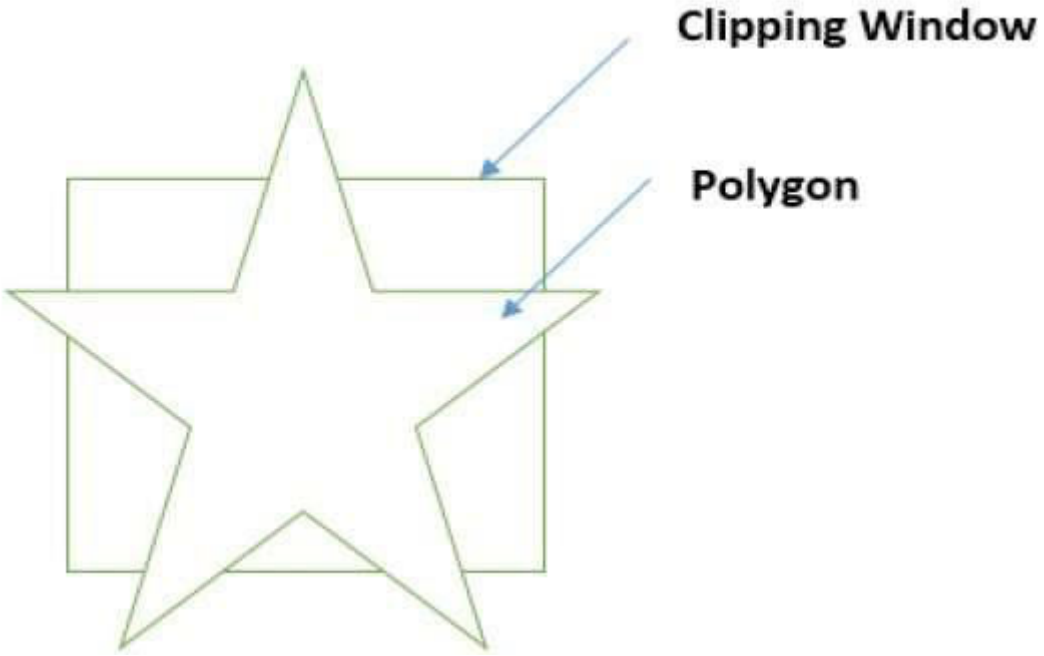
Step 3.2.4 – Repeat step 2 until we find a clipped line either trivially accepted or trivially rejected.

Step 4 – Repeat step 1 for other lines.

Polygon clipping algorithms Here, we'll study the so-called **Sutherland-Hodgman algorithm for polygon clipping**. The algorithm clips the polygon by the window side by side. With rectangular windows, the algorithm has four main iterations: clipping with the left side, clipping with the top side, clipping with the right side and clipping

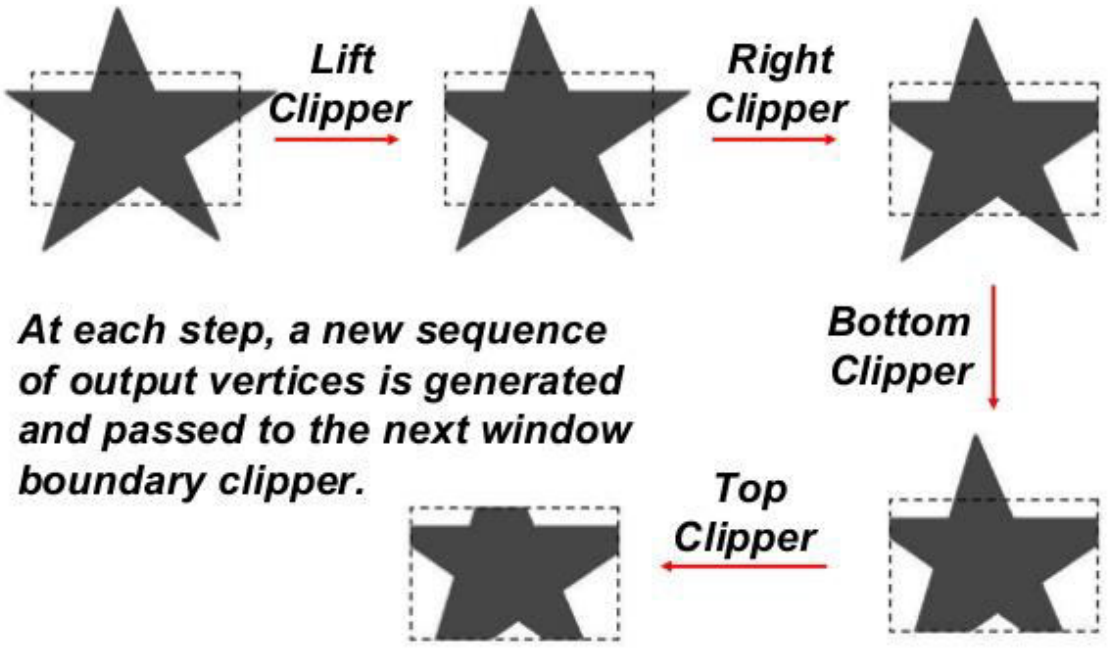
with the bottom side. The four iterations are similar; so, it is sufficient to describe the one 'generic' iteration. For this purpose, we need two utility functions:

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.



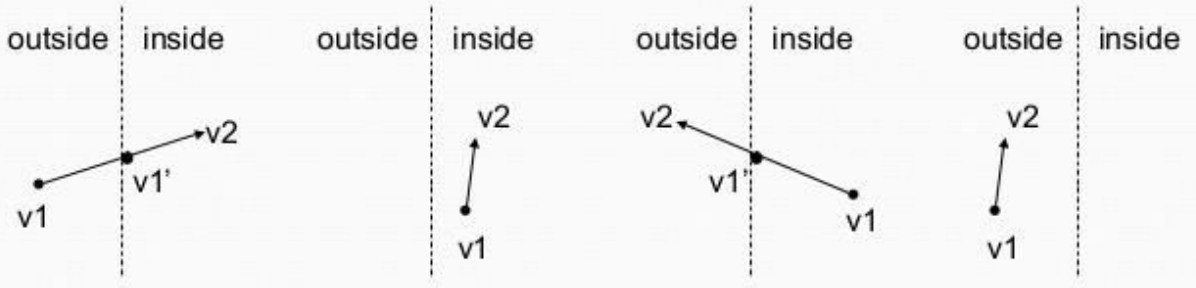
While processing an edge of a polygon with clipping window, an intersection point is found if edge is not completely inside clipping window and the a partial edge from the intersection point to the outside edge is clipped. The following figures show left, right, top and bottom edge clippings

Sutherland-Hodgman Polygon Clipping



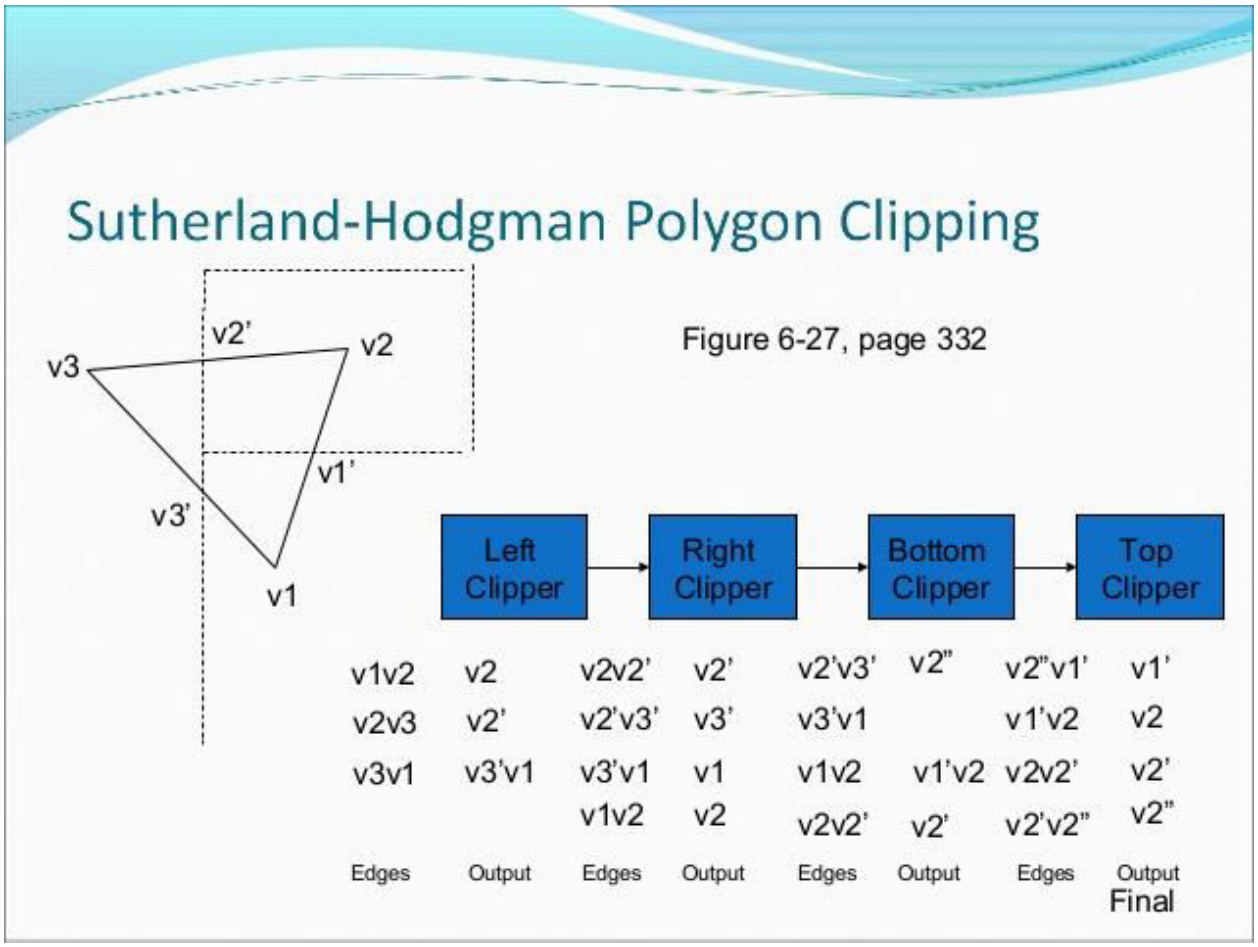
5

Sutherland-Hodgman Polygon Clipping: Four possible scenarios at each clipper



Sutherland-Hodgeman Polygon Clipping Algorithm:-

1. Read coordinates of all vertices of the Polygon.
2. Read coordinates of the clipping window
3. Consider the left edge of the window
4. Compare the vertices of each edge of the polygon, individually with the clipping plane.
5. Save the resulting intersections and vertices in the new list of vertices according to four possible relationships between the edge and the clipping boundary.
6. Repeat the steps 4 and 5 for remaining edges or the clipping window. Each time the resultant list of vertices is successively passed to process the next edge of the clipping window.
7. Stop.



[END OF UNIT – II]

UNIT – III

3D Graphics : 3D Representation Methods – 3D Transformations – Viewing And Projections – Parallel And Perspective Projections – Hidden Line Elimination – Hidden Surface Elimination.

3D Concepts

A three-dimensional coordinate system is just a fancy term for a system that measures objects with width, height, and depth (just like the real world). Similarly, 2-dimensional coordinate systems measure objects with width and height - ignoring depth properties (so unlike the real world).

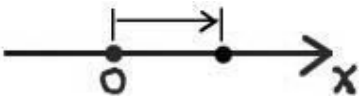
- It is the field of computer graphics that deals with generating and displaying three dimensional objects in a two-dimensional space(eg: display screen)
- In addition to color and brightness, a 3-D pixels adds a depth property that indicates where the point lies on the imaginary z-axis.
- When many 3-D pixels are combined, each with its own depth value, the result is a 3-D surface called a texture.

Coordinate Systems:

Coordinate systems are the measured frames of reference within which geometry is defined, manipulated and viewed. In this system, you have a well-known point that serves as the origin (reference point), and three lines(axes) that pass through this point and are orthogonal to each other (at right angles – 90 degrees).

Following are three types of the coordinate systems.

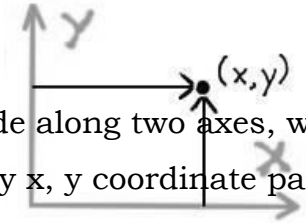
a) 1-D Coordinate Systems:



This system has the following characteristics:

- Direction and magnitude along a single axis, with reference to an origin
- Locations are defined by a single coordinate
- Can define points, segments, lines, rays
- Can have multiple origin

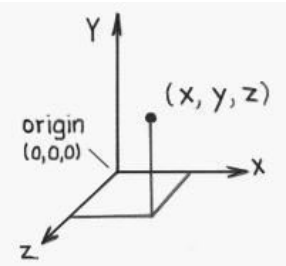
b) 2-D Coordinate Systems:



- Direction and magnitude along two axes, with reference to an origin
- Locations are defined by x, y coordinate pairs

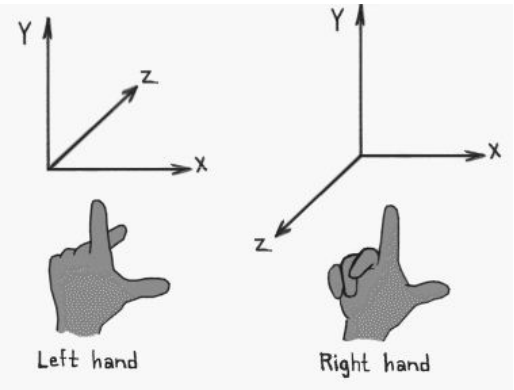
- Can define points, segments, lines, rays, curves, polygons, (any planar geometry)

b) 3-D Coordinate Systems:



- 3D Cartesian coordinate systems
- Direction and magnitude along three axes, with reference to an origin
- Locations are defined by x, y, z triples
- Can define cubes, cones, spheres, etc., (volumes in space) in addition to all one-and two-dimensional entities

Left-handed versus Right-handed

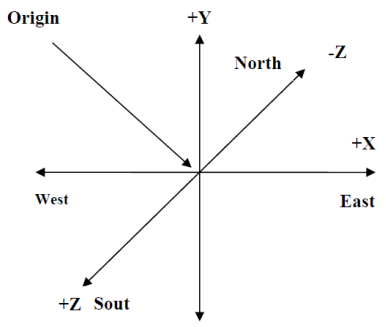


- Determines orientation of axes and direction of rotations
- Thumb = pos x, Index up = pos y, Middle out = pos z
- Most world and object axes tend to be right handed
- Left handed axes often are used for cameras.

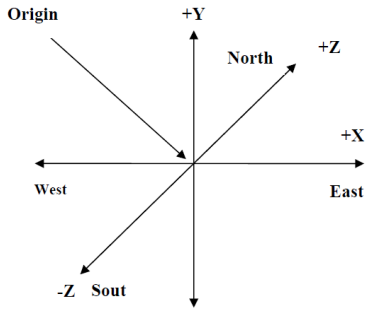
a) Right Handed Rule:

Right handed Cartesian coordinate system describes the relationship of the X,Y, and Z in the following manner:

- X is positive to the right of the origin, and negative to the left.
- Y is positive above the origin, and negative below it.
- Z is *negative* beyond the origin, and *positive* behind it.



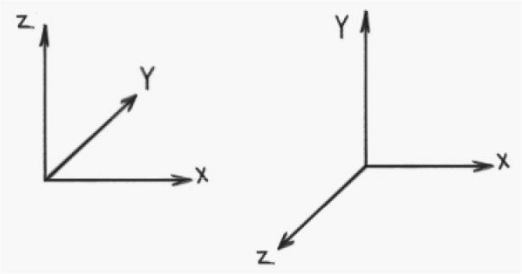
b) Left Handed Rule:



Left handed Cartesian coordinate system describes the relationship of the X, Y and Z in the following manner:

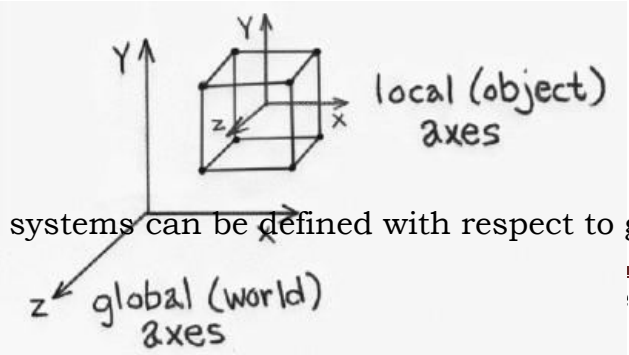
- X is positive to the right of the origin, and negative to the left.
- Y is positive above the origin, and negative below it.
- Z is positive beyond the origin, and negative behind it.

Y-up versus Z-up:



- z-up typically used by designers
- y-up typically used by animators

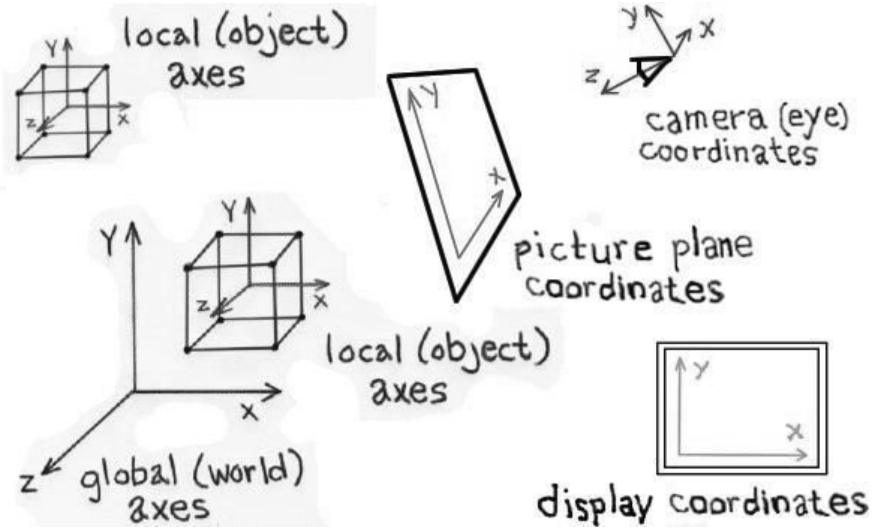
Global and Local Coordinate Systems:



- Local coordinate systems can be defined with respect to global coordinate system

- Locations can be relative to any of these coordinate systems
- Locations can be translated or "transformed" from one coordinate system to another.

Multiple Frames of Reference in a 3-D Scene:



3D display methods

Graphics is a field of many different kind of objects display on the screen. Types of objects: geometrical shapes, trees, terrains, clouds, rocks, glass, hair, furniture, human body, etc. Not possible to have a single representation for all.

In 2D object representations we can't access the proper view of an object, its only give the front view of the object. So to make the object clear there is a 3D object representation creating a proper view of any natural scene i.e., externally and internally. Now to design an object in 3D form we have to categorized the object into some model representation.

