Computer Graphics
Notes

EG3101CT

Year: III
Part: I

Total: 6 hours /week
Lecture: 3 hours/week
Tutorial: 1 hour/week
Practical: … hours/week
Lab: 2 hours/week

Course description:
This course deals with graphics hardware, two dimensional and three-dimensional graphics, fundamentals of animation techniques; graphical user interface design, web graphics design and graphics design packages.

Course objectives:
After completion of this course students will be able to:

1. Acquire the knowledge of computer graphics.
2. Familiarize with hardware involved in graphics.
3. Familiarize with the algorithms to generate two-dimensional and three-dimensional graphical objects and animations.

Course Contents:

Theory

Unit 1. Introduction **[3 Hrs.]**

1.1. History of Computer Graphics
1.2. Application of Computer Graphics
1.3. CAD and CAM

Unit 2. Graphics Hardware **[8 Hrs.]**

2.1. Input Hardware
  2.1.1. Keyboard, Mouse (mechanical & optical), Light pen, Touch panel (Optical, Sonic, and Electrical), Digitizers (Electrical, Sonic, Resistive), Scanner, Joystick
2.2. Output Hardware
  2.2.1. Monitors
  2.2.2. Monochromatic CRT Monitors
  2.2.3. Color CRT Monitors
  2.2.4. Flat Panel Display Monitors
2.3. Hardcopy Devices
  2.3.1. Plotters
  2.3.2. Printers
2.4. Raster and Vector Display Architectures, Principles and Characteristics

Unit 3. Two Dimensional Algorithms and Transformations **[10 Hrs.]**

3.1. Mathematical Line Drawing Concept
3.2. Line Drawing Algorithms
  3.2.1. Digital Differential Analyzer (DDA)
  3.2.2. Bresenham's Line Drawing Algorithm

3.3. Mid-point Circle Drawing
3.4. Mid-point Ellipse Drawing Algorithm
3.5. Review of Matrix Operations – Addition and Multiplication
3.6. Two-dimensional Transformations
  3.6.1. Translation
  3.6.2. Scaling
  3.6.3. Rotation
  3.6.4. Reflection
  3.6.5. Shearing
3.7. Two-Dimensional Viewing Pipeline

Unit 4. Three-Dimensional Graphics **[16 Hrs.]**

4.1. Three-dimensions transformations
  4.1.1. Translation
  4.1.2. Scaling
  4.1.3. Rotation
  4.1.4. Reflection
  4.1.5. Shearing
4.2. Three-dimensional Viewing Pipeline
4.3. Three-dimensions Projections
  4.3.1. Concept of Projection
  4.3.2. Projection of 3D Objects onto 2D Display Devices
  4.3.3. Three-dimensional Projection Methods
    4.3.3.1. Parallel Projection Method
    4.3.3.2. Perspective Projection Method
4.4. Three-dimensional Object Representations
  4.4.1. Polygon Surfaces
  4.4.2. Polygon Tables
4.5. Introduction to Hidden Line and Hidden Surface Removal Techniques
  4.5.1. Object Space Method
  4.5.2. Image Space Method
4.6. Introduction to Illumination/ Lighting Models
  4.6.1. Ambient Model
  4.6.2. Diffuse Model
  4.6.3. Specular Model
4.7. Introduction to Shading/ Surface Rendering Models
  4.7.1. Constant Shading Model
  4.7.2. Gouraud Shading Model
  4.7.3. Phong Shading Model

Unit 5. Web Graphics Designs and Graphics DesignPackages **[5 Hrs.]**

5.1. Introduction to graphics file formats
5.2. Principles of web graphics design – browser safe colors, size, resolurion, background, anti-aliasing
5.3. Type, purposes and features of graphics packages
5.4. Examples of graphics packages and libraries

Unit 6. Virtual Reality **[3 Hrs.]**

6.1. Introduction

6.2. Types of Virtual Reality

- 6.2.1. Non-immersive Virtual Reality
- 6.2.2. Semi-immersive Virtual Reality
- 6.2.3. Fully-immersive Virtual Reality
- 6.2.4. Augmented Virtual Reality
- 6.2.5. Collaborative Virtual Reality

6.3. Applications of Virtual Reality

Practical: **[30 Hrs.]**

As a part of the laboratory exercise, the students should implement all the algorithms studied in different chapters. At the end, students are required to integrate the codes they have written in earlier practical sessions to create a small project.

The lab contains few sessions dedicated to introduce the students to some of the popular professional graphics packages and CAD packages and explore their features. The course/lab instructor recommends packages to use.

Some algorithm implementation sessions may include:

1. Implementation of Digital Differential Analyzer (DDA), a line Drawing Algorithm.
2. Implementation of Bresenham's Line Drawing Algorithm.
3. Implementation of mid-point Circle Drawing Algorithm.
4. Implementation of mid -point Ellipse Drawing Algorithm.
5. Implementation of basic 2D transformation.
6. Implementation of basic 3D transformation.
7. Implementation of basic projections.

Final written exam evaluation scheme			
Unit	**Title**	**Hours**	**Marks Distribution***
1	Introduction	3	6
2	Graphics Hardware	8	15
3	Two Dimensional Algorithms and Transformations	10	20
4	Three-Dimensional Graphics	16	25
5	Web Graphics Designs and Graphics Design Packages	5	8
6	Virtual Reality	3	6
	Total	**45**	**80**

* There may be minor deviation in marks distribution.

Unit:1 - Introduction:

Computer graphics is an art of drawing pictures on computer screens with the help of programming. It involves computations, creation, and manipulation of data. In other words, we can say that computer graphics is a rendering tool for the generation and manipulation of images. Computer Graphics is used where a set of images needs to be manipulated or the creation of the image in the form of pixels and is drawn on the computer.

1.1 History of Computer Graphics

1951- whirlwind computer developed in MIT had computer driven CRT displays for output (round screen)
1962- sketchpad, by Ivan Sutherland (allowed users to interact with a computer using a light pen to draw on the screen.)
1964- CAD and CAM-shows graphical interaction and graphical activities
1968- flight simulators
1971-Gouraud Shading- rendering method (to simulate the differing shades of color on a smooth surface)
1974-1977 : Phong Shading (used for shading of surfaces to create highlights and reflections.)
1982- Ray Tracing (introduced the concept of ray tracing for rendering photorealistic images)
1993-Open GL- Open Graphics Library (cross-platform graphics API that is commonly used for rendering 2D and 3D vector graphics.)
1995- Microsoft, game playing API

1.2 Application of Computer Graphics:

1. Simulation and Training: Computer graphics are used in flight simulators, medical simulations, military training simulations, and virtual training environments to provide realistic scenarios for training purposes.

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

2. Design and Visualization: Computer graphics are used in architectural design, interior design, product design, and industrial design to create realistic 3D models and visualizations.

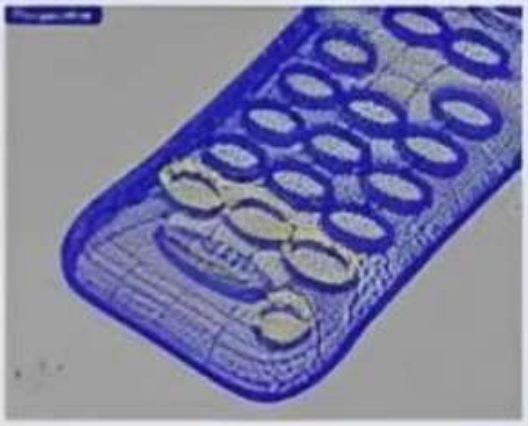
3. Data Visualization: Computer graphics are used to visualize complex data sets in a meaningful and interactive way, making it easier for users to analyze and interpret the data.

4. Virtual Reality and Augmented Reality: Computer graphics play a crucial role in creating immersive virtual reality (VR) and augmented reality (AR) experiences for various applications, including gaming, training, education, and simulations.

5. Advertising and Marketing: Computer graphics are used in advertising campaigns, product visualizations, and marketing materials to create compelling visual content that attracts and engages customers.

6. Medical Imaging: Computer graphics are used in medical imaging technologies such as MRI, CT scans, and ultrasound to create detailed 3D visualizations of the human body for diagnosis, treatment planning, and medical research.

1.3 CAD and CAM



CAD (Computer Aided Design)	CAM (Computer Aided Manufacture)
Represent part of geometry in our computer	Convert the geometry to machine tool or tool-path command

CAD (Computer Aided Design)

Computer Aided Design (CAD) is the use of computers for designing models of physical products which means computers are used to aid in creating the design, modifying the design, and analyzing the designing activities. Computer Aided Design is also known as Computer Aided Drafting. The purpose of CAD is to make 2D technical drawings and 3D models. So in Simple, we can say CAD represents your part geometry to the computer. Computer Aided Design (CAD) software is mostly used by an engineer.

Examples of CAD software include AutoCAD, Autodesk Inventor,

Computer + Designed Software = CAD

Features of CAD Software:

- 2D and 3D Modeling: CAD software allows designers to create both 2D and 3D models of their designs.

- Visualization: CAD software allows designers to view and analyze their designs from different angles and perspectives.
- Simulation: CAD software allows designers to simulate how their designs will perform in the real world and make changes accordingly.
- Collaboration: CAD software allows multiple users to work on the same design simultaneously and share their progress with each other.

CAM (Computer Aided Manufacturing)

Computer Aided Manufacturing (CAM) is the use of computer software to control machine tools in the manufacturing of modules. CAM transforms engineering designs into end products. CAM is different than conventional manufacturing as it implements automation in the manufacturing process. Computer Aided Manufacturing is also known as Computer-Aided Modeling or Machining. The purpose of CAM is to use 3D models to design machining processes. So in Simple, we can say CAM converts the geometry to the machine tool. So, without Computer-Aided Design (CAD), Computer-Aided Manufacturing (CAM) has no meaning. Computer Aided Manufacturing (CAM) software is mostly used by a trained machinist.

Examples of CAM software include Work NC, SolidCAM,

Manufacturing Tools + Computer = CAM

Features of CAM Software

- Toolpath Generation: CAM software generates toolpaths that machines can follow to create the desired product.
- Machine Control: CAM software controls the machines used in the manufacturing process, ensuring that they operate correctly and safely.
- Optimization: CAM software optimizes the manufacturing process, reducing waste and improving efficiency.
- Integration: CAM software can be integrated with other software systems, such as CAD software, to streamline the manufacturing process.

Difference between CAM and CAD:

CAD	CAM
It stands for Computer Aided Design.	It stands for Computer Aided Manufacturing.
It is the process of using computers to designing models of physical products.	It is a software that helps control machine tools in the process of manufacturing modules.
They help create design, modify the design and analyze the activity.	It helps transform the engineering designs into fully grown end products.
It is generally used by engineers.	It is generally used by a trained machinist.
It is used to represent the geometric part of the computer. It is an independent process.	CAM depends on CAD.
Examples of CAD software include AutoCAD, Autodesk Inventor,	Examples of CAM software include Work NC, Solid CAM.

Unit:2 – Graphics Hardware:

Graphics hardware is computer hardware that creates and shows computer graphics. It works with graphics software, such as a graphics card (video card), graphics processing unit (GPU), or display devices.

2.1 Input Device

An input device is any hardware component or peripheral device that allows users to input data, commands, or information into a computer or electronic system. These devices enable interaction between the user and the computer, allowing users to communicate with and control the system.

2.1.1 Various Input Devices

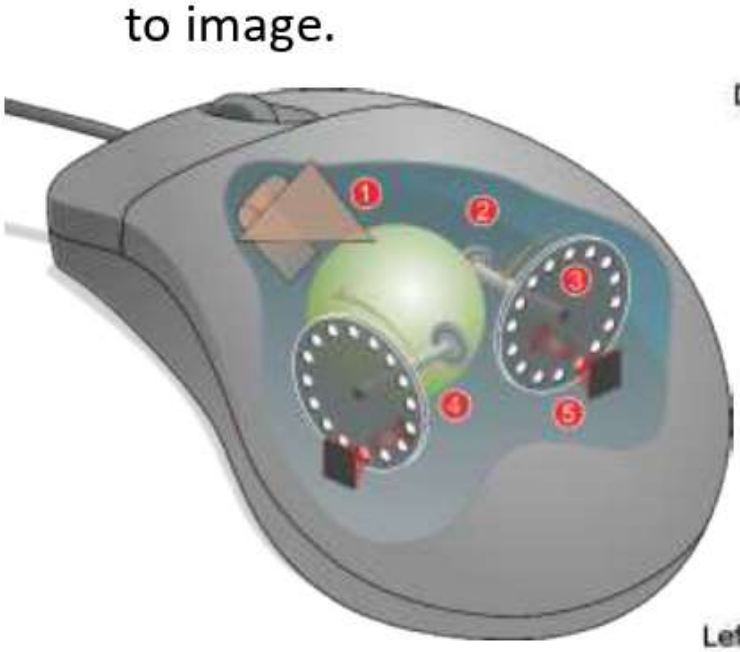
Keyboard: A keyboard is one of the primary input devices that allows users to input text into a computer or any other electronic machinery It consists of multiple buttons, which create numbers, symbols, and letters, and special keys like the Windows and Alt key. The design of the keyboard comes from the typewriter keyboards.

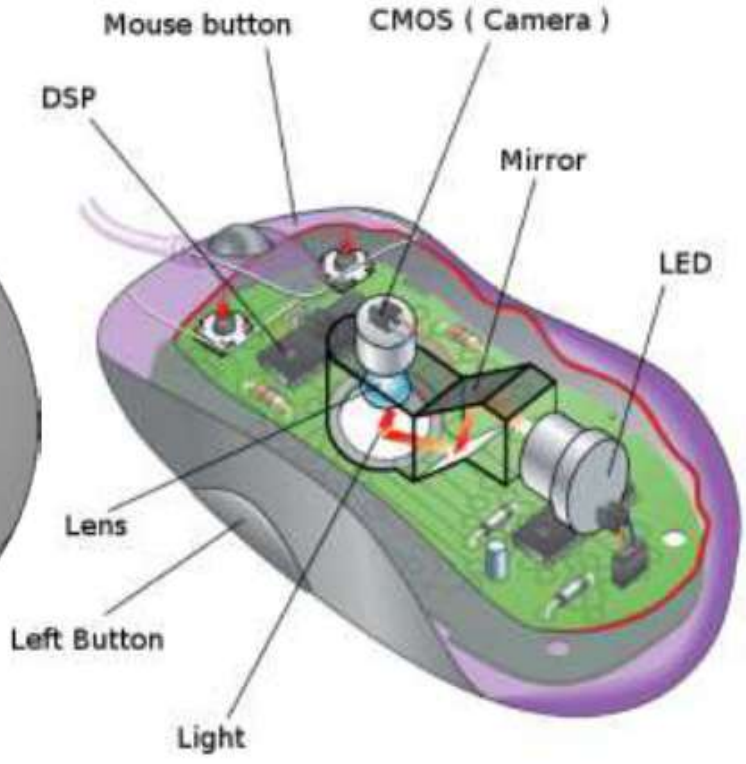
Mouse: A mouse is a small hardware input device used by hand. It controls the movement of the cursor on the computer screen and allows users to move and select folders, text, files, and icons on a computer.

Types:

a. **Mechanical Mouse:** A mechanical mouse is a computer mouse that contains a metal or rubber ball on its underside. When the ball is rolled in any direction, sensors inside the mouse detect this motion and move the on-screen mouse pointer in the same direction.

b. **Optical Mouse:** Optical mouse is a computer pointing device that uses a light-emitting diode (LED), optoelectronic sensor and digital signal processor (DSP) to detect changes in reflected light from image to image.





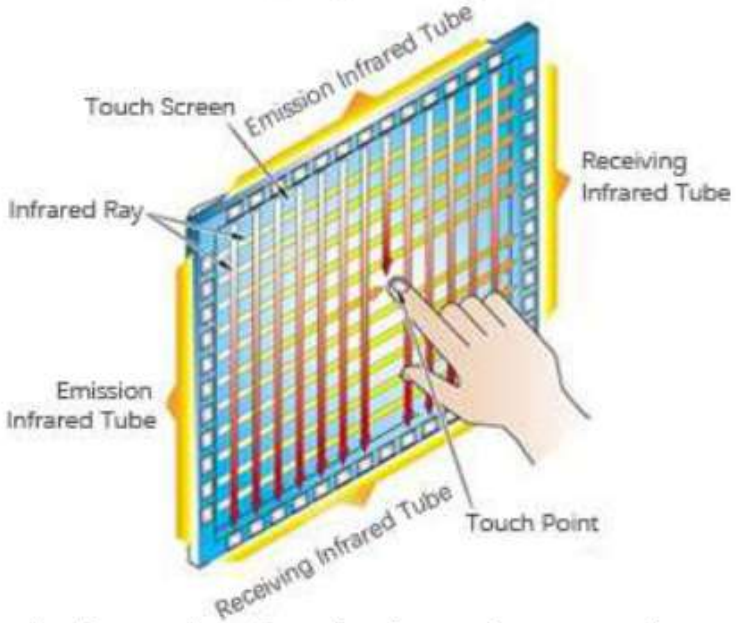
Light pen: It is a pencil shaped device to determine the coordinates of a point on the screen where it is activated such as pressing the button. It works by sensing the sudden small change in brightness of a point on the screen when the electron gun refreshes that spot. The light pen works well with CRT monitors.

Touch Panel: The touch panel allows the user to point at the screen directly with a finger to move the cursor around the screen or to select the icons. When a user touches the surface, the system records the change in the electrical current that flows through the display.

