MSc-IT

COMPUTER GRAPHICS AND MULTIMEDIA

GAUHATI UNIVERSITY
Institute of Distance and Open Learning

Semester- II

MSc-IT
Paper: INF 2046

Computer Graphics and Multimedia

www.idolgu.in

M.Sc.-IT-19-II-2046

GAUHATI UNIVERSITY
Institute of Distance and Open Learning

Second Semester

(under CBCS)

M.Sc.-IT

Paper: INF-2046
COMPUTER GRAPHICS AND MULTIMEDIA

Contents:

Contributors:

Dr. Kshirod Sarmah (Block I : Unit- 1)
Asstt. Prof., PDUAM, Goalpara

Mr. Rinku Basumatary (Block I: Units- 2 & 5)
Asstt. Prof., Bodoland University

Mr. Debojit Deori (Block I: Unit- 3)
Asstt. Prof., Sibsagar Commerce College, Sivasagar, Assam

Mr. Dipankar Dutta (Block I: Unit- 4, Block II: Unit 1)
Asstt. Prof., NERIM, Guwahati

Mrs. Jonalee Barman Kakati (Block II: Unit- 2)
Asstt. Prof., NERIM, Guwahati

Dr. Atowar ul Islam (Block II: Unit- 3, Block III: Unit- 1)
Associate Prof. (Part Time)
Dept. of CSE, USTM

Mr. Brajen Kumar Deka (Block II: Unit- 4)
Asstt. Prof., NERIM, Guwahati

Mr. Mridul Jyoti Roy (Block II: Unit- 5)
Asstt. Prof., Deptt. of
Computer Science and Engineering
Assam Engineering College, Guwahati

Dr. Swapnanil Gogoi (Block III: Unit- 3)
Asstt. Prof., GUIDOL

Mrs. Manasi Hazarika (Block III: Unit- 2)
Asstt. Prof., Assam Don Bosco University, Azara, Guwahati

Mrs. Jonalee Barman Kakati (Block III: Unit- 3)
Asstt. Prof., NERIM, Guwahati

Mrs. Jonalee Barman Kakati (Block III: Units- 3 & 4)
Asstt. Prof., NERIM, Guwahati

Ms. Mala Ahmed (Block III: Units- 5 & 6)
Asstt. Prof., GIMT, Azara

Mrs. Pinky Saikia Dutta (Block III: Unit- 7)
Asstt. Prof., GIMT, Azara

Content Editor:

Dr. Utpal Barman
Associate Prof., Faculty of Engineering and Technology
Kaziranga University, Jorhat

Course Coordination:

Prof. Dandadhar Sarma Director, IDOL, Gauhati University

Prof. Anjana Kakoti Mahanta Prof., Dept. Computer Science, G.U.

Cover Page Designing:

Bhaskar Jyoti Goswami IDOL, Gauhati University

May, 2022

Published on behalf of Institute of Distance and Open Learning, Gauhati University by the Director, and printed at Gauhati University Press, Guwahati-781014.

BLOCK I:
INTRODUCTION AND OVERVIEW OF GRAPHICS SYSTEMS AND OUTPUT PRIMITIVES

UNIT 1: INTRODUCTION TO COMPUTER GRAPHICS

Unit Structure:

Space for learners:

1.1 INTRODUCTION

Computer Graphics is the use of computers to create and manipulate pictures on a display device. It comprises software techniques to create, store, modify, and represent pictures.

Computer Graphics involves technology to access. The Process transforms and presents information in a visual form. The role of computer graphics is insensible. In today's life, computer graphics has now become a common element in user interfaces, T.V. commercial motion pictures.

Computer Graphics is the creation of pictures with the help of a computer. The end product of computer graphics is a picture it may be a business graph, drawing, and engineering.

In computer graphics, two or three-dimensional pictures can be created that are used for research. Many hardware devices algorithm has been developed for improving the speed of picture generation with time. It includes the creation and storage of models and images of objects. These models are for various fields like engineering, mathematics, and so on.

We know that we are fond of video games and we may be good at playing them. Have you seen the game of ping-pong? It's a game played by two people with a pair of video game controllers and a home television set. You can see that when a game is switched on, a small bright spot, representing a ball, is seen bouncing to and fro across the screen.

Now each player uses his video game controller to position a paddle to bounce the ball back to his opponent. The player who hits the ball past his opponent wins a point and the one who gains 15 points wins the game. Now how did you invent this video game? This has been done with the aid of Computer Graphics. Video games represent a major use in the home of computer graphics. Computer graphics help to create and manipulate pictures with the aid of computers.

Computer graphics is concerned with all aspects of producing images using a computer. It concerns the pictorial synthesis of real or imaginary objects from their computer-based models.

Space for learners:

1.2 UNIT OBJECTIVES

After going through this unit you will be able to:

- Understand the basic concepts of computer graphics and their applications.
- Know about the different categories of computer graphics.
- Know about the Image Representation and its application
- Give the basic concept image files and their different types.
- Understand Direct Coding and Lookup Table

Space for learners:

1.3 ORIGIN OF COMPUTER GRAPHICS

Years of research and development were made to achieve the goals in the field of computer graphics. In 1950 the first computer-driven display was used to generate only simple pictures. This display made use of a cathode ray tube similar to the one used in television sets. During the 1950s interactive computer graphics made little progress because the computers of that period were so unsuited for interactive use. These computers were used to perform only lengthy calculations.

The single vent that did the most to promote interactive computer graphics as an important new field was the publication in 1962 of a brilliant thesis by Ivan E. Sutherland. His thesis, entitled 'Sketchpad: A Man-Machine Graphical Communication System proved to many readers that interactive computer graphics was a viable, useful, and exciting field of research. By the mid -1960's large computer graphics research projects were undertaken at MIT, Bell Telephone Labs, and General Motors. Thus the golden age of computer graphics began. In 1970 the research began to bear fruit.

The instant appeal of computer graphics to users of all ages has helped it to spread into many applications throughout the world.

1.4 DIFFERENT TYPES OF COMPUTER GRAPHICS

Computer Graphics can be broadly divided into two types:

1.4.1 Non-Interactive or Passive Computer Graphics

In non-interactive computer graphics, the picture is produced on the monitor, and the user does not have any control over the image, i.e., the user cannot make any change in the rendered image. One example of its Titles shown on T.V.

Non-interactive Graphics involves only one-way communication between the computer and the user, User can see the produced image, and he cannot make any change in the image.

In noninteractive computer graphics otherwise known as passive computer graphics, the observer has no control over the image. Familiar examples of this type of computer graphics include the titles shown on TV and other forms of computer art. Interactive Computer Graphics: Interactive Computer Graphics involves two-way communication between computer and user.

Here the observer is given some control over the image by providing him with an input device for example the video game controller of the ping pong game. This helps him to signal his request to the computer. The computer on receiving signals from the input device can modify the displayed picture appropriately. To the user, it appears that the picture is changing instantaneously in response to his commands. He can give a series of commands, each one generating a graphical response from the computer. In this way, he maintains a conversation, or dialogue, with the computer.

1.4.2 Interactive Computer Graphics

In interactive Computer Graphics, users have some control over the picture, i.e., the user can make any change in the produced image. One example of it is the ping-pong game.

Interactive Computer Graphics require two-way communication between the computer and the user. A User can see the image and make any change by sending his command with an input device.

Interactive computer graphics affects our lives in several indirect ways. For example, it helps to train the pilots of our airplanes. We can create a flight simulator that may help the pilots to get trained not in a real aircraft but on the grounds at the control of the flight simulator.

The flight simulator is a mock-up of an aircraft flight deck, containing all the usual controls and surrounded by screens on which we have the projected computer-generated views of the

Space for learners:

terrain visible on take-off and landing. Flight simulators have many advantages over real aircraft for training purposes, including fuel savings, safety, and the ability to familiarize the trainee with a large number of the world's airports.

Advantages:

1. Higher Quality
2. More precise results or products
3. Greater Productivity
4. Lower analysis and design cost
5. Significantly enhances our ability to understand data and perceive trends.

Space for learners:

1.4.2.1 Working of Interactive Computer Graphics

The **modern graphics display** is very simple in construction. It consists of three components:

1. Frame Buffer or Digital Memory
2. A Monitor likes a home T.V. set without the tuning and receiving electronics.
3. Display Controller or Video Controller that passes the contents of the frame buffer to the monitor.

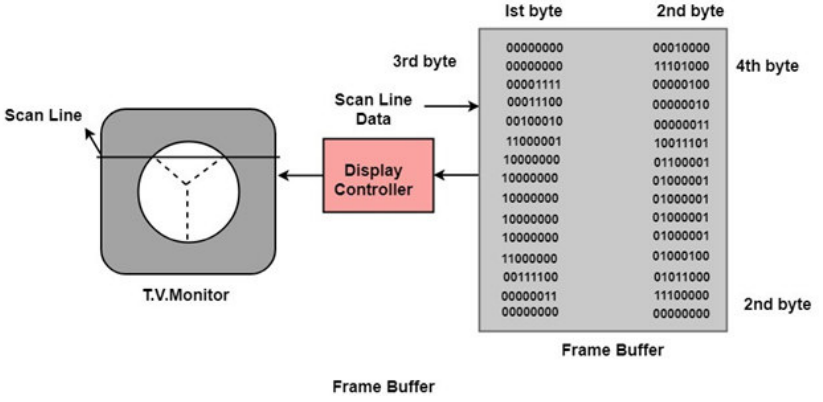


Fig. 1.1:Modern Graphics Display with Frame Buffer

Frame Buffer: A digital frame buffer is a large, contiguous piece of computer memory used to hold or map the image displayed on the screen.

- At a minimum, there is 1 memory bit for each pixel in the raster. This amount of memory is called a bit plane.
- A 1024 x 1024 element requires 2^{20} (2^{10}=1024;2^{20}=1024 x 1024) sq.raster or 1,048,576 memory bits in a single bit plane.
- The picture is built up in the frame buffer one bit at a time.
- A memory bit has only two states (binary 0 or 1), and a single bit plane yields a black and white (monochrome display).
- As frame buffer is a digital device write raster CRT is an analog device.

1.4.2.2 Properties of Video Monitor

- **Persistence:** Persistence is the duration of phosphorescence. Different kinds of phosphors are available for use in CRT. Besides color, a major difference between phosphors in their persistence is how they continue to emit light after the electron beam is removed.
- **Resolution:** Use to describe the number of pixels that are used on display images.
- **Aspect Ratio:** It is the ratio of width to its height. Its measure is a unit in length or number of pixels.
- Aspect Ratio = width unit/height unit

1.5 APPLICATIONS AND USES OF COMPUTER GRAPHICS

Computer graphics is useful in almost all part of our life. In the following sections, we are discussing some of the popular areas of computer graphics.

1.5.1 Design and Drawing

In almost all areas of engineering, be it civil, mechanical, electronic, etc., drawings are of prime importance. Drawing is said to be the language of engineers. The ability of computers to store

Space for learners:

complex drawings and display them on demand was one of the major attractions for using computers in graphic mode. However, these were further advantages.

Most of these drawings were the result of engineering calculations. Programs can be written to make these calculations and the results can be used to draw diagrams on the screen. If changes are to be made, one can get back to the design formulae and so on. Thus, the art of design and drawing was one of the earliest and most useful applications of graphics.

Space for learners:

1.5.2 Animation

But what brought the computers pretty close to the average individual is the concept of animation moving pictures. It is the well-known principle of moving pictures that a succession of related pictures, when flashed with sufficient speed will make the succession of pictures appears to be moving. In movies, a sequence of such pictures is shot and is displayed with sufficient speed to make them appear moving. Computers can do it in another way.

The properties of the picture can be modified at a fairly fast rate to make it appear moving. For example, if a hand is to be moved, say, the successive positions of the hand at different periods can be computed and pictures showing the position of the hand at these positions can be flashed on the screen. This led to the concept of "animation" or moving pictures. In the initial stages, animation was mainly used in computer games.

However, this led to a host of other possibilities. As we see later on in this course, computers not only allow you to display the figures but also offer you facilities to manipulate them in various ways, you can enlarge, reduce, rotate, twist, morph (make one picture gradually change to another – like an advertisement showing a cheetah change into a motorbike) and do a whole lot of other things.

Thus, a whole lot of films made use of computers to generate tricks. Several advertisement films and carton strips are built with no actors at all – only the computer-generated pictures.

1.5.3 Multimedia Applications

The use of sound cards to make computers produce sound effects led to other uses of graphics. The concept of virtual reality, where one can be taken through an unreal experience, like going through an unbuilt house (to see how it feels inside, once it is built) is possible by the use of computer graphics technology.

The ability of computers to convert electronic signals (0 & 1) to data and then on to figures and pictures have made it possible for us to get photographs of distant planets like mars being reproduced here on the earth in almost real-time.