- Polygon and quadric surfaces provide precise description for simple Euclidean objects like polyhedrons, ellipsoids.
- Spline surfaces and construction techniques are useful for designing aircraft wings, gears, and other engineering structures with curved surfaces
- Procedural methods such as fractals constructions and particle systems give us accurate representations for the natural objects like clouds, trees etc.
- Physically based modeling methods using systems for interacting forces can be used to describe the non-rigid behaviors of piece of jelly, or a piece of cloth.
- Octrees encoding are used to represent internal features of the objects, such as those obtained from medical CT images, volume renderings and other visualization techniques.

1) Polygon Surfaces

Set of adjacent polygons representing the object exteriors.

- All operations linear, so fast.
- Non-polyhedron shapes can be approximated by polygon meshes.
- Smoothness is provided either by increasing the number of polygons or interpolated shading methods.

This simplifies and speeds up the surface rendering and display of object since all surfaces can be described with linear equations. This method simplifies and speeds up the surface rendering and display of objects. Objects are represented as a collection of surfaces. 3D object representation is divided into two categories.

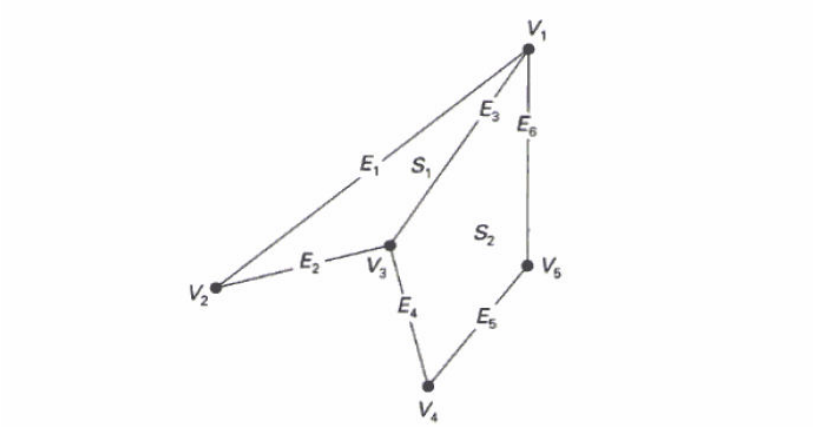
- **Boundary Representations (B-reps)**: It describes a 3D object as a set of surfaces that separates the object interior from the environment.
- **Space–partitioning representations:** It is used to describe interior properties, by partitioning the spatial region containing an object into a set of small, non-overlapping, contiguous solids (usually cubes).

The most commonly used boundary representation for a 3D graphics object is a set of surface polygons that enclose the object interior. Many graphics system use this method. Set of polygons are stored for object description. For other 3D objection representations, they are often converted into polygon surfaces before rendering

Polygon Tables

In this method, the surface is specified by the set of vertex coordinates and associated attributes. As shown in the following figure, there are five vertices, from v1 to v5. Each vertex stores x, y, and z coordinate information which is represented in the table as v1: x1, y1, z1.

- The Edge table is used to store the edge information of polygon. In the following figure, edge E1 lies between vertex v1 and v2 which is represented in the table as E1: v1, v2.
- Polygon surface table stores the number of surfaces present in the polygon.
- From the following figure, surface S1 is covered by edges E1, E2 and E3 which can be represented in the polygon surface table as S1: E1, E2, and E3.



VERTEX TABLE	
V_1:	x_1, y_1, z_1
V_2:	x_2, y_2, z_2
V_3:	x_3, y_3, z_3
V_4:	x_4, y_4, z_4
V_5:	x_5, y_5, z_5

EDGE TABLE	
E_1:	V_1, V_2
E_2:	V_2, V_3
E_3:	V_3, V_1
E_4:	V_3, V_4
E_5:	V_4, V_5
E_6:	V_5, V_1

POLYGON-SURFACE TABLE	
S_1:	E_1, E_2, E_3
S_2:	E_3, E_4, E_5, E_6

Figure: Polygon Table

Plane Equations

Plane Equations is the important topic of the Computer Graphics. Moreover, It is the Important subject of the Computer Science & Technological field.

The equation for plane surface can be expressed as:

$$Ax + By + Cz + D = 0$$

Where (x, y, z) is any point on the plane, and the coefficients A, B, C, and D are constants describing the spatial properties of the plane. We can obtain the values of A,B, C, and D by solving a set of three plane equations using the coordinate values for three non collinear points in the plane. Let us assume that three vertices of the plane are (x1, y1, z1), (x2, y2, z2) and (x3, y3, z3).

Let us solve the following simultaneous equations for ratios A/D, B/D, and C/D. You get the values of A, B, C, and D.

$$\frac{A}{D}x_1 + \frac{B}{D}y_1 + \frac{C}{D}z_1 = -1$$
$$\frac{A}{D}x_2 + \frac{B}{D}y_2 + \frac{C}{D}z_2 = -1$$
$$\frac{A}{D}x_3 + \frac{B}{D}y_3 + \frac{C}{D}z_3 = -1$$

Solving by determinant

$$A = \begin{vmatrix} 1 & y_1 & z_1 \\ 1 & y_2 & z_2 \\ 1 & y_3 & z_3 \end{vmatrix} \quad B = \begin{vmatrix} x_1 & 1 & z_1 \\ x_2 & 1 & z_2 \\ x_3 & 1 & z_3 \end{vmatrix} \quad C = \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix} \quad D = -\begin{vmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{vmatrix}$$

By expanding a determinant we can get values of A, B, C, D then stored in polygon data structure with other polygon data.

For any point (x, y, z) with parameters A, B, C, and D, we can say that –

If $Ax + By + Cz + D \neq 0$ the point (x,y,z) is not on that plane.

If $Ax + By + Cz + D < 0$ the point (x,y,z) is inside the surface.

If $Ax + By + Cz + D > 0$ the point (x,y,z) is outside the surface.

Polygon Meshes

- Construct the object from triangles or quadrilaterals by tiling
- Most models are constructed this way
- Easy to work with
- Easy to render
- Rates about 300 million triangles/sec available with ordinary graphics cards for PC computers

3D surfaces and solids can be approximated by a set of polygonal and line elements. Such surfaces are called **polygonal meshes**. In polygon mesh, each edge is shared by at most two polygons. The set of polygons or faces, together form the "skin" of the object.

Using a set of connected polygonally bounded planar surfaces to represent an object, which may have curved surfaces or curved edges. –

The wireframe display of such object can be displayed quickly to give general indication of the surface structure.

Realistic renderings can be produced by interpolating shading patterns across the polygon surfaces to eliminate or reduce the presence of polygon edge boundaries. –

Common types of polygon meshes are triangle strip and quadrilateral mesh.

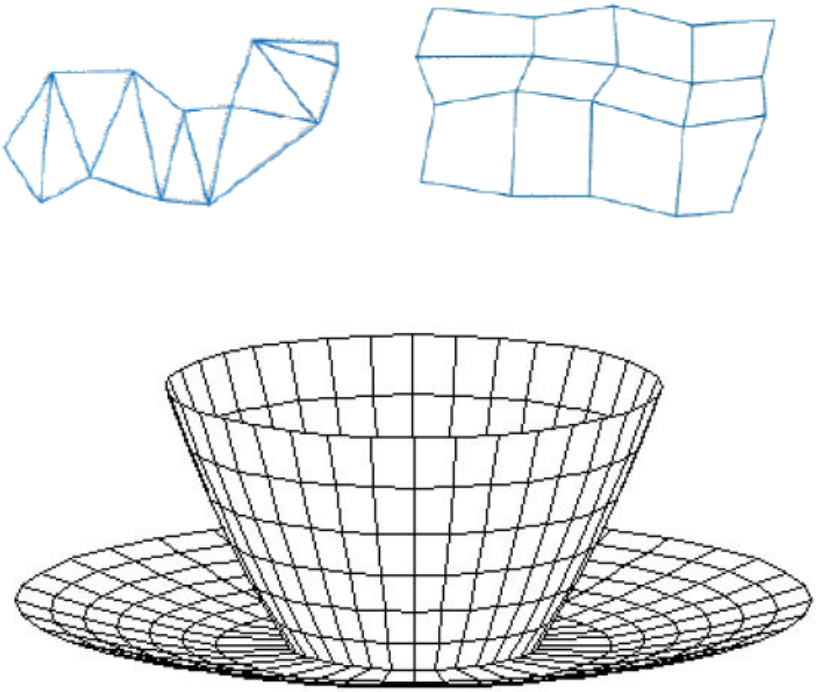


Figure: Polygon Mesh

Fast hardware-implemented polygon renderers are capable of displaying up to 1,000,000 or more shaded triangles per second, including the application of surface texture and special lighting effects.

Advantages

- It can be used to model almost any object.
- They are easy to represent as a collection of vertices.
- They are easy to transform.
- They are easy to draw on computer screen.

Disadvantages

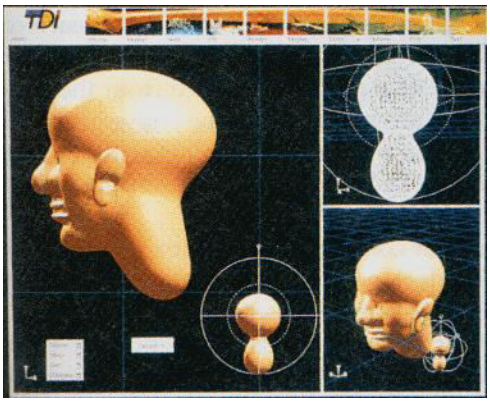
- Curved surfaces can only be approximately described.
- It is difficult to simulate some type of objects like hair or liquid.

Example for triangle mesh for an image

QUADRIC SURFACES

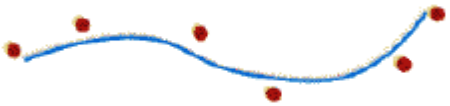
- These are frequently used class of objects. Quadric surfaces are described with second-degree equations. (Quadratics)
- The examples of quadric surfaces are spheres, ellipsoids, tori, paraboloids and hyperboloids.
- Spheres and ellipsoids are common elements of graphic scenes.

Some objects do not maintain a fixed shape, but change their surface characteristics in certain motions or when in proximity to other objects. Examples in this class of objects include molecular structures, water droplets and other liquid effects, melting objects, and muscle shapes in the human body. These objects can be described as exhibiting "blobbiness" and are often simply referred to as blobby objects, since their shapes show a certain degree of fluidity.



Spline Representations

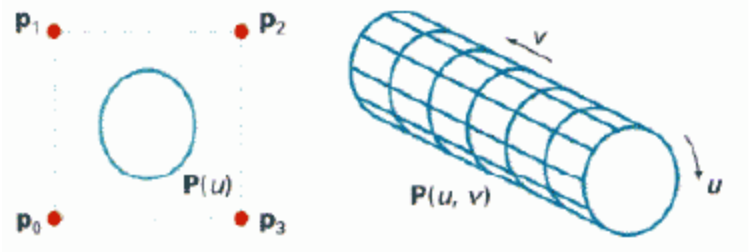
Spline means a flexible strip used to produce a smooth curve through a designated set of points.



Sweep representations

Sweep representations mean sweeping a 2D surface in 3D space to create an object. However, the objects created by this method are usually converted into polygon meshes and/or parametric surfaces before storing.

Translational Sweep:

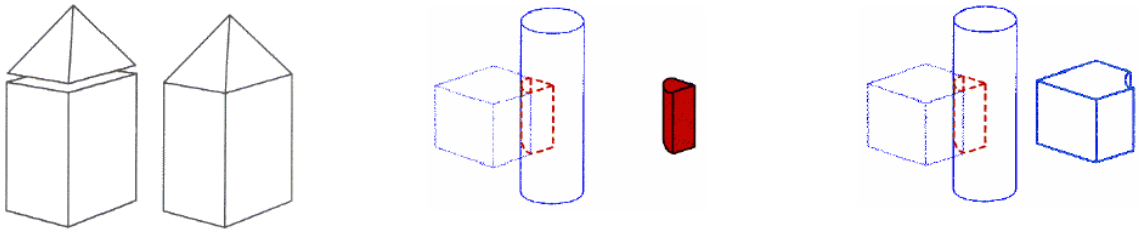


Constructive Solid-Geometry Methods

The Constructive Solid-Geometry Method (CSG) combines the volumes occupied by overlapping 3D objects using set operations:

- Union

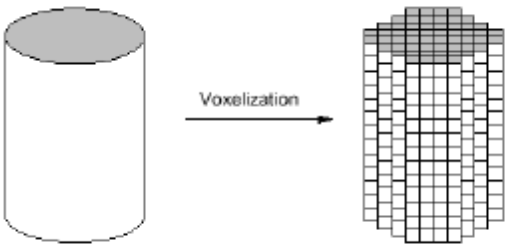
- Intersection
- Difference



Union **Intersection** **Difference**

voxel representation

In voxel representation, an object is decomposed into identical cells arranged in a fixed regular grid. These cells are called voxels (volume elements), in analogy to pixels.

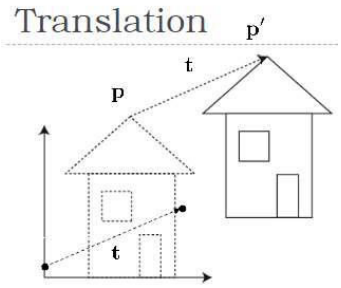


3D Tansformations

Translation

In 3D translation, we transfer the Z coordinate along with the X and Y coordinates. The process for translation in 3D is similar to 2D translation. A translation moves an object into a different position on the screen.

The following figure shows the effect of translation:



A point can be translated in 3D by adding translation coordinate (tx, ty, tz) to the original coordinate (X, Y, Z) to get the new coordinate (X', Y', Z').

$$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 1 \end{bmatrix}$$

$$P' = P \cdot T$$

$$[X' \quad Y' \quad Z' \quad 1] = [X \quad Y \quad Z \quad 1] \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 1 \end{bmatrix}$$

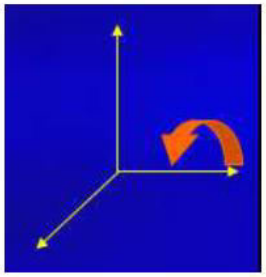
$$= [X + t_x \quad Y + t_y \quad Z + t_z \quad 1]$$

Rotation

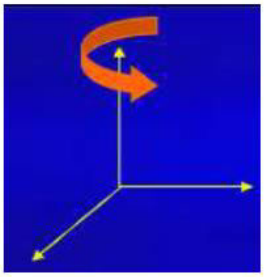
3D rotation is not same as 2D rotation. In 3D rotation, we have to specify the angle of rotation along with the axis of rotation. We can perform 3D rotation about X, Y, and Z axes. They are represented in the matrix form as below:

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

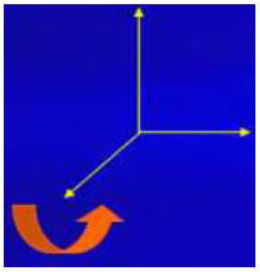
The following figure explains the rotation about various axes:



Rotation about x-axis



Rotation about y-axis



Rotation about z-axis

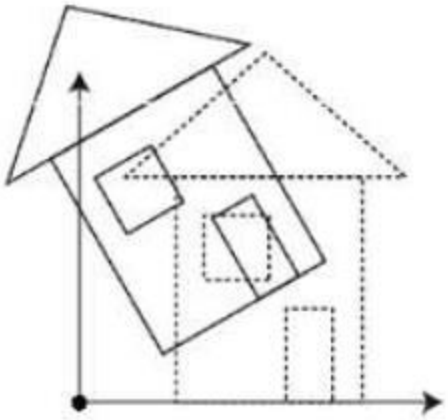


Figure: 3D Rotation

Scaling

You can change the size of an object using scaling transformation. 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. The following figure shows the effect of 3D scaling:

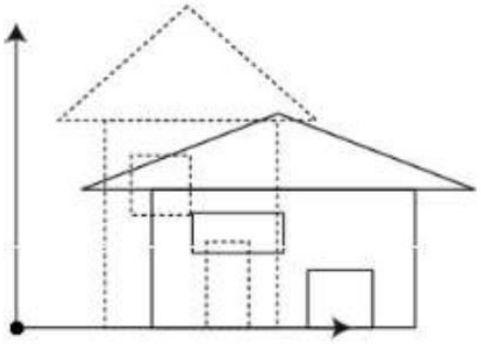


Figure: 3D Scaling

In 3D scaling operation, three coordinates are used. Let us assume that the original coordinates are (X, Y, Z), scaling factors are (SX, SY, Sz) respectively, and the produced coordinates are (X', Y', Z'). This can be mathematically represented as shown below:

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

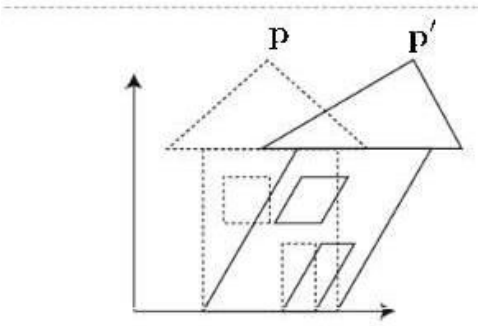
$$P' = P \cdot S$$

$$[X' \quad Y' \quad Z' \quad 1] = [X \quad Y \quad Z \quad 1] \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= [X \cdot S_x \quad Y \cdot S_y \quad Z \cdot S_z \quad 1]$$

Shear

A transformation that slants the shape of an object is called the **shear transformation**. Like in 2D shear, we can shear an object along the X-axis, Y-axis, or Z-axis in 3D.



As shown in the above figure, there is a coordinate P. You can shear it to get a new coordinate P', which can be represented in 3D matrix form as below:

$$Sh = \begin{bmatrix} 1 & Sh_x^y & Sh_x^z & 0 \\ Sh_y^x & 1 & Sh_y^z & 0 \\ Sh_z^x & Sh_z^y & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$P' = P \cdot Sh$$

$$X' = X + Sh_x^y\, Y + Sh_x^z\, Z$$

$$Y' = Sh_y^x\, X + Y + Sh_y^z\, Z$$

$$Z' = Sh_z^x\, X + Sh_z^y\, Y + Z$$

Transformation Matrices

Transformation matrix is a basic tool for transformation. A matrix with n x m dimensions is multiplied with the coordinate of objects. Usually 3 x 3 or 4 x 4 matrices are used for transformation. For example, consider the following matrix for various operation

$T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ t_x & t_y & t_z & 1 \end{bmatrix}$	$S = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$Sh = \begin{bmatrix} 1 & Sh_x^y & Sh_x^z & 0 \\ Sh_y^x & 1 & Sh_y^z & 0 \\ Sh_z^x & Sh_z^y & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Translation Matrix	**Scaling Matrix**	**Shear Matrix**

$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
Rotation Matrix		

3D Viewing and projection

Viewing in 3D involves the following considerations: - We can view an object from any spatial position, eg.