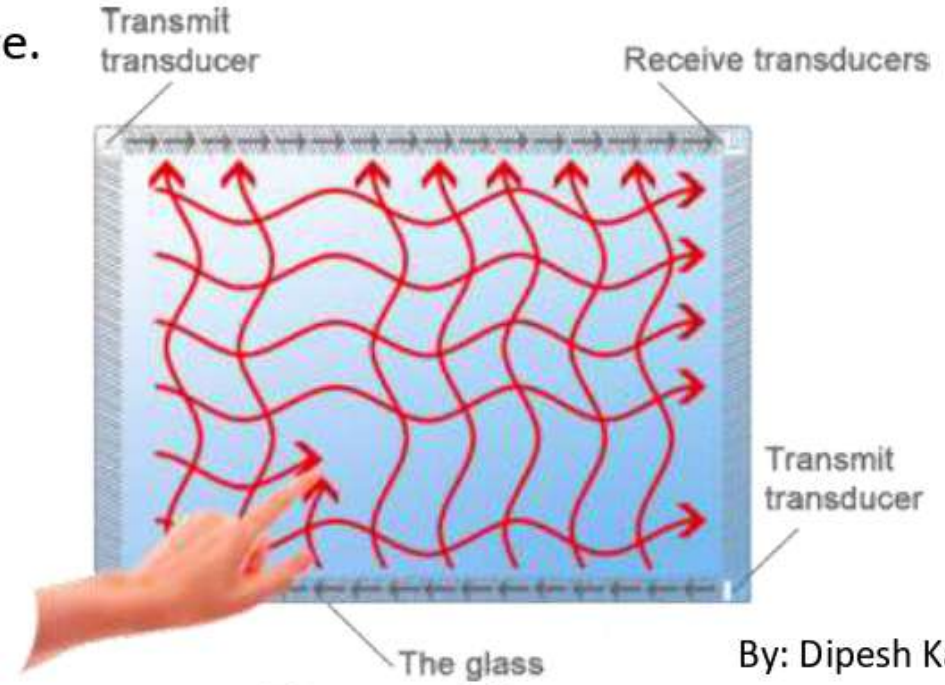
Types:

a. **Optical Touch Panel:** Optical touch panels use infrared light beams to detect touch input. These panels consist of an array of infrared LEDs (Light Emitting Diodes) along one or more edges of the display and infrared sensors along the opposite edges. When a user touches the screen, it interrupts the

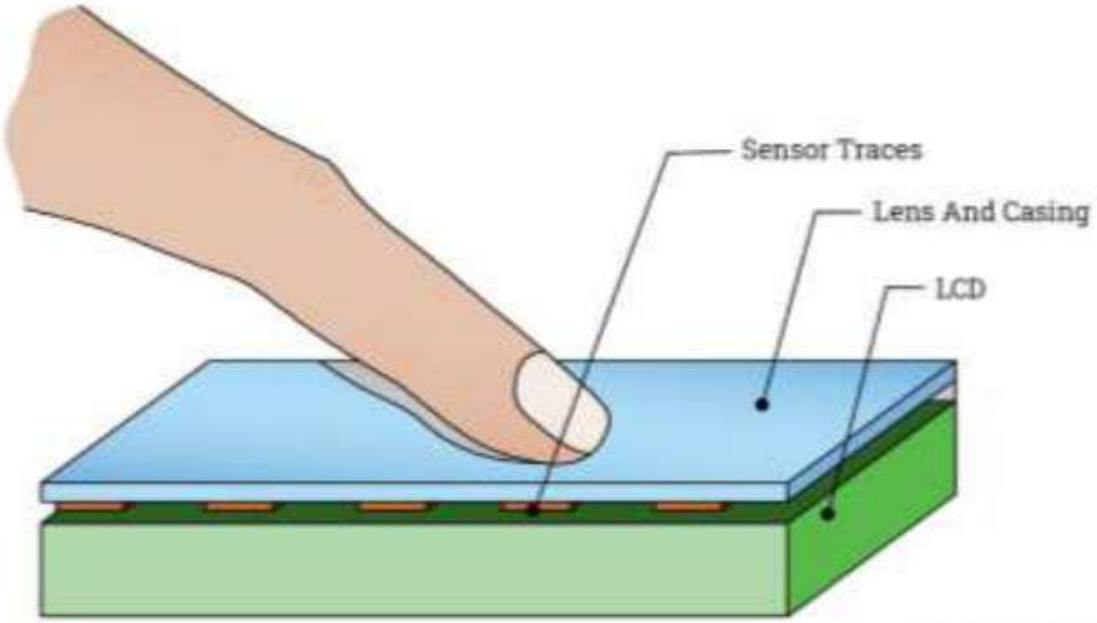
infrared light beams, and the sensors detect the interruption, determining the position of the touch.



b. **Sonic Touch Panels**: Sonic touch panels, or acoustic touch panels, use ultrasonic waves and transducers placed at the display's edges. When users touch the screen, it emits ultrasonic waves that travel through the panel. Transducers detect changes in these waves, pinpointing the touch location. These panels are resistant to environmental factors like dirt and moisture.



c. **Electrical Touch Panels:** Electrical touch panels, also called capacitive touch panels, use changes in capacitance to sense touch. They feature a layer of conductive material, like indium tin oxide (ITO), on a glass substrate. When users touch the screen, it alters the capacitance at that point, detected by sensors at the panel's corners. These panels are responsive and support multitouch.



Digitizer:

A digitizer in computer graphics is a device that converts analog information (such as drawings or sketches) into digital format that can be manipulated by a computer. This is commonly used in graphic design and animation to input hand-drawn images or sketches into digital software for editing and manipulation.

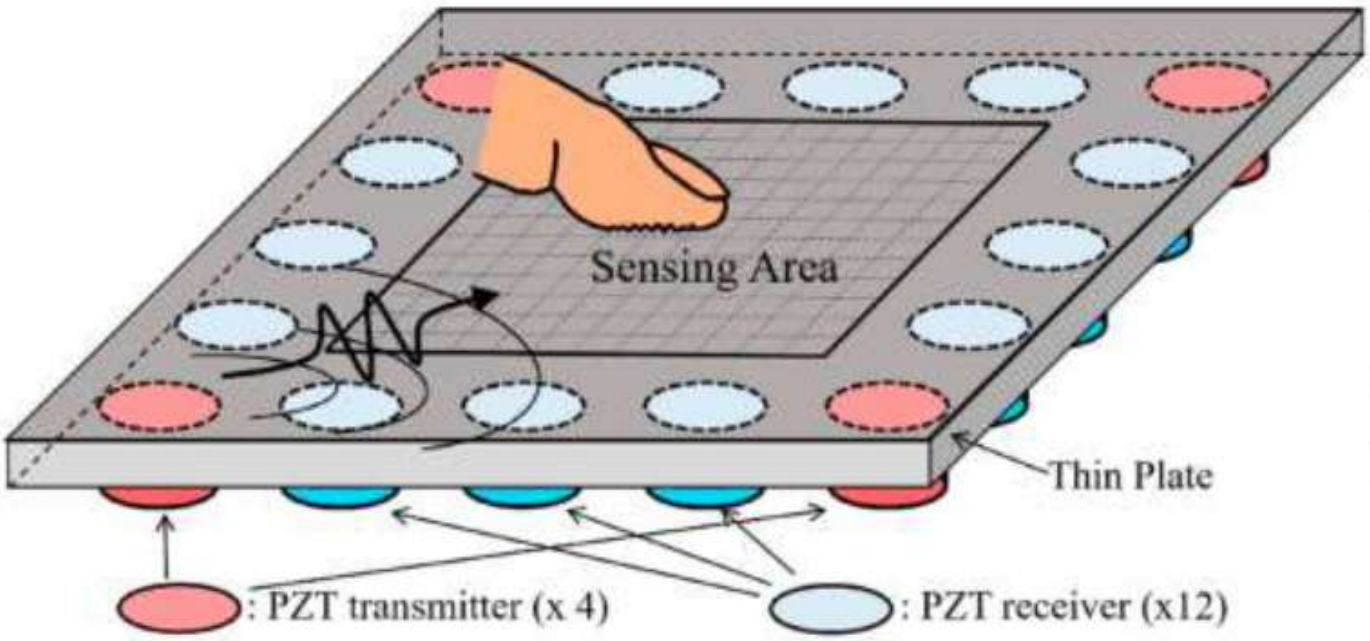
Types:

a. **Electrical Digitizer:**

An electrical digitizer typically works by sensing changes in electrical currents or voltages caused by the user's interactions with a stylus or touch-sensitive surface. Example: A graphics tablet, such as the Wacom Intuos, uses an electrical digitizer. When you draw on the tablet surface with a stylus, the tablet detects the position and pressure of the stylus and translates it into digital data that can be interpreted by graphic software.

b. Sonic Digitizer:

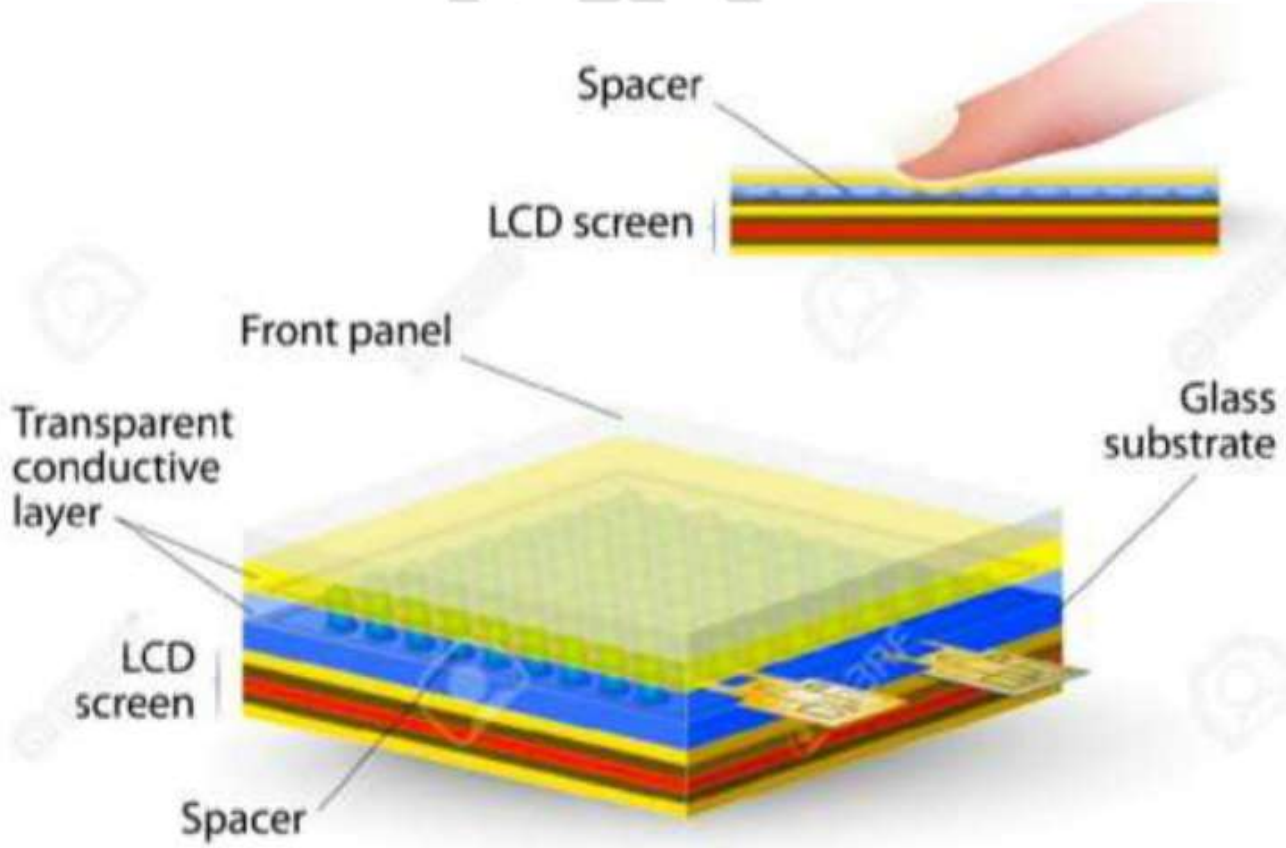
A sonic digitizer uses sound waves to determine the position of an object or stylus. It measures the time it takes for sound waves to travel from a transmitter to a receiver, which allows it to calculate the position accurately. Example: Ultrasound-based touchscreens use a sonic digitizer. When you touch the screen, it emits ultrasonic waves, and sensors detect the time it takes for the waves to bounce back. This information is used to determine the position of your touch.



c. Resistive Digitizer:

A resistive digitizer consists of multiple layers with a resistive coating. When pressure is applied, the layers make contact at the point of touch, creating a voltage drop that can be measured to determine the position. Example: Resistive touchscreens, commonly found in older smartphones or certain industrial applications,

use this technology. When you press on the screen, the layers make contact at that point, and the device detects the change in resistance to determine the touch location.



Scanner:

A scanner is a device used to convert physical documents or images into digital format. It typically works by capturing an image of the document using either a charge-coupled device (CCD) or a contact image sensor (CIS). The document is placed on a glass surface, and a light source illuminates it. The reflected light is then captured by the CCD or CIS, which converts it into digital data. This data is then processed and stored on a computer for viewing, editing, or printing.

Scanners can be flatbed (where the document lies flat on a glass surface) or sheet-fed (where the document is fed through the scanner).

Joystick:

A joystick is a hand-held input device used to control the movement or actions of a digital object on a computer screen. It typically consists of a handle or a lever that can be tilted or pushed in different directions to provide input to a computer or a gaming console. There are various types of joysticks available, including flight sticks, and gamepad-style joysticks. Flight sticks are specifically designed for flight simulators, and gamepad-style joysticks resemble traditional game controllers with added joystick functionality.

2.2 Output Hardware

Output hardware, also known as an output device, is a piece of computer hardware that receives data from a computer and converts it into another form. Output devices convert data from machine language to a human-understandable language. This can include text, graphics, audio, or video. For example, a monitor, printer, speaker, or projector are all output devices.

2.2.1 Monitors

Monitors, commonly called as Visual Display Unit (VDU), are the main output device of a computer. It forms images from tiny dots, called pixels that are arranged in a rectangular form. The sharpness of the image depends upon the number of pixels. This allows you to see the images produced by the computer. The quality of the graphics depends on the size and the resolution of the monitor.

2.2.2 Monochromatic CRT Monitors:

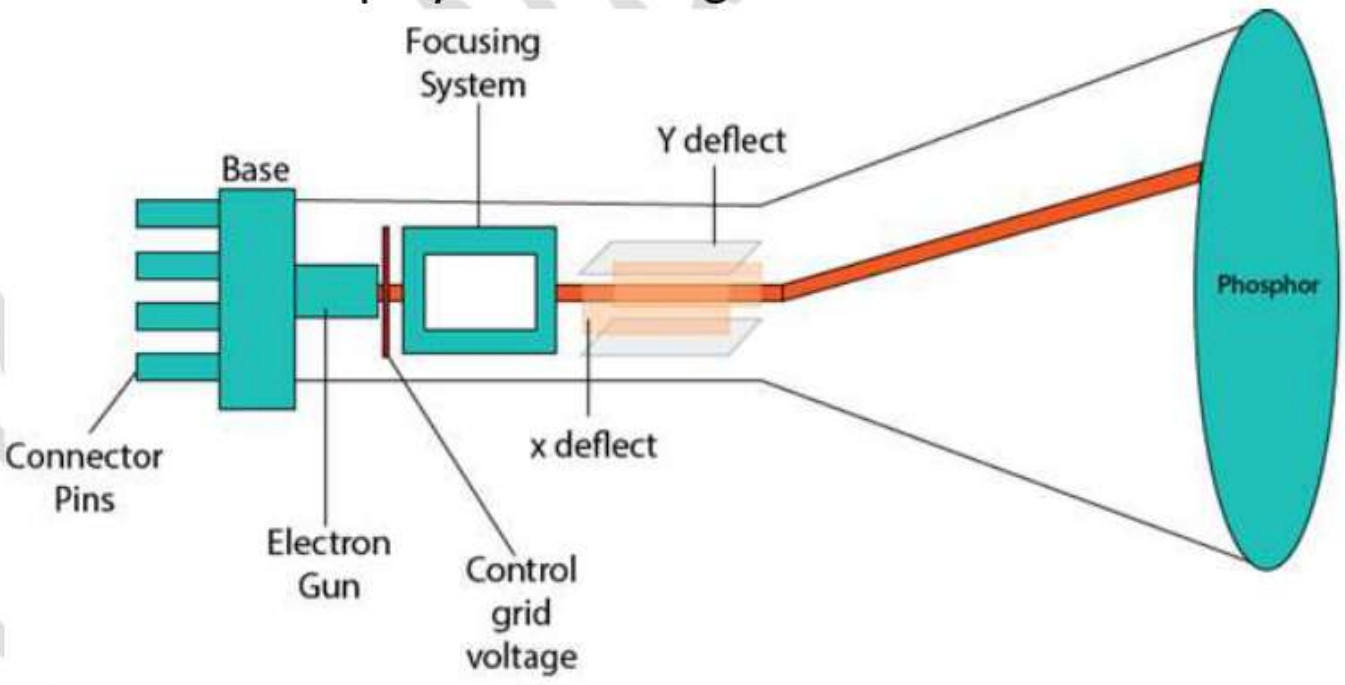
A type of display that can show graphics like images, texts, etc. only in gray shades and not in full color is referred to as a monochrome display. Monochrome displays are older visual displays, and they were used in older devices like CRT TVs, computer monitors, digital watches, calculators, etc. Sometimes, monochrome displays are also called as black and white displays. This is because they display graphics in black and white colors. A monochrome display consists of a grid of pixels. Each of these pixels is turned on and off to display different shades of gray color.