Space for learners:

1.5.4 Simulation

The other revolutionary change that graphics made was in the area of simulation. Simulation is a mockup of an environment elsewhere to study or experience. The availability of easily interactive devices (mouse is one of them, we are going to see a few others later in the course) made it possible to build simulators.

One example is flight simulators, wherein the trainee, sitting in front of a computer, can operate on the interactive devices as if he were operating on the flight controls and the changes he is expected to see outside his window are made to appear on the scree so that he can master the skills of flight operations before actually trying his hand on the actual flights.

The graphic capabilities of computers are used in a very large variety of areas like criminology (to recreate faces of victims, andand assailants etc.,) medical fields (recreating pictures of internal cavities, using signals sent by miniature cameras), a a recreation of Satellite pictures,, etc.

1.5.5 Education and Training

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

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

1.5.6 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.

1.5.7 Use in Biology

A molecular biologists can display a picture of molecules and gain insight into their structure with the help of computer graphics.

1.5.8 Computer-Generated Maps

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

1.5.9 Architect

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

1.5.10 Presentation Graphics

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

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

Space for learners:

1.5.11 Computer Art

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

1.5.12 Entertainment

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

1.5.13 Visualization

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

1.5.14 Educational Software

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

1.5.15 Printing Technology

Computer Graphics is used for printing technology and textile design.

1.6 OVERVIEW OF IMAGE REPRESENTATION

In computer science, we can represent an image in various forms. Most of the time, it refers to the way that brings information, such as color is coded digitally, and how the image is stored, i.e., how an image file is structured. Several open standards were recommended to create, manipulate, store, and exchange digital images. The rules described the format of image files, the algorithms of image encoding, and the form of additional information often named metadata.

A digital image is the composition of individual pixels or picture elements. The pixels are arranged in the form of rows and columns

Space for learners:

to form a picture area. The number of pixels in an image is a function of the size of the image and the number of pixels per unit length (e.g., inch) in horizontal as well as vertical directions.

Space for learners:

1.6.1 Image Processing

It is a method to implement some operations on an image. It is also used to get an enhanced image or to access some useful information from an image. It is a type of processing in which the input is an image, and the output may be the image or characteristics/features correlated with that image.

For example– photographs, and frames of video.

Most image processing techniques consider the image to a two-dimensional and apply standard signal-processing techniques to it.

Pixel: Pixel is the smallest unit of a picture displayed on the computer screen. A pixel includes its Intensity and Name or Address. The size of the image is defined as the total number of pixels in the horizontal direction times the total number of pixels in the vertical direction (512 x 512,640 x 480, or 1024 x 768).

The ratio of an image's width to its height, we can measure it in unit length or number of pixels, is known as the **aspect ratio** of the image.

For example– A 2 x 2inch image and a 512 x 512 image have an aspect ratio of 1/1, whereas a 6 x 4inch image and a 1024 x 768 image have an aspect ratio of 4/3.

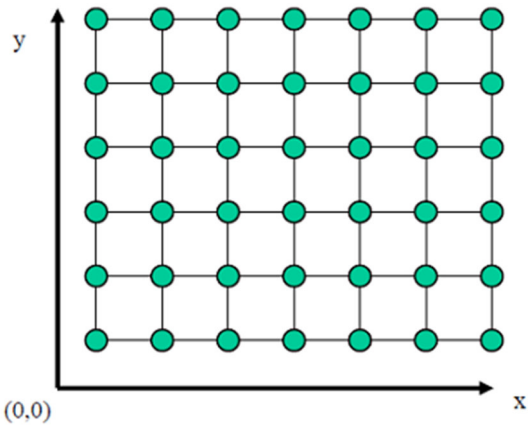


Fig.1.2: Representation of pixels

Resolution: It is the number of separate pixels displayed on a screen expressed in terms of pixels on the horizontal axis and vertical axis. The sharpness of the picture on display depends on the resolution and the size of the monitor. The number of pixels per unit is called the resolution of the image.
It also includes-
Image Resolution: It is defined as the distance between two pixels.
Screen Resolution: It is defined as the number of horizontal and vertical pixels displayed on the screen is called Screen Resolution."

For Example– 640 x 480, 1024 x 768 (Horizontal x Vertical)

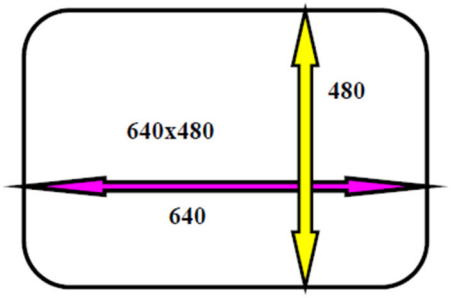


Fig.1.3: Block diagram of screen resolution 640x480.

Aspect Ratio: "The ratio of an image's width to its height is known as the aspect ratio of an image**.**" The height and width of an image are measured in length or number of pixels.
For Example: If a graphic has an aspect ratio of 2:1, it means the width is twice large as the height.
It includes–
Frame aspect ratio: Horizontal /Vertical Size
Pixel aspect ratio: Width of Pixel/Height of Pixel

1.6.2 Applications of Image Processing

Some application areas of Image Processing are as follows:

- Computerized Photography
- Space Image Processing (e.g., Hubble space telescope image, Interplanetary probe images)
- Medical/ Biological Image Processing
- Automatic Character Recognition
- Fingerprint/ Face/ Iris Recognition

Space for learners:

- Remote sensing
- Industrial application

Space for learners:

1.7 IMAGE FILES

In computer science, the representation of an image can take many forms. Most of the time, it refers to the way that the conveyed information, such as color, is coded digitally and how the image is stored, i.e., how is structured an image file. A file that contains graphics data for example a GIF or PNG file is known as an image file. It is a file that is copied to a hard disk, CD, DVD, or BD bit for bit. An image is a virtual representation of something. An image is a picture that has been created or copied and stored in electronic form which can be described in terms of vector graphics or raster graphics. All digital image files fall into one of two categories: vector or raster.

1.7.1 Format of Image Files

There are some different types of images which are mentioned as:

- **JPEG (Joint Photographic Experts Group):** It is used for digital images, especially for those images which are composed of digital photography. The '.jpeg' filename extension is used to save the file.
- **PNG (Portable Network Graphics):** These files are commonly used to store graphics for web images. PNG was developed to enhance the non-registered replacement for Graphics Interchange Format. The '.png' filename extension is used to save the file.
- **GIF (Graphics Interchange Format):** It is a file format for storing graphical images up to 256 colors. PNG is based on a lossless compression method, which makes higher quality output. PNG was created as a more powerful option for the GIF file format. The '.gif' filename extension is used to save the file.
- **TIFF/ TIF (Tagged Image File):** These files can be saved in a collection of color formats and many forms of compression. TIFF file is used to maintain image integrity and clarity. It is often used for professional photography. The '.tif' filename extension is used to save the file.

- **PSD (Photoshop Document):** It is a layered image file used in Adobe PhotoShop. It is a default format that is used by PhotoShop for saving data. PSD is a custody file that allows the user to work with the images' separate layers even after the file has been saved. The '.psd' filename extension is used to save the file.
- **PDF (Portable Document Format):** It is used to share the documents between computers and across operating system platforms when the user needs to save files that cannot be altered. The '.pdf' filename extension is used to save the file.
- **EPS (Encapsulated Postscript):** It is a graphics file format, which is used in vector-based images. In Windows, the user needs graphics software to open the EPS file (i.e., Adobe Illustrator, Coral Draw). The '.eps' filename extension is used to save the file.
- **AI (Adobe Illustrator Document):** It is a file format developed by the Adobe system. It is used to represent single-page vector-based drawings in EPS or PDF formats. The '.ai' filename extension is used to save the file.

Space for learners:

1.8 COLOR MODEL

The color model is a specification of a color coordinate system and the subset of visible colors in this coordinate system. Conversion formulas between the color models have to be given.

The RGB color model is an additive color model in which the red, green, and blue primary colors of light are added together in various ways to reproduce a broad array of colors. The name of the model comes from the initials of the three additive primary colors, red, green, and blue. It is the process of mixing 3 primary colors, red, green, and blue, together in different proportions to make more different colors. In the RGB color model, the combination of primary colors creates different colors that we perceive by stimulating the different cone cells simultaneously.

The main purpose of the RGB color model is for the sensing, representation, and display of images in electronic systems, such as televisions and computers, though it has also been used in conventional photography. Before the electronic age, the RGB color model already had a solid theory behind it, based on human perception of colors.

RGB is a device-dependent color model: different devices detect or reproduce a given RGB value differently, since the color elements (such as phosphors or dyes) and their response to the individual red, green, and blue levels vary from manufacturer to manufacturer, or even in the same device over time. Thus an RGB value does not define the same color across devices without some kind of color management.

Typical RGB input devices are color TV and video cameras, image scanners, and digital cameras. Typical RGB output devices are TV sets of various technologies (CRT, LCD, plasma, OLED, quantum dots, etc.), computer and mobile phone displays, video projectors, multicolor LED displays, and large screens such as the Jumbotron. Color printers, on the other hand, are not RGB devices, but subtractive color devices

The RGB color model is one of the most widely used color representation methods in computer graphics. It uses a color coordinate system with three primary colors:

R(red), G(green), B(blue)

Each primary color can take an intensity value ranging from 0(lowest) to 1(highest). Mixing these three primary colors at different intensity levels produces a variety of colors. The collection of all the colors obtained by such a linear combination of red, green, and blue forms the cube-shaped RGB color space.

The corner of the RGB color cube that is at the origin of the coordinate system corresponds to black, whereas the corner of the cube that is diagonally opposite to the origin represents white. The diagonal line connecting black and white corresponds to all the gray colors between black and white, which is also known as the gray axis.
In the RGB color model, an arbitrary color within the cubic color space can be specified by its color coordinates: (r,g.b).

Space for learners:

1.9 DIRECT CODING

In Computer Graphics, Direct coding is an algorithm that provides some amount of storage space for each pixel so that pixel is coded with color. Image representation is essentially the representation of

pixel colors. Using direct coding we allocate a certain amount of storage space for each pixel to code its color.

For example, we may allocate 3 bits for each pixel, with one bit for each primary color. This 3-bit representation allows each primary to vary independently between two intensity levels: 0 (off) or 1 (on). Hence each pixel can take on one of the eight colors that correspond to the corners of the RGB color cube.

Bit 1: r	Bit 2: g	Bit 3: b	Color name
0	0	0	black
0	0	1	blue
0	1	0	green
0	1	1	cyan
1	0	0	red
1	0	1	magenta
1	1	0	yellow
1	1	1	white

Fig 1.4: Direct coding of color using 3 bits.

A widely accepted industry standard uses 3 bytes, or 24 bits, per pixel, with one byte for each primary color. This way we allow each primary color to have 256 different intensity levels, corresponding to binary values from 00000000 to 11111111. Thus a pixel can take on color from 256 x 256 x 256 or 16.7 million possible choices.

The 24-bit format is commonly referred to as the true color representation, for the difference between two colors that differ by one intensity level in one or more of the primaries is virtually undetectable under normal viewing conditions. Hence a more precise representation involving more bits is of little use in terms of perceived color accuracy.

A notable special case of direct coding is the representation of black-and-white (bilevel) and
gray-scale images, where the three primaries have the same value and hence need not be coded separately.

A black-and-white image requires only one bit per pixel, with bit values 0 representing black and 1 representing white.

A gray-scale image is typically coded with 8 bits per pixel to allow a total of 256 intensity or gray levels.

Space for learners:

Although this direct coding method features simplicity and has supported a variety of applications, we can see a relatively high demand for storage space when it comes to the 24-bit standard.

For example, a 1000 x 1000 true-color image would take up to three million bytes. Furthermore, even if every pixel in that image had a different color, there would only be one million colors in the image.

In many applications, the number of colors that appear in any one particular image is much less. Therefore the 24-bit representation's ability to have 16.7 million different colors appear simultaneously in a single image seems to somewhat overkill.

Space for learners:

1.10 LOOKUP TABLE

A lookup table (LUT) is an array of numbers that may be referred to by subscript. In computer graphics, lookup tables are used to store the starting addresses of each line and the values corresponding to the placement of pixels within a byte. This avoids the recalculation of values each time a number in a table must be referred to. In computer graphics, lookup tables are used to store the starting addresses of each line and the values corresponding to the placement of pixels within a byte.

Professionals working with video or image-editing software may need to calibrate their monitors to achieve the high level of color accuracy their work requires. Often, the graphics card needs to have more than one LUT to calibrate more than one monitor at a time. As color calibration hardware and software become more mainstream, we recognize the importance of having this information available and easy to find. Intel is currently working on making lookup table information available.