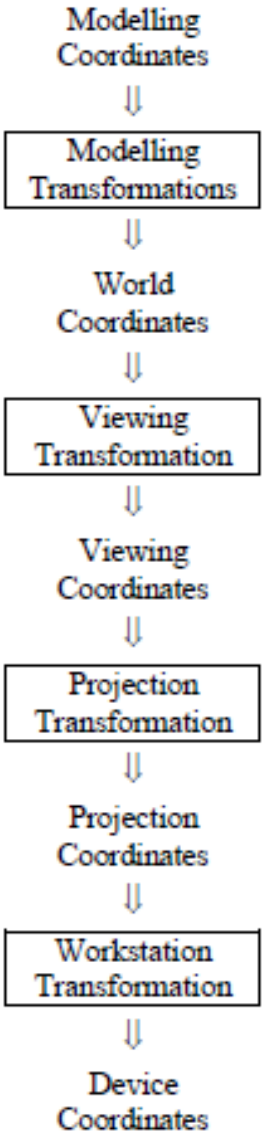
- In front of an object
- Behind the object
- In the middle of a group of objects
- Inside an object, etc.

We have already seen how to display a 2 dimensional picture on a view window, we have seen line clipping and polygon clipping that can he used to clip those regions of a 2 dimensional picture that are outside the window boundaries. We have seen window to view port transformation for mapping a 2 d scene to view port coordinates.

- For 3 dimensional applications, display of 3 dimensional objects involves a number of steps.
- Generating a view of a 3d scene is similar to taking a photograph of a scene using a camera. To take a photo, first we place the camera at a particular position. Then we change the direction of the camera. We can point the camera to a particular scene and we can rotate the camera around the 3d object. Finally, when we put the switch, the scene will be adjusted to the size of the window of the camera and light form the scene will be projected on to the camera film.

Steps involved in 3d viewing (conceptual)

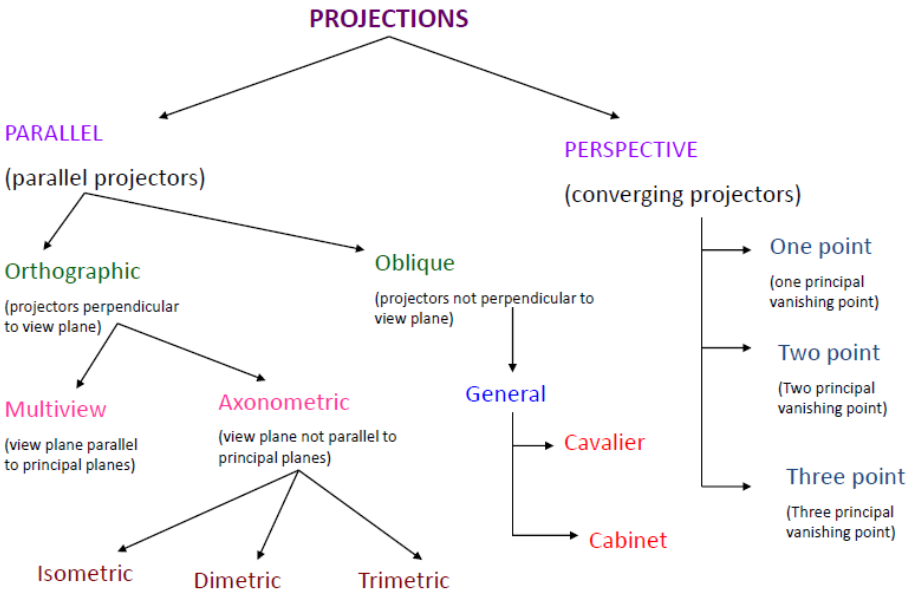
- In 3 dimensional viewing, we specify a view volume. Then a projection on to a projection plane and a mapping of this projected scene on to the view surface. Objects in 3d are clipped against the view volume and are then projected on to a plane.
- The contents in the projection plane are mapped (transformed) in to the view port for display.



Projection

For centuries, artists, engineers, designers, drafters, and architects have been facing difficulties and constraints imposed by the problem of representing a three-dimensional object or scene in a two-dimensional medium - the problem of projection. The implementers of a computer graphics system face the same challenge.

We have seen the method to convert world coordinate description of objects to viewing coordinates. Next we have to project 3 dimensional objects on to the 2 dimensional view planes.

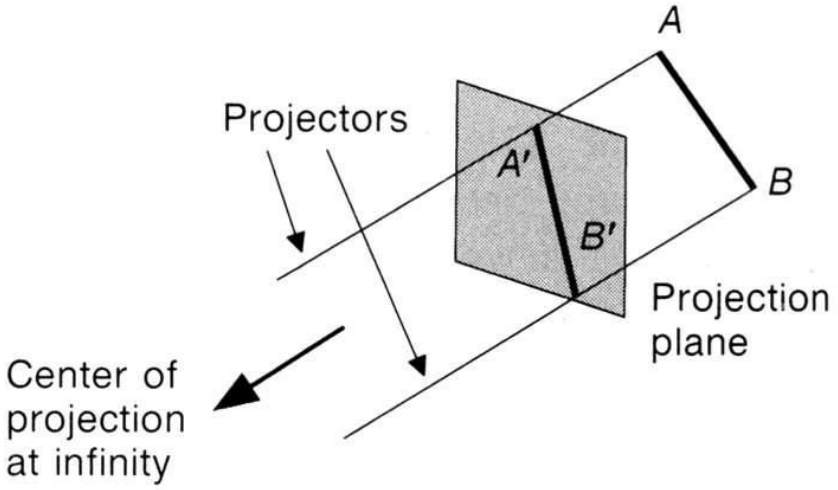


There are two basic methods of projection1) Parallel Projection 2) Perspective Projection. Key factor is the center of projection.

If distance to center of projection is finite : Perspective, If infinite : Parallel

Parallel Projection

In a parallel projection, coordinate positions are transformed to the view plane along parallel lines. 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. 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.

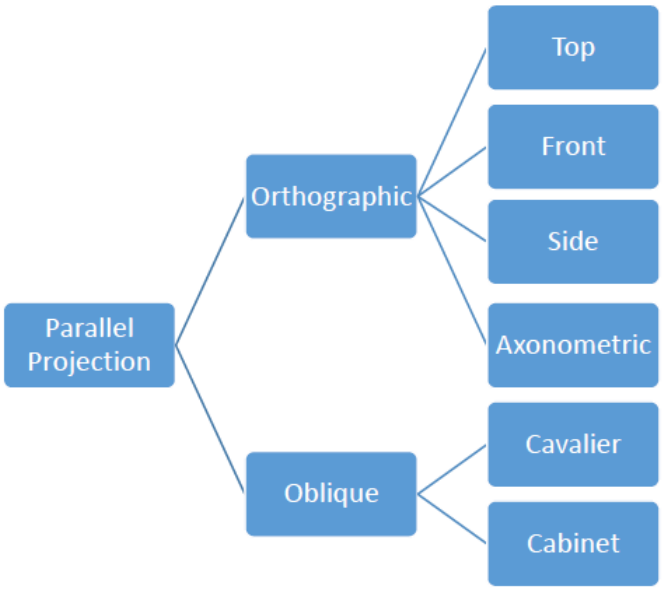
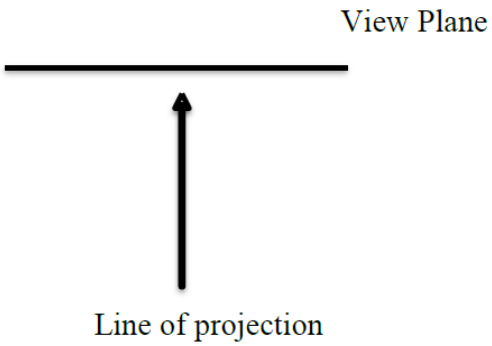


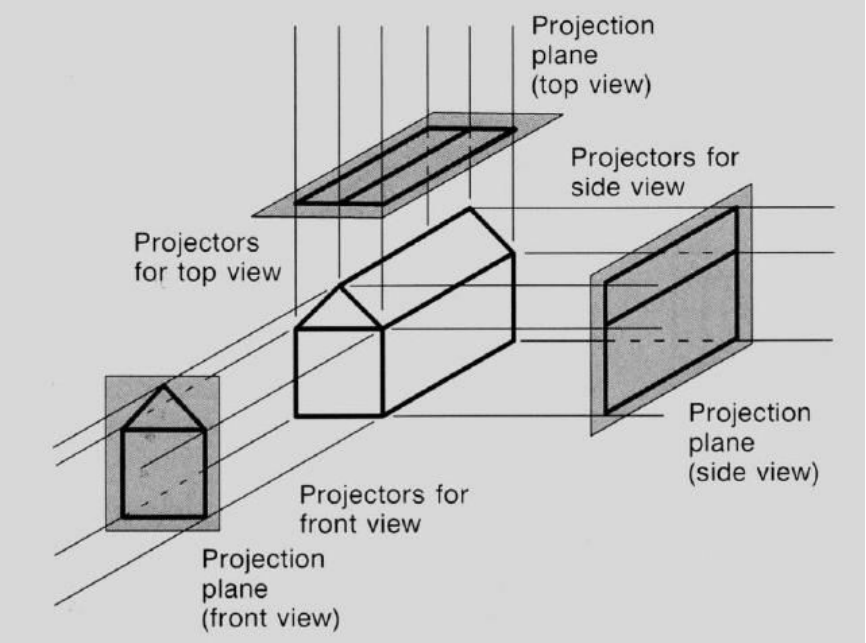
Figure: Types of Parallel Projection

Orthographic parallel projection

When the projection is perpendicular to the view plane, we have an orthographic parallel projection.



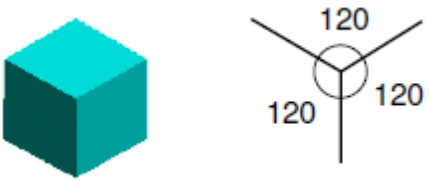
Orthographic projections are most often used to produce the front views and top view of the object. Front, side, and rear orthographic projections of an object are called elevations; and a top orthographic projection is called a plan view. Engineering and architectural drawings commonly employ these orthographic projections, because lengths and angles are accurately depicted, and can be measured from the drawings.



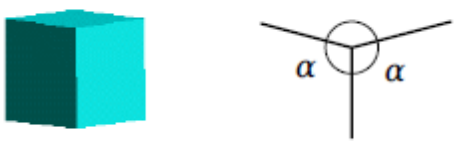
Axonometric Projections

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:

- Isometric: all angles between principal axes are equal



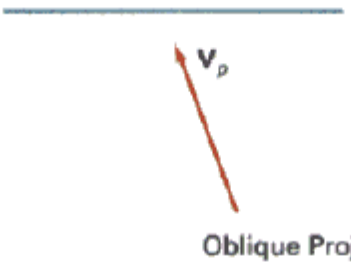
- Dimetric: angles between two principal axes are equal



- Trimetric: all angles different

Oblique parallel projection

An oblique projection is obtained by projecting points along parallel lines that are not perpendicular to the projection plane. Objects can be visualized better then with orthographic projections. Common oblique parallel projections are ***Cavalier*** and ***Cabinet***



Oblique 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. The Cabinet projection makes 63.4° angle with the projection plane. In Cabinet projection, lines perpendicular to the viewing surface is projected at ½ their actual length.

Cavalier: Angle between projectors and projection plane is 45°. Depth is projected full scale.

Cabinet: Angle between projectors and projection plane is 63.4°. Depth is projected ½ scale.

Both the projections are shown in the following figure:

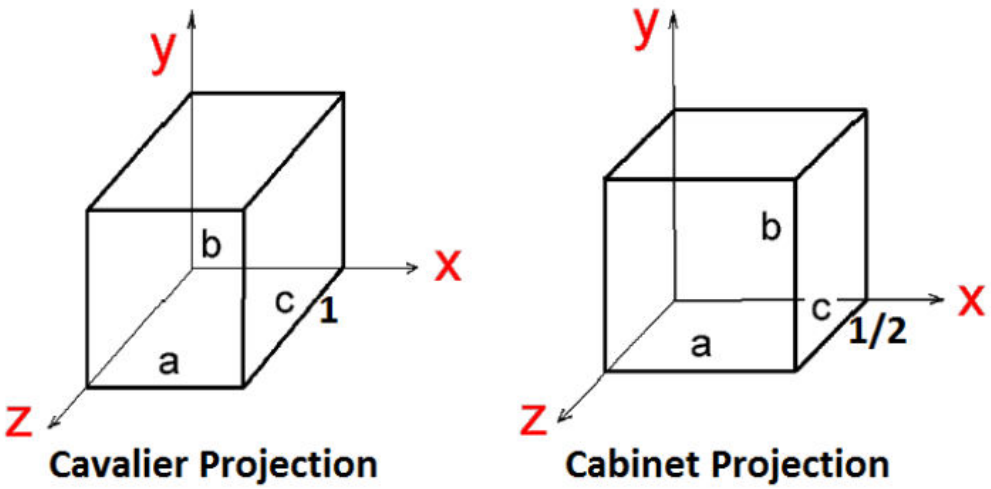
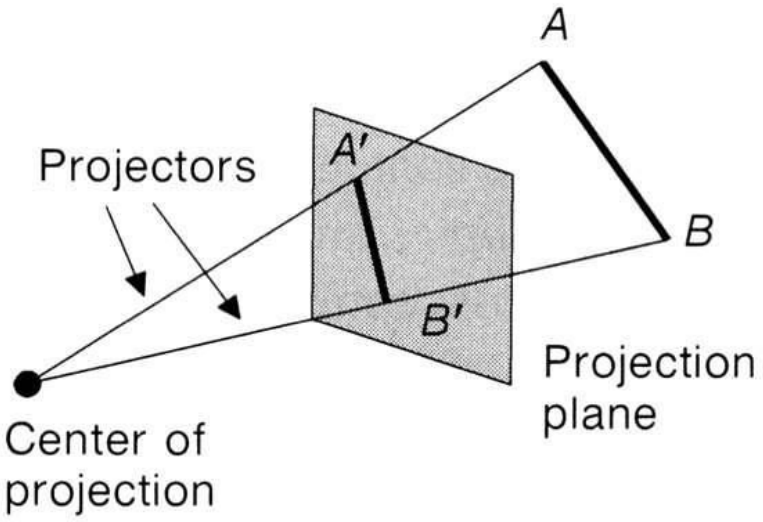


Figure: Cavalier & Cabinet Projection

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. Visual effect is similar to human visual system.

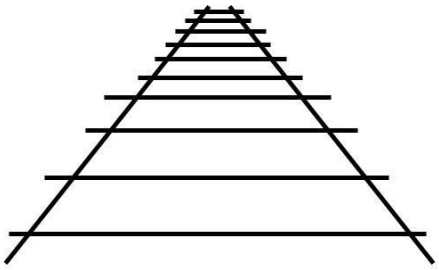
- In perspective projection, object positions are transformed to the view plane along lines that converge to a point called projection reference point (center of projection).



- Single point centre of projection (i.e. projection lines converge at a point).
- Shapes are projected smaller as their distances to the view plane increase.
- More realistic (human eye is a perspective projector).
- Depending on number of principal axes intersecting the viewing plane: 1, 2 or 3 vanishing points.

Vanishing points

Under perspective projections, any set of parallel lines that are not parallel to the PP will converge to a vanishing point.



Vanishing points of lines parallel to a principal axis x, y, or z are called principal vanishing points.

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 only z axis intersects single vanishing point. One point perspective projection is simple to draw.

Two point perspective x and z axes intersect two vanishing points. Two point perspective projection gives better impression of depth.

Three point perspective all axes intersect three vanishing points. Three point perspective projection is most difficult to draw.

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

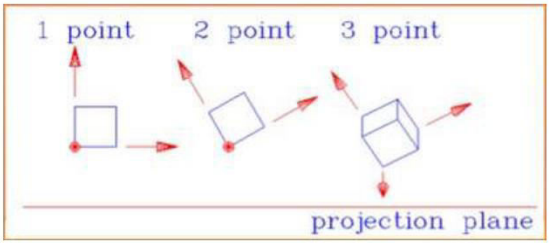
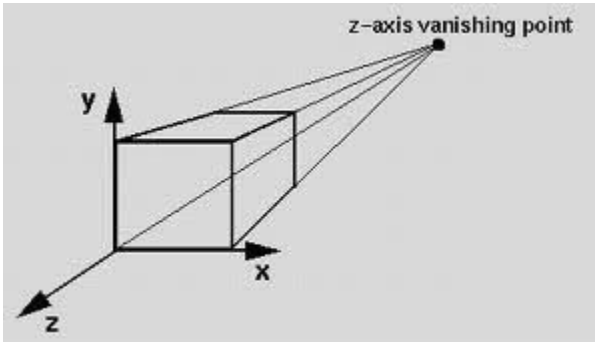
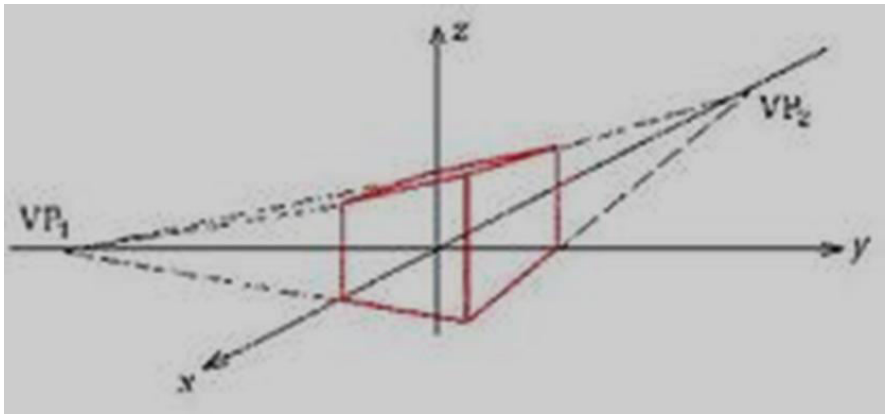


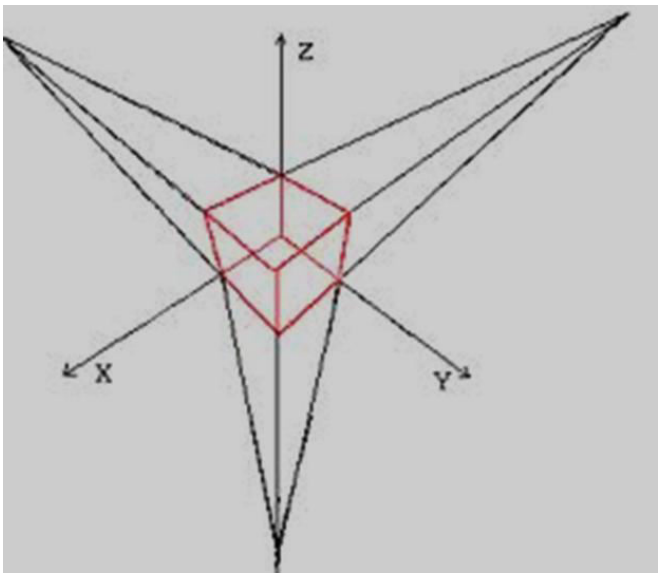
Figure: 1-point, 2-point, 3-point perspective projection



One-Point Perspective

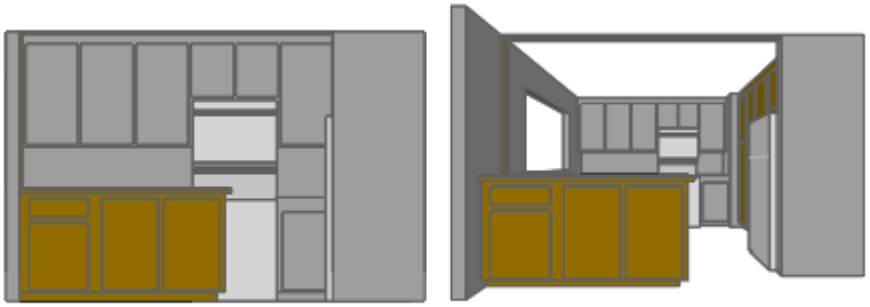


Two-Point Perspective



Three-Point Perspective

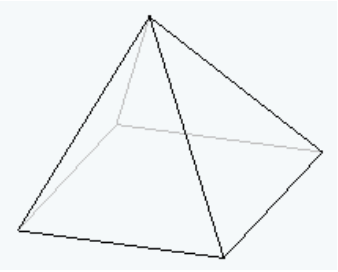
The following illustration shows the same model in both a parallel projection and perspective projection. Both are based on the same viewing direction.