2.2.3 Color CRT Monitor

A color CRT (Cathode Ray Tube) monitor is a type of display device that was commonly used in older computer monitors and television sets. It uses a cathode ray tube to display images by directing electron beams onto a phosphorescent screen. The screen is coated with phosphor dots that emit light when struck by the electron beams.

p

In a color CRT monitor, the screen is divided into thousands or millions of tiny pixels, each capable of emitting different colors of light. By controlling the intensity of the electron beams and the arrangement of the phosphor dots, the monitor can display a wide range of colors and shades.



2.2.4 Flat Panel Display Monitors

A flat-panel display monitor is a type of display device that uses flat-panel technology to produce images. Unlike traditional CRT monitors, which use cathode ray tubes, flat-panel displays are thin and lightweight, making them well-suited for modern computer monitors, televisions, and mobile devices.

There are several types of flat-panel display technologies:

1. **LCD (Liquid Crystal Display):** LCD monitors use a layer of liquid crystal molecules sandwiched between two transparent electrodes. When an electric current is applied, the crystals twist to control the amount of light passing through them, creating images. LCD monitors are popular due to their thin profile, low power consumption, and relatively low cost.

2. **LED (Light Emitting Diode):** LED monitors are a type of LCD monitor that uses light-emitting diodes (LEDs) to provide backlighting. Compared to traditional cold cathode fluorescent lamps (CCFLs) used in older LCD monitors, LEDs offer better energy efficiency, improved brightness, and longer lifespans.

3. **OLED (Organic Light Emitting Diode):** OLED monitors use organic compounds that emit light when an electric current is applied. Unlike LCD monitors, which require a backlight, each pixel in an OLED display emits its own light, allowing for deeper blacks and higher contrast ratios. OLED monitors are known for their vibrant colors and high-quality image reproduction.

2.3 Hardcopy devices

Hardcopy devices are types of output devices that produce physical copies of digital documents or images on paper or other tangible media. These devices are commonly used to create permanent or easily shareable copies of electronic information. Examples of hardcopy devices include printers, photocopiers, and plotters.

2.3.1 Plotter

A plotter is basically a type of printer technology. It is a computer output device which is used to create high-quality images, large-formats images and diagrams and drawings on flat surfaces or papers.

There are various types of a plotter, which are as follows:

Drum Plotters: Drum plotters, as name suggests, they use a rotating drum to move the paper left and right while one or more fixed pens write up and down. These plotters are capable of producing large-format images and were popular in the past for applications like map making and technical drawings.

Drum plotter

Flatbed plotters: A flatbed plotter works with paper that is put on a flat surface in a stationary position. In this plotter, the writing pen moves in both the x and y axes. Flatbed plotter pens are available in a number of sizes and colors. It works by moving a pen over paper. The paper's size is governed by the size of the flat surface on which it is placed. Larger flatbed plotters have the ability to print up to 60-inches in length on the paper.

Flatbed plotter

Cutting Plotter: The cutting plotter is a large-scale cutting machine that uses blades instead of pens to cut the design. It creates mylar or vinyl lettering and graphics that are pre-cut. The plotter's flat surface is used to place the to-be-cut paper. The plotter receives a command from the computer, and the knife executes it to cut the media to the appropriate dimensions.

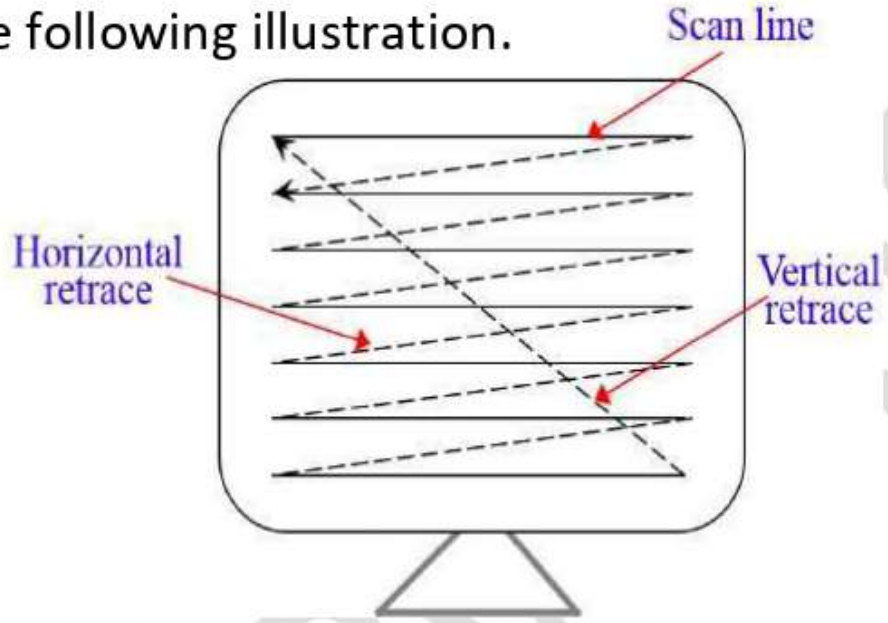
Inkjet Plotters: These types of plotters use inkjet technology and colored inkjet pens to provide high-quality color images and graphics. They are best suitable for applications that require vibrant, detailed output, precision work such as graphic design, architectural renderings, and fine art in a less time. They are commonly used for a large printer, like billboards, banners, and big signs that are used for roadside indication.

2.4 Raster and Vector Display Architectures, Principles and Characteristics

Raster Display

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

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



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

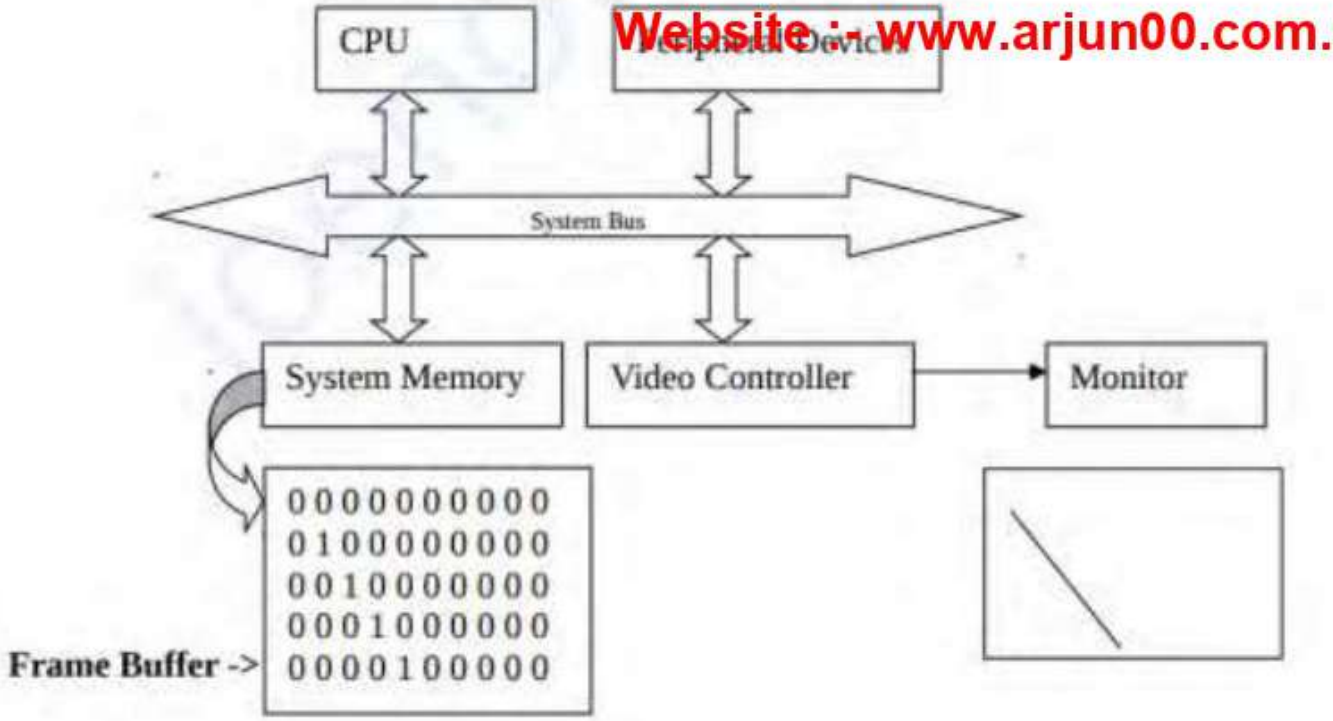


Fig. Architecture of Vector Display

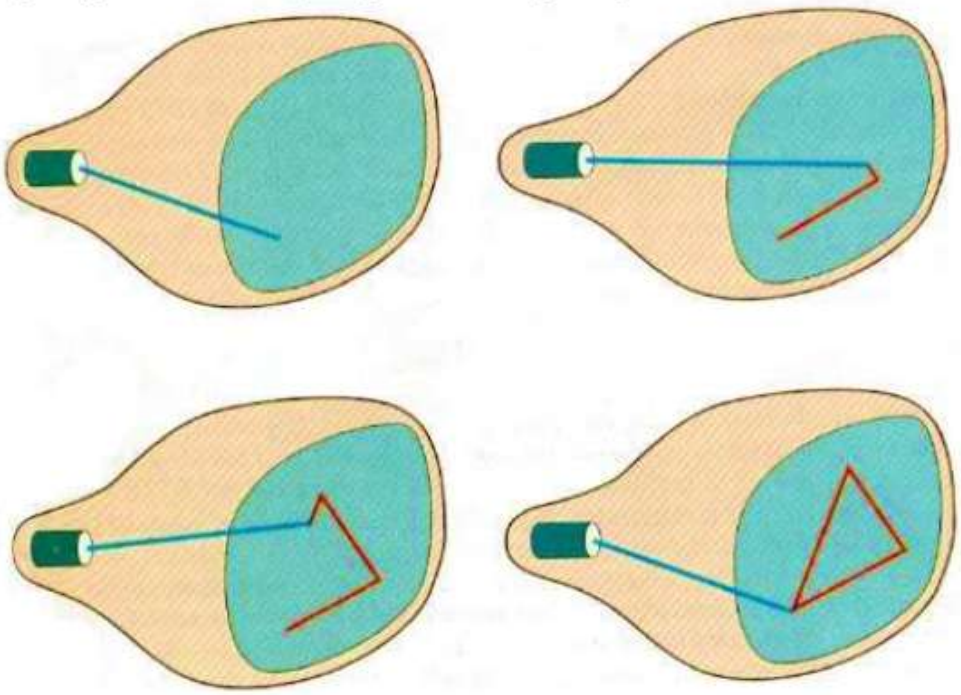
A pixel in a frame buffer may be represented by one bit where each pixel on CRT screen is either on '1' or off '0'

Characteristics:

1. **Pixel-Based:** Images are composed of pixels, individually controlled for color and intensity.
2. **Electron Beam Scanning:** CRTs use electron beams to illuminate phosphors; LCDs and OLEDs use electronic signals.
3. **Grid Formation:** The screen is divided into a grid of rows and columns, with each intersection representing a pixel.
4. **Sequential Activation:** Pixels are activated sequentially, usually row by row or column by column, resulting in the formation of images.
5. **Image Quality:** Renders detailed images, photos, and videos with high resolution and color depth.

Vector Scan (Random Scan)

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



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

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

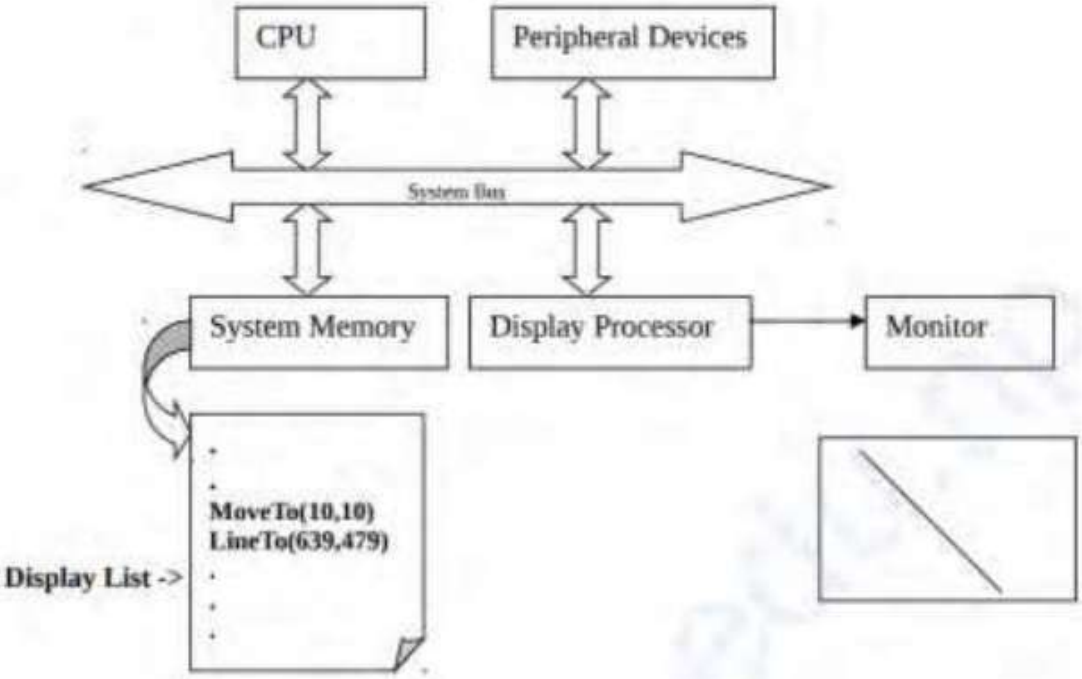


Fig. Architecture of Random Display

Characteristics:

1. **Vector Drawing:** Vector scan displays utilize vector graphics, which define images using mathematical equations or commands to draw lines and shapes on the screen.
2. **Electron Beam Control**: An electron beam is used to draw lines or shapes on the screen, with the beam's intensity and position controlled to create the desired graphics.
3. **Continuous Drawing:** It draw lines continuously from one point to another, allowing for smooth rendering of geometric shapes and curves. This continuous drawing method results in crisp and precise graphics with no pixelation.
4. **Scalability:** Vector displays are more scalable because the complexity of the image does not depend on the number

of pixels. They can handle images of various sizes without loss of quality, making them suitable for applications requiring variable image dimensions.

5. **Applications:** Commonly used in early computer graphics, flight simulators, radar displays, and oscilloscopes, ideal for precise line drawing tasks.

Unit:3 – Two Dimensional Algorithms and Transformations:

3.1 Mathematical Line Drawing Concept

In computer graphics, Mathematical line drawing refers to the process of generating visual representations of lines on a digital display or canvas using mathematical algorithms. These algorithms calculate the appropriate pixel positions to create the illusion of a continuous line between specified endpoints.

The primary goal of mathematical line drawing algorithms is to efficiently and accurately determine which pixels should be activated (or colored) to represent the line between two given points.

3.2 Line Drawing Algorithm

Various algorithms, such as Digital Differential Analyzer (DDA) and Bresenham's Line Algorithm are commonly used for line drawing in computer graphics.

3.2.1 Digital Differential Analyzer (DDA)

The Digital Differential Analyzer (DDA) is a simple algorithm used for generating digital representations of straight lines on a raster display device, such as a computer monitor. It

calculates the coordinates of points along the line path between two given endpoints.

The basic principle of the DDA algorithm involves incrementally stepping from one endpoint to the other in either the x or y direction (whichever has the greater distance), while linearly interpolating the other coordinate based on the slope of the line.

Advantages of DDA Algorithm:

- It is a simple and easy-to-implement algorithm.
- It avoids using multiple operations which have high time complexities.
- It is faster than the direct use of the line equation because it does not use any floating-point multiplication and it calculates points on the line.

Disadvantages of DDA Algorithm:

- It deals with the rounding off operation and floating-point arithmetic so it has high time complexity.
- As it is orientation-dependent, so it has poor endpoint accuracy.
- Due to the limited precision in the floating-point representation, it produces a cumulative error.

Uses of DDA Algorithm:

- Creating basic graphics: It can be used to draw simple shapes such as lines, polygons, and rectangles.
- Computer-aided design (CAD): It is used to draw lines between two points.
- Image processing: It can be used in image processing for tasks such as edge detection and image segmentation.
- Video game development: It is used for rendering lines and polygons in real-time graphics rendering for video games.

DDA Algorithm

1. Define the starting point of the line as (x1, y1) and the ending point as (x2, y2).
2. Calculate the slope of the line: m = (y2 - y1) / (x2 - x1).
3. If the slope (m) is less than 1 (m < 1), increment x by 1 and calculate y as y1 = y1 + m.
4. If the slope (m) is greater than 1 (m > 1), increment y by 1 and calculate x as x1 = x1 + 1 / m.
5. If the slope is equal to 1 (m = 1), increment both x and y by 1: x1 = x1 + 1 and y1 = y1 + 1.
6. Repeat the above steps until the end point of the line (x2, y2) is reached.