Image representation using a lookup table can be viewed as a compromise between our desire to have a lower storage requirement and our need to support a reasonably sufficient number of simultaneous colors. In this approach pixel values do not code colors directly. Instead, they are addresses or indices in a table of color values. The color of a particular pixel is determined by the color value in the table entry that the value of the pixel references.

CHECK YOUR PROGRESS

1. Computer Graphics is the use of computers to

(a) create pictures (b) producing images

(b) manipulate pictures (d) all of these.

2. In non-interactive computer graphics which of the following is true

(a) the picture is produced on the monitor

(b) the user does not have any control over the image

(c)both (a) and (b)

(d) none of the above.

3. Interactive Computer Graphics require

(a) one-way communication between the computer and the user.

(b)two-way communication between the computer and the user.

(c)multiple-way communication between the computer and the user.

(d) none of the above.

4. Aspect Ratio is

(a) the ratio of width to its height.

(b) the ratio of height to its width.

(c)both (a) and (b)

(d) none of the above.

5. The smallest unit of a picture displayed on the computer screen is known as

(a) point (b) line (c) pixel

(d) buffer

Space for learners:

1.11 SUMMING UP

- **Computer Graphics** is the use of computers to create and manipulate pictures on a display device. It comprises software techniques to create, store, modify, and represent pictures. Computer Graphics is the creation of pictures with the help of a

computer. The end product of computer graphics is a picture it may be a business graph, drawing, and engineering.

- **In non-interactive computer graphics**, the picture is produced on the monitor, and the user does not have any control over the image, i.e., the user cannot make any change in the rendered image.
- **In interactive Computer Graphics,** users have some control over the picture, i.e., the user can make any change in the produced image. Interactive Computer Graphics require two-way communication between the computer and the user. A User can see the image and make any change by sending his command with an input device.
- In computer science, the representation of an image can take many forms. Most of the time, it refers to the way that the conveyed information, such as color, is coded digitally and how the image is stored, i.e., how is structured an image file. A file that contains graphics data for example a GIF or PNG file is known as an image file. It is a file that is copied to a hard disk, CD, DVD, or BD bit for bit.
- An image is a virtual representation of something. An image is a picture that has been created or copied and stored in electronic form which can be described in terms of vector graphics or raster graphics. All digital image files fall into one of two categories**: vector or raster.**
- The **modern graphics display** is very simple in construction. It consists of three components: (i) a Frame Buffer or Digital Memory, (ii) A Monitor like a home T.V. set without the tuning and receiving electronics, and (iii) a Display Controller or Video Controller that passes the contents of the frame buffer to the monitor.
- Applications and uses of computer Graphics in different areas are Design and Drawing Animation, Multimediaapplications, Simulation, Education and Training, Flight Simulator, Use in Biology, Computer-Generated Maps, Architect, Presentation Graphics, Computer Art, Entertainment, Visualization, Educational Software, Printing Technology
- **Pixel** is the smallest unit of a picture displayed on the computer screen. A pixel includes its Intensity and Name or Address.

Space for learners:

- **Resolution** is the number of separate pixels displayed on a screen expressed in terms of pixels on the horizontal axis and vertical axis.
- **Aspect Ratio is** the ratio of image's width to its height is known as the aspect ratio of an image**.**" The height and width of an image are measured in length or number of pixels.
- **The color model** is a specification of a color coordinate system and the subset of visible colors in this coordinate system. Conversion formulas between the color models have to be given.
- **The RGB color model** is an additive color model in which the red, green, and blue primary colors of light are added together in various ways to reproduce a broad array of colors. The name of the model comes from the initials of the three additive primary colors, red, green, and blue. It is the process of mixing 3 primary colors, red, green, and blue, together in different proportions to make more different colors. In the RGB color model, the combination of primary colors creates different colors that we perceive by stimulating the different cone cells simultaneously.
- **Direct coding** is an algorithm that provides some amount of storage space for each pixel so that pixel is coded with color. Image representation is essentially the representation of pixel colors. Using direct coding we allocate a certain amount of storage space for each pixel to code its color.
- **A lookup table (LUT)** is an array of numbers that may be referred to by subscript. In computer graphics, lookup tables are used to store the starting addresses of each line and the values corresponding to the placement of pixels within a byte.

Space for learners:

1.12 ANSWER TO CHECK YOUR PROGRESS

Answer:

1(d), 2(c), 3(b), 4(a), 5(c).

1.13 MODEL QUESTIONS

1. What do you mean by Computer Graphics? Explain its different types.

2. Describe briefly the various applications of computer graphics.
3. Explain the working principle of The RGB color model.
4. What are image files? What are its different types?
5. What do you mean by Direct coding and Lookup Table in Computer Graphics?

Space for learners:

1.14 REFERENCES AND SUGGESTED READINGS

- Computer Graphics C version, 2nd Edition, Pearson by Donald D. Hearn and M.P. Baker.

UNIT 2: GRAPHIC I/O DEVICES

Unit Structure:

Space for learners:

2.1 INTRODUCTION

Several devices are available to input and output data/information for general-purpose computers with graphical facilities and sophisticated workstations designed for graphics applications. Without these devices, the users will not be able to interact with the computer. Some of the input devices are an alphanumeric keyboard, mouse, light pen, joystick, touch panel, data glove, image scanner, trackball, digitizer, and voice systems. Some of the output devices are monitors which are an integral part of the computer system and different kinds of printers and plotters. While talking about display devices, there are different types of display devices available and they use different techniques to display text, graphics, videos, etc.

Space for learners:

2.2 UNIT OBJECTIVES

At the end of this unit, you should be able to describe various types of input-output devices and their working. You shall also be able to discuss different types of display devices like CRT, DVST, plasma, LED, LCD, etc., and be able to describe different displaying techniques like raster and random scan displaying methods.

2.3 INPUT DEVICES

Keyboard: Keyboard is the most common integral part of a computer system. Keyboards are used to type texts, and documents, use the shortcut keys, access menus, and play games. Based upon the manufacturer keyboards can have different types of keys, depending on the operating system and for which they are meant desktop or laptop. Most of the keyboards have 80 to 110 keys, which are: A to Z letters, numbers 0 to 9, characters like < , + - ! @ # $ = etc, functions keys F1 to F12 control keys Ctrl Alt Delete Home Enter etc. Function keys are used to enter frequently used operations in a single keystroke and control keys are used to control the cursor and screen.

Fig 2.1: Keyboard

Fig reference:
https://www.apple.com/in/shop/product/MQ052HN/A/magic-keyboard-with-numeric-keypad-usenglish

Mouse: The mouse is also an important part of the computer system. It is a handheld pointing device, designed to be placed underhand by the user and to detect movement relative to its two-dimensional supporting surface. It has become an inseparable part of the computer system. A cursor in the shape of an arrow or crosshair is always associated with a mouse. The mouse is used to move the cursor or drag and drop something, resize some object or activate a file, folder, or menu, etc., draw and design figures, shapes, etc. in a mechanical mouse a roller (ball) assembly is used for detecting motion in X and Y direction. An optical mouse use LED and photodiodes to detect the movement of the underlying surface. The laser mouse uses a small laser. The mouse may have two or three buttons.

Fig 2.2: Mouse

Fig reference: https://www.uow.edu.au/student/learning-co-op/technology-and-software/mouse-buttons/

Space for learners:

Touch screen: A touch screen is a display device that accepts user input through a touch screen. Provide input by touching the displayed button, menu, or icon with the finger. In optical touch panels, LEDs are installed on adjacent edges (one vertical, one horizontal). The opposing pair of adjacent edges contains light detectors. These detectors will immediately recognize which two orthogonal beams emitted by the LED are blocked by a finger or other pointing device, thereby recording the x and y coordinates of the touch screen position for selection. However, due to poor resolution, touch screens cannot be used to select very small graphical objects or precise screen positions. In an electric (or capacitive) touch panel, two glass plates coated with suitable conductive and resistive materials are similarly placed face to face to capacitor plates. Touching a point on the display panel generates a force that changes the spacing between the panels. This, in turn, will result in a change in the capacitance across the board, which is converted to the coordinate value of the selected screen position. In one type of acoustic touch panel, similar to light, sound waves are generated from the horizontal and vertical edges of the screen. By placing your finger in the designed position on the screen, the sound beam will be blocked or reflected. Determine the position of the fingertip from the travel time of the beam. Touch screens are widely used in bank ATMs, electronic game consoles, rail or travel information systems, etc.

Fig 2.3 Touch Screen

Fig reference: https://www.amazon.com/Raspberry-Touchscreen-Monitor-1024x600-Speakers/dp/B07S51QDTG

Space for learners:

Voice system: The voice system or voice recognition system is a complex input device that accepts the user's voice or voice input and converts it into digital data, which can be used to trigger graphical operations or enter data into a specific field. Create a voice dictionary for a specific operator (user) by recording the frequency patterns of the voice commands (spoken words) and the corresponding functions to be performed. Later, when the same operator issues a voice command, the system searches the dictionary for a frequency pattern match and, if found, triggers the corresponding operation. If a different operator wants to use the system, the dictionary must be rebuilt using the new operator's voice mode.

Fig 2.4: Voice Recognition System

Fig reference: https://www.indiamart.com/proddetail/voice-recognition-system-10207431155.html

Joystick: The joystick is used as a personal computer peripheral or as a general control device. It consists of a handheld joystick that rotates around the base and guides the screen cursor to move. The joystick can be 2D or 3D. The 2D joystick has two axes of movement similar to that of the mouse: joystick left or right means to move along the X-axis, and up or down means to move along the Y-axis. The 3D joystick is set for 3D movements such as turning the joystick left or turning the joystick right or right representing movement along the Z-axis. In some joysticks, an optical sensor is used instead of an analog potentiometer to digitally read the movement of the joystick.

Space for learners:

Fig 2.5: Joystick

Fig reference: https://www.amazon.in/Joystick-YF2009-Controller-Vibration-Throttle/dp/B06XGBL1HL

Digitizer: Digitizer is a locator device used for drawing, painting or interactively selecting coordinate positions on an object. A graphics tablet is a kind of digitizer that consists of a flat surface upon which the user can draw an image using an attached stylus, a pen-like drawing instrument. The image doesn't appear on the tablet but is displayed on the monitor.

Fig 2.6: Digitizer

Fig reference: https://www.123rf.com/photo_3854698_working-with-digitizer.html

Trackball: A trackball is a pointing device that consists of a sphere, which is located in a receptacle that contains a sensor to detect the rotation of the sphere around two axes (in 2 dimensions). It is the same as an inverted mouse with an exposed protruding ball. The user must roll the ball with their finger or palm to move the cursor.

Space for learners:

Fig 2.7: Trackball

Fig reference: https://www.adesso.com/product/imouse-t40-adesso-2-4-ghz-wireless-4-button-desktop-trackball/

Light Pen: A lightpen is a pointing device, shaped like a pencil, connected to a computer. The tip of the stylus contains a photosensitive element (photocell), when it is resting on the screen, it will detect the light from the screen, so that the computer can recognize the position of the pen on the screen. It allows the user to point to the displayed object or draw on the screen, similar to a touch screen, but with higher positioning accuracy. The stylus can be used with any CRT-based display, but cannot be used with LCD screens, projectors, or other display devices. When the electron gun cools the spot, the light pen works by detecting small sudden changes in the brightness of the spot on the screen. By looking at the exact position the scan reached that moment, the x and y positions of the pencil can be resolved. The position of the pen is updated every time the screen is updated.

Fig 2.8: Light Pen

Space for learners:

Fig reference: https://www.c64-wiki.com/wiki/Light_pen

Data Glove: Data Glove is an interface device that uses position tracking sensors and fiber optic cables that run the length of each finger and is connected to a compatible computer; hand and finger movements are displayed on the computer monitor in real-time, allowing users to virtually touch objects displayed on the same monitor. With animated objects, the user (with data gloves) appears to be able to pick up an object and do things with it, just as with real objects. In modern data glove devices, touch sensors are used to provide users with additional tactile sensation or pressure or force exerted by the fingers or hands, even if the user is not touching anything. Therefore, Data Glove is an agent that transports users to virtual reality.