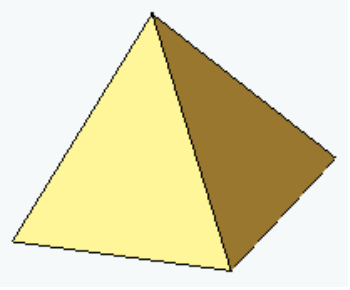


Parallel projection Perspective projection

Visible line identification (or) Hidden line removal

- Highlight the visible lines or display them in different color
- Display non-visible lines as dashed lines
- Remove the non-visible lines





Visible Surface Detection Methods (Hidden surface elimination)

Visible surface detection is the major concern for a realistic graphics for identifying those parts of scene which are visible from the chosen viewing position. Several algorithms have been developed. Some require more memory, some require more processing time & some apply only to the special types of objects.

These methods are broadly classified according to how they deal with objects or with their projected images. These two approaches are:

Object-Space methods: Compares objects & parts of objects to each other within scene definition to determine that surface, as whole, we should label as visible.

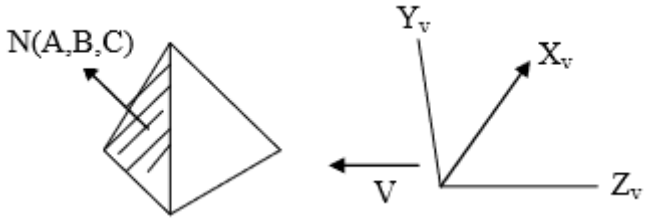
Image-Space methods: Visibility is decided point-by-point at each pixel position on projection plane. Most visible surface detection algorithm uses this method but in some cases object space methods are also used for it.

1) Back-face detection (Plane Equation method)

In a solid object, there are surfaces which are facing the viewer (front faces) and there are surfaces which are opposite to the viewer (back faces). These back faces contribute to approximately half of the total number of surfaces. Since we cannot see these surfaces anyway, to save processing time, we can remove them before the clipping process with a simple test. Each surface has a normal vector. If this vector is pointing in the direction of the center of projection, it is a front face and can be seen by the viewer. If it is pointing away from the center of projection, it is a back face and cannot be seen by the viewer. The test is very simple, if the z component of the normal vector is positive, then, it is a back face. If the z component of the vector is negative, it is a front face. Note that this technique only caters well for non overlapping convex polyhedral. For other cases where there are concave polyhedra or overlapping objects, we still need to apply other methods to further determine where the obscured faces are partially or completely

A fast & simple object space method used to remove the hidden surface from 3D object drawing is known as "Plane equation method" or back-face detection method. A point (x,y,z) is inside the polygon surface

if $Ax + By + Cz + D < 0$



We can simplify this test by considering normal vector N to polygon surface that has Cartesian components (A,B,C)

If V is vector in viewing direction from an eye position then this polygon is back face if, V.N>0

If , V,N < 0, then this polygon is front face .

So, simply our hidden surface removal routine computing V.N, visible surface are detected.

2) Depth-buffer-method (*z*-buffer method):

Depth Buffer Method is commonly used image space method for detecting visible surface. It is also known as *z*-buffer method. It compares surface depths at each pixel position on a projection plane. It is called *z*-buffer method as object depth is usually measured from view plane along the *z*-axis of the viewing system.

This approach compares surface depths at each pixel position on the projection plane. Object depth is usually measured from the view plane along the *z* axis of a viewing system. This method requires 2 buffers: one is the image buffer and the other is called the *z*-buffer (or the depth buffer). Each of these buffers has the same resolution as the image to be captured. As surfaces are processed, the image buffer is used to store the color values of each pixel position and the *z*-buffer is used to store the depth values for each (x,y) position.

Algorithm: Z-buffer

1. Initialize depth buffer & refresh buffer so that for all buffer position (x,y)
   depth (x,y) = -1, refresh (x,y) = $I_{background}$
2. For each position on each polygon surface, compare depth values to the previously stored value in depth buffer to the determine visibility.
   Calculate depth Z for each (x,y) position on polygon,
   If Z> depth (x,y) then
   depth (x,y) =Z
   refresh (x,y) = $I_{surface}$ (x,y)

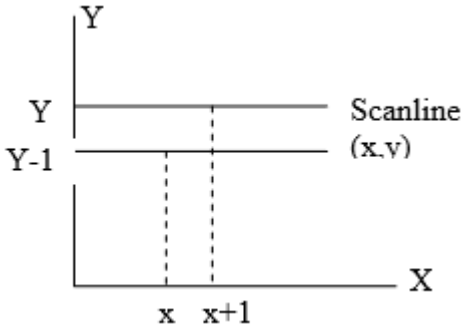
Where $I_{background}$ is an intensity value for background & $I_{surface}$ (x,y) is an intensity value for surface at pixel position (x,y) on the projected plane. After all surfaces are processed, depth buffer contains depth value of the visible surface & refresh buffer contains corresponding intensity values for those surface. The depth value of a surface position (x,y) are calculated by plane equation of surface.

$$Z = \frac{-Ax - By - D}{C}$$

Let Depth Z' at position (x+1,y)

$$Z' = \frac{-A(x+1) - By - D}{C}$$

$$\Rightarrow Z' = Z - A/C \quad \text{——} \quad (1)$$



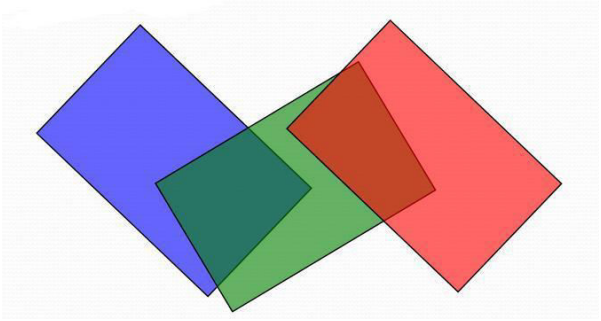
-A/C is constant for each surface so succeeding depth value v_i across the scan line are obtained from preceding values by simple calculation.

3) A-buffer method:

The depth-buffer method identifies only one visible surface at each pixel position: Cannot accumulate color values for more than one transparent and translucent surfaces. A buffer (also known as anti-aliased, area-averaged, accumulation buffer) is an extension to the Z buffer. A buffer algorithm was developed by Pixar. A buffer method could be used effectively for medium scale virtual memory computers. It is named so since z-buffer is used for depth semantics and in some sense a-buffer has the nature of the other end of alphabet. The same algorithm used by Z buffer is used with A buffer. However, A buffer provides anti-aliasing in addition to what Z buffer does. In A buffer, each pixel is made up of a group of sub pixels. The final color of a pixel is computed by summing up all of it sub pixels. A buffer gets the name accumulation buffer due to this accumulation taking place at sub pixel level.

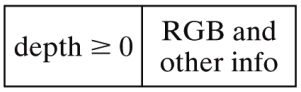
In Depth-Buffer method, only one visible surface can be identified at each pixel position, i.e. transparent surfaces cannot be displayed.

By expanding Z-Buffer method so that each position in depth-buffer can refer to linked list of surfaces also combinations of different surface properties e.g. transparency, can be computed to give final pixel intensity. The depth buffer is now called accumulation buffer instead.

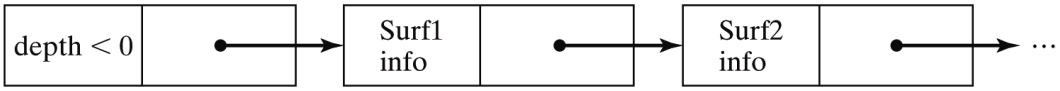


Each position in it has 2 fields:

- Depth field, stores the real number (≥ 0 or < 0)
- Intensity field, stores surface intensity information or pointer value



(a)



(b)

- If depth field ≥ 0, then the record with Number - depth of surface in that pixel position

  Surface data - surface color
- If depth field < 0, then multiple surfaces in those pixel positions. The intensity field then stores a pointer to a linked List of surface data. Color field stores the pointer to next surface and Surface information is included for each surface.

  Surface information in the A-buffer includes:

  a. RGB intensity components

  b. Opacity parameter (0-1, level of transparency)

  c. Depth

  d. Amount of area coverage

  e. Surface identifier

  f. Other surface material properties

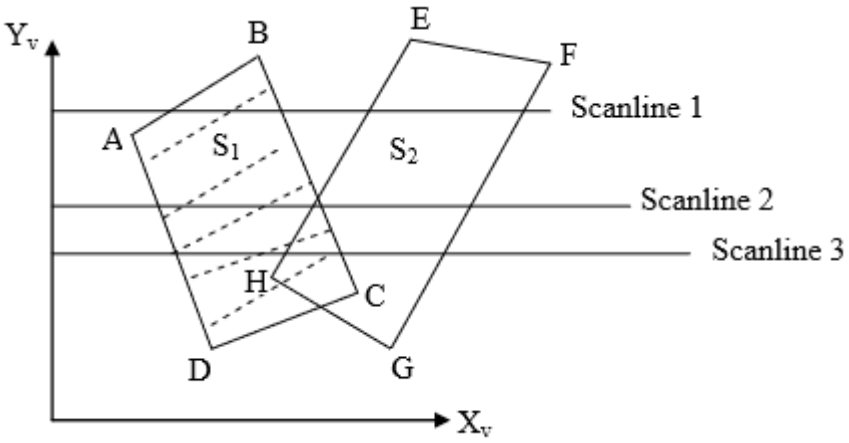
The algorithm proceeds just like the depth buffer algorithm. The depth and opacity values are used to determine the final colour of a pixel.

Difference between Z-Buffer and A-Buffer

Z buffer and A buffer are two of the most popular visible surface detection techniques. In fact, A buffer is an extension to Z buffer, which adds anti-aliasing. Typically, A buffer has a better image resolution than Z buffer, because it uses an easily computable Fourier window. However, A buffer is slightly costly than Z buffer.

4) Scan line method:

Scan line method is image space method for removing hidden surface which is extension of scan line polygon filling for polygon interiors. Instead of filling one surface we deal with the multiple surfaces here. As each scan line is processed, all polygon surfaces intersecting that line are examined to determine which are visible. We assume that polygon table contains co-efficient of a plane equation for each surface as well as vertex edge, surface information, intensity information for surface, & possibly pointers to edge table.



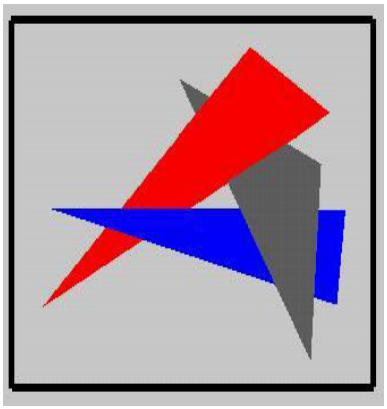
- In above figure, active edge list for scan line 1 contains information from edge table for edge AB, BC, EH, FG.
- For positions along scan line between edge AB & BC, only flag for surface S1 is on
- Therefore, no depth calculation must be made using plane co-efficient for 2 surfaces & intensity information for surface S1 is entered from polygon table into refresh buffer.
- Similarly between EH & FG. Only flag for S2 is on. No other positions along scan line 1 intersect surface. So intensity values in other areas are set to the background intensity.
- For Scan line 2 & 3, active edge list contains edges AD, EH BC, FG. Along-scan line 2 from edge AD, to the edge EH only surface flag for S1 is on, but between

edges EH & BC, flags for both surface is on. In this interval, depth calculation is made using plane coefficients for 2 surfaces.

For example, From viewer, if Z of surface S1 is less than the surface S2, So intensity of S1 is loaded into the refresh buffer until boundary BC is encountered. Then flag for surface S1 goes off & intensities for surface S2 are stored until edge FG is passed. Any number of overlapping surface is processed with these scan line methods.

5) Depth sorting method: (Painter Algorithm)

This method uses both object-space & image-space method. In this method surface representation of three dimension object are sorted in decreasing depth from viewer. Then sorted surface are scan converted in order, starting with surface of the greatest depth for a viewer.



Depth Sort Algorithm:

- Check if polygon is in front of any other.
    - If no, render it.
- If yes, Sort the polygons in the scene by their depth (Sort polygons by farthest depth).
    - Draw them back to front

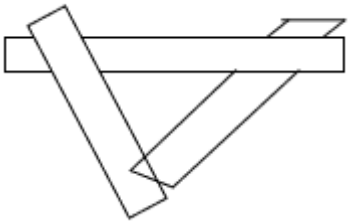
Steps:

1. Sort all surfaces according to their distances from the view point.
2. Render the surfaces to the image buffer one at a time starting from the farthest surface.
3. Surfaces close to the view point will replace those which are far away.
4. After all surfaces have been processed, the image buffer stores the final image.

In this method, newly displayed surface is partly or completely obscures previously displayed surface. Essentially, we are sorting surface into priority order such that surface with the lower priority can be hidden by those with higher priority.

This algorithm is also known as "Painter's Algorithm" as it simulates how the painter typically produces his painting by starting with background & then progressively adding new objects to a canvas.

Problem: One of the major problems in this algorithm is intersecting polygon surfaces. As shown in figure below:



Different polygons may have the same depth.

- The nearest polygon could be farthest.

We cannot use the simple depth sorting to remove hidden surfaces in the images.

Solution: For the intersecting polygons, we can split one polygon into two (or more) polygons which can then be painted from back to front. This needs more time to compute the intersection between polygons. So it becomes more complex algorithm for such surface existence.

The basic idea of this method is simple. When there are only a few objects in the scene, this method can be very fast. However, as the number of objects increases, the sorting process can become very complex and time consuming.

Example: Assuming we are viewing along the z axis. Surface S with the greatest depth is then compared to other surfaces in the list to determine whether there are any overlaps in depth. If no depth overlaps occur, S can be scan converted. This process is repeated for the next surface in the list. However, if depth overlap is detected, we need to make some additional comparisons to determine whether any of the surfaces should be reordered.

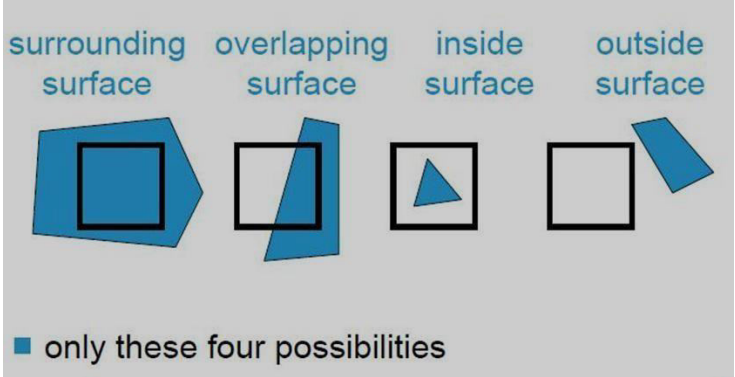
Area Subdivision Algorithms

The area-subdivision method takes advantage of area coherence in a scene by locating those view areas that represent part of a single surface. The total viewing area is successively divided into smaller and smaller rectangles until each small area is simple, ie. it is a single pixel, or is covered wholly by a part of a single visible surface or no surface at all. We continue this process until the subdivisions are

easily analyzed as belonging to a single surface or until they are reduced to the size of a single pixel.

An easy way to do this is to successively divide the area into four equal parts at each step. There are four possible relationships that a surface can have with a specified area boundary

- **Surrounding surface**-One that completely encloses the area.
- **Overlapping surface**-One that is partly inside and partly outside the area.
- **Inside surface**-One that is completely inside the area.
- **Outside surface**-One that is completely outside the area.



The tests for determining surface visibility within an area can be stated in terms of these four classifications.

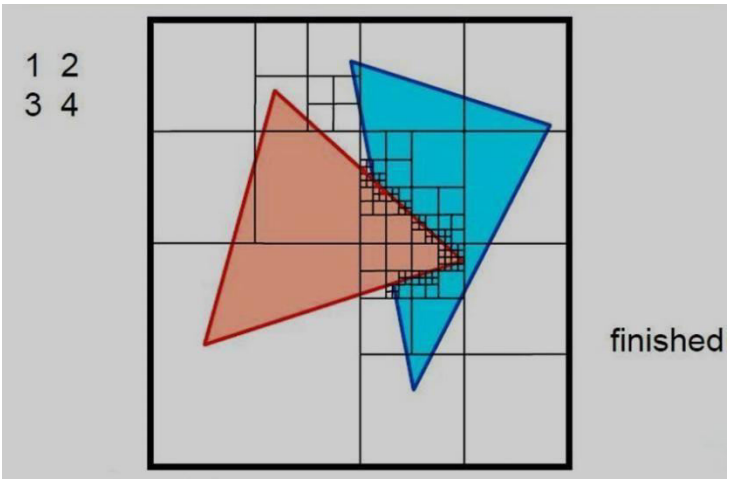
No further subdivisions of a specified area are needed if one of the following conditions is true:

- All surfaces are outside surfaces with respect to the area.
- Only one inside, overlapping, or surrounding surface is in the area.
- A surrounding surface obscures all other surfaces within the area boundaries

- if all three tests fail ⇒ do *subdivision*
  - subdivide area into four equal subareas
  - outside and surrounding surfaces will remain in this status for all subareas
  - some inside and overlapping surfaces will be eliminated $Ax + By + Cz + D < 0$
- no further subdivision possible (pixel resolution reached)
  - sort surfaces and take intensity of nearest surface

Example for **Area-Subdivision Method**

0



Once a single inside, overlapping or surrounding surface has been identified, its pixel entities are transferred to the appropriate area within the frame buffer.

Area Subdivision (Warnock's algorithm) – try to make an easy decision about which polygon is visible in a section of the image. If a decision cannot be made, subdivide the area recursively until one can be made. – work at image precision for subdivision, and at object precision for depth comparison.