Draw a line from point (1,1) to the point (8,7) using DDA

Here,

(x1,y1) = (1,1) and (x2,y2) = (8,7)

First, we calculate dif x and dif y:

dif x = |x2-x1| = |8-1| = 7

dif y = |y2-y1| = |7-1| = 6

slope of the line:

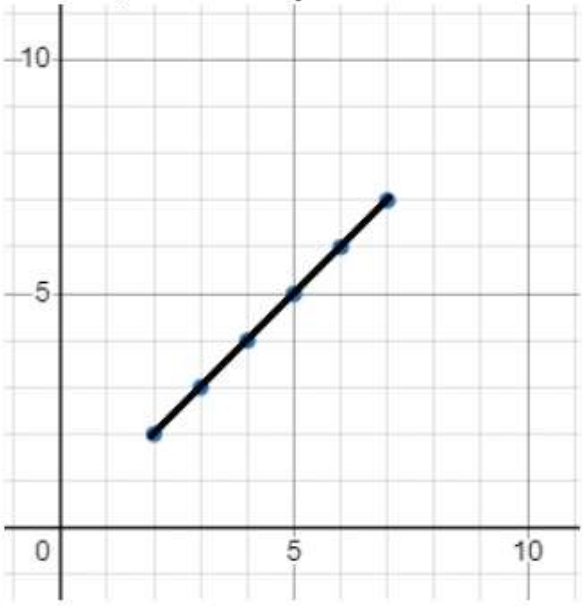
m = dif y / dif x = 6/7 = 0.9 (ie. m<1)

as m<1, x is increase by 1 and y is calculated.

The steps will be x1=x1+1 and y1=y1+m

No. of steps	X	Y	X1=x1+1	Y1=y1+m	Roundoff (x1,y1)
1	1	1	1+1=2	1+0.9=1.9	(2,2)
2	2	2	2+1=3	2+0.9=2.9	(3,3)
3	3	3	3+1=4	3+0.9=3.9	(4,4)
4	4	4	4+1=5	4+0.9=4.9	(5,5)
5	5	5	5+1=6	5+0.9=5.9	(6,6)
6	6	7	6+1=7	6+0.9=6.9	(7,7)

Here, we stop as x value reach 7



Try it yourself:

1. Draw a line using the DDA from the point (0,0) to (4,6).
   Answer= (0,0), (1,1), (1,2), (2,3), (3,4), (3,5), (4,6)
2. Draw a line using the DDA from the point (0,0) to (7,7).
   Answer = (2,2), (3,3), (4,4), (5,5), (6,6), (7,7)
3. Draw a line using the DDA from the point (2,3) to (9,8).
   Answer = (2,3), (3,4), (4,4), (5,5), (6,6), (7,7), (8,7), (9,8)

3.2.2 Bresenham's Line Drawing Algorithm:

Bresenham's line drawing algorithm is a fundamental method used in computer graphics to draw a line between two points on a grid or raster display. The algorithm works by determining which pixels to turn on or "plot" to approximate the line between two endpoints. It iteratively calculates the positions of pixels that lie closest to the ideal line path.

Advantages of Bresenham's Line Drawing Algorithm:

- Efficiency: It's fast due to its use of integer arithmetic, making it suitable for real-time applications.
- Accuracy: It generates precise results, ensuring the drawn line closely follows the intended path.
- Versatility: It can be adapted to draw various geometric shapes efficiently, enhancing its usefulness in computer graphics.

Disadvantages of Bresenham's Line Drawing Algorithm:

- Limited to Straight Lines: It's primarily designed for drawing straight lines and requires modification for curves or more complex shapes.
- Fixed-Width Lines: It produces lines with a fixed width, which may not be suitable for all graphical applications.
- Restricted to Octants: It operates optimally within specific octants of the coordinate plane, requiring additional logic for lines in other directions.

Uses of Bresenham's Line Drawing Algorithm:

- Computer Graphics: It's widely used in graphics libraries and software for rendering straight lines efficiently on displays.
- Rasterization: It forms the basis for rasterization techniques used in computer graphics to render geometric primitives like lines, circles, and ellipses.
- Drawing Applications: It's employed in drawing applications and graphic design software for line drawing and rendering vector graphics.
- Image Processing: It's utilized in image processing algorithms for tasks such as edge detection and feature extraction.

Bresenham's Line Drawing Algorithm:

Starting coordinates = (X_0, Y_0)
Ending coordinates = (X_n, Y_n)

Step-01:
Calculate:
$\Delta X = X_n - X_0$
$\Delta Y = Y_n - Y_0$
$2\,\Delta Y = 2 * \Delta Y$
$2\,\Delta Y - 2\,\Delta X$

Step-02:
Calculate the decision parameter Pk as:
$P_0 = 2\Delta Y - \Delta X$

Step-03:
Suppose the current point is (X_k, Y_k) and the next point is (X_{k+1}, Y_{k+1}).
Find the next point depending on value of decision parameter P_k.
Follow the below two cases:

Case 1: If $P_k<0$ than
Point to plot (X_k+1, Y_k)
$P_{k+1}=P_k + 2\Delta Y$

Case 2: If $P_k>=0$ than
Point to plot (X_k+1, Y_k+1)
$P_{k+1}=P_k + 2\Delta Y - 2\Delta X$

Step-04:
Keep repeating Step-03 until the end point is reached.

Calculate the points between the starting coordinates (9, 18) and ending coordinates (14, 22)

Here,

Starting coordinates = (X0, Y0) = (9, 18)

Ending coordinates = (Xn, Yn) = (14, 22)

Step-01:

Calculate :

$\Delta X = Xn - X0 = 14 - 9 = 5$

$\Delta Y = Yn - Y0 = 22 - 18 = 4$

$2\Delta Y = 2*4=8$

$2\Delta Y - 2\Delta X = (4*2)-(5*2) = 8-10 = -2$

Step-02:

Calculate the decision parameter $P_0 = 2\Delta Y - \Delta X$

$= 2 \times 4 - 5 = 3$

Step-03:

As Pk >= 0, so case 2 is satisfied.

Next point to plot = $(X_k+1, Y_k+1) = (9+1, 18+1) = (10,19)$

& $P_1 = P_0 + 2\Delta Y - 2\Delta X = 3 + (-2) = 1$

As Pk >= 0, so case 2 is satisfied.

Next point to plot = $(X_k+1, Y_k) = (11,20)$

& $P_2 = P_1 + 2\Delta Y = 1 + (8) = 9$

As Pk >= 0, so case 2 is satisfied.

Next point to plot = $(X_k+1, Y_k+1) = (12,20)$

& $P_3 = P_2 + 2\Delta Y - 2\Delta X = 9 + (-2) = 7$

As Pk >= 0, so case 2 is satisfied.

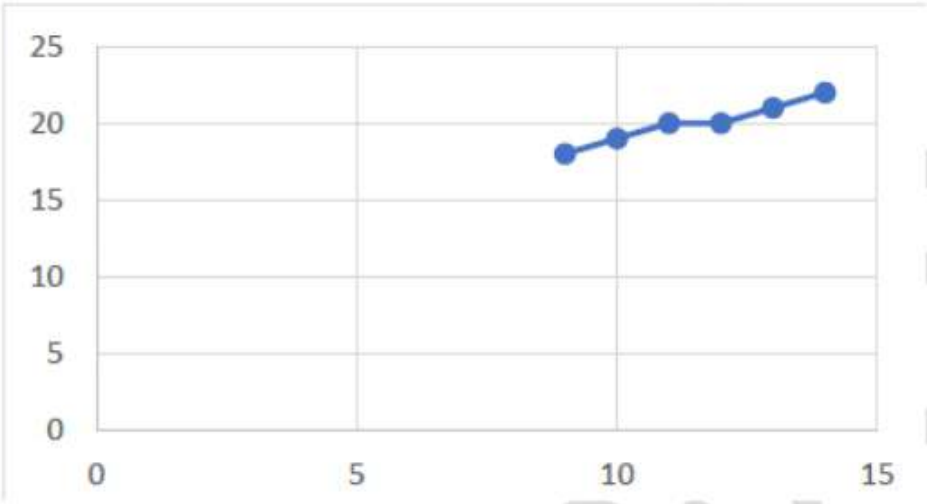
Next point to plot = $(X_k+1, Y_k+1) = (13,21)$

& $P_4 = P_3 + 2\Delta Y - 2\Delta X = 7 + (-2) = 5$

As Pk >= 0, so case 2 is satisfied.

Next point to plot = $(X_k+1, Y_k+1) = (14,22)$

P	P_k	X_{k+1}	Y_{k+1}
		9	18
P0	3	10	19
P1	-1	11	20
P2	7	12	20
P3	5	13	21
P4	3	14	22



Try it yourself:

1. Calculate the points between the starting coordinates (20, 10) and ending coordinates (30, 18).
   Answer = [redacted]
2. Calculate the points between the starting coordinates (1, 1) and ending coordinates (8, 5).
   Answer = [redacted]

3.3 Mid-Point Circle Drawing

Midpoint circle drawing algorithm is a technique used in computer graphics to draw a circle on a raster display device like a computer screen. The algorithm works by dividing a circle into eight equal parts, each corresponding to a different octant. It starts drawing the circle from a point (x0, y0) in the first octant and then mirrors the plotted pixels in the other octants based on symmetry. The key idea behind the algorithm is to use the midpoint of the previously plotted pixel to decide which pixel to plot next.

Algorithm:

Given,

- Centre point of Circle = (X_0, Y_0)
- Radius of Circle = R

Step-01:

Assign the starting point coordinates (X_0, Y_0) as-

$X_0 = 0$ $Y_0 = R$

Step-02:

Calculate the value of initial decision parameter P_0 as- $P_0 = 1 - R$

Step-03:

Suppose the current point is (X_k, Y_k) and the next point is (X_{k+1}, Y_{k+1}).

Find the next point of the first octant depending on the value of decision parameter P_k.

Follow the below two cases-

Case 1: If $P_k < 0$

$X_{k+1} = X_k + 1, \quad Y_{k+1} = Y_k, \quad P_{k+1} = P_k + 2 * X_{k+1} + 1$

Case 2: If $P_k >= 0$

$X_{k+1} = X_k + 1, \quad Y_{k+1} = Y_k - 1, \quad P_{k+1} = P_k - 2 * Y_{k+1} + 2 * X_{k+1} + 1$

Step-04:

If the given centre point (X_0, Y_0) is not (0, 0), then do the following and plot the point-

- $X_{plot} = X_c + X_0$
- $Y_{plot} = Y_c + Y_0$

Here, (X_c, Y_c) denotes the current value of X and Y coordinates.

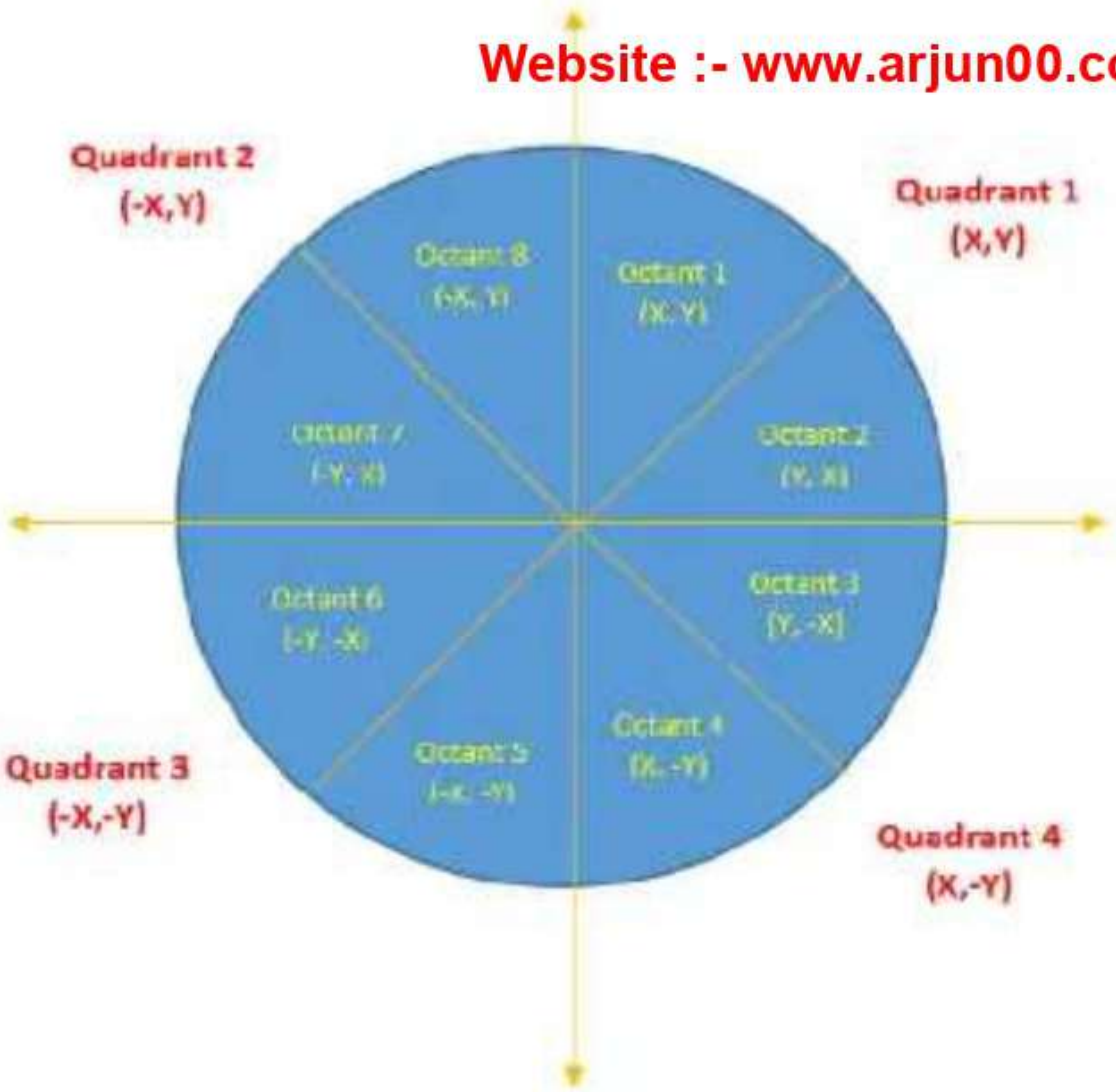
Step-05:

Keep repeating Step-03 and Step-04 until $X_{plot} >= Y_{plot}$.

Step-06:

Step-05 generates all the points for one octant.

To find the points for other seven octants, follow the eight-symmetry property of circle.



Given the center point coordinates (0, 0) and radius as 10, generate all the points to form a circle.

Given-

- Centre Coordinates of Circle $(X_0, Y_0) = (0, 0)$
- Radius of Circle = 10

Step-01:

Assign the starting point coordinates (X_0, Y_0) as $X_0 = 0$ and $Y_0 = R = 10$

Step-02:

Calculate the value of initial decision parameter P_0 as: $P_0 = 1 - R = 1 - 10$, $P_0 = -9$

Step-03:

As $P_{initial} < 0$, so case-01 is satisfied. $X_{k+1} = X_k + 1 = 0 + 1 = 1$

$Y_{k+1} = Y_k = 10$ and $P_{k+1} = P_k + 2 \times X_{k+1} + 1 = -9 + (2 \times 1) + 1 = -6$

Step-04:

This step is not applicable here as the given center point coordinates is (0, 0).

Step-05:

Step-03 is executed similarly until $X_{k+1} >= Y_{k+1}$ as follows

P_k	P_{k+1}	(X_{k+1}, Y_{k+1})
		(0, 10)
-9	-6	(1, 10)
-6	-1	(2, 10)
-1	6	(3, 10)
6	-3	(4, 9)
-3	8	(5, 9)
8	5	(6, 8)
Algorithm Terminates, these are all points for Octant-1.		

Algorithm calculates all the points of octant-1 and terminates. Now, the points of octant-2 are obtained using the mirror effect by swapping X and Y coordinates.

Octant-1 Points	Octant-2 Points
(0, 10)	(8, 6)
(1, 10)	(9, 5)
(2, 10)	(9, 4)
(3, 10)	(10, 3)
(4, 9)	(10, 2)
(5, 9)	(10, 1)
(6, 8)	(10, 0)
These are all points for Quadrant-1.	

Now, the points for rest of the part are generated by following the signs of other quadrants. The other points can also be generated by calculating each octant separately.
Here, all the points have been generated with respect to quadrant-1-

Quadrant-1 (X,Y)	Quadrant-2 (-X,Y)	Quadrant-3 (-X,-Y)	Quadrant-4 (X,-Y)
(0, 10)	(0, 10)	(0, -10)	(0, -10)
(1, 10)	(-1, 10)	(-1, -10)	(1, -10)
(2, 10)	(-2, 10)	(-2, -10)	(2, -10)
(3, 10)	(-3, 10)	(-3, -10)	(3, -10)
(4, 9)	(-4, 9)	(-4, -9)	(4, -9)
(5, 9)	(-5, 9)	(-5, -9)	(5, -9)
(6, 8)	(-6, 8)	(-6, -8)	(6, -8)
(8, 6)	(-8, 6)	(-8, -6)	(8, -6)
(9, 5)	(-9, 5)	(-9, -5)	(9, -5)
(9, 4)	(-9, 4)	(-9, -4)	(9, -4)
(10, 3)	(-10, 3)	(-10, -3)	(10, -3)
(10, 2)	(-10, 2)	(-10, -2)	(10, -2)
(10, 1)	(-10, 1)	(-10, -1)	(10, -1)
(10, 0)	(-10, 0)	(-10, 0)	(10, 0)
These are all points of the Circle.			