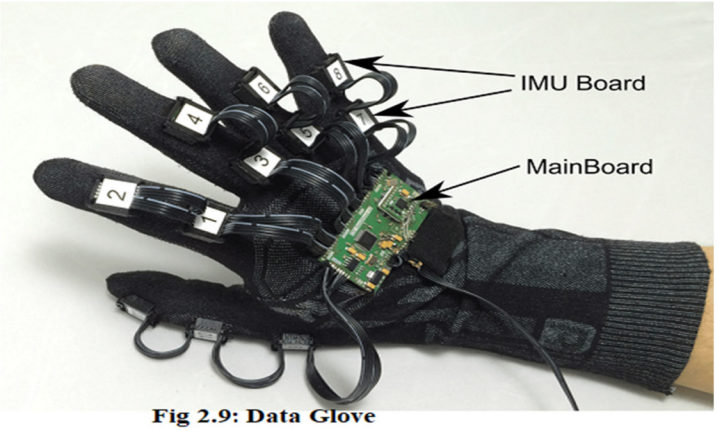


Fig 2.9: Data Glove

Fig reference: https://www.researchgate.net/figure/Fully-functional-CIE-DataGlove-with-all-IMU-sensors-attached_fig2_314135437

Scanner: Scanner is a kind of input device which gives input to the computer by taking pictures of the images, just like the digital cameras. The pictures are converted into digital form which is then sent to the computer and can be displayed on the screen. The images may be pictures of handwritten texts, hard copy text graphics, photographs, etc. There are several kinds of scanners present; flatbed scanners, hand-held scanners, and 3D scanners. In a flatbed scanner, the image is placed upon a glass window and scanned. In a handheld scanner, the device is moved by hand. 3D scanners are used for gaming and industrial design, etc.

Space for learners:

Fig 2.10: Flatbed & Handheld Scanner

Fig reference:
https://www.indiamart.com/proddetail/hp-scanjet-flatbed-scanner-15066650062.html
https://www.indiamart.com/proddetail/retsol-ls-450-barcode-scanner-9875844788.html

CHECK YOUR PROGRESS

1. Which one of the following is not a pointing device?
   a. Mouse
   b. Trackball
   c. Keypad
   d. Touchpad
2. Which one of the following devices is used to control video games, training simulators, flight simulators, control robots?
   a. Light pen
   b. Digitizer
   c. Keyboard
   d. Joystick
3. Which one of the following is a kind of scanner?
   a. Flat bed
   b. Flat led
   c. Hand led
   d. Compact

Space for learners:

2.4 OUTPUT-DEVICES AND DISPLAY DEVICES

Space for learners:

Output devices are part of computer systems that gives the output of the processed information to the users in different forms like text, pictures, signals, audio, and video. The most basic output devices are a monitor, printer, and plotter.

Printer: The printer is an important part of any computer system, especially the graphics system. Most graphics created using computer graphics are used in print. The prints can be used for documents, exhibitions, or publications in print media or books. Therefore, the quality of the print must be clear and easy to understand. Depending on the available printing technology, the main factors that control the quality of the printer are the size of the individual dots on the paper and the number of dots per inch (dpi). The smaller the point size, the better the reproduced graphic details. Higher dpi values increase the clarity and detail of graphics and increase the number of intensity levels supported by the printer. Other important factors when choosing a printer are the print speed and the print area or memory of the printer. There are several important printing technologies available. Printers that use a mechanism to form a character surface by pressing an ink ribbon onto the paper to create images are called impact printers, such as dot-matrix printers and line printers. Printers that do not touch the paper but use laser technology, inkjet, and electrostatic printing processes to create images on paper are called non-impact printers, such as laser printers, inkjet printers, and electrostatic printers, drum plotters, flatbed tracers.

Dot-matrix printer: It refers to a computer printer whose print head (usually contains 9 to 24 pins) runs back and forth on the page and prints on impact, hitting the ink-soaked fabric ribbon on the paper, like a typewriter. But apart from typewriters or daisy-wheel printers, letters are drawn from a dot matrix, so various fonts and arbitrary graphics can be generated. Since printing involves mechanical pressure, these printers can create carbon copies. The print head usually prints along each raster line of the printing paper, and the printed color is the color of the ink on the ribbon. Each point is produced by a small and strong metal rod, also known as a "wire" or "pin", which is propelled forward by the power of a small electromagnet or solenoid directly or through a

small lever (ratchet). The pins are typically arranged vertically and edge offsets are provided between the columns to reduce dot pitch. The position of the needle in the print head limits the quality of this printer. Although dot matrix printer hardware enhancements have increased transport speed, added more font options, increased dot density (from 60 dpi to 240 dpi), and added pseudo-color printing using multi-colored ribbons. These printers cannot still print computer-generated images of acceptable quality. It is suitable for printing text on continuous paper. The term "dot-matrix" is inappropriate here, because almost all inkjet, thermal, and laser printers produce dot matrices. However, in general, they are rarely called "dot-matrix" printers to avoid confusion with dot-matrix impact printers.

Fig 2.11: Dotmatrix

Fig reference: https://www.indiamart.com/proddetail/hvp-printer-dot-matrix-printer-hvp1410-22926934788.html

Line printer: A-line printer is a high-speed impact printer that prints one type of line at a time. In a typical design, the fixed font character set is engraved on the periphery of several printing wheels, and the number matches the number of columns (the letters on a line). The wheels rotate at high speed, and the paper and ribbon move out of the printing position. When the required characters of each column pass the printing position, the hammer will hit the paper and the ribbon to record the required characters on the continuous paper. The print type is set to a fixed position, and a row can contain any number of character positions, the most common is 132 columns, but the 80, 128, and 160 column types are also used. Other variants of line printers have the type of moving rods or horizontal rotating chains. Line printer technology is generally faster and cheaper (wholly owned) than laser printers. It is

Space for learners:

used for medium-volume accounting and other large enterprise applications, where print volume and speed take precedence over quality. Due to the limited set of characters engraved on the wheels and the fixed font spacing, this technology has never been used in readable materials such as books or newspapers.

Fig 2.12: Line Printer

Fig reference: https://www.indiamart.com/proddetail/computer-line-printer-9913028188.html

Inkjet printer: An inkjet printer is a non-impact printer that places very small drops of ink on paper to create an image. These printers are popular due to their lower cost and the ability to produce attractive graphic results. The dots sprayed on the paper are very small (typically 50-60 microns in diameter), and the positioning is very precise, with a resolution of up to 1440 x 720 dpi. These dots can combine different colors to create photo-quality images. The core of an inkjet printer is a printhead that contains a series of nozzles to eject ink droplets. Ink is contained in a variety of ink cartridge combinations, such as separate black and color ink cartridges, or separate ink cartridges for each ink color. The stepping motor moves the print head assembly (print head and ink cartridge) back and forth on the paper. The mechanical operation of the printer is controlled by a small circuit board that contains a microprocessor and memory. There are two main inkjet technologies currently used by printer manufacturers.

Space for learners:

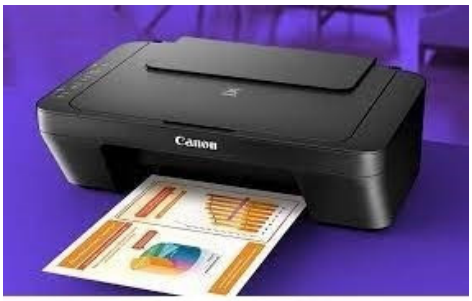


Fig 2.13: Inkjet Printer

Fig reference: https://fabrikbrands.com/best-inkjet-printer-for-mac/

Hot bubble (or bubble jet) - used by manufacturers such as Canon and Hewlett-Packard. In thermal inkjet printers, small resistors generate heat, which causes the ink to evaporate to create bubbles. As the bubble expands, some ink is sprayed from the nozzle onto the paper. When the bubble bursts, a vacuum is created. This will put more ink into the print head of the cartridge. A typical bubble jet print head has 300 or 600 micro-nozzles, all of which can jet a drop at the same time.

Fig 2.14: Thermal Printer

Fig reference: https://www.indiamart.com/proddetail/thermal-recipt-printer-21609056991.html

Piezoelectric-This technology is patented by Epson and uses piezoelectric crystals. Each nozzle has a glass on the back of the ink tank. The crystal receives a small electric charge which causes it to vibrate. When the glass vibrates inward, it ejects a small amount of ink from the nozzle. When it vibrates, it puts more ink into the container to replace the ejected ink.

Space for learners:

Space for learners:

Fig 2.15: Piezoelectric Printer

Fig reference: https://www.indiamart.com/proddetail/water-based-eco-solvent-piezoelectric-printer-9764534888.html

Laser printers: Laser printers use technology similar to photocopiers. The laser beam focuses on the positively charged selenium-coated rotating drum. The laser gun removes the positive charge from the photosensitive drum outside the area to be printed (the black part of the paper). In this way, the laser draws letters and images to print them as a pattern of charges, that is, an electrostatic image. The negatively charged black toner first adheres to the positively charged area (image) on the photosensitive drum, from where it is transferred to the rolling white paper. Before the paper roll is under the drum, it is given a stronger positive charge than the positive charge of the electrostatic image, so the paper can remove the toner dust. Then gently heat the paper to melt the loose toner and fix it on the paper. Laser printers are mainly two-layer printers. In the case of a color laser, this process is repeated 3 times. For the printer driver and the host computer to communicate, they must use the same page description language. The main printing languages these days are HP's Printer Command Language (PCL) and Adobe's Postscript. Both languages describe pages in vector form, that is, as mathematical values of geometric shapes, rather than a series of points (bitmap images).In addition to the image data, the printer driver also receives all the commands that tell the printer what to do: what paper to use, how to format the page, how to handle fonts, and so on. Therefore, the driver sets the margins of the text, sorts the words, and places any graphics. When the page is organized, the raster image processor (RIP) takes the data from the page as a whole or as a part and splits it into a matrix of small dots so that the

laser can write it onto the Photoreceptor drum. In most laser printers, the driver saves all print job data in its memory. This allows the driver to queue different print jobs so that the user can process them one at a time.

Fig 2.16: Laser Printer

Fig reference: https://en.wikipedia.org/wiki/Laser_printing

Electrostatic printer: In an inkjet printer a single print head moves from left to right and prints as it moves. In contrast, electrostatic printers have many print heads, which cover the entire width of 36"paper. Therefore, electrostatic printers print the entire width of the page at once, rather than a single print head moving across the width of the media. The medium (paper, kraft paper, film) is electrostatically charged (fed). The toner solution circulates through the media and "sticks" to the energized parts of the media, producing very fast, high-quality images. The printer creates color prints by breaking down the color data into three basic colors (cyan, magenta, and yellow) in addition to black and printing one color at a time. In 5-pass printing mode, the combination of cyan, magenta, yellow, and black provides a variety of different colors. Using the registration marks printed during the preliminary registration process ensures beautiful color tracking without misalignment.

Space for learners:

Fig 2.17: Electrostatic Printer

Fig reference: https://dir.indiamart.com/impcat/electrostatic-printing.html

Plotter: Plotters are like printers that accept the user commands from the computer and transform it into make drawing on a paper. A plotter may have one or more pens, on the color plotters, the pen holder supports multiple pens of different colors and widths. The plotter microprocessor receives instructions from the host computer and executes instructions such as "move" (moving the pen holder up to a certain position) and "drawing" (drawing geometric entities such as points, lines, arcs, and circles with the pen down). Since the plotter is a vector device, it can go directly to a specific position on the recording paper without following the order of the hatch line. In a flatbed plotter, when the pen is moved from one position to another on the paper, the paper remains flat and still. But in a drum plotter, the paper itself slides on the cylindrical drum and the pen moves on the drum.

Fig 2.18: Plotter

Fig reference: https://www.youtube.com/watch?v=pGNbbItif9c

Monitor: The computer monitor is a device that is the most commonly used display device that displays information in text, graphics, or video form. It is also called a visual display unit as the monitor can display

Space for learners:

visuals, some manufacturers of the monitor have included an audio system also. The earlier monitor uses Cathode Ray Tubes, but the later versions use many new technologies like Liquid Crystal Display, Thin Film Transistor (Liquid Crystal Display), Light Emitting Diodes, Electroluminescent Display, Plasma, OLED, and many more. Some of these devices are discussed later in this unit.

Fig 2.19: Computer Monitors

Fig reference: https://en.wikipedia.org/wiki/Computer_monitor

https://www.online-tech-tips.com/computer-tips/why-would-you-want-a-crt-monitor-in-2019/

CHECK YOUR PROGRESS

1. The primary output device in a computer system is_____
   a) Scanner
   b) Video monitor
   c) Neither a nor b
   d) Printer
2. What is output device?
   a) It allow data to be typed into a computer
   b) It allow to print data
   c) It allow to store data
   d) It allow to read internal data for processing
3. Which output device of a computer is generally used for presentations?
   a) Multimedia projectors
   b) Plotters
   c) Mouse
   d) Scanner

Space for learners:

2.5 DISPLAY TECHNIQUES

Space for learners:

The most important part of the personal computer is the display system, for which the graphics are visible. The display system can be connected to a personal computer to display characters, images, and video output. Some common types of display systems on the market are Raster scan screens, Random scan screens, Direct view storage tubes, Flat panel screens, Three-dimensional display devices, and Virtual and stereoscopic reality systems. Screen systems are also called video monitors or visual display units (VDUs). The most common video monitor that is generally provided with a personal computer is the raster scan type. However, each display system has three basic parts: a display adapter that creates and stores image information, a monitor that displays that information, and a cable that transmits image data between the display adapter and the monitor.