For each area do the following tests

1. are all polygons disjoint from area?

if yes, display background color

2. only one intersecting or contained polygon.

if yes, fill with background color, and then draw contained polygon or intersecting portion

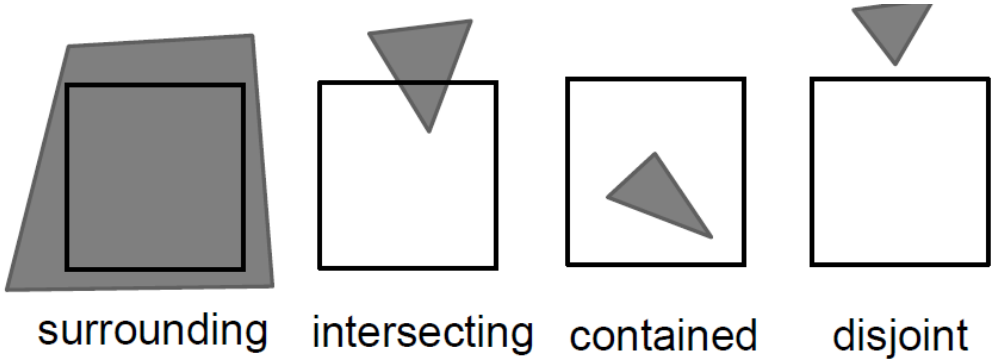
3. one single surrounding polygon, no intersecting or contained polygons.

if yes, draw area with that polygon's color

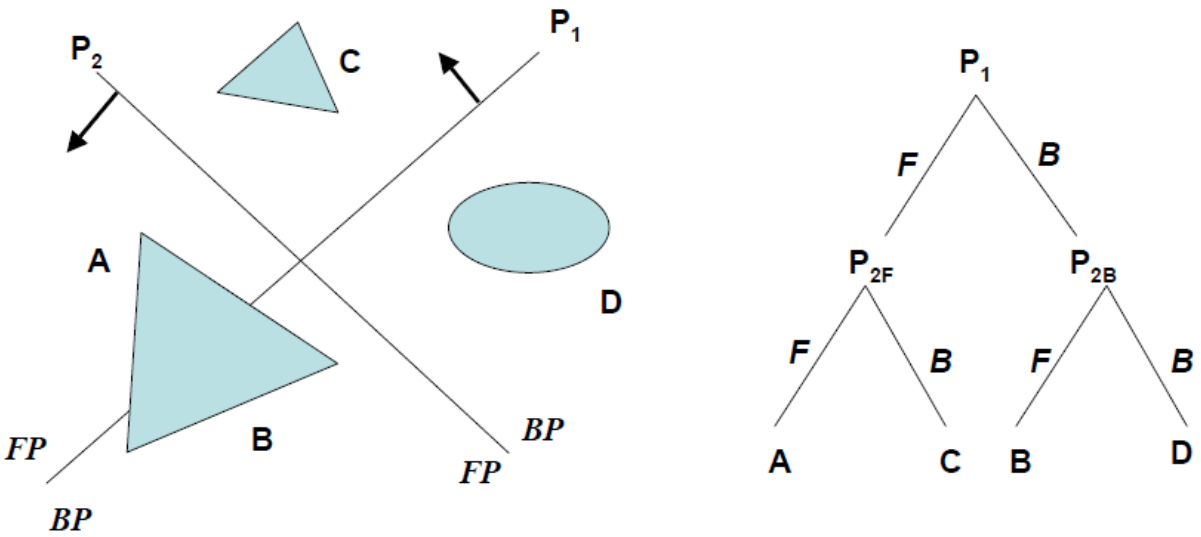
4. more than one polygon is intersecting, contained in, or surrounding, but only one polygon is surrounding the area and is in front of others

if yes draw area with that front polygon's color

5. otherwise, subdivide the area into 4 equal areas and recursive.



BSP (Binary Space Partition) trees Method



Salient features:

- Identify surfaces that are inside/front and outside/back the partitioning plane at each step of the space division, relative to the viewing direction.

- Start with any plane and find one set of objects behind and the rest in the front.
- In the tree, the objects are represented as terminal nodes with front objects as left branches and back objects as right branches.
- BSP tree's root is a polygon selected from those to be displayed.
- One polygon each from the root polygon's front and back half plane, becomes its front and back children
- The algorithm terminates when each node contains only a single polygon.

[END OF UNIT III]

Unit–IV : Graphics Modelling

Syllabus:

Curves, Surface And Solid Modeling – Color Model – Ray Tracing Methods – Graphic File Formats.

Curves, Surface And Solid Modeling

Curves

A curve is an infinitely large set of points. The points in a curve have a property that any point has 2 neighbours, except for a small number of points that have one neighbour (these are the endpoints). Some curves have no endpoints, either because they are infinite (like a line) or they are closed (loop around and connect to themselves).

The main aim of computer graphics is to display an arbitrary surface so that it looks real. The first step toward this goal is an understanding of curves. The user starts by entering the coordinates of points, either by scanning a rough image of the desired shape and digitizing certain points on the image, or by drawing a rough shape on the screen and selecting certain points with a pointing device such as a mouse. After the curve has been drawn, the user may want to modify its shape by moving, adding, or deleting points. Such points can be employed in two different ways:

1. We may want the curve to pass through them. Such points are called **data points** and the curve is called an **interpolating curve**.
2. We may want the points to control the shape of the curve by exerting a "pull" on it. A point may pull part of the curve toward it, allowing the user to change the shape of the curve by moving the point. Generally, however, the curve does not

pass through the point. Such points are called **control points** and the curve is called an **approximating curve**.

The problem that we need to address is how to describe a curve - to give "names" or representations to all curves so that we can represent them on a computer. For some curves, the problem of naming them is easy since they have known shapes: line segments, circles, elliptical arcs, etc. A general curve that doesn't not have a "named" shape is sometimes called a free-form curve. Because a free-form curve can take on just about any shape, they are much harder to describe.

There are three main ways to describe curves mathematically:

Explicit curve

A mathematical function y = f(x) can be plotted as a curve. Such a function is the explicit representation of the curve. The explicit representation is not general, since it cannot represent vertical lines and is also single-valued. For each value of x, only a single value of y is normally computed by the function.

Implicit curve

The implicit representation of a curve has the form f(x, y) = 0. It can represent multivalued curves (more than one y value for an x value). A common example is the circle, whose implicit representation is $x^2 + y^2-R^2= 0$.

Parametric Curves

Curves having parametric form are called parametric curves. The explicit and implicit curve representations can be used only when the function is known. In practice the parametric curves are used. A two-dimensional parametric curve has the following form -

P(t) = f(t), g(t) or P(t) = x(t), y(t)

The functions f and g become the (x, y) coordinates of any point on the curve, and the points are obtained when the parameter t is varied over a certain interval [a, b], normally [0, 1].

BEZIER CURVE and B-SPLINE

In computer graphics, a smooth curve that passes through two or more points. Splines are generated with mathematical formulas. Two of the most common types of splines are *Bezier curves* and *b-spline curves.*

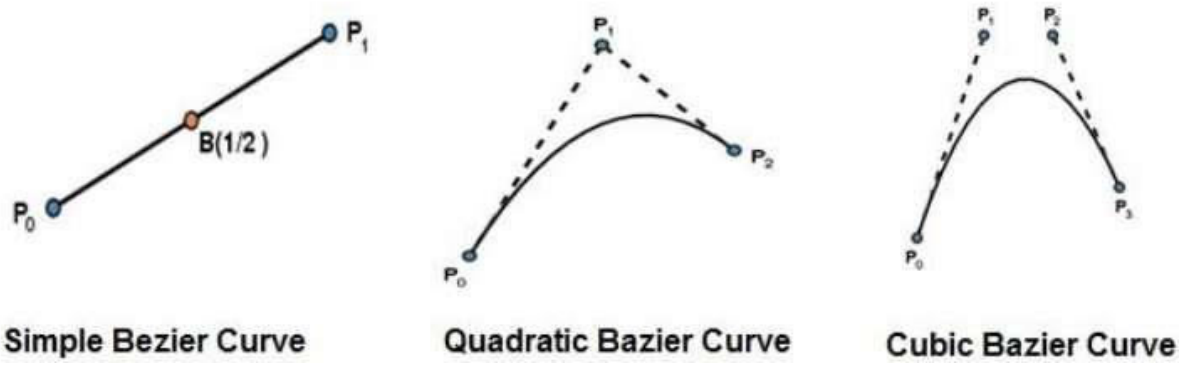
Bezier Curves

French Engineer Pierre Bézier has discovered Bezier curve. The curves are generated under the control of other points. Curve is generated by using approximate tangents. Mathematically the curve is represented as:

$$x(u) = \sum_{i=0}^{i=n} n_{c_i} \ (1-\mathrm{u})^{\mathrm{n-i}}\mathrm{u}^{\mathrm{i}} \ \mathrm{x_i}$$

$$0 \leq u \ \leq 1$$

The simplest Bézier curve is the straight line from the point P0 to P1. Three control points determine a quadratic Bezier curve and four control points determine cubic Bezier curve.



Simple Bezier Curve Quadratic Bazier Curve Cubic Bazier Curve

Properties of Bezier Curves

The following are some of the properties of Bezier curve:

- It takes the shape of the control polygon consisting of segments that join the control points.
- They pass through the first and last control points.
- The convex hull of the control points contains these curves.
- The degree of the polynomial defining the curve segment is one less that the number of defining polygon point. Therefore, for 4 control points, the degree of the polynomial is 3, i.e. cubic polynomial.
- The shape of the defining polygon is usually followed by the Bezier curve.
- The vector that determines the first and last segments is the same as that of the direction of the tangent vector.
- The convex hull property for a Bezier curve ensures that the polynomial smoothly follows the control points.
- No straight line intersects a Bezier curve more times than it intersects its control polygon.

- Under an affine transformation, they are very invariant.
- Global control is exhibited by Bezier curves. In the sense that the shape of the whole curve is altered by the movement of a control point.
- A Bezier curve can be subdivided at a point t=t0 into two Bezier segments which join together at the point corresponding to the parameter value t=t0.

B-Spline Curves

In Bezier curve , if we want to define control points, its order is fixed. Suppose if we take 4 control popints then we have a cubic Bezier curve or we take 5 control points then we have a quartic Bezier curve and so on.

In B-Spline curves, these are basically a part of this attempt where given a set of control points, the order which we choose for the curve is independent of the control points. We have a freedom to basically fix whether we want a curve which is quadratic or cubic or higher order as and when they are required.

-Curve is defined by n+1 control points and the order (k) of the curve.

- The curve has an advantage that it has local propagation unlike global propagation properties of Bezier curves.

- The curve can be used to define both open and closed curves.

In Bezier curve , Changing any control point will change the entire shape of the curve. Because every point on the curve is defined by all thye control points. This is called global propagation of the curve where as here the curve is divided into number of segments. Infact a B- Spline curve is not a single curve, you can call it as a composite curve, number of curve which are consisting requirements. Now the advantage of defining a curve in terms of segments is that a particular segment of a curve is influenced only by a certain control points.

changing one of the control point then only a some portion of the curve only a few, we can say pieces of the curve , few segments of the curve they are influenced. That is a major advantage of this particular curve.

Mathematical definition

$$x(u) = N_{0,3}(U)x_0 + N_{1,3}(U)x_1 + N_{2,3}(U)x_2 + N_{3,3}(U)x_3 + N_{4,3}(U)x_4 + N_{5,3}(U)x_5$$

B-Spline curve

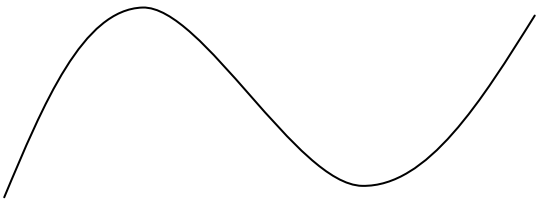
$$x(u) = \sum_{i=0}^{i=n} n_{i\,k}(\mathrm{u})\,\mathrm{x_i}$$

$$0 \le u \le n - k + 2$$

k is the order of the polynomial segments of he B-spline curve. Order k means that the curve is made up of piecewise polynomial segments of degree k - 1,

Parameter range is not from 0 to 1. Depending on how many control points which we choose and depending what is the value of K which we choose, we will have a different parameter range. Take for example, if we have n is equal to 6 that means i have 7 control points and the K is equal to 3 then what is the range of u? So here this is N-K+2.So this is 6 minus 3 and plus 2, so that is 5. So there are five segments of the curve, the first segment goes from U is equal to 0 to 1.Second one has a parameter range which is equal to 1 to 2; third as 2 to 3; and so on. We have about five segments of the curve, which will define the complete curve.

Non Unniform B-Spline curve



B-spline: Benefits

The use of B-splines has become a basic design tool in many graphics systems (e.g., trueSpace, LightWave 3D, 3D Studio and Maya) and is widely used in many interdisciplinary areas

Properties of B-spline Curve

The following are the properties of B-spline curves:

- The sum of the B-spline basis functions for any parameter value is 1.
- Each basis function is positive or zero for all parameter values.
- Each basis function has precisely one maximum value, except for k=1.
- The maximum order of the curve is equal to the vertices that define the polygon.
- The number of vertices defining the polygon and the degree of B-spline polynomial are independent.
- The local control over the curve surface is allowed by B-spline, since each of the vertex affects the shape of the curve and where the associated basis function is nonzero.
- The variation diminishing property is exhibited by the curve.
- The shape of the defining polygon is followed.

- By applying to the vertices of defining polygon, an affine transformation is applied to the curve.
- The curve line within the convex hull of its defining polygon.

The definition of B- Spline curves to define what we call as a non – uniform rational B-Spline. (or) NURBS curve.

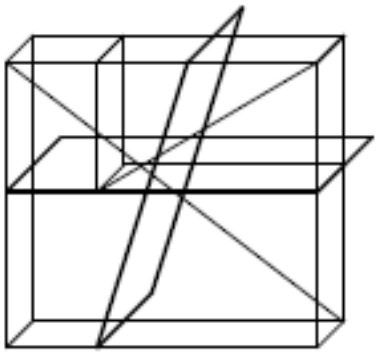
Solid Modelling

Three types of 3D modeling are supported: **wireframe**, **surface**, and **solid**. Each type has its own creation and editing techniques.

A wireframe model is a skeletal description of a 3D object. There are no surfaces in a wireframe model; it consists only of points, lines, and curves that describe the edges of the object. Because each object that makes up a wireframe model must be independently drawn and positioned, this type of modeling can be the most time-consuming.

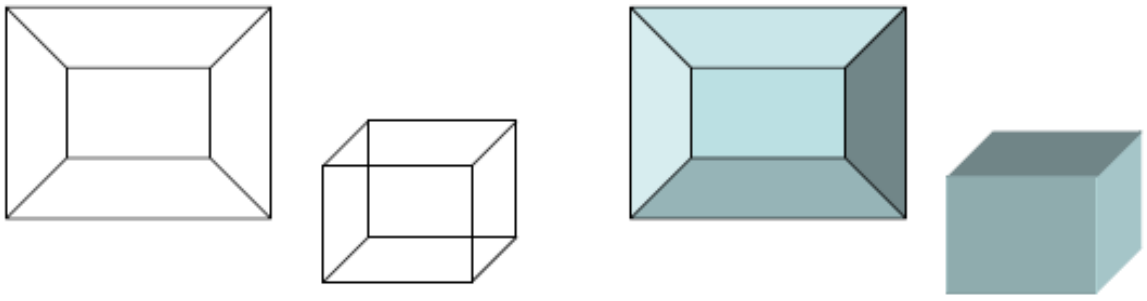
Disadvantages of Wireframe Model

1. Ambiguous in the way to represent an object.
2. Not suitable for
    a. Mass property calculations
    b. Hidden surface Removal
    c. Shaded images generation



Surface modeling is more sophisticated than wireframe modeling in that it defines not only the edges of a 3D object, but also its surfaces. **Solid modeling is the easiest type of 3D modeling to use**. With the solid modeler, you can make 3D objects by creating basic 3D shapes: boxes, cones, cylinders, spheres, wedges, and tori (donuts). You can then combine these shapes to create more complex solids by joining or subtracting them or finding their intersecting (overlapping) volume. You can also create solids by sweeping a 2D object along a path or revolving it about an axis.

Solid modeling gives a complete and unambiguous definition of an object, describing not only the shape of the boundaries but also the object's interior and exterior regions.



Wireframe Model Solid Model

Basics of Solid Modeling Theory

The fundamental geometric principles are **Geometry and topology** –

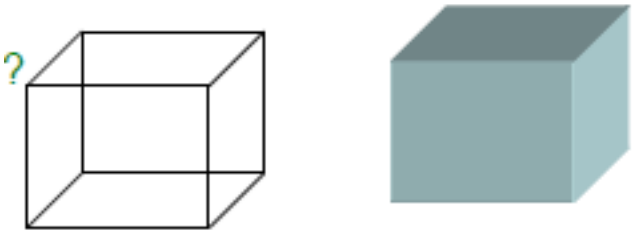
Geometry: Geometry relates to the information containing shape-defining parameters, such as the coordinates of the vertices in a polyhedral object, dimension, length, angle, area and transformations

Topology: Topology describes the connectivity among the various geometric components, i.e. the relational information between the different parts of an object The invisible information about the connectivity, neighbourhood, associatively etc.,

Advantages of Solid Models

Unlike wireframes and surface representations which contain only geometrical data, the solid model uses topological information in addition to the geometrical information to represent the object unambiguously and completely.

Solid model results in accurate design, helps to further the goal of CAD/ CAM like CIM, Flexible manufacturing leading to better automation of the manufacturing process.



Geometry Vs Topology

Definition of a Solid Model

A solid model of an object is a more complete representation than its surface (wireframe) model. It provides more topological information in addition to the geometrical information which helps to represent the solid unambiguously.

Types of Solid Modelling

1. Half Spaces.
2. Boundary Representation (B-Rep)
3. Constructive Solid Geometry (CSG)
4. Sweeping
5. Analytical Solid Modeling (ASM)
6. Cell decomposition
7. Spatial Enumeration
8. Octree encoding
9. Primitive Instancing

Among these, the 3 most popular schemes are: B-Rep, CSG, Sweeping

COLOR MODEL

Any method for explaining the properties or behavior of color within some particular context is called a Color Model.

Additive color

Uses light to display color. Mixing begins with black and ends with white; as more color is added, the result is lighter and tends to white. Used for computer displays

Example: The RGB colors are light primaries and colors are created with light.

subtractive color

Uses ink to display color. Mixing means that one begins with white and ends with black; as one adds color, the result gets darker and tends to black. Used for printed material

It is called 'subtractive' because its wavelength is less than sum of the wavelengths of its constituting colors.

Example: The CMYK color system is the color system used for printing.