3.5 Review of Matrix Operations – Addition and Multiplication

Addition

Two matrices can be added/subtracted, iff (if and only if) the number of rows and columns of both the matrices are same, or the order of the matrices are equal.

Matrix Addition of 2*2 Matrices

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} + \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11}+b_{11} & a_{12}+b_{12} \\ a_{21}+b_{21} & a_{22}+b_{22} \end{bmatrix}$$

Example:

$$A = \begin{bmatrix} 2 & 1 \\ 5 & 4 \end{bmatrix}$$

$$B = \begin{bmatrix} 2 & 0 \\ 7 & 3 \end{bmatrix}$$

$$A + B = \begin{bmatrix} 2 & 1 \\ 5 & 4 \end{bmatrix} + \begin{bmatrix} 2 & 0 \\ 7 & 3 \end{bmatrix} = \begin{bmatrix} 4 & 1 \\ 12 & 7 \end{bmatrix}$$

Multiplication

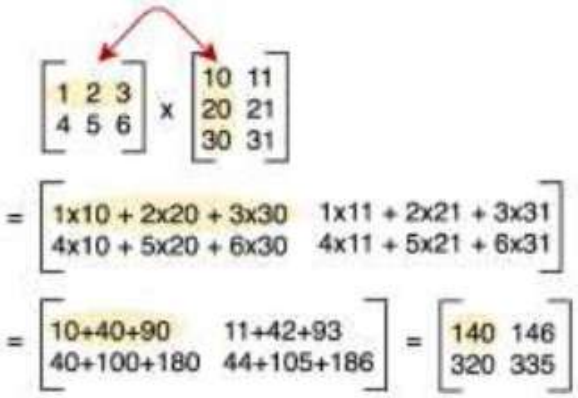
Matrix can be Multiplied two ways:

1. Scalar Multiplication
2. Multiplication with another matrix

1. **Scalar Multiplication**: It involves multiplying a scalar quantity to the matrix. Every element inside the matrix is to be multiplied by the scalar quantity to form a new matrix.

$$5 \times \begin{bmatrix} 5 & 7 \\ 12 & 3 \\ 6 & 2 \end{bmatrix} = \begin{bmatrix} 25 & 35 \\ 60 & 15 \\ 30 & 10 \end{bmatrix}$$

2. **Multiplication of a matrix with another matrix:** Two matrices can be multiplied iff the number of columns of the first matrix is equal to the number of rows of the second matrix.



3.6 Two-dimensional Transformations

2D Transformations take place in a two-dimensional plane. Transformations are helpful in changing the position, size, orientation, shape etc of the object.

3.6.1 Translation

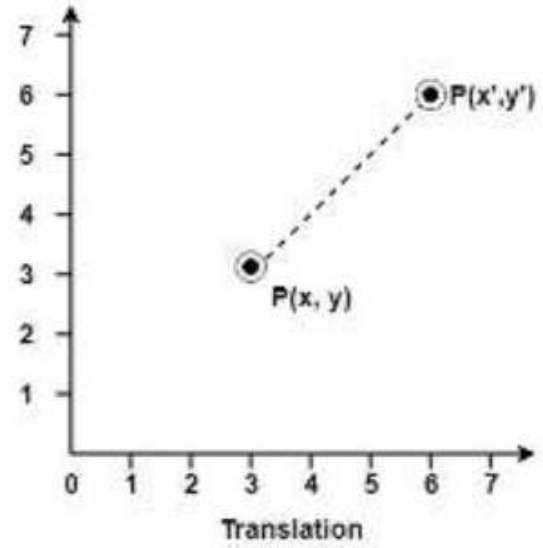
It is the straight-line movement of an object from one position to another is called Translation. Here the object is positioned from one coordinate location to another.

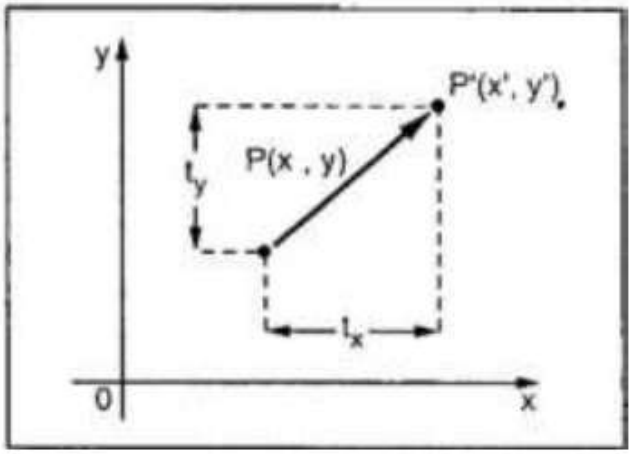
Translation of point:

To translate a point from coordinate position (x, y) to another (x_1 y_1), we add algebraically the translation distances T_x and T_y to original coordinate.

$x_1 = x + T_x$ $y_1 = y + T_y$

The translation pair (T_x,T_y) is called as shift vector.





Matrix for Translation:

$$\begin{vmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ t_x & t_y & 1 \end{vmatrix} \quad \text{Or} \quad \begin{vmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{vmatrix}$$

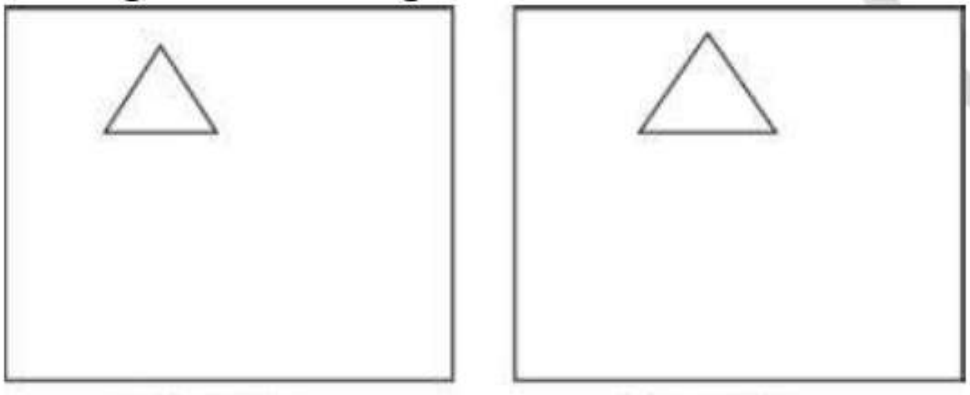
3.6.2 Scaling

It is used to alter or change the size of objects. The change is done using scaling factors. There are two scaling factors, i.e. Sx in x direction Sy in y-direction. If the original position is x and y. Scaling factors are Sx and Sy then the value of coordinates after scaling will be x1 and y1.

Enlargement: If $T_1=\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$,If (x_1 y_1)is original position and T_1 is translation vector then (x_2 y_2) are coordinated after scaling

$$[x_2 y_2] = [\, x_1 y_1] \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} = [2x_1 2y_1]$$

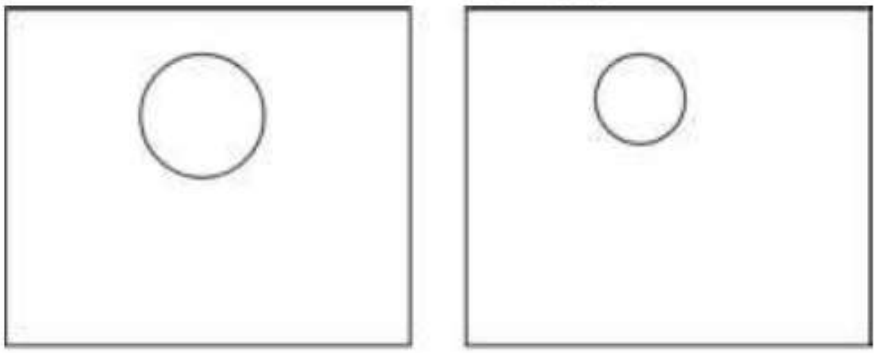
The image will be enlarged two times



Original Image Enlarged Image

Reduction: If $T_1=\begin{bmatrix}0.5 & 0\\ 0 & 0.5\end{bmatrix}$. If $(x_1\ y_1)$ is original position and T_1 is translation vector, then $(x_2\ y_2)$ are coordinates after scaling

$$[x_2y_2] = [\ x_1y_1]\begin{bmatrix}0.5 & 0\\ 0 & 0.5\end{bmatrix} = [.5x_1 . 5y_1]$$

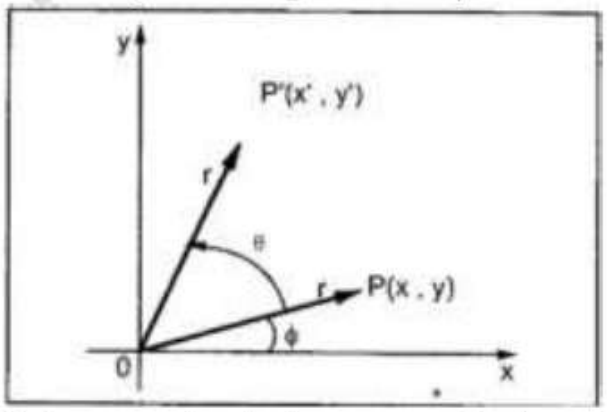


Original Image Reduction Image

3.6.3 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 PX,Y is located at angle ϕ from the horizontal X coordinate with distance r from the origin.

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



Using standard trigonometric the original coordinate of point PX,Y can be represented as $X=r\cos\phi \ldots\ldots(1)$

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

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

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

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

Substituting equation 1 & 2 in 3 & 4 respectively, we will get

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

Representing the above equation in matrix form,

$$[X'Y'] = [XY]\begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix} \quad \text{OR} \quad P' = P \cdot R$$

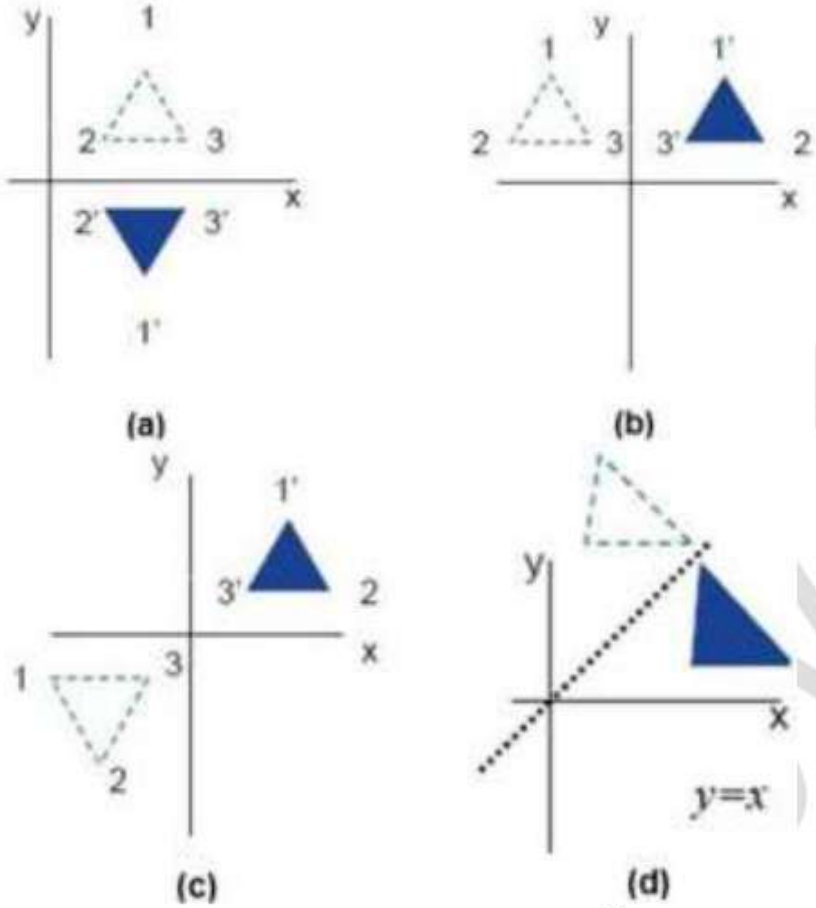
Where R is the rotation matrix

$$R = \begin{bmatrix} cos\theta & sin\theta \\ -sin\theta & cos\theta \end{bmatrix}$$

3.6.4 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.

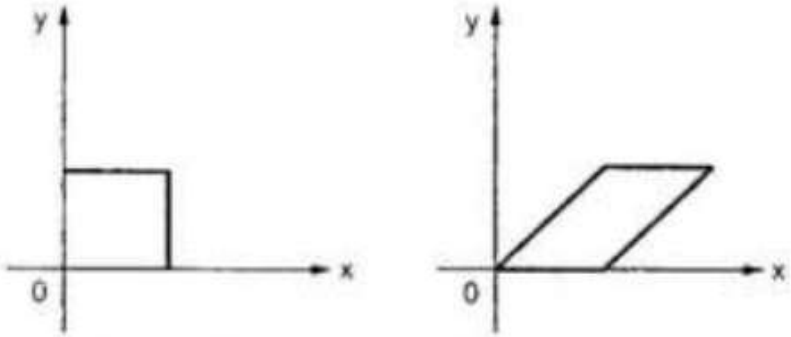


3.6.5 Shearing

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.



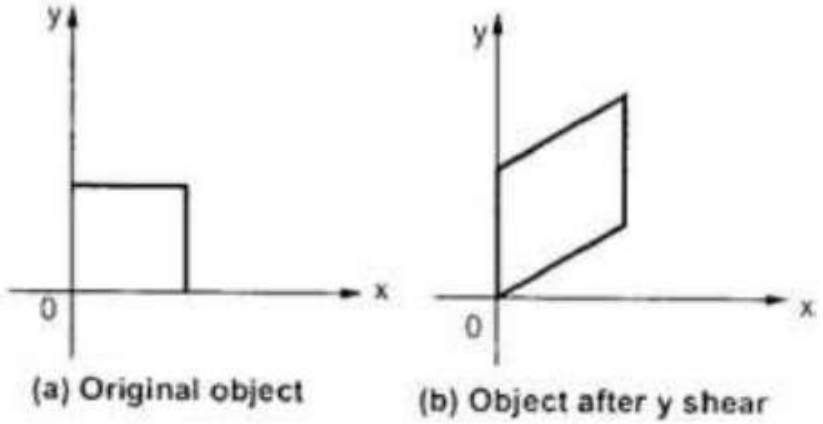
(a) Original object (b) Object after x shear

The transformation matrix for X-Shear can be represented as

$$X_{sh} = \begin{bmatrix} 1 & 0 & 0 \\ Sh_x & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Y-Shear:

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



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

The Y-Shear can be represented in matrix from as

$$Y_{sh} = \begin{bmatrix} 1 & Sh_y & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3.7 Two-Dimensional Viewing Pipeline

The 2D viewing pipeline in computer graphics refers to the process of transforming 2D objects from their model coordinates to screen coordinates for display on a 2D screen or viewport. The 2D viewing pipeline in computer graphics is a crucial process that maps world coordinates (actual coordinates of objects in the real world) to device coordinates (the actual coordinates of objects on an output screen, such as a monitor).

The 2D viewing pipeline in computer graphics involves several essential steps:

1. **Modeling Transformation:** Applying transformations like translation, rotation, scaling, and shearing to position and orient 2D objects as desired.

2. **Viewing Transformation:** Mapping transformed objects from model space to a specific viewpoint or camera position in world space.

3. **Projection Transformation:** Projecting 3D world coordinates onto a 2D plane, representing the screen or viewport. Techniques include perspective and orthographic projection.

4. **Clipping:** Removing parts of objects outside the viewing volume or viewport to render only visible portions.

5. **Viewport Transformation:** Mapping transformed and clipped objects to 2D screen coordinates, scaling and translating them to fit within screen boundaries.

Unit:4 – Three-Dimensional Graphics:

4.1 3D transformation

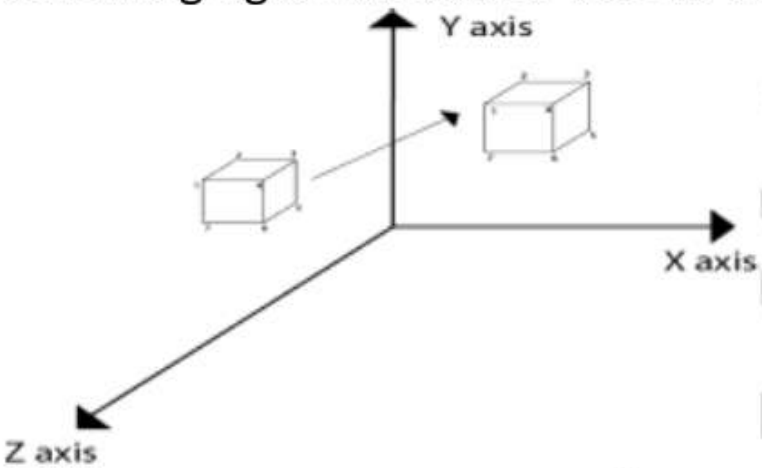
The geometric transformations play a vital role in generating images of 3D objects with the help of these transformations, the location of objects relative to others can be easily expressed. Sometimes viewpoint changes rapidly, or sometimes objects move in relation to each other. For this number of transformations can be carried out repeatedly. Transformations are helpful in changing the position, size, orientation, shape etc of the object.