2.5.1 Raster Scan Display (Raster scan screen)

This type of screen uses a cathode ray tube (CRT) or LCD panel for display. CRT works like a picture tube of a television. Its viewing surface is coated with a series of fluorescent (phosphor) dots. At the back of the CRT, there is a set of the electron gun (cathodes), which generates a controlled flow of electrons. When hit by these high-energy electrons, the phosphor emits light. The frequency and intensity of the emitted light depend on the type of fluorescent material used and the energy of the electrons. To produce an image on the screen, these directional electron beams start from the top of the screen and scan from left to right along the rows of phosphor dots. Then it goes back to the leftmost position and scans down one line again, repeating this operation to cover the entire screen. The process of returning the beam direction from the rightmost to the leftmost line is called horizontal backward movement (or horizontal retrace), during which the electron flow is cut off. During this sweeping or sweeping motion, the electron gun is controlled by the video data stream from the graphics card into the display, thereby changing the intensity of the electron beam at each position on the screen. The immediate control of the intensity of the electron beam at each point is the reason for controlling the color and brightness of each pixel on the screen. All of this happens very quickly,

and the entire screen is drawn in a fraction (for example, 1/60) of a second. The image on the raster scan screen is composed of a set of points and lines; the lines are displayed by highlighting the points (with the desired color) that are as close as possible to the shortest path between the endpoints of the lines. After completing the first frame, i.e., when the scan reaches the rightmost lower corner, then the cursor will jump leftmost upper corner again and the process continues, this is called vertical retrace.

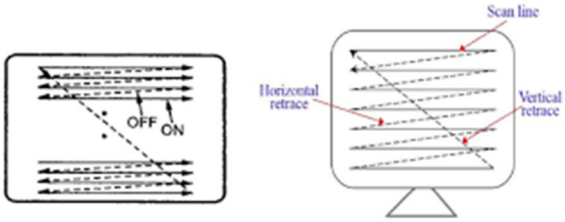


Fig 2.20: Raster Scan Display Method

2.5.2 Random Scan Display (Random Scan Screen)

In the random scanning technology, the electron beam is directly directed at one or more specific points on the screen where the image is to be produced. Image is generated by drawing a set of random straight lines, just like a pencil can be moved on a piece of paper to draw an image: strokes from one point to another, one line at a time. This is why this technique is also called vector drawing or stroke writing or calligraphy visualization. The vector system does not contain a bit plane for mapping pixel values. Instead, the display buffer stores a set of line-drawing commands and endpoint coordinates in the display list or display program created by the graphics package.

The main difference between the random scan and raster scan lies in the technology used to generate images on fluorescent-coated CRT screens. In the raster scanning method, the electron beam scans across the screen, just like you write a whole page of text on a notebook, one word by word, one character, from left to right, from top to bottom. In a random scan, every point has the address of the next point so the pointer will move randomly from one point to the other.

Space for learners:

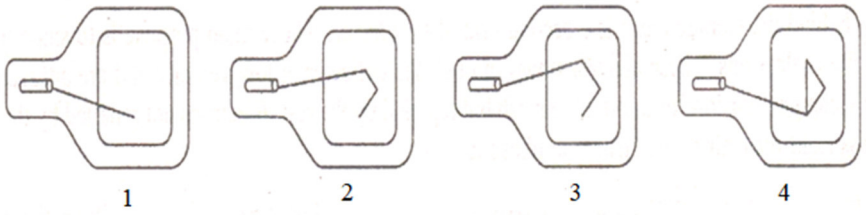


Fig 2.21: Random Scan Display Method

2.6 COLOUR DISPLAY TECHNIQUES

The Cathode Ray Tube monitor screen uses a combination of phosphors. Matches come in different colors. There are two popular methods for using CRT to produce color displays: the Shadow Mask method and the Beam penetration method.

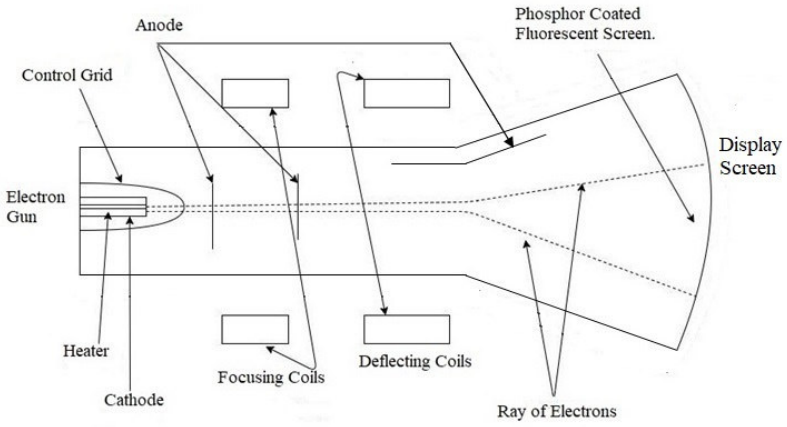


Fig 2.22: Cathode Ray Tube

Shadow Mask method: The shadow mask method is commonly used in the Raster Scan system because it produces a wider color spectrum beam penetration than the Raster Scan method. It is used in most color TVs and monitors. The CRT shadow mask has three fluorescent color dots at each pixel position. The phosphor point emits red, green, and blue light. This type of CRT has three electron guns, an electron gun for each color point, and a shadow mask grid behind the phosphor-coated screen. The shadow mask grid is perforated with small circular holes in a triangular pattern. The deflection system of the CRT acts on the three electron beams simultaneously; the three electron beams are deflected and focused on the shadow mask, which contains a series of holes

Space for learners:

aligned with the phosphor dot pattern. When the three rays pass through a hole in the shadow mask, they activate a dotted triangle, which appears as a small colored dot on the screen. The phosphor points in the triangle are arranged so that each electron beam can only activate its corresponding color point as it passes through the shadow mask. The three electron guns can also be configured into the in-line arrangement, in which the three electron guns and the corresponding red, green, and blue dots on the screen are arranged along a scan line instead to be arranged in a triangle. The in-line arrangement of this electron gun is easier to maintain alignment and is typically used for high-resolution color CRTs.

Beam Penetration Method: The Beam Penetration method has been used to scan monitors randomly. In this method, the CRT screen is coated with two layers of red and green phosphor, and the color displayed depends on the distance the electron beam penetrates the phosphor layer. This method produces only four colors, red, green, orange, and yellow. A beam of slow electrons only excites the outer red shell; therefore the display only shows red. A high-speed electron beam excites the inner green shell. Therefore, the screen is green. Advantages: Realistic images can generate millions of different colors. Shadow scenes are possible. These are expensive as compared to monochrome CRT and also have convergence problems of relatively low resolution.

Space for learners:

2.7 DIRECT VIEW STORAGE TUBES

Direct View Storage Tube (DVST) is a rarely used display system these days. DVST marks a major technological change in generally updated displays. In raster scan and random scan systems, the screen image is maintained (flicker-free) by redrawing or refreshing the screen several times per second by circulating the image data stored in the update buffer. There is no update or refresh buffer in DVST. These images are created by drawing vectors or line segments using a relatively slow-moving beam of electrons. The beam design is not drawn directly onto the phosphor, but onto a thin metal mesh (called a storage mesh) coated with a dielectric and installed behind the screen. A pattern of positive charges is deposited on the grid and the pattern is transferred to the phosphor-coated screen by a continuous large number of electrons

emitted from a separate flood gun. Just behind the storage grid, there is a second grid, the collector, whose main purpose is to smooth the flow of electrons. These electrons pass through the collector at low speed and are attracted to the positively charged part of the storage mesh, but are repelled by the rest. Electrons that are not repelled by the storage network pass directly through it and hit the phosphor. To increase the energy of these slow-moving electrons and produce a bright image, the screen is kept at a high positive potential. The storage tube saves the generated image until it is erased. Therefore, there is no need to update, and the image is flicker-free. One of the main disadvantages of DVST in interactive computer graphics is that it cannot selectively erase certain parts of the image from the screen. To erase a line segment from the displayed image, you must first erase the entire image and then redraw, omitting the line segment. However, DVST supports very high resolution, which is good for displaying complex images.

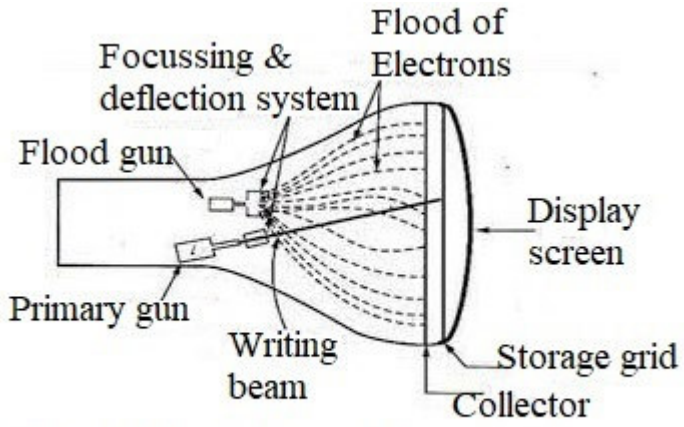


Fig 2.23: Direct View Storage Tube

2.8 EMISSIVE & NON-EMISSIVE FLAT PANEL DISPLAYS

Flat-panel displays are electronic display devices that are used to display text, pictures, graphics, and videos. Flat-panel displays are lighter than the traditional CRT monitors. They can be used in several devices such as; computer monitors, mobile phones screens, display signboards, hoardings, etc. flat panel displays can be categorized into:

Space for learners:

Emissive display: Emission screen or transmitter is a device that converts electrical energy into light energy. Examples: plasma panels, and LEDs (light-emitting diodes).
Non-Emissive Display: Non-luminous displays are devices that use optical effects to convert sunlight or other light sources into graphic patterns. Example: LCD (liquid crystal display), Thin Film Transistor (TFT-LCD).

Space for learners:

2.8.1 Plasma Panels

In plasma panels there is a layer of gas (usually neon gas) sandwiched between two glass plates, vertical thin (column), the conductive strips pass through one plate, and the horizontal conductors (row), rise and fall on the other plate. By applying high voltage to a pair of horizontal and vertical conductors, a small part of the gas (tiny lights neon) at the intersection of the conductors decomposes into luminous electrons and ion plasma. Therefore, in the bulb array, each bulb can be set to the "on" state (illumination) or the "off" state by adjusting the voltage across the appropriate pair of conductors. Once set to "on", the bulb will remain in that state until it is clearly "off" by temporarily reducing the voltage applied to the pair of conductors. Therefore, it is not necessary to update. Due to their excellent brightness, contrast, and scalability to larger sizes, plasma panels are very attractive.

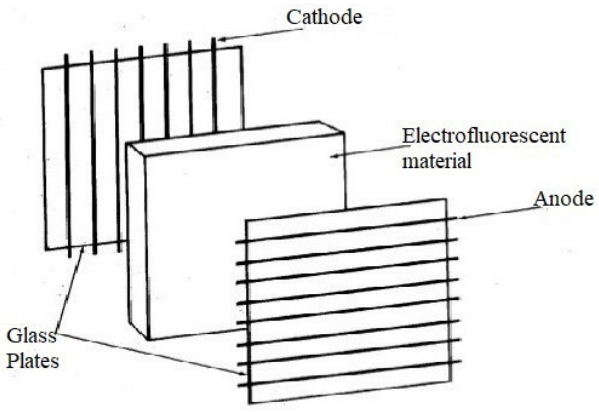


Fig 2.24: Plasma Panel Display

2.8.2 Thin Film Display

An electroluminescent screen (ELD) is a flat-screen made by interposing a layer of electroluminescent material (such as GaAs (gallium arsenide)) between two layers of conductors. When current flows, the material layer emits radiation in the form of visible light. Electroluminescence (EL) is an optical and electrical phenomenon in which a material emits light in response to an electric current or strong electric field passing through it. The term "electroluminescent display" describes displays that do not use LED or OLED devices, but use traditional electroluminescent materials. “Beneq” and “Mordovia State University” jointly manufactured TFEL displays (Thin Film Electroluminescence Displays) and TASEL displays (Transparent Electroluminescent Displays) with the brand name LUMINEQ using Atomic Layer Deposition (ALD) technology. The structure of TFEL is similar to that of passive matrix LCD or OLED displays, while TASEL displays are essentially transparent TEFL displays with transparent electrodes. TASSEL screen transparency can reach 80%. Both TEFL and TASEL displays use chip-on-glass technology, which mounts the display driver IC directly to one of the edges of the display. The TASEL screen can be embedded in the glass panel. Unlike LCD screens, TFEL is more durable and can operate at a temperature of 60-105°C. Unlike OLED, TFEL can operate for 100,000 hours without long aging and only loses about 80% of its initial brightness. Electroluminescent materials are deposited using an atomic deposition layer, which is a process of depositing a thick layer of the atom.