RGB stands for the three primary colors of light – Red, Green, and Blue. RGB can be described as the computer's native color space for capturing images and displaying them. As human eyes are sensitive to these primary colors – red, green, and blue – all colors are perceived as a combination of these three colors. The RGB color model,

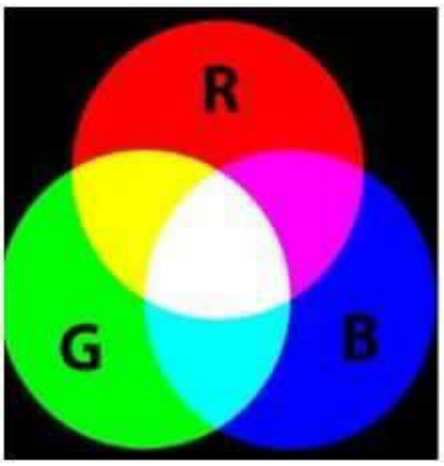
based on a Cartesian coordinate system, is considered as an addictive model in which red, green, and blue, are combined in several methods to reproduce all other colors.

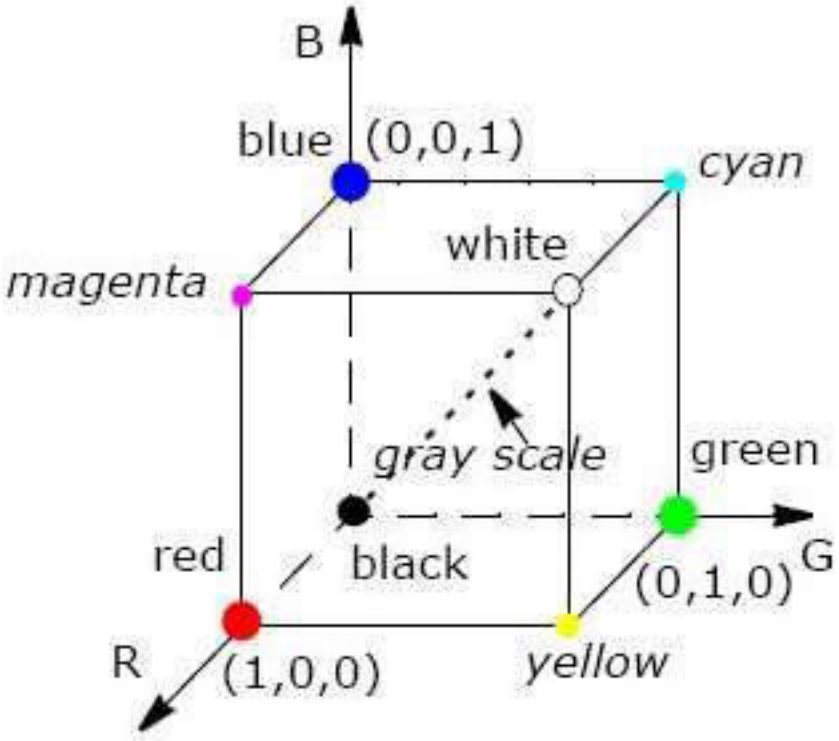
RGB and Displays

One of the most widely used application of the RGB color model is the display of colors on CRT, LCD, or plasma display, such as a television or a computer's monitor. Cameras and scanners also work in the same manner; it captures color with sensors which record the varying intensities of RGB at each pixel location in the frame.

Some examples of 24-bit representation of colors:

- (255, 255, 255) represents white
- (0, 0, 0) represents black
- (255, 0, 0) represents red
- (0, 0, 255) represents blue
- (0, 255, 0) represents green
- (255, 255, 0) represents yellow
- (255, 0, 255) represents magenta
- (0, 255, 255) represents cyan





Applications of RGB

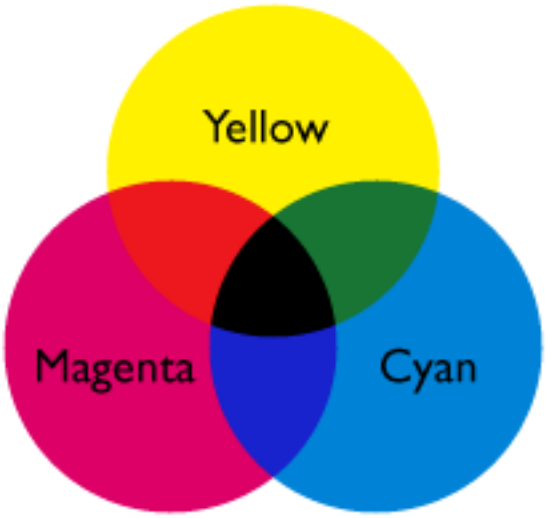
The common applications of RGB model are

- Cathode ray tube (CRT)
- Liquid crystal display (LCD)
- Plasma Display or LED display such as a television
- A computer monitor or a large scale screen

CMYK

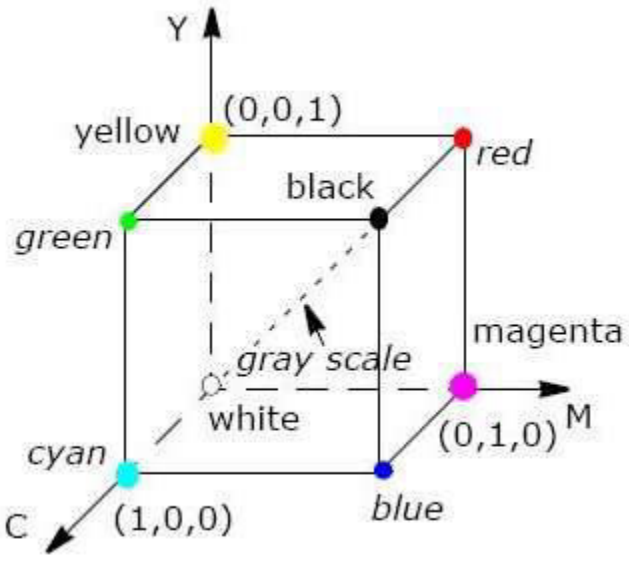
RGB is not used for printing on paper, instead CMYK – also called CYM or YMCK – is a subtractive color model used in color printing. The combinations of cyan, magenta, yellow and key (black) ink make up all the colors needed for printing.

In this model, Cyan, Magenta and Yellow are added together to get the resultant color BLACK.



Subtractive primaries and color mixing:

Magenta + Yellow = Red
Magenta + Cyan = Blue
Cyan + Yellow = Green
Magenta + Yellow + Cyan = Black
None = White



Each color point within the bounds of the cube is represented as the triple (C,M,Y).where value for C,M,Y are also assigned in the range from 0 to 1.

Here CMY color also place together at 120 degree. CYAN+ MAGENTA + YELLO = BLACK(contribute)

All other colors are generated from these three primary colors.

$$\begin{pmatrix} C \\ M \\ Y \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

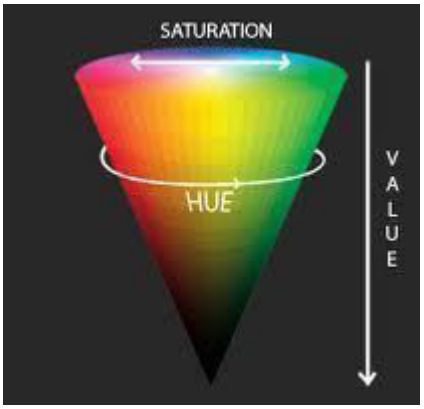
HSV Color Model

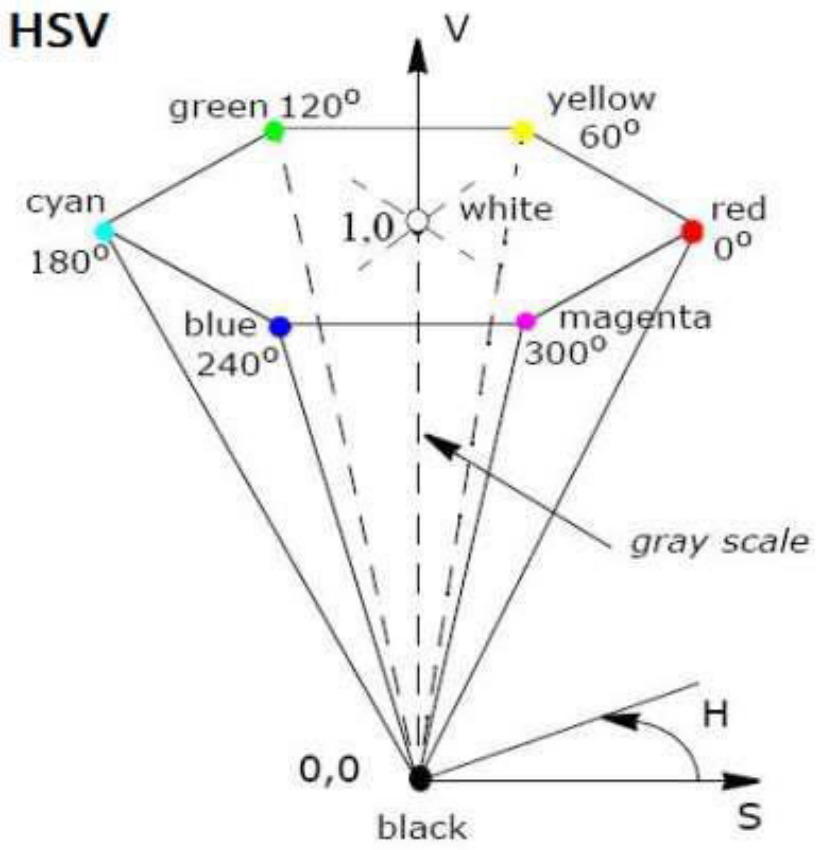
Hue, Saturation, Value or **HSV** is a color model that describes colors (**hue** or tint) in terms of their shade (**saturation**) and their brightness (**value**)., HSV is used today in **color pickers**, in **image editing software**, and less commonly in **image analysis** and **computer vision**. 'Value' is sometimes substituted with 'brightness' and then it is known as HSB. The HSV model was created by Alvy Ray Smith in 1978. HSV is also known as the hex-cone color model.

Hue

In HSV, hue represents color. In this model, hue is an angle from 0 degrees to 360 degrees.

Angle	**Color**
0-60	Red
60-120	Yellow
120-180	Green
180-240	Cyan
240-300	Blue
300-360	Magenta





Saturation

Saturation indicates the range of grey in the color space. It ranges from 0 to 100%. Sometimes the value is calculated from 0 to 1. When the value is '0,' the color is grey and when the value is '1,' the color is a primary color. A faded color is due to a lower saturation level, which means the color contains more grey.

Value

Value is the brightness of the color and varies with color saturation. It ranges from 0 to 100%. When the value is '0' the color space will be totally black. With the increase in the value, the color space brightness up and shows various colors.

Applications of HSV

The HSV color space is widely used to generate high quality computer graphics. In simple terms, it is used to select various different colors needed for a particular picture. An HSV color wheel is used to select the desired color. A user can select the particular color needed for the picture from the color wheel. It gives the color according to human perception.

HSV Representations

The HSV color wheel is used to pick the desired color. Hue is represented by the circle in the wheel. A separate triangle is used to represent saturation and value. The

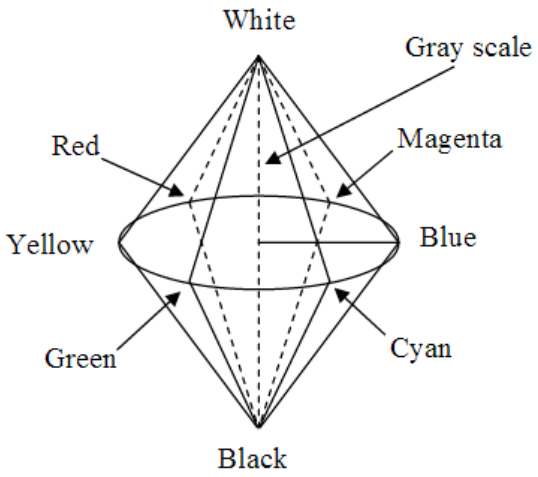
horizontal axis of the triangle indicates value and the vertical axis represents saturation. When you need a particular color for your picture, first you need to pick a color from the hue (the circular region), and then from the vertical angle of the triangle you can select the desired saturation. For brightness, you can select the desired value from the horizontal angle of the triangle.

Advantages of HSV

The HSV color space is quite similar to the way in which humans perceive color. The other models, except for HSL, define color in relation to the primary colors. The colors used in HSV can be clearly defined by human perception, which is not always the case with RGB or CMYK.

HLS COLOR MODEL

This model, Hue, Lightness, and Saturation, was popularized by Tektronix who used it to define the color effects on its monitors. It uses a double cone, as shown below:



The hues are specifies by angles, as they were for HSV, but in this model Blue is at 0°, Magenta is at 60°, Red is at 120°, Yellow is at 180°, Green is at 240°, and Cyan is at 300°. So the order on which the colors appear is the same as before, and complementary colors are still on opposite sides of the circle, separated by 180°, but the color sequence begins with blue instead of red. The angle is measured from above, as before, beginning at the line shown from medium gray to blue.

The hue definitions now lie on a circle, as compared to the hexagon that was used for HSV. This is much easier to deal with since full saturation of any hue will now have an S value of 1.0, as compared to, for example, the $\sqrt{3}/2$ that we had to use for the S value for orange using HSV.

Once again, gray scales appear on the center line of symmetry, with L= 0 at the bottom and L= 1 at the top. In this mode l the line is twice as long as in HSV.
Pure colors have an L value of 0.5. So, for example, pure orange is at an HLS triple of (150°, 0.5, 1.0). Overall HSV seems to be the preferred method for interactive selection of colors.

CIE XYZ Color Model

The XYZ color space is an international standard developed by the CIE (Commission Internationale de l'Eclairage). This model is based on three hypothetical primaries, XYZ,and all visible colors can be represented by using only positive values of X, Y, and Z. The CIE XYZ primaries are hypothetical because they do not correspond to any real light wavelengths. The Y primary is intentionally defined to match closely to luminance, while X and Z primaries give color information. The main advantage of the CIE XYZ space (and any color space based on it) is that this space is completely device-independent. Intel IPP functions use the following basic equations; to convert between gamma corrected R'G'B' and CIE XYZ models:

X = 0.412453*R' + 0.35758 *G' + 0.180423*B'

Y = 0.212671*R' + 0.71516 *G' + 0.072169*B'

Z = 0.019334*R' + 0.119193*G' + 0.950227*B'

The equations for X,Y,Z calculation are given on the assumption that R',G', and B' values are normalized to the range [0..1].

R' = 3.240479 * X - 1.53715 * Y - 0.498535 * Z

G' = -0.969256 * X + 1.875991 * Y + 0.041556 * Z

B' = 0.055648 * X - 0.204043 * Y + 1.057311 * Z

The equations for R',G', and B' calculation are given on the assumption that X,Y, and Z values are in the range [0..1].

YIQ Color Model

YIQ is the system used for US TV broadcast (PAL is the most common system used in other countries). The primary goals of the system were to provide a signal that could be directly displayed by black and white TVs, while also providing easy coding and decoding of RGB signals.

The conversions from RGB to YIQ and back are given by the matrices:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} .299 & .587 & .114 \\ .701 & -.587 & -.114 \\ -.299 & -.587 & .886 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

and

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.000 & 1.000 & .000 \\ 1.000 & -.509 & -.194 \\ 1.000 & .000 & 1.000 \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix}$$

where obviously the two matrices are inverses. The Y component, which is the same as the Y value in the CIE system, is the signal that is used directly by black and white TVs.

Y is said to convey the luminance information and is transmitted on a separate carrier signal from the chromaticity components, I and Q. This encoding is of far more importance for film & TV people than it is for computer graphics people.

Ray tracing methods

Ray tracing is a rendering technique that can produce incredibly realistic lighting effects. Essentially, an algorithm can trace the path of light, and then simulate the way that the light interacts with the virtual objects it ultimately hits in the computer-generated world

Ray tracing is a rendering technique that creates a graphical image based on virtual light and how that lighting source would interact with virtual objects along an image plane. The issue, however, is that ray tracing requires a significant amount of power to render the effects.

An evolution in ray casting rendering came in 1979 when Turner Whitted continued the ray casting process by introducing reflection, refraction, and shadows. A reflected ray continues on in the mirror-reflection direction from a shiny surface. The reflected color is determined by the intersection of the reflected rays with objects in the scene. A refracted ray is created similarly to reflect rays except its direction is into the object and can eventually exit the object. The shadow is computed by creating shadow rays which originates from the intersection to all lights. If the shadow ray intersects an object before it reaches the light, then that intersection point is shadowed from that particular light.

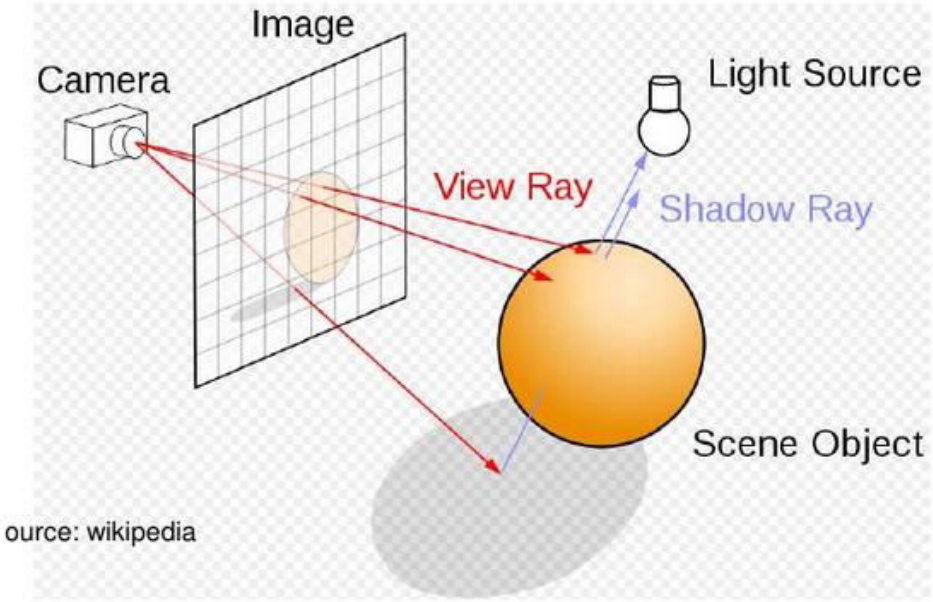
Advantage/Disadvantage The main advantage of ray tracing is its realistic rendering of reflections, refractions and shadows. Once the ray object intersection is coded, reflection, refractions, and shadows are able to be added very easily. In addition, anti-aliasing and depth of field effects are easily achieved using ray tracing. The most serious disadvantage is the heavy computational requirement of the ray tracing algorithm. Advanced lighting effects such as caustics are difficult to render using ray tracing.