4.1.1 Translation

The translation moves an object in 3D space along the x, y, and z axes. Suppose (x,y,z,1) is a point in 3D space and (x′,y′,z′,1) is a transformed point, then the translation equation would be:

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

Here, d (tx,ty,tz,1)represents the translation vector. The following figure visualizes the 3D translation:

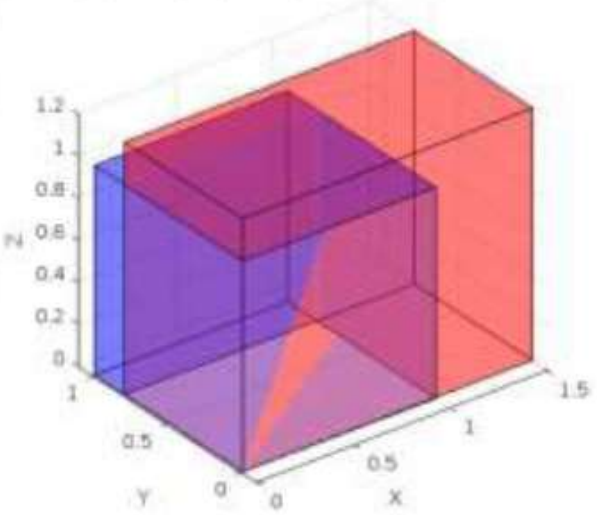


4.1.2 Scaling

Scaling is the process of resizing an object in 3D space. Suppose (x,y,z,1) is a point in 3D space and (x′,y′,z′,1) is a transformed point, then the scaling equation would be:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

where, S_x, S_y, and S_z represent the scaling factors along the x, y, and z axes, respectively. The following figure visualizes the 3D scaling:



4.1.3 Rotation

Rotation involves changing the orientation of an object around one or more axes. The rotation occurs along an axis. It also includes the angle of rotation θ that determines the extent to which the object will be turned about that axis. If θ is positive, the rotation would be counterclockwise.

Suppose (x,y,z,1) is a point in 3D space and (x′,y′,z′,1) is a transformed point, then the rotation along the x-axis would be:

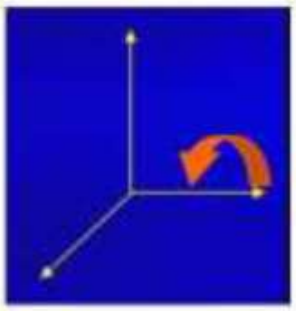
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\theta & -sin\theta & 0 \\ 0 & sin\theta & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

The rotation along the y-axis would be:

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

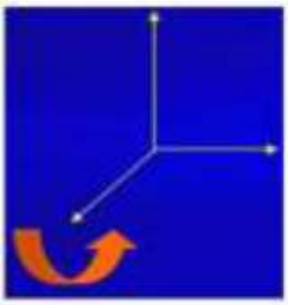
The rotation along the z-axis would be:

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



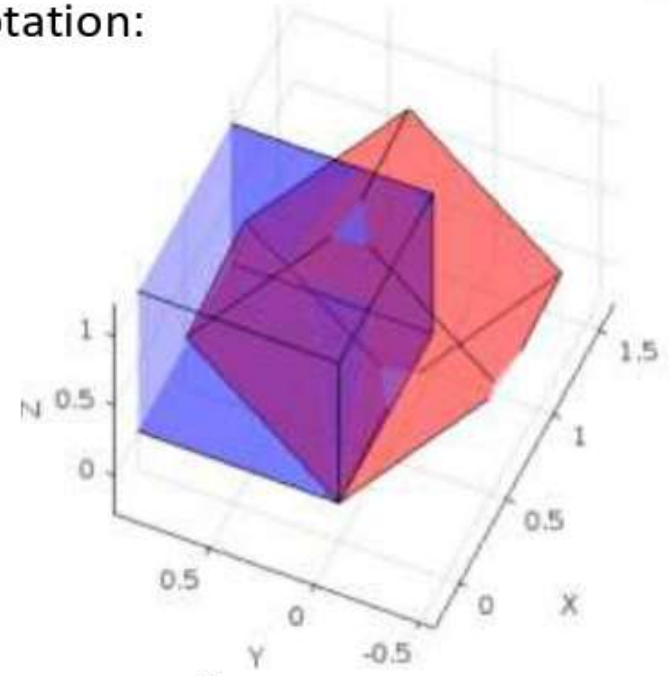
Rotation about x-axis

Rotation about y-axis



Rotation about z-axis

Here, θ represents the angle of rotation. The following figure visualizes the 3D rotation:

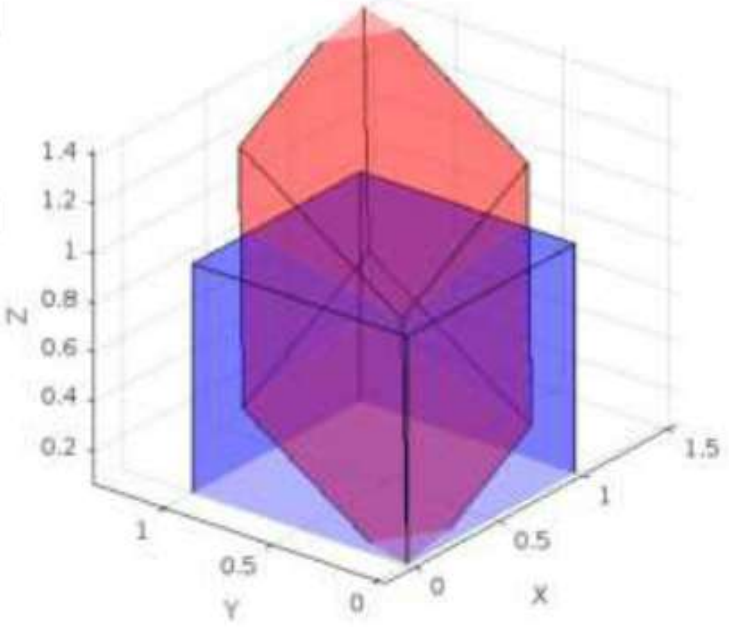


4.1.4 Shearing

The shear transformation distorts the shape of an object along one or more axes. Suppose (x,y,z,1) is a point in 3D space and (x′,y′,z′,1) is a transformed point, then the shearing equation would be:

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

where s_x,s_y,and s_z are the shear factors along the x, y, and z axes, respectively. The following figure visualizes the 3D shearing:



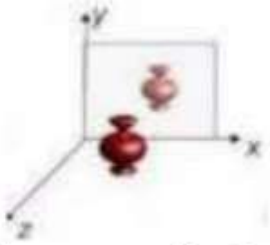
4.1.5 Reflection

Reflection is a kind of rotation where the angle of rotation is 180 degrees. The reflected object is always formed on the other side of mirror. The size of reflected object is same as the size of original object. For reflection, plane is selected

(xy,xz or yz). Following matrices show reflection respect to all these three planes.

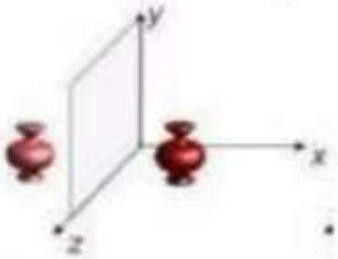
Reflection relative to XY plane or Z=0 plane:

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



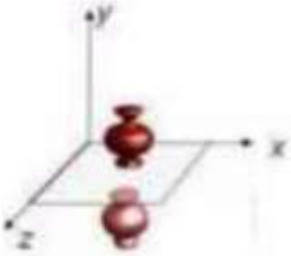
Reflection relative to YZ plane or X=0 plane:

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



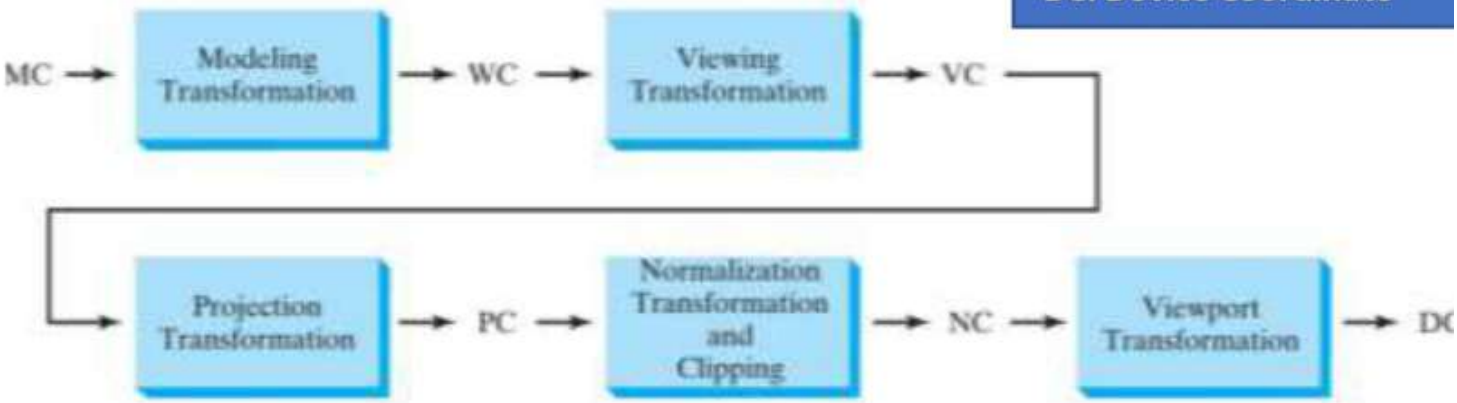
Reflection relative to ZX plane or Y=0 plane:

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



MC: Modeling Coordinate
WC: World Coordinate
VC: Viewing Coordinate
PC: Projection Coordinate
NC: Normalized Coordinat
DC: Device Coordinate

4.2 Three-dimensional Viewing Pipeline



The three-dimensional (3D) viewing pipeline in computer graphics refers to the process of transforming 3D objects and scenes into 2D images that can be displayed on a two-dimensional screen.

1. **Modeling**: This initial stage involves creating or importing 3D models of objects or scenes using modeling software. These models are represented as collections of vertices, edges, and faces.
2. **World Coordinates**: In this stage, the vertices of the 3D models are transformed from their local coordinate systems (object space) to a common coordinate system known as world space. This transformation involves applying translation, rotation, and scaling operations.
3. **Viewing Transformation**: The viewing transformation stage involves transforming the objects from world space into a coordinate system relative to the viewer or camera. This transformation typically includes setting up a camera viewpoint and applying operations such as translation, rotation, and possibly scaling to simulate the perspective.
4. **Clipping:** Clipping is the process of removing any objects or parts of objects that lie outside the view frustum, which is the region of space that is visible to the camera. This stage ensures that only the objects within the camera's field of view are rendered.
5. **Projection:** In this stage, the 3D coordinates of the objects are projected onto a 2D plane to create the final image.

6. **Viewport Transformation:** The projected 2D coordinates are mapped to the screen or viewport coordinates, which represent the actual pixels on the display device. This transformation involves scaling and translating the coordinates to fit the dimensions of the viewport.

4.3 Three dimensions Projections

4.3.1 Concept of Projection

In computer graphics, projection refers to the process of transforming a three-dimensional (3D) scene into a two-dimensional (2D) representation, typically for display on a screen. It's essentially a way to take a three-dimensional world and flatten it onto a two-dimensional plane. This transformation is necessary because computer screens, like paper, are inherently two-dimensional surfaces, and thus cannot directly represent 3D objects without some form of mapping.

4.3.2 Projection of 3D Objects onto 2D Display Devices

Our eyes perceive the world in 3D, but computer displays are inherently 2D (width and height only). Projecting a 3D object onto this flat surface requires some mathematical tricks to convey depth and realism.

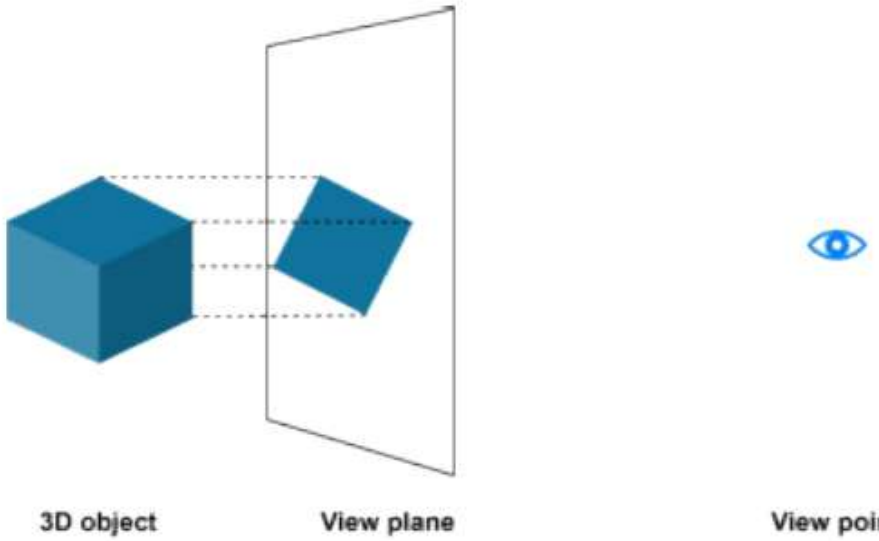
There are several steps involved in transforming a 3D object into a 2D image:

1. **Modeling:** The 3D object is defined using points in space (vertices) and how they connect (faces).
2. **Transformation:** The object's position, orientation, and scale are adjusted in the 3D world relative to a virtual camera.
3. **Projection:** Through a chosen projection method (perspective or parallel), a series of matrices are applied to each vertex of the object. These matrices essentially translate the 3D location of each point into a corresponding 2D location on the view plane.
4. **Clipping:** Parts of the object that fall outside the viewing frustum (the pyramid-shaped viewing area) are discarded.
5. **Rasterization:** Finally, the projected points are converted into pixels on the screen, determining the final image we see.

4.3.3 Three-dimensional Projection Methods

4.3.3.1 Parallel Projection Method

Parallel projections are a class of projections where projection lines are parallel to each other. Unlike perspective projection, parallel projections do not simulate the viewer's perspective, and therefore, objects do not appear smaller as they move farther away from the viewer. Instead, objects maintain their relative sizes and shapes regardless of their distance from the viewer.



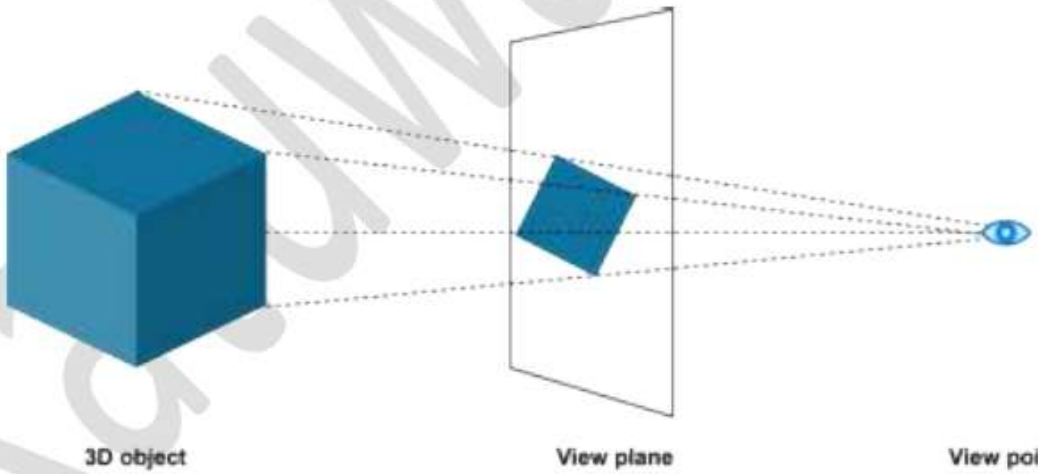
Two common types of parallel projections are:

- **Orthographic Projection:** In orthographic projection, the projection lines are perpendicular to the projection plane. This results in objects being projected onto the plane without any foreshortening or distortion. Orthographic projection is commonly used where accurate representation of shapes and proportions is crucial.
- **Oblique Projection:** Oblique projection is a type of parallel projection where the projection lines are not perpendicular to the projection plane. Instead, they are at an angle, resulting in a distorted representation of the objects.

4.3.3.2 Perspective Projection Method

Perspective projection is a method used to represent three-dimensional (3D) objects and scenes onto a two-dimensional (2D) surface, such as a computer screen or a piece of paper. It simulates the way objects appear in the

real world by taking into account the viewer's perspective. Perspective projection is widely used in applications such as 3D computer graphics, video games, virtual reality, and architectural visualization, where realism and immersion are important.