Reference: https://www.easytechjunkie.com/what-is-a-tft-lcd-monitor.htm

Another kind of Thin Film Display is the Thin-FilmTransistor Liquid Crystal Display, which is a flat-screen display that can be used as a computer monitor, television, or mobile phone screen. TFT LCD is the short form for ThinFilm Transistor LiquidCrystalDisplay.In most cases, manufacturers shorten the term for this type of screen to LCD and remove TFT from the name because this acronym only refers to the type of LCD screen, and TFT is easily the most accepted type. Thinfilm transistors consist of thin sheets of semiconductor material applied to a glass substrate. Each pixel has its transistor and liquid crystal material. Liquid crystal material not only exhibits the characteristics of a liquid, because it can change rapidly, but also exhibits the characteristics of a crystal because it can remain in the aligned position. The transistor

Space for learners:

applies a voltage to the pixel to determine its color and intensity. Pixel is an abbreviation for picture elements. These small pixels are mixed to create an image on the screen. Another name for TFT LCDs is active matrix LCD. Although TFT is not the only active matrix technology, it is the most common type, which is why some people use the two terms interchangeably. However, TFT is only a small part of the active matrix LCD screen. The term active matrix refers to the screen's ability to control individual pixels and change them rapidly. Active matrix LCDs differ from passive-matrix LCDs in several ways. At least compared to passive matrix displays, they have a high refresh rate, high contrast, and high response time. Passive matrix LCDs are commonly found in the calculator or digital watch displays, where the display contains a limited number of segments and does not require full color. Active matrix displays are typically high-resolution,full-color LCDs and include those used in computer monitors, cell phones, and televisions. Several different types of thin-film transistor technology can be found in TFT LCDs. The most common computer monitors and televisions are called twisted nematic (TN) monitors, which have fast response times. However, TN displays are not great in terms of screen viewing angle and color reproduction. Another common monitor technology is IPS, which stands for In-PlaneSwitching. IPS displays provide excellent colors and good viewing angles, but their refresh rate is very slow.

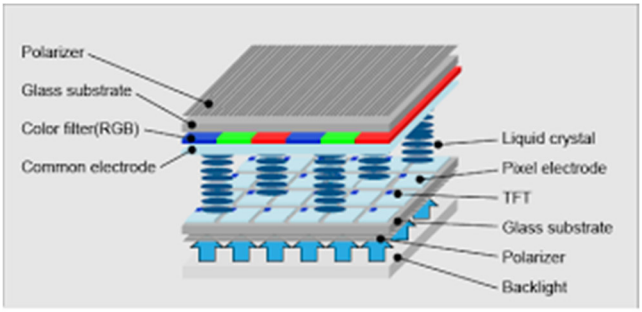


Fig 2.25: TFT-LCD

Fig reference: https://www.orientdisplay.com/knowledge-base/tft-basics/what-is-thin-film-transistor-tft/

2.8.3 Light Emitting Diode

Space for learners:

The LED display is a flat panel display, where arrays of diodes are arranged to form pixel positions on the screen, and the image definition is stored in the update buffer. The data is read from the update buffer and converted to a voltage level applied to the diode to produce a light pattern on the screen. It uses a series of light-emitting diodes as the pixels of the video display. Their brightness allows them to be used for storing signs and outdoor billboards where they are visible in the sun. They are also used for destination signs on public transportation and variable information signs on highways, airports, or railway stations. LED displays can provide general non-visual lighting such as stage lighting or other decorative (contrary to information) purposes. LED screens can provide higher contrast than projectors, making them an alternative to traditional projection screens. They can be used for large, uninterrupted video walls (a single screen frame does not produce a visible grid). A Micro LED display is an LED display with smaller LEDs.

Fig 2.26: LED

Fig reference: https://create.arduino.cc/projecthub/miklas/arduino-led-display-45e9e1

2.8.4 Liquid Crystal Display

LCD consists of a layer of liquid crystal, interposed between two polarizing plates. The polarizers are arranged perpendicular to each other (one vertical and the other horizontal), so the light that hits the first polarizer will be blocked by the second. Because polarizers only pass photons (quanta of light), their electric field is parallel to the polarization direction of the polarizer. The LCD is arranged in a matrix. The rows of the matrix are defined by a thin layer of horizontal transparent conductors, while the columns are defined by another thin layer of vertical transparent conductors; these layers are placed between the LCD layer and the corresponding polarizer. The intersection of the two conductors defines a pixel location. This means that each pixel on

Space for learners:

the screen needs a separate LCD component, and the CRT can provide multiple points for each pixel. The liquid crystal material is composed of rod-shaped crystalline molecules containing cyanobiphenyl units. The individual polar molecules in the helical (parallel but not arranged in well-defined planes) liquid crystal layer are generally arranged in a spiral fashion so that the direction of polarization of the polarized light passing through it rotates 90 degrees. Light from the internal light source (backlight) enters the first polarizer (e.g. horizontal) and is polarized (horizontally) accordingly. When light passes through the LC layer, it is twisted 90 degrees (aligned with vertical) to allow it to pass through the rear polarizer (vertical) and then reflected from the reflector behind the rear polarizer. When the reflected light returns to the viewer's eyes, the LCD screen appears bright. When the current passes through the LCD layer, the crystal molecules line up parallel to the direction of the light, so there is no polarization effect. Due to the misalignment of the polarization direction, the light entering from the front polarizer cannot pass through the rear polarizer. The result is that the light reflection is zero, and the LCD is black. On a color LCD screen, there are three layers of liquid crystal panels, one on top of the other. Each one is filled with a colored liquid crystal (red, green, or blue). Each has its own set of horizontal and vertical conductors. Each layer absorbs the adjustable portion of the single-color light passing through it. The main advantage of this design is that it helps to create as many screen pixels as there are intersections, making higher resolution LCD panels possible. Each pixel includes three color units or sub-pixel elements. The image drawing operation on the LCD panel is different from the CRT, although both are raster scan types. A simple LCD panel illuminates an entire row of screen pixels at a time. Then the next line and so on, until the entire screen image is completed. The image definition is stored in the update buffer, and the screen is usually updated at a rate of 60 frames per second. After setting, the screen pixels will maintain a fixed brightness until reset. Compared with CRT, it takes longer to establish pixel brightness. LCDs are widely used in mobile phones TV, monitors, etc.

Space for learners:

Fig 2.27: LCD

2.9 THREE DIMENSIONAL VIEWING DEVICES

Three-dimensional transformation is an extension of two-dimensional transformation. Two coordinates, x, and y are used in 2D, and three coordinates x, y, and z are used in 3D. For three-dimensional objects and images, three-dimensional transformations are required. It is about translation, scaling, and rotation. They are also called basic transformations because they are represented by matrices. It uses 3D matrices to handle more complex transformations. 2D can display two-dimensional objects. Such as bar charts, pie charts, and graphs. But some more natural objects can be rendered in 3D. Using 3D, different shapes of objects in different parts can be seen. In 3D, when translating, three rotation factors are needed, which are part of the three rotations. Each can be done along three Cartesian axes. 3D can also represent a series of transformations as a single matrix. Computer graphics uses Computer-Aided Design(CAD). CAD allows the manipulation of three-dimensional machine parts. It also provides research on car bodies and aircraft parts. All of these activities require realism. For realism, 3D is necessary. It is difficult to create realistic 3D scenes from 2D. It requires third dimension, namely depth.

Fig 2.28: 3D Viewing Glass

Space for learners:

Fig reference: https://www.livelaw.in/news-updates/3d-glasses-3d-movies-free-of-cost-kerala-state-consumer-disputes-redressal-commission-172796

Space for learners:

2.10 DISPLAY SYSTEMS ARCHITECTURE

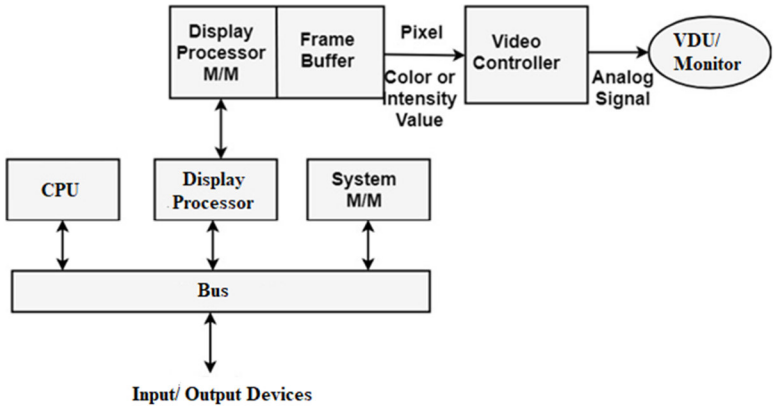


Fig 2.29(a): Architecture of Raster Scan Method

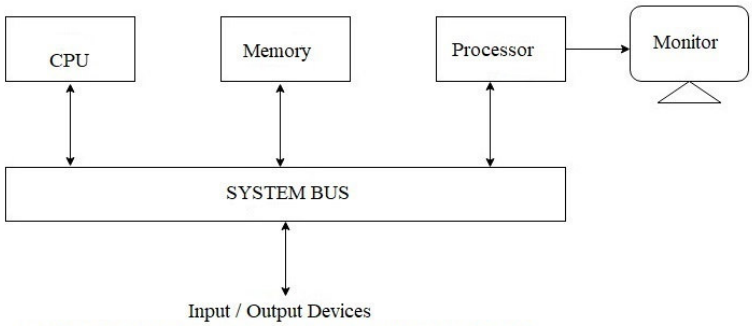


Fig 2.29(b): Architecture of Random Scan Display Method

STOP TO CONSIDER

In this sub section we discussed about the working principle of various display devices and several display techniques with covering to the details about Plasma Panels, Liquid Crystal Displays, Electroluminescent Displays, Thin Film Transistor and techniques like Random Scan and Raster Scan methods, three dimensional viewing devices and display architectures.

CHECK YOUR PROGRESS

1. _____ stores the picture information as a charge distribution behind the phosphor-coated screen.
   a) Cathode ray tube
   b) Direct-view storage tube
   c) Flat panel displays
   d) 3D viewing device
2. The devices which converts the electrical energy into light is called
   a) Liquid-crystal displays
   b) Non-emitters
   c) Plasma panels
   d) Emitters
3. In which system, the Shadow mask methods are commonly used
   a) Raster-scan system
   b) Random-scan system
   c) Only b
   d) Both a and b
4. Which display devices allows walking around an object and viewing it from different sides.
   a) Direct view storage tubes
   b) Three-dimensional devices
   c) Flat panel display devices
   d) Plasma panel display devices
5. On raster system, lines are plotted with
   a) Lines
   b) Dots
   c) Pixels
   d) None of the mentioned

Space for learners:

2.11 SUMMING UP

This chapter gave a detailed overview of the working of various input and output devices, without which the computer would never have been able to communicate with the outside world. The users could never be able to use the computer. As these devices are used to send data (command) into the computer and also get back processed information from the computer in the form of words, pictures, signals, audio, and videos. These devices are also called peripheral devices because they can be attached to computers. Later at the end, we discussed some of the display devices and display techniques.

Space for learners:

2.12 ANSWERS TO CHECK YOUR PROGRESS

Question 1: Answer: c
Explanation: All except the keypad are point-and-draw devices. They are used to rapidly point to and select a graphic icon or menu item from multiple options displayed on the GUI of a screen.

Question 2: Answer: d
Explanation: Joystick is the device used for the same. It is a point-and-draw device. It has a click button, a stick, a ball, a socket as well as a light indicator.

Question 3: Answer: a
Explanation: Image scanners are the input devices that translate the paper documents into an electronic format for storage in a computer. Stored image can be altered or manipulated with image-processing software.

Question 4: Answer: b
Explanation: The video monitor is the commonly used output device.

Question 5: Answer: b

Explanation: The output devices allow data to be printed out from the computer to the outside world.

Question 6: Answer: a

Explanation: Multimedia projectors are used to display presentations on a screen.

Question 7: Answer: b
Explanation: Instead of refreshing, DVST stores the picture information behind the screen.

Question 8: Answer: d
Explanation: Emissive displays are devices that convert electrical energy into light.