Algorithm

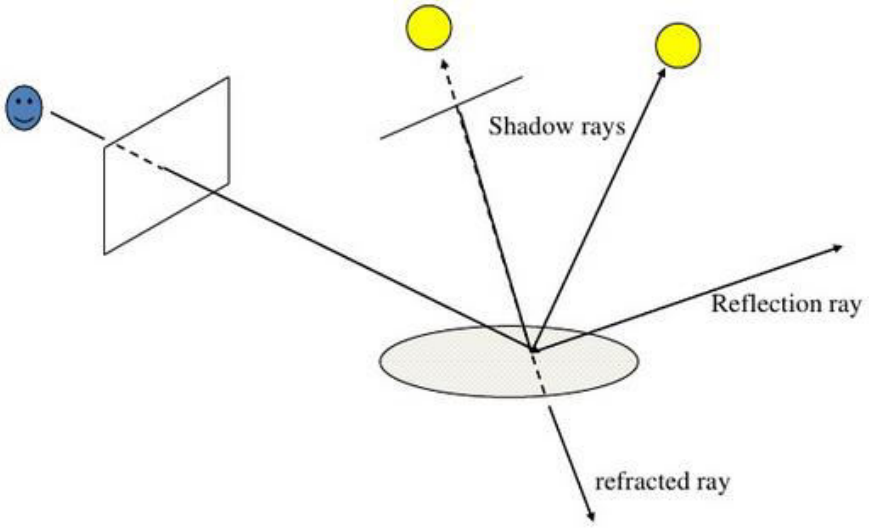
The general algorithm of ray tracing is performed by tracing a path from an imaginary eye through each pixel in a virtual screen, and calculating the color of the object visible through it. Each ray is tested for intersection with objects and once the nearest intersection has been identified, the algorithm estimates the incoming light at the point of intersection, examine the material properties of the object, and combine this information to calculate the final color of the pixel. The algorithm is divided into five sections: Camera Ray casting, Ray-object intersection, dealing with object transformation, lighting calculations, and recursive ray tracing.

Ray tracing is a technique for rendering three dimensional graphics with very complex light interactions. This means you can create pictures full of mirrors, transparent surfaces, and shadows, with stunning results.

Ray Tracing Model



Ray tracing



Ray tracing algorithm

- Builds the image pixel by pixel
- Cast additional rays from the hit point to determine the pixel color
  - Shoot rays toward each light. If they hit something, the object is shadowed from that light, otherwise use "standard model" for the light
  - Reflection rays for mirror surfaces, to see what should be reflected in the mirror
  - Refraction rays to see what can be seen through transparent objects
  - Sum all the contributions to get the pixel color

Recursive ray tracing

- When a reflected or refraction ray hits a surface, repeat the whole process from that point
  - Send more out shadow rays
  - Send out new reflected rays (if required)
  - Send out a new refracted ray (if required)
  - Generally, reduce the weight of each additional ray when computing the contributions to the surface color
  - Stop when the contribution from a ray is too small to notice or maximum recursion level has been reached

Common Image File Formats

There are numerous image file types out there so it can be hard to know which file type best suits your image needs. Some image types such a TIFF are great for printing while others, like JPG or PNG, are best for web graphics.

The list below outlines some of the more common file types and provides a brief description, how the file is best used, and any special attributes the file may have.

TIFF (.tif, .tiff)

TIFF or Tagged Image File Format are lossless images files meaning that they do not need to compress or lose any image quality or information (although there are options for compression), allowing for very high-quality images but also larger file sizes.

Compression: Lossless - no compression. Very high-quality images.

Best For: High quality prints, professional publications, archival copies

Special Attributes: Can save transparencies

Bitmap (.bmp)

BMP or Bitmap Image File is a format developed by Microsoft for Windows. There is no compression or information loss with BMP files which allow images to have very high quality, but also very large file sizes. Due to BMP being a proprietary format, it is generally recommended to use TIFF files.

Compression: None

Best For: High quality scans, archival copies

JPEG (.jpg, .jpeg)

JPEG, which stands for Joint Photographic Experts Groups is a "lossy" format meaning that the image is compressed to make a smaller file. The compression does create a loss in quality but this loss is generally not noticeable. JPEG files are very common on the Internet and JPEG is a popular format for digital cameras - making it ideal for web use and non-professional prints.

Compression: Lossy - some file information is compressed or lost

Best For: Web Images, Non-Professional Printing, E-Mail, PowerPoint

Special Attributes: Can choose amount of compression when saving in image editing programs like Adobe Photoshop or GIMP.

GIF (.gif)

GIF or Graphics Interchange Format files are widely used for web graphics, because they are limited to only 256 colors, can allow for transparency, and can be animated. GIF files are typically small is size and are very portable.

Compression: Lossless - compression without loss of quality

Best For: Web Images

Special Attributes: Can be Animated, Can Save Transparency

PNG (.png)

PNG or Portable Network Graphics files are a lossless image format originally designed to improve upon and replace the gif format. PNG files are able to handle up to 16 million colors, unlike the 256 colors supported by GIF.

Compression: Lossless - compression without loss of quality

Best For: Web Images

Special Attributes: Save Transparency

EPS (.eps)

An EPS or Encapsulated PostScript file is a common vector file type. EPS files can be opened in many illustration applications such as Adobe Illustrator or CorelDraw.

Compression: None - uses vector information

Best For: Vector artwork, illustrations

Special Attributes: Saves vector information

RAW Image Files (.raw, .cr2, .nef, .orf, .sr2, and more)

RAW images are images that are unprocessed that have been created by a camera or scanner. Many digital SLR cameras can shoot in RAW, whether it be a .raw, .cr2, or .nef. These RAW images are the equivalent of a digital negative, meaning that they hold a lot of image information, but still need to be processed in an editor such as Adobe Photoshop or Light room.

Compression: None

Best For: Photography

Special Attributes: Saves metadata, unprocessed, lots of information

[END OF UNIT - IV]

Unit–V : User Interface Design

Syllabus : Interactive Handling Models – Input And Output Handling In Window Systems.

Introduction

As hardware cost is plummeting, which is considered as the major bottleneck for the progress; now communication devices is more listened for better development. For that reason, techniques for developing high-quality user interfaces are moving to the forefront in computer science and are becoming the "last frontier" in providing computing to a wide variety of users—as other aspects of technology continue to improve, but the human users remain the same. Interest in the quality of user-computer interfaces is a recent part of the formal study of computers. The emphasis until the early 1980s was on optimizing two scarce hardware resources, computer time and memory. Program efficiency was the highest goal. With today's plummeting hardware costs and powerful graphics-oriented personal computing environments the focus turns to optimizing user efficiency rather than computer efficiency. Thus, although many of the ideas presented in this chapter require additional CPU cycles and memory space, the potential rewards in user productivity and satisfaction well outweigh the modest additional cost of these resources. The quality of the user interface often determines whether users enjoy or despise a system, whether the designers of the system are praised or damned, whether a system succeeds or fails in the market. Actually, a poor user interface such as in air traffic control or in nuclear power plant monitoring can lead to catastrophic consequences.

The desktop user-interface metaphor, with its windows, icons, and pull-down menus, all making heavy use of raster graphics, is popular because it is easy to learn and requires little typing skill. Most users of such systems are not computer programmers and have little sympathy for the old-style difficult-to-learn keyboard-oriented command-language interfaces that many programmers take for granted. The designer of an interactive graphics application must be sensitive to users' desire for easy-to-learn yet powerful interfaces. In this chapter, we discuss the three basic low-level elements of user interfaces: input devices, interaction techniques, and interaction tasks. **Interaction techniques are the primitive building blocks from which a user interface is crafted.**

We focus in this chapter on input devices—those pieces of hardware by which a user enters information into a computer system. Input devices for the earliest computers were switches and knobs, jumper wires placed in patch boards, and punched cards. These were followed by the teletype, the text-only forerunner of today's interactive terminals. The mouse and keyboard now predominate, but a wide variety of input devices can be used. **An interaction task is the entry of a unit of information by the user. Basic interaction tasks are *position*, *text*, *select*, and *quantify*.** The unit of information that is input in a position interaction task is of course a position; the text task yields a text string; the select task yields an object identification; and the quantify task yields a numeric value. A designer begins with the interaction tasks necessary for a particular application. For each such task, the designer chooses an appropriate interaction device and interaction technique. Many different *interaction techniques* can be used for a given interaction task, and there may be several different ways of using the same device to perform the same task. For instance, a selection task can be carried out by using a mouse to select items from a menu, using a keyboard to enter the name of the selection, pressing a function key, circling the desired command with the mouse, or even writing the name of the command with the mouse. Similarly, a single device can be used for different tasks: A mouse is often used for both positioning and selecting.

Interaction tasks are defined by *what* the user accomplishes, whereas logical input devices categorize *how* that task is accomplished by the application program and the graphics system. Interaction tasks are user-centered, whereas logical input devices are a programmer and graphics-system concept. By analogy with a natural language, single actions with input devices are similar to the individual letters of the alphabet from which words are formed. The sequence of input-device actions that makes up an interaction technique is analogous to the sequence of letters that makes up a word. A word is a unit of meaning; just as several interaction techniques can be used to carry out the same interaction task, so too words that are synonyms convey the same meaning. An interactive dialogue is made up of interaction-task sequences, just as a sentence is constructed from word sequences.

Concept of Positioning and Pointing

Most display terminals provide the user with an alphanumeric keyboard with which to type commands and enter data for the program. For some applications, however, the keyboard is inconvenient or inadequate. For example, the user may wish to indicate

one of a number of symbols on the screen, in order to erase the symbol. If each symbol is labeled, he can do so by typing the symbol's name; by pointing at the symbol, however, he may be able to erase more rapidly, and the extra clutter of labels can be avoided.

Another problem arises if the user has to add lines or symbols to the picture on the screen. Although he can identify an items's position by typing coordinates he can do so even better by pointing at the screen, particularly if what matters most is the items's position relative to the rest of the picture.

These two examples illustrate the two basic types of graphical interaction: pointing at items already on the screen and positioning new items. The need to interact in these ways has stimulated the developed of a number of different types of graphical input device, some of which are described in this chapter.

Ideally a graphical input device should lend itself both to pointing and to positioning. In reality there are no devices with this versatility. Most devices are much better at positioning than at pointing; one device, the light pen, is the exact opposite. Fortunately, however we can supplement the deficiencies of these devices by software and in this way produce hardware-software system that has both capabilities. Nevertheless the distinction between pointing and positioning capability is extremely important.

Another important distinction is between devices that can be used directly on the screen surface and devices that cannot. The latter might appear to be less useful, but this is far from true. Radar operators and air-traffic controllers have for years used devices like the joystick and the tracker ball neither of which can be pointed at the screen. The effectiveness of these input devices depends on the use of visual feedback: the x and y outputs of the device control the movement of a small cross, or cursor, displayed on the screen. The user of the device steers the cursor around the screen as if it were a toy boat on the surface of a pond. Although this operation sounds as if it requires a lot of skill, it is in fact very easy.

The use of visual feedback has an additional advantage: just as in any control system, it compensates for any lack of linearity in the device. A linear input device is one that faithfully increases or decreases the input coordinate value in exact proportion to the user's hand movement. If the device is being used to trace a graph or a map. Linearity is important. A cursor, however, can be controlled quite easily even if the device behaves in a fairly nonlinear fashion. For example, the device may be much less

sensitive near the left – hand region of its travel: a 1 – inch hand movement may change the x value by only 50 units, whereas the same movement elsewhere may change x by 60 units. The user will simply change his hand movement to compensate, often without even noticing the no linearity. This phenomenon has allowed simple, inexpensive devices like the mouse to be used very successfully for graphical input.

Interactive Graphic Devices

Various devices are available for data input on graphics workstations. Most systems have a keyboard and one or more additional devices specially designed for interactive input. These include a mouse, trackball, spaceball, joystick, digitizers, dials, and button boxes. Some other input devices used in particular applications are data gloves, touch panels, image scanners, and voice systems.

Keyboards

The well-known QWERTY keyboard has been with us for many years. It is ironic that this keyboard was originally designed to *slow down* typists, so that the typewriter hammers would not be so likely to jam. Studies have shown that the newer Dvorak keyboard , which places vowels and other high-frequency characters under the home positions of the fingers, is somewhat faster than is the QWERTY design. It has not been widely accepted. Alphabetically organized keyboards are sometimes used when many of the users are non typists. But more and more people are being exposed to QWERTY keyboards, and experiments have shown no advantage of alphabetic over QWERTY keyboards .In recent years, the chief force serving to displace the keyboard has been the shrinking size of computers, with laptops, notebooks, palmtops, and personal digital assistants. The typewriter keyboard is becoming the largest component of such pocket-sized devices, and often the main component standing in the way of reducing its overall size. The *chord keyboard* has five keys similar to piano keys, and is operated with one hand, by pressing one or more keys simultaneously to "play a chord." With five keys, 31 different chords can be played. Learning to use a chord keyboard (and other similar stenographer style keyboards) takes longer than learning the QWERTY keyboard, but skilled users can type quite rapidly, leaving the second hand free for other tasks. This increased training time means, however, that such keyboards are not suitable substitutes for general use of the standard alphanumeric keyboard. Again, as computers become smaller, the benefit of a

keyboard that allows touch typing with only five keys may come to outweigh the additional difficulty of learning the chords. Other keyboard-oriented considerations, involving not hardware but software design, are arranging for a user to enter frequently used punctuation or correction characters without needing simultaneously to press the control or shift keys, and assigning dangerous actions (such as delete) to keys that are distant from other frequently used keys.

Touch Panels

As the name implies, touch panels allow displayed objects or screen positions to be selected with the touch of a finger. A typical application of touch panels is for the selection of processing options that are represented with graphical icons. Other systems can be adapted for touch input by fitting a transparent device with a touch-sensing mechanism over the video monitor screen. Touch input can be recorded using optical, electrical, or acoustical methods.

Optical touch panels employ a line of infrared light-emitting diodes (LEDs) along one vertical edge and along one horizontal edge of the frame. The opposite vertical and horizontal edges contain light detectors. These detectors are used to record which beams are interrupted when the panel is touched. The two crossing beams that are interrupted identify the horizontal and vertical coordinates of the screen position selected. Positions can be selected with an accuracy of about inch. With closely spaced LEDs, it is possible to break two horizontal or two vertical beams simultaneously. In this case, an average position between the two interrupted beams is recorded. The LEDs operate at infrared frequencies, so that the light is not visible to a user. An electrical touch panel is constructed with two transparent plates separated by a small distance. One of the plates is coated with a conducting material, and the other plate is coated with a resistive material. When the outer plate is touched, it is forced into contact with the inner plate. This contact creates a voltage drop across the resistive plate that is converted to the coordinate values of the selected screen position.

In acoustical touch panels, high-frequency sound waves are generated in the horizontal and vertical directions across a glass plate. Touching the screen causes part of each wave to be reflected from the finger to the emitters. The screen position at the point of contact is calculated from a measurement of the time interval between the transmission of each wave and its reflection to the emitter.

Light pens

The pencil-shaped devices 's are used to select screen positions by detecting the light coming from point on the CRT screen. They are sensitive to the short burst of light emitted from the phosphor coating at the instant the electron beam strikes a particular point. Other light sources, such as the background light in the room, are usually not detected by a light pen. An activated light pen, pointed at a spot on the screen as the electron beam lights up that spot, generates an electrical pulse that causes the coordinate position of the electron beam to be recorded. As with cursor-positioning devices, recorded light-pen coordinates can be used to position an object or to select a processing option. Although light pens are still with us, they are not as popular as they once were since they have several disadvantages compared to other input devices that have been developed. For one, when a light pen is pointed at the screen, part of the screen image is obscured by the hand and pen. And prolonged use of the light pen can cause arm fatigue. Also, light pens require special implementation for some applications because they cannot detect positions within black areas. To be able to select positions in any screen area with a light pen, we must have some nonzero intensity assigned to each screen pixel. In addition, light pens sometime give false readings due to background lighting in a room.

Graphics Tablets

One type of digitizer is the graphics tablet (also referred to as a data tablet), which is used to input two-dimensional coordinates by activating a hand cursor or stylus at selected positions on a flat surface. A hand cursor contains cross hairs for sighting positions, while a stylus is a pencil-shaped device that is pointed at positions on the tablet. This allows an artist to produce different brush strokes with different pressures on the tablet surface. Tablet size varies from 12 by 12 inches for desktop models to 4 by 60 inches or larger for floor models. Graphics tablets provide a highly accurate method for selecting coordinate positions, with an accuracy that varies from about 0.2 mm on desktop models to about 0.05 mm or less on larger models. Many graphics tablets are constructed with a rectangular grid of wire embedded in the tablet surface. Electromagnetic pulses are generated in sequence along the wires, and an electric signal is induced in a wire coil in an activated stylus or hand cursor to record a tablet position. Depending on the technology, a their signal strength, coded pulses, or phase shifts can be used to determine the position on the tablet.

Joysticks

A joystick consists of a small, vertical lever (called the stick) mounted on a base that is used to steer the screen cursor around. Most joysticks select screen positions with actual stick movement; others respond to pressure on the stick. The distance that the stick is moved in any direction from its center position corresponds to screen-cursor movement in that direction. Potentiometers mounted at the base of the joystick measure the amount of movement, and springs return the stick to the center position when it is released. One or more buttons can be programmed to act as input switchs to signal certain actions once a screen position has been selected.

Mouse

A mouse is small hand-held box used to position the screen cursor. Wheels or rollers on the bottom of the mouse can be used to record the amount and direction of movement. Another method for detecting mouse motion is with an optical sensor. For these systems, the mouse is moved over a special mouse pad that has a grid of horizontal and vertical lines. The optical sensor detects movement across the lines in the grid.

Since a mouse can be picked up and put down at another position without change in cursor movement, it is used for making relative changes in the position of the screen cursor. One, two, or three buttons are usually included on the top of the mouse for signaling the execution of some operation, such as recording cursor position or invoking a function. Most general-purpose graphics systems now include a mouse and a keyboard as the major input devices.

Voice Systems

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

A dictionary is set up for a particular operator by having the operator speak the command words to be used into the system. Each word is spoken several times, and the system analyzes the word and establishes a frequency pattern for that word in the dictionary along with the corresponding function to be performed. Later, when a voice command is given, the system searches the dictionary for a frequency-pattern match. Voice input is typically spoken into a microphone mounted on a headset. The microphone is designed to minimize input of other background sounds. If a different

operator is to use the system, the dictionary must be reestablished with that operator's voice patterns. Voice systems have some advantage over other input devices, since the attention of the operator does not have to be switched from one device to another to enter a command.