Key characteristics of perspective projection include:

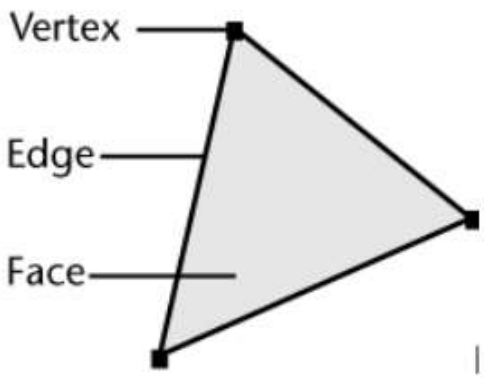
1. **Foreshortening:** Objects appear smaller as they move farther away from the viewer. This phenomenon creates a sense of depth in the rendered scene.
2. **Vanishing Point:** Parallel lines that recede into the distance appear to converge at a single point called the vanishing point. This effect is particularly noticeable in scenes with linear elements such as roads, railway tracks, or buildings.
3. **Depth Perception:** Perspective projection accurately conveys the relative distance between objects in the scene, allowing viewers to perceive the depth and spatial relationships between them.

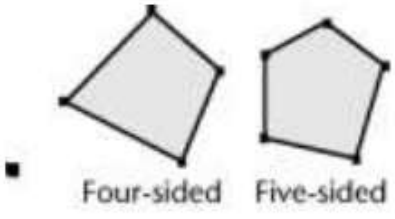
4.4 Three-dimensional Object Representations

In computer graphics, representing three-dimensional (3D) objects involves creating digital models that accurately depict the shape, appearance, and spatial characteristics of real-world objects. These digital models are then rendered onto a two-dimensional (2D) screen for viewing.

4.4.1 Polygon Surface

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. The polygon surfaces are common in design and solid-modeling applications, since their wireframe display can be done quickly to give general indication of surface structure. Then realistic scenes are produced by interpolating shading patterns across polygon surface to illuminate.

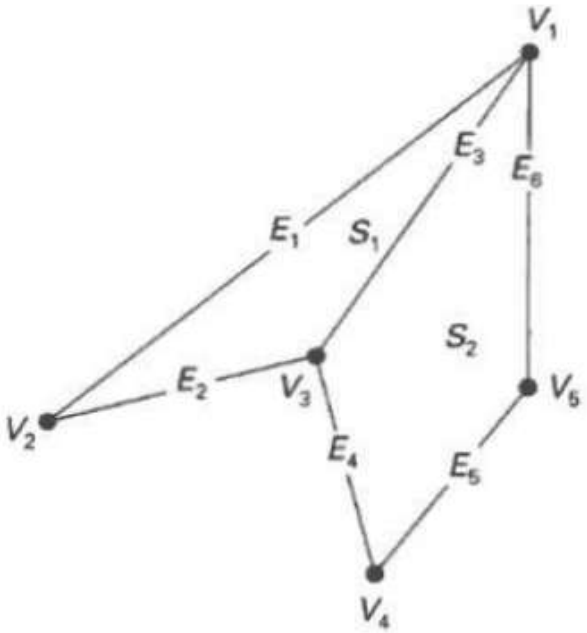




Polygonal models are composed of many separate polygons combined into a polygon mesh.

4.4.2 Polygon Tables

Polygon table is the specification of polygon surfaces using vertex coordinates and other attributes. It can be organized into two groups: geometric tables and attribute table.
For storing geometric data, we create three lists: a vertex table, an edge and a polygon table. Coordinates values for each vertex in the object are stored in vertex table. The vertices for each polygon edges, And the polygon table contains pointers back into the edges table to identify the edges for each polygon.
Attribute table contains qualitative properties like degree of transparency, surface reflectivity etc.



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

4.5. Introduction to Hidden Line and Hidden Surface Removal Techniques

Hidden surface removal (HSR) is a crucial technique in computer graphics to determine which parts of a 3D scene are visible to the viewer and which are obscured by other objects. It essentially removes hidden surfaces from the final image, creating a realistic representation of depth and visibility.

4.5.1 Object Space Method

In computer graphics, the Object Space Method is a technique used for hidden surface removal, where visibility relationships are determined directly in the object's model space before rendering. This method involves analyzing the geometry of objects in the scene to determine which surfaces are visible from the viewer's perspective. Only the visible surfaces of each object are rendered, while the hidden surfaces are discarded.

Advantages:

- Object space methods can be computationally efficient because visibility determination is performed directly on the objects themselves.
- They are suitable for static scenes where visibility relationships do not change frequently.

Disadvantages:

- Preprocessing time can be significant, especially for complex scenes with many objects.
- They may not be suitable for dynamic scenes where visibility relationships change frequently.

4.5.2 Image Space Method

In computer graphics, the Image Space Method is a technique used for hidden surface removal, where visibility is determined during or after the rendering process directly on the pixel values in the image space. This method involves rendering the scene as usual, producing a color buffer or depth buffer containing information about the visible surfaces. Algorithms are then applied to the buffers to determine which surfaces are visible from the viewer's perspective, and any hidden surfaces are removed in post-processing to generate the final image.

Advantages:

- Image space methods are more flexible and can handle dynamic scenes efficiently since visibility determination is performed directly on the rendered image.

- They do not require preprocessing of the scene geometry, making them suitable for dynamic scenes and real-time applications.

Disadvantages:

- They may be less efficient than object space methods for static scenes, as visibility determination is performed during or after rendering.
- They may introduce artifacts or errors in the final image, especially in scenes with complex geometry or occlusion.

4.6 Introduction to Illumination/ Lighting Models

In computer graphics, illumination models, also known as lighting models or shading models, are the secret sauce behind creating realistic and visually appealing images. They essentially dictate how light interacts with the surfaces of 3D objects in a virtual scene, determining their color, brightness, and shadows.

There are three factors on which lighting effect depends on:

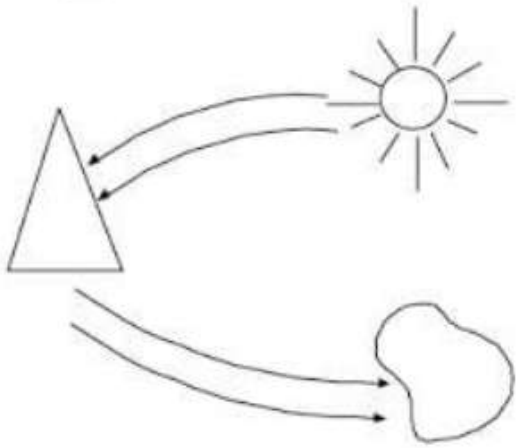
- **Light Source :** Light source is the light emitting source.
- **Surface :** When light falls on a surface part of it is reflected and part of it is absorbed. Now the surface structure decides the amount of reflection and absorption of light.

- **Observer :** The observer's position and sensor spectrum sensitivities also affect the lighting effect

Types of Illumination Models:

4.6.1 Ambient Model

The ambient lighting model simulates the overall illumination present in a scene. It represents the light that is scattered and bounced around the environment, providing a base level of illumination even in shadowed areas. Ambient lighting is typically uniform and lacks directionality. In a computer graphics context, ambient lighting is often used to prevent scenes from appearing completely dark and to provide a basic level of visibility.



If a surface is exposed only to ambient light, then the intensity of the diffuse reflection at any point on surface is $I=K_a I_a$; where I_a is the intensity of ambient light and K_a is the ambient coefficient reflection.

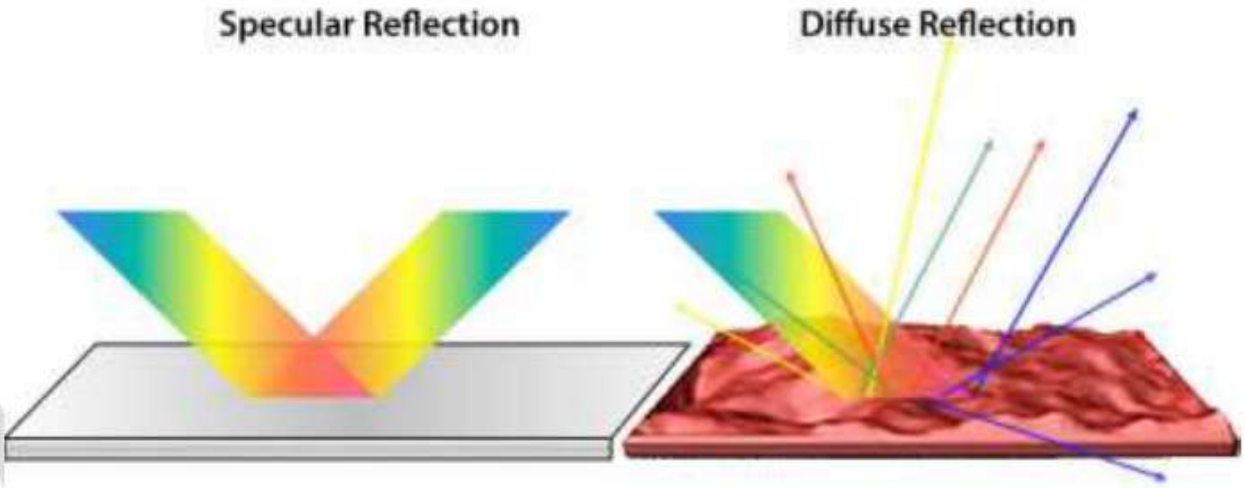
4.6.2 Diffuse Model

The diffuse lighting model simulates how light interacts with rough or matte surfaces. It accounts for the scattering of light in all directions upon hitting a surface. Diffuse lighting is essential for portraying the color and brightness of objects in a scene based on their surface properties and the angle of incident light. In this model, the brightness of a surface is determined by the angle between the light source and the surface normal (a line perpendicular to the surface). Surfaces facing the light source directly appear brighter, while those facing away from it appear darker. Diffuse lighting is critical for creating realistic shading and depth in computer-generated images.

4.6.3. Specular Model

The specular lighting model simulates the reflection of light off shiny or glossy surfaces. It accounts for the concentrated reflection of light in specific directions, such as the reflection of a light source off a polished surface like metal or glass. Specular highlights are typically small, bright areas that appear on surfaces where the angle of incidence equals the angle of reflection. The intensity and size of specular highlights depend on factors like the surface roughness and the properties of the light source. Specular lighting adds realism to computer-generated images by mimicking the way light interacts with smooth surfaces, creating highlights and reflections that contribute to the

perception of material properties such as shininess and glossiness.



4.7 Introduction to Shading/ Surface Rendering Models

Shading models, sometimes referred to as surface rendering models, are essential components of computer graphics systems. They're essentially sets of rules or algorithms that dictate how light behaves when it interacts with surfaces in a virtual environment. In simpler terms, they determine how objects look when light shines on them in a computer-generated scene.

4.7.1 Constant Shading Model

It is a simple method of polygon rendering. It is also called Flat shading. It is the simplest shading model, where each polygon in a 3D scene is rendered with a single, uniform color. This color is typically computed based on the angle between the surface normal (a line perpendicular to the

surface) and the direction of the light source. Flat shading does not account for variations in lighting across the surface of a polygon, resulting in a faceted appearance.

4.7.2 Gouraud Shading Model

Gouraud shading computes the color of each vertex of a polygon and interpolates these colors across the polygon's surface. It computes lighting at each vertex and then interpolates the resulting colors across the polygon using techniques like linear interpolation. Gouraud shading produces smoother results compared to flat shading, but it can still exhibit some visual artifacts, especially on surfaces with sharp changes in lighting.

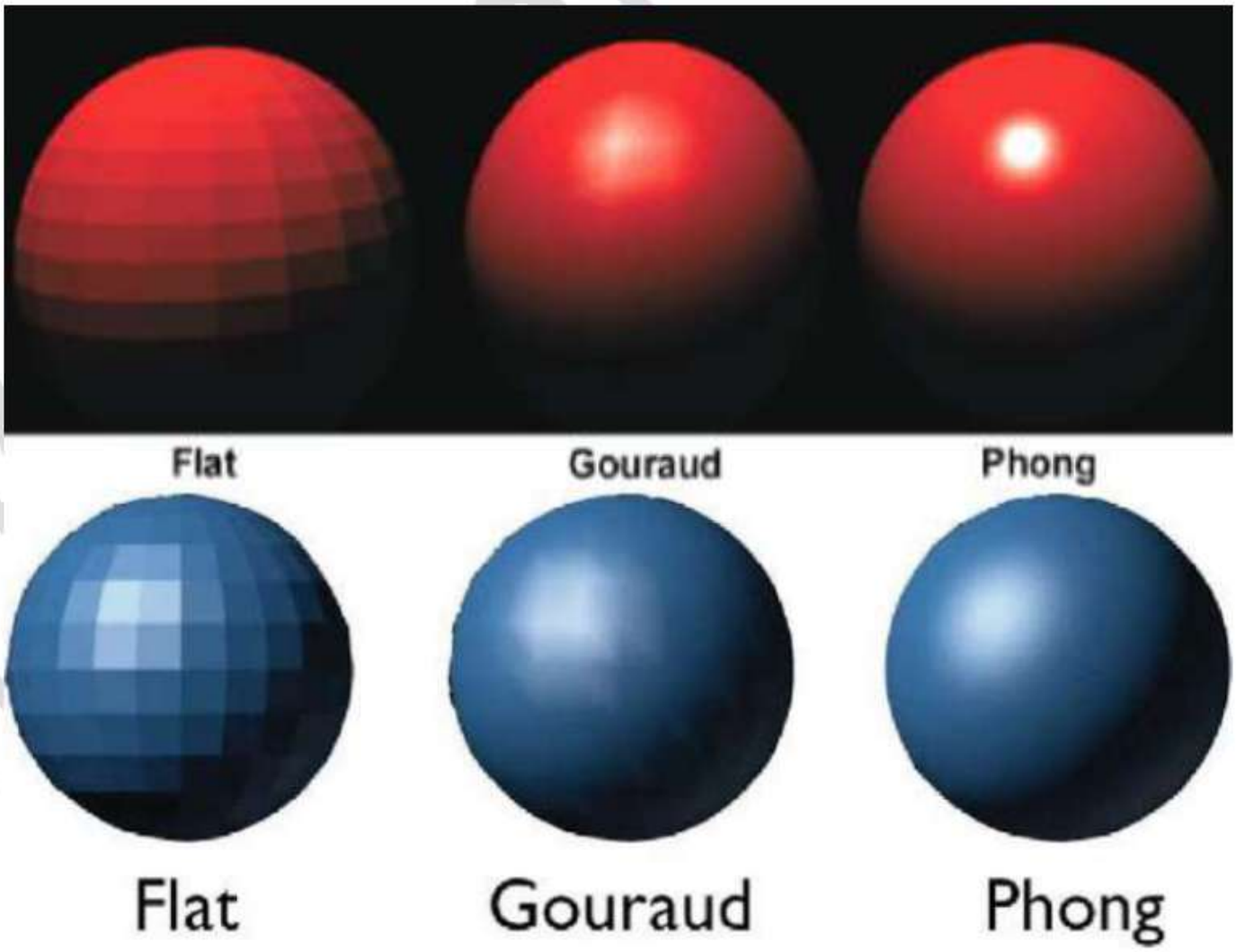
4.7.3 Phong Shading

Phong shading improves upon Gouraud shading by interpolating surface normals across a polygon's surface instead of just colors. This allows for more accurate calculations of lighting at each pixel, resulting in smoother highlights and better handling of specular reflections. Phong shading is widely used in computer graphics due to its ability to produce high-quality, realistic-looking images.

In summary:

- **Constant Intensity Shading:** Simple but unrealistic.
- **Gouraud Shading:** More realistic, but less accurate than Phong shading.

- **Phong Shading:** Accurate and provides realistic highlights.



Unit:5 – Web Graphics Designs and Graphics Design Packages:

5.1 Introduction to graphics file formats

Computer graphics file formats are essential for storing and displaying digital images. Some graphics formats are:

1. **JPEG (Joint Photographic Experts Group):**
   - JPEG is widely used for photographs and images with complex color gradients.
   - It uses lossy compression, which reduces file size but sacrifices some image quality.
   - Great for web images and photography due to its balance between quality and size
2. **PNG (Portable Network Graphics):**
   - PNG supports lossless compression and is ideal for images with transparency (alpha channels).
   - Commonly used for web graphics, logos, and icons.
   - Larger file sizes compared to JPEG but maintains high quality
3. **GIF (Graphics Interchange Format):**
   - GIF is suitable for simple animations and images with limited colors.
   - Supports transparency and animation loops.
   - Uses lossless compression but has a limited color palette.

4. **WebP:**
- Developed by Google, WebP combines both lossy and lossless compression.
- Provides better compression than JPEG and PNG for web images.
- Not as widely supported as other formats

5. **TIFF (Tagged Image File Format):**
- TIFF is commonly used for high-quality images, especially in printing and professional photography.
- Supports both lossless and lossy compression.
- Larger file sizes but excellent image fidelity.

6. **BMP (Bitmap):**
- BMP is a simple format that stores pixel data without compression.
- Commonly used in Windows applications.
- Large file sizes and not recommended for web use.