Question 9: Answer: a
Explanation: Raster-scan system uses shadow-mask method because they produce wide range of colors.

Question 10: Answer: b
Explanation: 3D display devices allows user to view the object from different sides.

Question 11: Answer: c
Explanation: Plotting can be done using pixels lines.

Space for learners:

2.13 MODEL QUESTIONS

1. What are the differences between positioning and pointing devices?
2. Distinguish the working of a digitizer and plotter?
3. Compare the working method of touch panel and light pen.

2.14 REFERENCES AND SUGGESTED READINGS

1. Computer Graphics: C Version by Donald Hearn & M. Pauline Baker.
2. Computer Graphics Principles & Practice by John F. Hughes, Andries Van Dam, Morgan Mcguire, David F. Sklar, James D. Foley, Steven K. Feiner, Kurt Akeley
3. https://www.apple.com/in/shop/product/MQ052HN/A/magic-keyboard-with-numeric-keypad-us-english
4. https://www.uow.edu.au/student/learning-co-op/technology-and-software/mouse-buttons/
5. https://www.amazon.com/Raspberry-Touchscreen-Monitor-1024x600-Speakers/dp/B07S51QDTG

6. https://www.indiamart.com/proddetail/voice-recognition-system-10207431155.html
7. https://www.amazon.in/Joystick-YF2009-Controller-Vibration-Throttle/dp/B06XGBL1HL
8. https://www.123rf.com/photo_3854698_working-with-digitizer.html
9. https://www.adesso.com/product/imouse-t40-adesso-2-4-ghz-wireless-4-button-desktop-trackball/
10. https://www.c64-wiki.com/wiki/Light_pen
11. https://www.researchgate.net/figure/Fully-functional-CIE-DataGlove-with-all-IMU-sensors-attached_fig2_314135437
12. https://www.indiamart.com/proddetail/hp-scanjet-flatbed-scanner-15066650062.html
13. https://www.indiamart.com/proddetail/retsol-ls-450-barcode-scanner-9875844788.html
14. https://www.indiamart.com/proddetail/hvp-printer-dot-matrix-printer-hvp1410-22926934788.html
15. https://www.indiamart.com/proddetail/computer-line-printer-9913028188.html
16. https://fabrikbrands.com/best-inkjet-printer-for-mac/
17. https://www.indiamart.com/proddetail/thermal-recipt-printer-21609056991.html
18. https://www.indiamart.com/proddetail/water-based-eco-solvent-piezoelectric-printer-9764534888.html
19. https://en.wikipedia.org/wiki/Laser_printing
20. https://dir.indiamart.com/impcat/electrostatic-printing.html
21. https://www.youtube.com/watch?v=pGNbbItif9c
22. https://en.wikipedia.org/wiki/Computer_monitor
23. https://www.online-tech-tips.com/computer-tips/why-would-you-want-a-crt-monitor-in-2019/

Space for learners:

UNIT 3: GRAPHICS SOFTWARE

Space for learners:

Unit Structure:

3.1 INTRODUCTION

In this unit, we will learn some fundamental aspects of Graphics software. we will learn about their features, uses, and types. we will learn what kind of files the graphics software deals in. we will be able to learn on what basis this software is classified, what are they, what are the advantages and disadvantages of Raster files and Vector files, and what are the various graphics standard available, why it came. we will be able to learn about the various output primitives and their attributes. We will be able to answer about the objectives behind the development of various graphics standards like GKS, PHIGS, PHIGS+, and basic parts of these graphics packages. We will discuss various standard graphics functions like- Polylines, Polymarkers, FillArea, Text, CellArray, etc. we will be able to learn different graphics functions use in a programming language like C and C++ with their purpose of use, syntax, and with examples. We will also discuss CAD and its advantages.

3.2 UNIT OBJECTIVES

After going through this unit, you will be able to

- define graphics software
- know the various types of graphics software
- discuss the most popular graphics software available with their purpose.
- understand various aspects of graphics standards
- understand GKS, PHIGS, and PHIGS+ packages and their advantages, and disadvantages.

3.3 GRAPHICS SOFTWARE

Graphics are any image media, usually pictures or movies that are produced with the help of computer hardware and software. So the software which is been used to create, display and edit computer

Space for learners:

graphics is called **Graphics Software.** This software is used to manage two-dimensional photos, logos, web graphics, clip art, or any other digital images. This software mainly provides tools to create digital images.

Space for learners:

3.3.1 Features of Graphics Software

Graphics Software features include-

1. **Photo editing:** With the help of graphics software we can edit an existing image or enhance it as per our requirements.
2. **Advanced font options:** We can enter text into a graphics file and then give a different font style to it.
3. **Freehand drawing:** It provides tools to create your models or figures.
4. **High-quality templates:** It provides high-quality templates that are very helpful for a graphics designer
5. **Art brushes:** Special brushes like an airbrush can be used to draw a picture or form an image with a different paint effect.
6. **Clip art:** It has a built-in library of clipart pictures.
7. **Zooming:** It provides us the tool to zoom to a particular position of the image and see its details.
8. **3D capabilities:** It provides us the feature to create and design 3D models and figures.
9. **Exporting and importing files:** It has the feature to create a graphics file and then export that file to an application program. It also has the features of importing a file to graphics software or package.

3.3.2 Examples of Graphics Software

Software like MS Word, MS Excel, and database programs also support graphics but in a limited way what they can do with the graphics. Like MS Word, one can draw simple line art or display images alongside text. Spreadsheet programs can be used to form

graphs and charts. But this software will give us the full ability to edit graphics in detail.

Some of the most popular graphics editors software are:- Adobe Photoshop, Illustrator, Paint shop Pro, Corel Draw, Adobe Lightroom, Digital image suite, etc.

Space for learners:

3.3.3 USE OF GRAPHICS SOFTWARE?

The following are the applications of graphics software:-

1. Editing &Sharing digital photos
2. Creating logos
3. Drawing and modifying clip art
4. Creating digital fine art.
5. Creating Web Graphics
6. Designing advertisements and product packaging
7. Touching up scanned photos
8. Drawing maps or other complex diagrams
9. Editing video in Photoshop or 3D drawing in Illustrator.

3.3.4 Types (Classification)

There are mainly two general classifications of graphics software:-

1. General Programming Packages
2. Special purpose application packages.

   Features of Graphics Packages:

3.3.4.1 General Programming Packages

It provides an extensive set of graphics functions that can be used in high-level programming languages, like C, C++, PASCAL, and FORTRAN. Some of the common graphics programming packages

include GKS, PHIGS, PHIGS+, 3D GKS, and the Graphics Library (GL) system of silicon graphics equipment. Graphics packages require coordinates specification to be given concerning Cartesian reference frames. Each object of a scene can be defined in a separate modeling Cartesian coordinate system, which is first mapped to a world, coordinates to construct the scene, and then from the world coordinate, objects are transferred to normalized device coordinates and finally to display device coordinates.

Space for learners:

3.3.4.1.1 Functions of General Programming Packages

Following are the main functions of general programming packages-

i. Generating picture components (like straight lines, polygons, circles, oval ellipses and other figures)
ii. Setting color and intensity values
iii. Setting views
iv. Applying transform actions.

3.3.4.1.2 Taxonomy of the Principal Feature of Graphics Packages

a. Graphical Output:
b. Attributes
c. Geometric and modeling transformations
d. Transformations and Viewing
e. Structure operations
f. Input functions
g. Control operation

a. Graphical output: Graphics package provides a small number of graphical output primitives like- Polyline, Polymarker, text, fill area, cell array, and generalized drawing primitives.

b. Attributes: Each output primitives has some number of attributes like- color, thickness, height, width, etc. For E.g. the line has the attributes of thickness, height, width, and color.

c. Structuring the Model: Structure elements are grouped to form a named list called structures and a special structure element may be used to define the link between structures to create a hierarchical network structure.

d. **Editing the model:** When a model has been defined, it may be interactively edited by inserting and deleting structure elements and deleting the entire structure. Changes made to the definition of the model are automatically reflected in the display image.
e. **Transformations and viewing:** A complex model is built up from simpler components that may be independently using the most appropriate co-ordinates systems, which are known as modeling coordinates, and are combined using modeling transformations to create the overall picture in world coordinates. To display a 3D picture on a workstation, PHIGS provides a general-purpose mechanism for creating parallel and perspective views, including all the views in common use in Computer-Aided Design (CAD).
f. **Graphical Input:** Standard graphic package provides several facilities for the user to provide a graphical input to the application program, using essentially the same model of logical input developed for GKS. Six types of logical input devices are locator, stroke, choice, pick, string, and valuator, each of which may be used in several styles, using request, sample, and event modes.
g. **Model and picture storage:** The structured network may be stored and retrieved using archive files that are extended to PHIGS. 2D images may be stored on metafiles.

Space for learners:

3.3.4.2 Special Purpose Application Package

They are designed for non-programmers, so that user needs not worry about the internal knowledge of how they function. They provide the user interface to communicate with the program in their style.
E.g.: Artist painting programs, business CAD systems, graphing packages, and visualization programs

3.4 TYPES OF FILES GRAPHICS SOFTWARE DEAL WITH

1. Raster files
2. Vector files

3.4.1 Raster Files

Raster files use a matrix of pixels to form an image. The pixels or dots are created at a certain size, so enlarging the image can make its appearance blur and by shrinking, it can erase important details. Raster files are usually best for photographs or complex images, as long as they do not need to be sized very often. Common raster graphic file formats include BMP (bitmap), JPEG (Joint Photographic Expert), TIFF (Tagged Image File Format), and GIF (Graphic Interchange Format)

- BMP are resolution-dependent and support 24-bit color
- JPEGs are compressed and sacrifice image quality in exchange for a reduction in image size.
- GIFS display up to 256 colors, feature build-in compression, and work with most web browsers.
- TIFFs are used with high-resolution images.
- Graphics software like Microsoft paint, Adobe Photoshop, PC paintbrush, and Paintshop Pro works on Raster files.

3.4.1.1 Advantages of Raster Files

1. Here individual pixels can be modified which helps in editing the picture to a greater extent.
2. Raster files are best for photographs or complex images.

3.4.1.2 Disadvantages of Raster Files

1. Enlarging the image can make it blur and shrinking the image leads to the loss of important details in the image.
2. Individual parts of an image cannot be resized only the whole image can be resized.
3. Here the information of every pixel has to be stored so it uses larger amounts of backing storage space.

3.4.2 Vector Files

Space for learners:

Vector files use mathematical equations to determine how certain elements should be placed within the design. By using these equations, it allows vector graphics to be scaled up or down without losing clarity. Vector graphics software is used in an application designed to perform digital diagramming or CAD. Vector graphics software includes- Adobe Illustrator, CorelDraw, and Inkspace.
E.g. of vector graphics file formats are DXF (Data Exchange Format) and WMF (Windows Meta File), Micrographix Designer.

Space for learners:

3.4.2.1 Advantages of Vector Files

1. They use less memory space than bitmap graphics.
2. Each part of an image is treated as a separate object so it can be easily modified.

3.4.2.2 Disadvantages of Vector Files

1. It does not look as realistic as a bitmap image or as Raster files.
2. Not suitable for complex graphics display.
3. Rasterization is required for display.

CHECK YOUR PROGRESS

1. MS Word, MS Excel, and database programs also _____ graphics.
   a. support
   b. don't support
2. Spreadsheet programs can be used to form _____ and _____.
3. Each object of a scene can be defined in a separate modeling _____coordinate system.
4. Graphics software deal with _____ files and _____ files.
5. Raster files use a _____ of pixels to form an image.
6. _____ files are usually best for photographs or complex images.
7. Vector files use _____ _____ to determine the position of each object.

3.5 SOFTWARE STANDARDS

Without standards, programs designed for one hardware system often cannot be transferred to another system without extensive re-writing of the programs. So, to overcome the above problems, graphics software packages are designed with standard graphics functions, so that software can be moved easily from one hardware system to another and used in different implementations and applications.

International and national standards planning organizations in many countries came together and put in the effort which led to the development of the **Graphical Kernel System (GKS)**. This system was adopted as the first graphics software standard by the International Standards Organization (ISO) and by various national standards organizations, including (ANSI) American Standards Institute. GKS was originally designed as a 2D graphics package and later on a three-dimensional GKS extension was subsequently developed.

The second software standard to be developed and approved by the standard organization **was PHIGS (Programmer's Hierarchical Interactive Graphics Standard)**, which is an extension of GKS. Another extension of PHIGS was developed which is **PHIGS+** which provides a **3D graphics package.**

3.5.1 GKS (Graphics Kernel System)

The Graphic Kernel System is an international standard for graphics files that defines how the graphics are handled by the graphics software. It allows us to create computer graphics files and then move them to a different computer platform or software.