Other Devices

Here we discuss some of the less common, and in some cases experimental, 2D interaction devices. Voice recognizers, which are useful because they free the user's hands for other uses, apply a pattern-recognition approach to the waveforms created when we speak a word. The waveform is typically separated into a number of different frequency bands, and the variation over time of the magnitude of the waveform. in each band forms the basis for the pattern matching. However, mistakes can occur in the pattern matching, so it is especially important that an application using a recognizer provide convenient correction capabilities. Voice recognizers differ in whether or not they must be trained to recognize the waveforms of a particular speaker, and whether they can recognize connected speech as opposed to single words or phrases. Speaker-independent recognizers have very limited vocabularies—typically, they include only the ten digits and 50 to 100 words. Some discrete word recognizers can recognize vocabularies of thousands of different words after appropriate training. But if the user has a cold, the recognizer must be retrained. The user of a discrete word recognizer must pause for a fraction of a second after each word to cue the system that a word end has occurred. The more difficult task of recognizing connected speech from a limited vocabulary can now be performed by off-the-shelf hardware and software, but with somewhat less accuracy. As the vocabulary becomes larger, however, artificial-intelligence techniques are needed to exploit the context and meaning of a sequence of sentences to remove ambiguity. A few systems with vocabularies of 20,000 or more words can recognize sentences such as "Write Mrs. Wright a letter right now!" Voice synthesizers create waveforms that approximate, with varying degrees of realism, spoken words. The simplest synthesizers use *phonemes*, the basic sound units that form words. This approach creates an artificial-sounding, inflection-free voice. More sophisticated phoneme-based systems add inflections. Other systems actually play back digitized spoken words or phrases. They sound realistic, but require more memory to store the digitized speech. Speech is best used to augment rather than to replace visual feedback, and is most effective when used sparingly. For instance, a training application could show a student a graphic animation of some

process, along with a voice narration describing what is being seen. See for additional guidelines for the effective application of speech recognition and generation in user-computer interfaces, and for an introduction to speech interfaces, and for speech recognition technology. The data tablet has been extended in several ways. Many years ago, Herot and Negroponte used an experimental pressure-sensitive stylus : High pressure and a slow drawing speed implied that the user was drawing a line with deliberation, in which case the line was recorded exactly as drawn; low pressure and fast speed implied that the line was being drawn quickly, in which case a straight line connecting the endpoints was recorded. Some commercially available tablets sense not only stylus pressure but orientation as well. The resulting 5 degrees of freedom reported by the tablet can be used in various creative ways. For example, Bleser, Sibert, and McGee implemented the GWPaint system to simulate various artist's tools, such as an italic pen, that are sensitive to pressure and orientation. An experimental touch tablet, developed by Buxton and colleagues, can sense multiple finger positions simultaneously, and can also sense the area covered at each point of contact The device is essentially a type of touch panel, but is used as a tablet on the work surface, not as a touch panel mounted over the screen. The device can be used in a rich variety of ways . Different finger pressures correlate with the area covered at a point of contact, and are used to signal user commands: a light pressure causes a cursor to appear and to track finger movement; increased pressure is used, like a button-push on a mouse or puck, to begin feedback such as dragging of an object; decreased pressure causes the dragging to stop.

Interactive Graphical Techniques

There are several techniques that are incorporated into graphics packages to aid the interactive construction of pictures. Various input options can be provided, so that coordinate information entered with locator and stroke devices can be adjusted or interpreted according to a selected option. For example, we can restrict all lines to be either horizontal or vertical. Input coordinates can establish the position or boundaries for objects to be drawn, or they can be used to rearrange previously displayed objects.

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. We interactively select coordinate positions with a pointing device, usually by positioning the screen cursor. Just how the object or text-string positioning is performed depends on the selected

options. With a text string, for example, the screen point could be taken as the center string position, or the start or end position of the string, or any of the other string-positioning options. For lines, straight line segments can be displayed between two selected screen positions:

As an aid in positioning objects, numeric values for selected positions can be echoed on the screen. Using the echoed coordinate values as a guide, we can make adjustments in the selected location to obtain accurate positioning.

Constraints

With some applications, certain types of prescribed orientations or object alignments are useful. A constraint is a rule for altering input-coordinate values to produce a specified orientation or alignment of the displayed coordinates. There are many kinds of constraint functions that can be specified, but the most common constraint is a horizontal and vertical alignment of straight lines. This type of constraint, shown in Figs. 6.1 and 6.2, is useful in forming network layouts. With this constraint, we can create horizontal and vertical lines without worrying a-bout precise specification of endpoint coordinates.

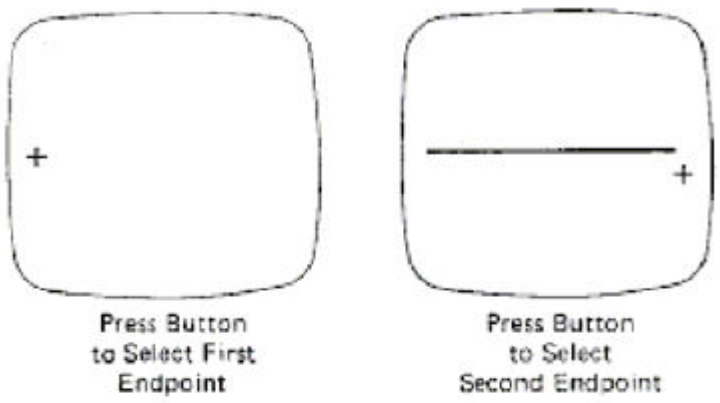


Figure 6.1: Horizontal line constraint

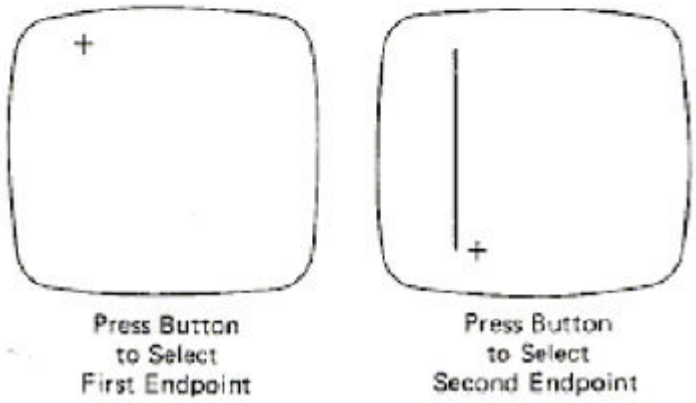


Figure 6.2: Vertical line constraint

A horizontal or vertical constraint is implemented by determining whether any two input coordinate endpoints are more nearly horizontal or more near vertical. If the difference in the y values of the two endpoints is smaller than the difference in x values, a horizontal line is displayed. Otherwise, a vertical line is drawn. Other kinds of constraints can be applied to input coordinates to produce a variety of alignments. Lines could be constrained to have a particular slant, such as 45°, and input coordinates could be constrained to lie along predefined paths, such as circular arcs.

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. Figure 6.3 illustrates line drawing with grid. Each

of the two cursor positions is shifted to the nearest grid intersection point, and the line is drawn between these grid points. Grids facilitate object constructions,

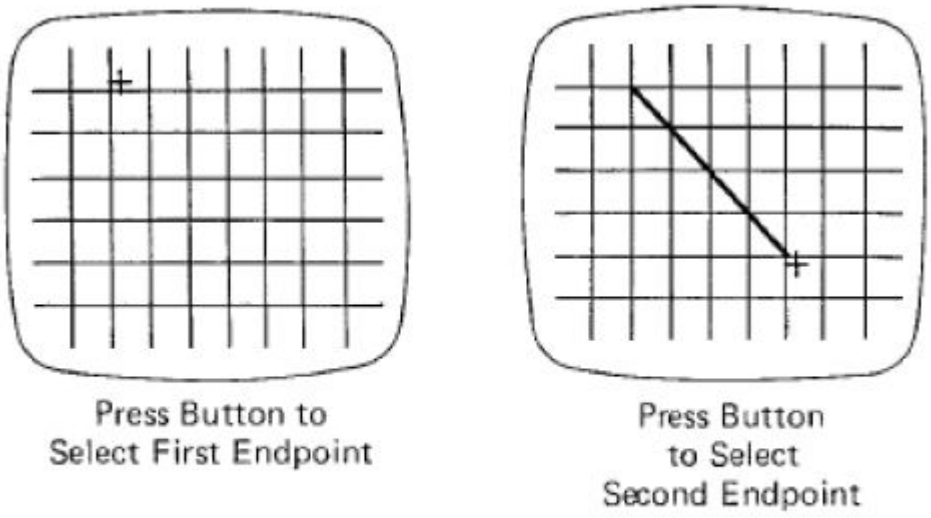


Figure 6.3: Line drawing using a grid

because a new line can be joined easily to a previously drawn line by selecting any position near the endpoint grid intersection of one end of the displayed line.

Figure 6.3: Line drawing using a grid

Spacing between grid lines is often an option that can be set by the user. Similarly, grids can be turned on and off, and it is sometimes possible to use partial grids and grids of different sizes in different screen areas.

Gravity Field

In the construction of figures, we sometimes need to connect lines at positions between endpoints. Since exact positioning of the screen cursor at the connecting point can be difficult, graphics packages can be designed to convert any input position near a line to a position on the line.

This conversion of input position is accomplished by creating a gravity field area around the line. Any selected position within the gravity field of a line is moved ("gravitated") to the nearest position on the line. A gravity field area around a line is illustrated with the shaded boundary shown in Fig. 6.4. Areas around the endpoints are enlarged to make it easier for us to connect lines at their endpoints. Selected positions in one of the circular areas of the gravity field are attracted to the endpoint in that area. The size of gravity fields is chosen large enough to aid positioning, but small enough to reduce chances of overlap with other lines. If many lines are displayed,

gravity areas can overlap, and it may be difficult to specify points correctly. Normally, the boundary for the gravity field is not displayed.

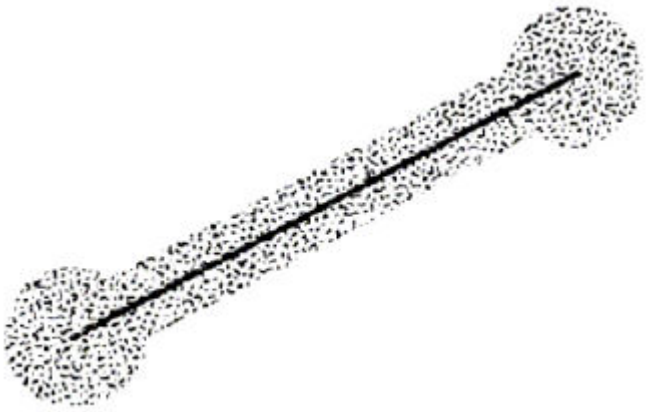


Figure 6.4: Gravity field around a line. Any selected point in the shaded area is shifted to a position on the line

Rubber-Band Methods

Straight lines can be constructed and positioned using rubber-band method which stretch out a line from a starting position as the screen cursor is move Figure 6.5 demonstrates the rubber-band method. We first select a screen position for one endpoint of the line. Then, as the cursor moves around, the line displayed from the start position to the current position of the cursor. When we finally select a second screen position, the other line endpoint is set.

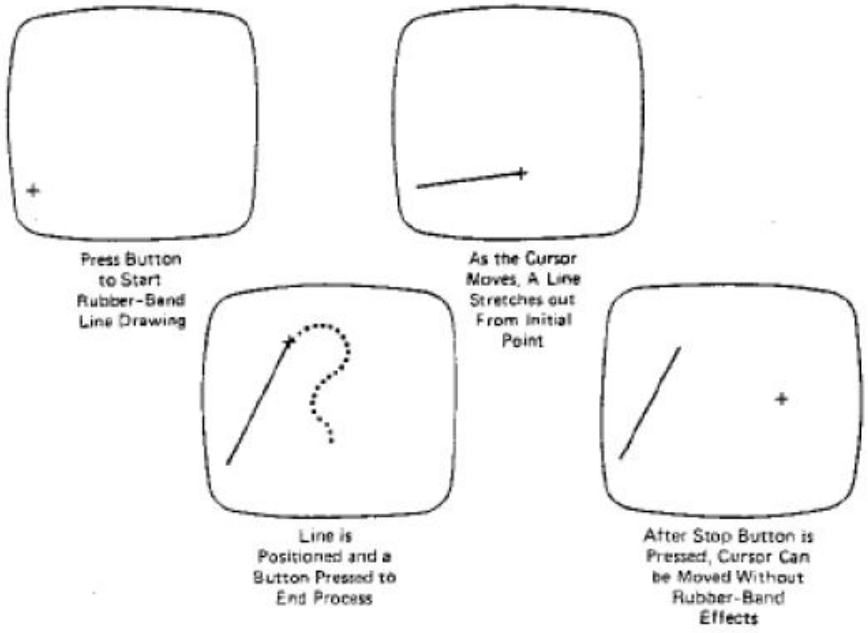


Figure 6.5: Rubber-band method for drawing and positioning a straight line segment

Rubber-band methods are used to construct and position other objects besides straight lines. Figure 6.6 demonstrates rubber-band construction of a rectangle, and Fig. 6.7 shows a rubber-band circle construction.

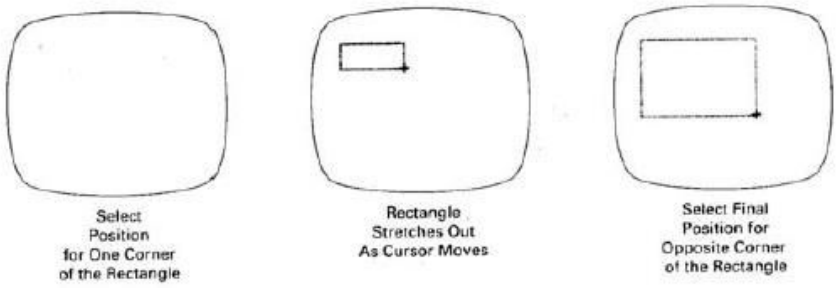


Figure 6.6: Rubber-band method for constructing a rectangle

Figure 6.7: Constructing a circle using a rubber-band method

Sketching

Options for sketching, drawing, and painting come in a variety of forms. Straight lines, polygons, and circles can be generated with methods discussed in the previous sections. Curve-drawing options can be provided using standard curve shapes, such as circular arcs and splines, or with freehand sketching procedures. Splines are interactively constructed by specifying a set of discrete screen points that give the general shape of the curve. Then the system fits the set of points with a polynomial curve. In freehand drawing, curves are generated by following the path of a stylus on a graphics tablet or the path of the screen cursor on a video monitor. Once a curve is displayed, the designer can alter the curve shape by adjusting the positions of selected points along the curve path.

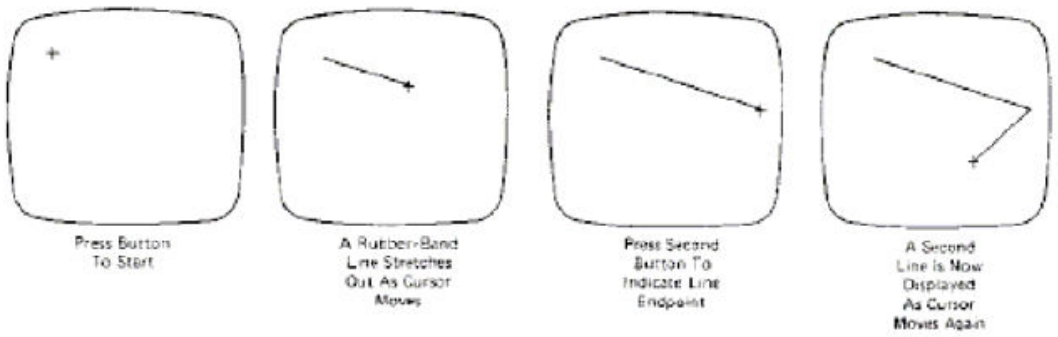


Figure 6.8 Uses rubber band methods to create objects consisting of connected line segments

Line widths, line styles, and other attribute options are also commonly found in painting and drawing packages. Various brush styles, brush patterns, color combinations, object shapes, and surface-texture patterns are also available on many systems, particularly those designed as artist's workstations. Some paint systems vary the line width and brush strokes according to the pressure of the artist's hand on the stylus.

Dragging

A technique that is often used in interactive picture construction is to move objects into position by dragging them with the screen cursor. We first select an object, then move the cursor in the direction we want the object to move, and the selected object follows the cursor path. Dragging objects to various positions in scene is useful in applications where we might want to explore different possibilities before selecting a final location.

Inking and Painting

If we sample the position of a graphical input device at regular intervals and display a dot at each sampled position, a trail will be displayed of the movement of the device. This technique, which closely simulates the effect of drawing on paper, is called inking. For many years the main use of inking has been in conjunction with on-line character-recognition programs. With the advent of high-quality raster displays the technique has found wider use for painting purposes.

Painting

A raster display incorporating a random-access frame buffer, can be treated as a painting surface for interactive purposes. As the user moves the cursor around, a trace of its path can be left on the screen. The user can build up freehand drawings of surprisingly good quality.

It is possible to provide a range of tools for painting on a raster display: these tools take the form of brushes that lay down trails of different thick nesses and colors. For example, instead of depositing a single dot at each sampled input position, the program can insert a group of dots so as to fill in a square or circle: the result will be a much thicker trace. On a "black-and-white display the user needs brushes that paint in both black and white, so that information can be both added and removed (Figure 6.9). When a color display is used for painting, a menu of different colors can be provided.

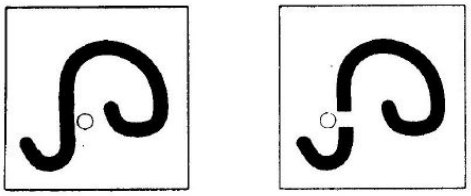


Figure 6.9: Erasing with a white brush

Summary

• An interaction technique is a way of using a physical input/output device to perform a generic interaction task in a human-computer dialogue. It represents an abstraction of some common class of interactive task, for example, choosing one of several objects shown on a display screen, so it is not bound to a single application.

• The basic interaction tasks for interactive graphics are positioning, selecting, entering text, and entering numeric quantities.

• Input functions available in a graphics package can be defined in three input modes. Request mode places input under the control of the application program. Sample mode allows the input devices and program to operate concurrently. Event mode allows input devices to initiate data entry and control processing of data. Once a mode has been chosen for a logical device class and the particular physical device to be used to enter this class of data, input functions the program are used to enter data values into the program. An application program can make simultaneous use of several physical input devices operating in different modes.

• Interactive picture-construction methods are commonly used in a variety applications, including design and painting packages. These methods provide users with the capability to position objects, to constrain figures to predefine orientations or alignments, to sketch figures, 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.

INPUT AND OUTPUT HANDLING IN WINDOWS SYSTEM :

Input Handling:

The GS is also responsible for handling input from the user, since it sits between the application and the devices.Again, think of this as a mapping/transformation - we're taking the physical input and mapping it to logical devices (remember that?) so that applications can make sense of it. How does the GS know what to do with an input? Applications usually have to explicitly express interest in something (e.g. a right mouse button click) - then the GS will pass it on to that app (assuming that other conditions are met - such as window was in focus, etc.)

This leads to a model of programming called event-driven programming (anyone have any experience with this?) -different from normal sequential program -structure: init, run event loop, quit -the GS basically renders the scene, waits for an event, passes it on, and re-renders the scene Any problems with this setup (e.g. for animation/VR - can't wait on events)

Output Handling:

Once the application has specified primitives and attributes, the GS is responsible for realizing those primitives in terms of output on the screen.Again, you can think of this as a mapping or transformation - from the more abstract primitive descriptions to actual pixel values. Called rendering of primitives. The idea of different spaces also comes up here - we're mapping from model/world space into screen space (different coordinate systems!) A final way to think of this job is providing the user a certain view into the model (a window on the internal world of the application) - that is often the point of computer graphics - visualization of something that otherwise is only present in bits.

Window management:

Window managers/window systems are usually separate entities from the graphics package. Their job is to manage the available screen space and mediate this space between multiple applications This is where logical output devices come in - each application only sees its canvas(es) and doesn't need to worry about everyone else. The window manager takes care of saving portions of windows that get covered up, dividing the space among windows, deciding the size and position of windows, etc. (GS-Graphics System)

[END OF UNIT- V]