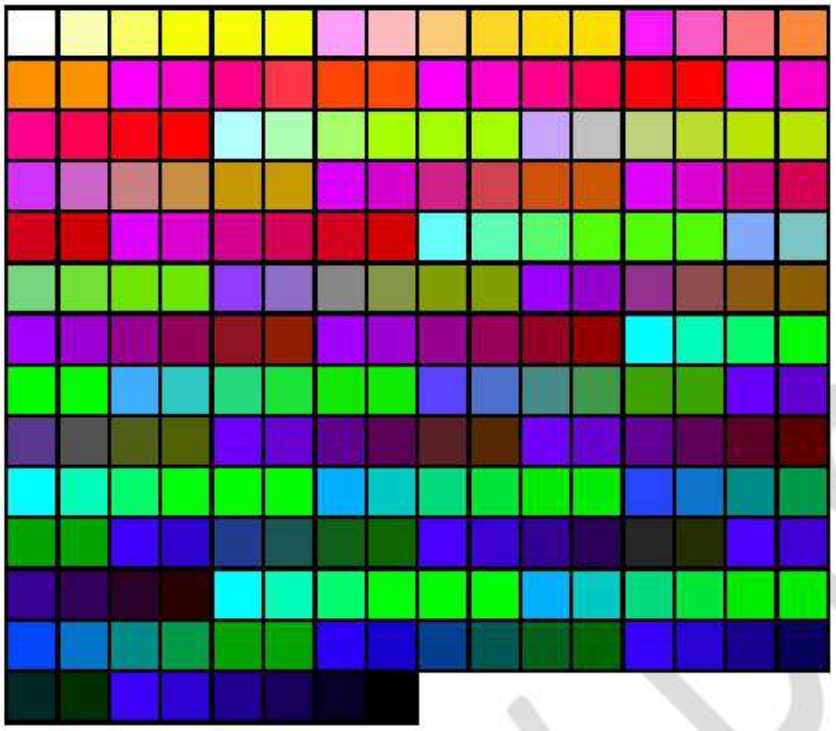
7. **HEIF (High Efficiency Image File Format):**
- HEIF is a modern format that provides better compression than JPEG.
- Supports both lossy and lossless compression.
- Used by Apple devices and some Android phones.

8. **SVG (Scalable Vector Graphics):**
- SVG is a vector format that defines images using geometric shapes and paths.
- Ideal for logos, icons, and graphics that need to scale without losing quality.
- Not suitable for complex photographic images.

5.2 Principles of web graphics design – browser safe colors, size, resolution, background, anti-aliasing

Browser-Safe Colors:

- The browser-safe color palette consists of 216 colors that display consistently on any computer monitor capable of showing at least 8-bit color (256 colors) 12.
- These colors avoid dithering and ensure a uniform appearance across different platforms and browsers.
- When designing web graphics, consider using these safe colors to maintain consistency.



Browser-Safe Colors Palette (216 colors)

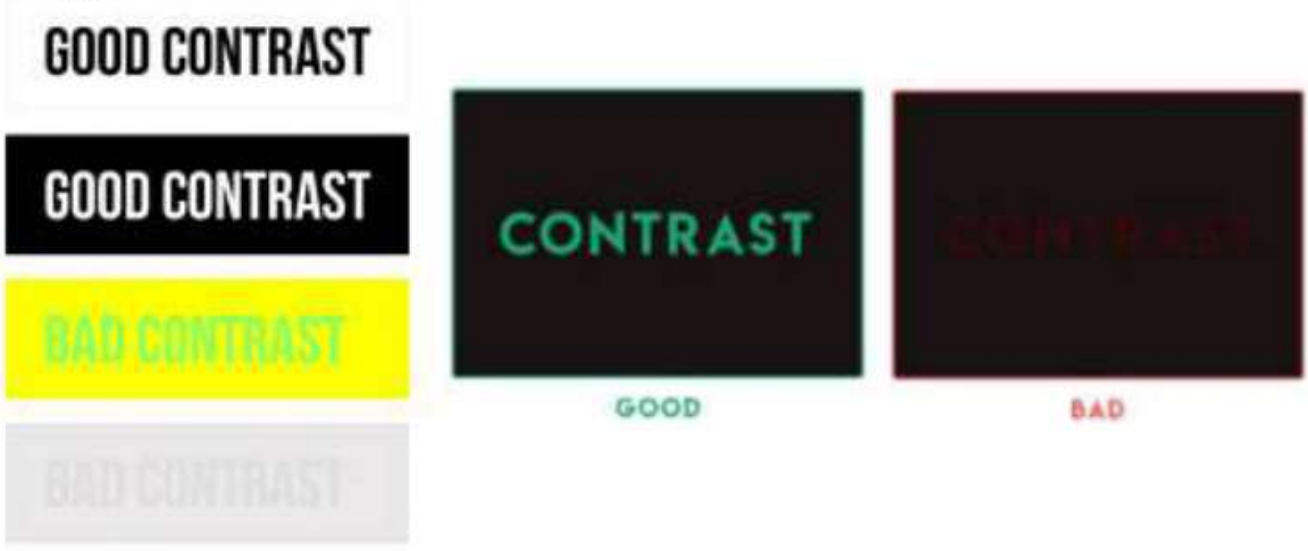
Size and Resolution:

The size and resolution of web graphics are crucial for optimal performance and user experience. Images should be optimized for the web to ensure fast loading times without sacrificing quality.

- **Size:** Keep file sizes manageable to improve loading times. Use compressed formats like JPEG or WebP for photographs and PNG for images with transparency.
- **Resolution:** For screen display, use 72 DPI (dots per inch). Higher resolutions are unnecessary and increase file size without improving quality.

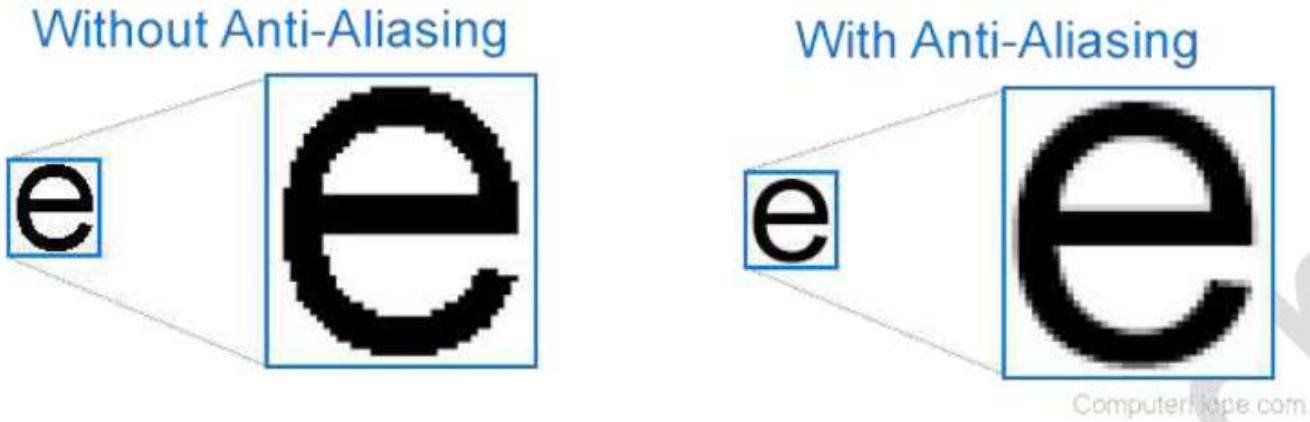
Background:

The choice of background for a website can significantly impact its aesthetics and readability. Designers should consider factors such as contrast, readability of text overlaid on the background, and overall visual harmony with the website's theme. Avoid distracting or overly busy backgrounds that compete with the main content.



Anti-aliasing:

Anti-aliasing is a technique used to smooth the jagged edges of graphics, particularly when resizing or displaying at non-native resolutions. It helps to create a more polished and visually appealing appearance. Most modern web browsers support anti-aliasing. It can be especially important for text and icons displayed at small sizes. However, excessive anti-aliasing can slightly blur the image and increase file size.



5.3 Type, purposes and features of graphics packages

Types:

Graphics software programs can be broadly categorized into two main types:

1. **Pixel-Based Image Editors (Raster Graphics):**
   - These editors manipulate images using a grid of small dots called pixels.
   - Common examples include Adobe Photoshop, GIMP, and Corel PaintShop Pro.

- **Features:** Editing, retouching, and enhancing digital photos; creating web graphics; adjusting brightness, contrast, and color levels; applying filters; and more.

2. **Path-Based Image Editors (Vector Graphics):**

- These editors use mathematical commands to create and modify images.
- Vector graphics are resolution-independent and can be scaled without loss of quality.
- Examples include Adobe Illustrator and Inkscape.
- **Features:** Creating logos, illustrations, diagrams, and scalable graphics; precise control over shapes and lines; and more.

Purpose:

- **Editing and Sharing Digital Photos**:
  - Adjusting exposure, color balance, and removing imperfections.
- **Creating Logos and Branding**:
  - Designing unique logos for businesses and organizations.
- **Drawing and Modifying Clip Art**:
  - Customizing existing clip art or creating new ones.
- **Digital Fine Art**:
  - Creating original artwork using digital tools.

- **Web Graphics**:
  - Designing banners, buttons, and other web elements.
- **Advertisements and Product Packaging**:
  - Crafting visually appealing ads and packaging designs.
- **Touching Up Scanned Photos**:
  - Enhancing old or damaged photographs.
- **Creating Maps and Diagrams**:
  - Designing maps, flowcharts, and technical diagrams

Features:

- Cropping, Resizing, and Rotating Images.
- Adjusting Brightness, Contrast, and Color Levels.
- Removing blemishes or unwanted elements.
- Applying Filters or Effects.
- Creating and editing vector paths (for vector graphics).
- Managing layers and blending modes.
- Exporting images in various formats.

5.4 Examples of graphics packages and libraries

Graphic packages are standalone applications that provide a user-friendly interface for creating and editing graphics. They typically are for wide range of users, from beginners to professionals. Here are some examples:

- **Adobe Photoshop(raster graphics editor):** The industry standard for photo editing and manipulation. Offers a vast array of tools for image enhancement, retouching, and creative effects.

- **GIMP(raster graphics editor):** A free and open-source alternative to Photoshop with a wide range of features for image manipulation, photo editing, and digital painting.

- **Inkscape(vector graphics editor):** A free and open-source vector graphics editor for creating scalable illustrations, logos, and icons.

- **CorelDraw(vector graphics editor):** A commercial vector graphics editor known for its design, illustration, and page layout capabilities.

Graphics Libraries:

Graphics libraries are collections of code that provide programmers with tools for creating and manipulating graphics within software applications. They offer more control and flexibility than graphic packages but require

programming knowledge to use. Here are some examples:

- **OpenGL(cross-language graphics API):** An industry-standard cross-language graphics API for rendering 2D and 3D graphics. Used in games, simulations, and other graphics-intensive applications.

- **Direct3D(Microsoft graphics API):** A proprietary graphics API from Microsoft used for rendering 2D and 3D graphics in games and other applications on Windows and Xbox platforms.

SDL(Simple DirectMedia Layer): A cross-platform multimedia library for creating games, multimedia players, and other graphics-intensive applications.

Unit:6 – Virtual Reality:

6.1 Introduction

Virtual reality is a simulated 3D environment that enables users to explore and interact with a virtual surrounding in a way that approximates reality, as it is perceived through the users' senses. The environment is created with computer hardware and software, although users might also need to wear devices such as helmets or goggles to interact with the environment. The more deeply users can immerse themselves in a VR environment -- and block out their physical surroundings -- the more they are able to suspend their belief and accept it as real, even if it is fantastical in nature.

6.2 Types of Virtual Reality

VR systems can vary significantly from one to the next, depending on their purpose and the technology used, although they generally fall into one of the following three categories:

6.2.1 Non-immersive VR

A non-immersive VR is fashioned to impart a computer-generated environment where the user can control activities without direct interaction. The type of VR is commonly used in everyday life, and it completely relies on

a computer or a video game console to build an environment. One common example of a non-immersive VR would be a video game where you can control the character without direct interaction.

Non-immersive virtual reality is a type of VR in which you interact with a virtual environment, usually through a computer, where you can control some characters or activities within the experience, but the virtual environment is not directly interacting with you.

6.2.2 Semi immersive VR

A semi-immersive VR allows a virtual tour while connecting to the physical surroundings. The technology is somewhere in between non-immersive and fully-immersive VR. With

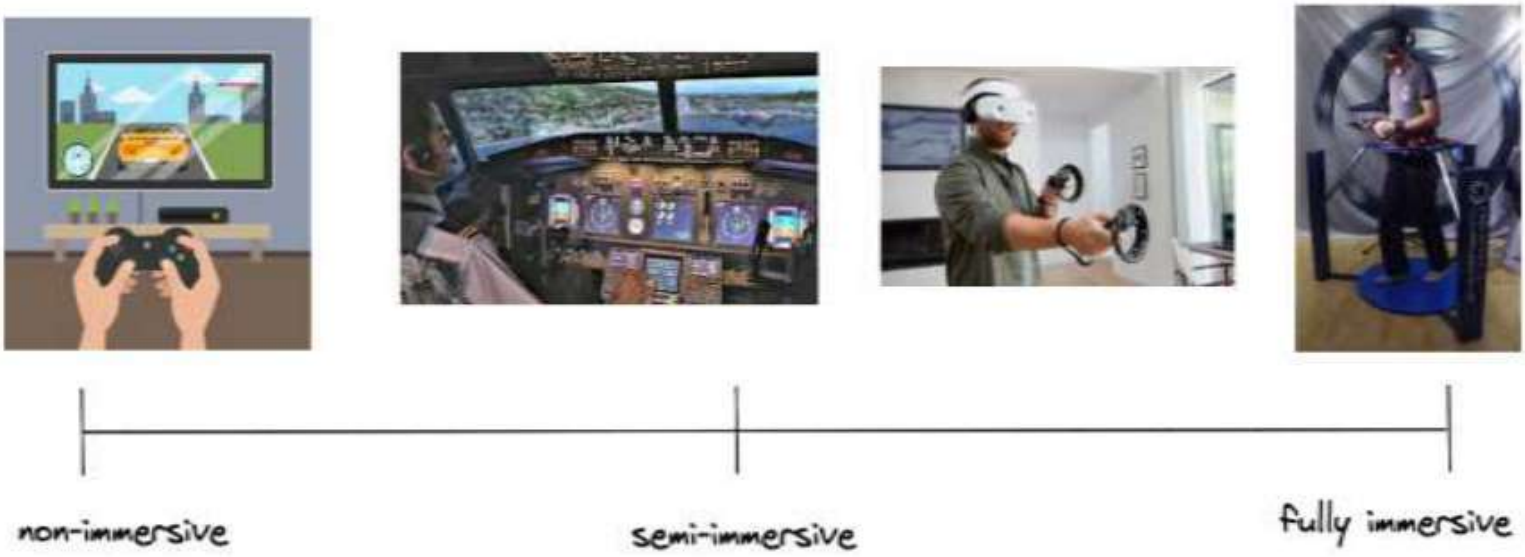
the help of VR glasses, the user can experience a virtual environment without any physical sensation. It simply means a semi-immersive VR allows the user to be in a different reality while being connected to their physical surroundings. The technology depends on high-resolution and powerful simulators to impart realism. This type of virtual reality is often used for education or training purposes.

6.2.3 Fully immersive VR

A fully immersive VR has incorporated technological advancements for a complete virtual tour from sound to sight. This type of virtual reality is completely confined and away from the physical surroundings. It is commonly adapted for gaming and entertainment. In a fully immersive VR, the user would feel physically present in the virtual world and would experience events like they are first-hand. The artificial environment is created with special equipment

like VR glasses, body detectors, gloves, and sense detectors. The display creates a stereoscopic 3D effect for a believable experience.

A common example of a fully immersive VR would be a virtual gaming zone which involves the players interacting with the virtual environment and playing against each other. The breakthrough technology stimulates as many senses as possible to provide a true-to-life experience.



6.2.4 Augmented Virtual Reality

Augmented Reality (AR) seamlessly integrates digital content into the real world, offering users enhanced experiences through devices like smartphones, tablets, or AR glasses. By overlaying computer-generated information onto physical environments, AR enables interactive and immersive encounters. This technology, leveraging real-time tracking and display capabilities, finds applications across diverse fields. In gaming, AR transforms surroundings into playgrounds for virtual characters and gameplay elements, engaging users in dynamic experiences. Retail utilizes AR for virtual try-on, enabling customers to visualize products in their own environment before purchase.

6.2.5 Collaborative Virtual Reality

Collaborative virtual reality (VR) refers to a technology that allows multiple users to interact with each other in a shared virtual environment. This can be achieved through various VR platforms and applications that enable users to communicate, collaborate, and engage with each other in real-time within the virtual space. For example, video game called PUBG (Players Unknown Battle-Ground), where tons of players come into existence as individual virtual characters that they can control.

6.3 Applications of Virtual Reality

Virtual reality (VR) technology has a wide range of applications across various industries. Here are some examples:

1. **Gaming:** This is one of the most well-known applications of VR. VR gaming immerses players into virtual worlds where they can interact with the environment and experience gameplay in a highly immersive way. Examples are Beat Saber, Half-Life: Alyx.
2. **Training and Simulation:** VR is extensively used for training purposes in fields such as military, aviation, medicine, and heavy industry. For instance, flight simulators allow pilots to practice flying in different conditions without the risk of real-world accidents, also medical students can simulate surgeries.
3. **Education:** VR offers immersive learning experiences that can enhance understanding of complex subjects. Students can explore historical events or travel to distant planets. Platforms like Google Expeditions provide virtual field trips to places around the world.
4. **Healthcare:** VR is being used in various healthcare applications, including physical therapy, and treating phobias. It helps individuals overcome fears through exposure therapy.
5. **Entertainment and Media:** Beyond gaming, VR is used in entertainment and media for immersive

experiences such as virtual concerts, 360-degree videos, and virtual theme park rides.

6. **Architecture and Design:** Architects and designers use VR to create immersive walkthroughs of buildings, allowing clients to experience spaces before they are built.