3.5.1.1 Objectives Behind the Development of GKS

1. To control all types of graphics devices like – Plotters and display devices in an efficient manner.
2. To provide the complete range of graphical facilities in 2D, including the capabilities to interact.
3. To provide a better algorithm.

Space for learners:

4. To make the system portable.
5. To avoid the rewriting of the code.
6. To assist in the learning of graphics systems by application programmers.

Space for learners:

3.5.1.2 Features of Graphics Kernel System

- GRAPHICS FUNCTIONS: It provides graphics function to draw 2D and 3D model
- DEVICE INDEPENDENCE: It does not assume that the input and output devices have any kind of restrictions or features
- DISPLAY MANAGEMENT: It provides a complete suite of display management functions.
- TEXT or ANNOTATIONS: All the text or annotations are in natural languages like English.

3.5.1.3 Layer Model of Graphics Kernel System

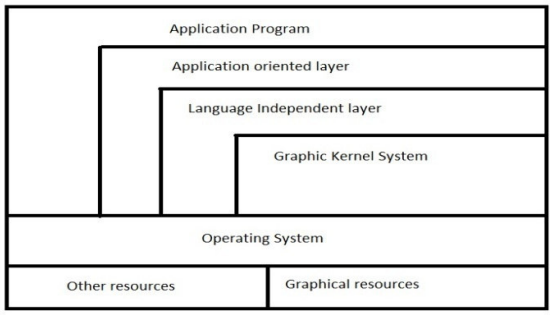


Fig 3.1

3.5.1.4 Basic Parts of GKS

The GKS consists of three basic parts:-

1. The casual exhibition of the substances of the standard which is placed, how polygonal areas are to be filled, and so on.
2. Formalization of the expository material in (1) by way of abstraction of the ideas into discrete functional descriptions. These functional descriptions contain such information as

descriptions of input and output parameters, a precise description of the effect each function should have, a reference to the expository material in (1), and a description of error conditions. The functional descriptions in this section are language-independent.

3. Language bindings. These are an implementation of the abstract functions described in (2) in a specific computer language like C, FORTRAN, etc.

Space for learners:

3.5.1.5 Advantages of GKS

1. It makes the system portable.
2. It saves time as rewriting of code is not required.
3. It provides an improved algorithm.

3.5.1.6 GKS Primitives

The GKS primitives are Polyline, Polymarker, Text, and Fill-area.

3.5.2 PHIGS (Programmer's Hierarchical Interactive Graphics Standard)

It is the second standard to be developed after the GKS.PHIGS is the extension of GKS which provides a 3D graphics package.PHIGS includes some additional functions for object Modeling, color specification, Surface Rendering, and Picture manipulation.

PHIGS acts as a device-independent interface between the graphics subsystem and the application program like GKS. It provides specifications for basic graphics functions. It does not provide a standard method for storing and transmitting images or pictures. It provides functions for application modeling using 3D interactive computer graphics. It gives the ability to construct models that have a hierarchical logical structure and to create pictures of a model or design for the programmer to interact with. The model may be developed in a modular fashion, using either a top-down or bottom-up approach. Since the design is modular, parts of the model may be manipulated independently by other parts. Data may be shared by

defining structures that are instanced by other structures. It defines a set of functions and data structure for the graphics programmer to manipulate and display 3D graphics objects. Like GKS, PHIGS also

has same primitives, attributes, the workstation concept and the viewing and input model.

Space for learners:

3.5.2.1 Facilities of PHIGS

The PHIGS has the following facilities-

1. To create and edit a hierarchical application model.
2. It has the facility of device-independent.
3. It can perform real-time manipulation of pictures and their underlying models.
4. It has the facility of graphical interactions.
5. Pictures and images can be stored on an offline storage device.

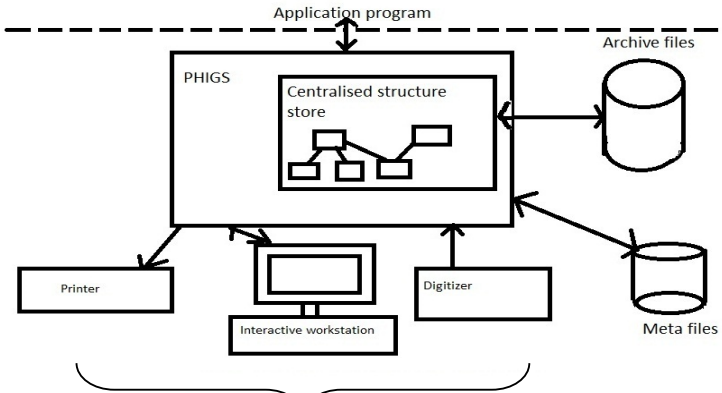


Fig: 3.2 PHIGS in relation to the application program

3.5.3 PHIGS+

PHIGS+ is the extension of PHIGS having additional functionality of providing three-dimensional surface shading capabilities that were absent in PHIGS. It can render illuminated scenes. It also has additional methods for lighting 3D scenes.

PHIGS+ has the following extensions –

- It has some additional output primitives for curves and surfaces in 2D and 3D, including triangle net, quadrilateral mesh, and non-uniform rational B-spline curves and surfaces.
- It has illumination and shading for controlling how output primitives are rendered; ambient, diffuse, specular, and spotlight sources are provided, and various shading methods, including those of Gouraud and Phong, the application may also define colors (representing extra data) to be interpolated across surfaces in conjunction with colors resulting from illumination calculations.
- It does not produce an image that is indistinguishable from photographs of real scenes or photorealistic images and hence it has no facilities for transparency and shadows or reflections between objects in a scene.

Space for learners:

3.5.4 PHIGS Workstations

Workstations are computer systems mainly designed for a single user with many input and output devices. In GKS and PHIGS, it is used to identify the various combinations of graphics software and hardware. A PHIGS workstation can be a single input device, an output device, a combination of both input and output devices, a file, or a window display on a video monitor.

The following statement gives a general structure of a PHIGS function.

```
        openPhigs (errorFile, memorySize)
        openWorkstation (ws, connection, type)
{
        Create and display pictures
}
closeWorkstation (ws)
closePhigs
```

Where the parameter error file is to contain any error messages that are generated and memory size specifies the size of an internal storage area. The workstation identifier is given in parameter ws, and parameter connection states the access mechanism for the workstation, such as an input device, an output device, a combination of output and input device, or an output or an input metafile. Many workstations can be open in a particular application,

with input coming from various open input devices and output directed to all the open output devices.

Space for learners:

3.6 Output Primitives

- Pictures are considered to be constructed from many basic building blocks called primitives.
- The functional set of primitives has the word names POLYLINES, POLYMARKER, FILL AREA, TEXT and
- CELL AREA, even though a few implementations widen this basic set.

3.7 Standard Graphics Functions

Standard Graphics functions are defined as a set of specifications that is independent of any programming language. A language binding is then defined for a particularly high-level programming language. This binding gives the syntax for accessing the various standard graphics functions from this language. Graphics functions can be divided into the following categories: Output primitives, attributes, geometric and modeling transformations, viewing transformations, structure operations, input functions and control operations.

3.7.1 Polylines

The function for drawing line segments is called 'POLYLINE'. The POLYLINE command takes an array of X-Y co-ordinates and creates line segments joining them. The elements that are organize the look of a POLYLINE are-

a. Line Type: Solid, dotted or dashed.
b. Line width scale factor: thickness of the line
c. Polyline color index: color of the line

In PHIGS, GKS and some other graphics packages, the two dimension line function is –

Polyline (n, Wcpoints) where the parameter n is an assigned integer number equal to the number of coordinate positions to be

input an Wcpoints is the array of input world coordinates values of the line segment endpoints. This function is used to define a set of (n-1) connected straight line segment. In order to display a single straight line segment, we set n=2 and list the x and y values of the two endpoint coordinates in wcPoints.

E.g.the following statements will generate two connected line segments, with endpoints at (100, 100), (200, 280), and (280, 100)

```
wcPoints.x[1]=100;
wcPoints.y[1]=100;
wcPoints.x[2]=200;
wcPoints.y[2]=280;
wcPoints.x[3]=280;
wcPoints.y[3]=100;
polyline (3, wcPoints);
```

The polyline procedure is implemented by accomplishing first performing a series of coordinate transformations, then making a sequence of calls to a device-level line drawing routine. In PHIGS, the input line endpoints are actually specified in modeling coordinates, which are then converted to world coordinates, the world coordinates are then converted to normalized coordinates and finally to device coordinates.

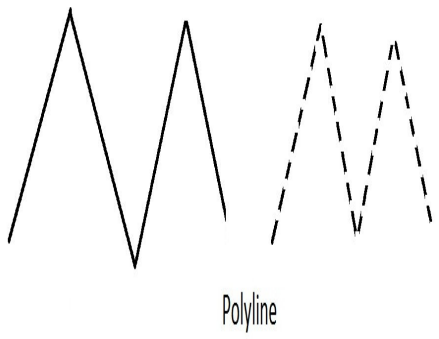


Fig:3.3

3.7.2 Polymarkers

Space for learners:

The 'POLYMARKER' function permits to draw symbol of marker centered at co-ordinates points. This features that control the look of 'POLYMARKER' are-

a. Marker size scale factor: Size of marks
b. Marker Characters: dot, plus, asterisk, circle or cross
c. Polymarker color index: color of the marker.

POLYMARKER (N, XPTS, YPTS)

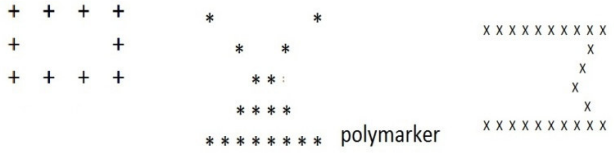


Fig: 3.4

The function polymarker (n, wcPoints) where the parameter n is an assigned integer number equal to the number of coordinate positions to be input an Wcpoints is the array of input world coordinates values of the line segment endpoints just same as polyline function. This function places a designated character, called a marker symbol, at one or more selected positions.

3.7.3 Fillarea

The FILLAREA function permits to denote a polygonal shape of a zone to be filled with various interior shapes. The features that control the look of fill area are-

a. *FILLAREA interior style:* solid color, hatch pattern
b. *FILLAREA style index:* horizontal lines, vertical lines, left slant lines, right slants lines, horizontal and vertical lines of left slant and right slant lines.
c. *FILLAREA color index:* color of the fill pattern/solid areas.

The function fillArea (n, wcVertices) defines a polygon fill area with n number of vertices specified in world coordinates array wcVertices. The fillAreaset( ) function allows a series of polygons to be displayed by specifying the list of vertices for each polygon.

Space for learners:

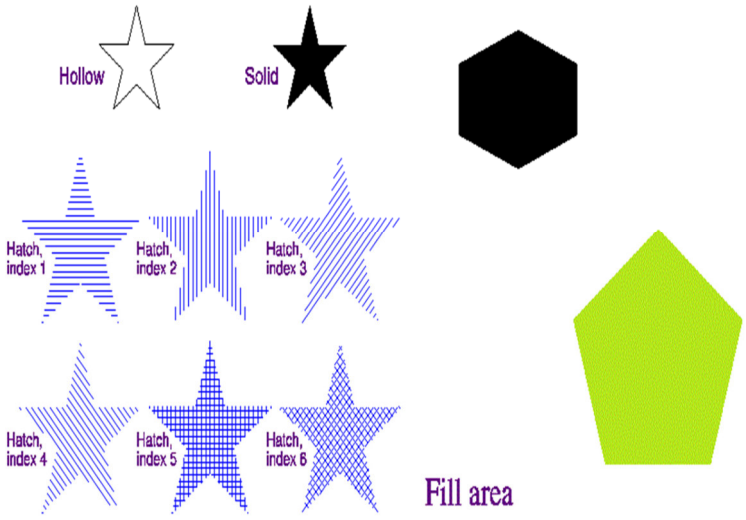


Fig: 3.5

3.7.4 TEXT:TEXT(X, Y, String) The GKS text function allow us to draw a text string at a specified coordinate position. The attributes that control the appearance of text area are-

a. *Text font and precision:* It specifies what text font should be used for the characters and how precisely their representation should adhere to the settings of the other text attributes.
b. *Character expansion factor:* It controls the height: width ratio of each plotted character.
c. Character spacing: It specifies how much additional white space should be inserted between characters in a string.
d. Text color index: It specifies what color the text string should be.
e. Character height: It specifies the height of the characters.
f. Character up vector: It specifies the angle of the text.
g. Text path: It specifies the direction of the text.(right, left, up, down)
h. Text alignment: Vertical and horizontal centering options for the text string.

The function text (wcPoints, string) displays a specified character string. World coordinates position wcPoints specifies a reference point for displaying the string according to the attributes settings.

Space for learners:

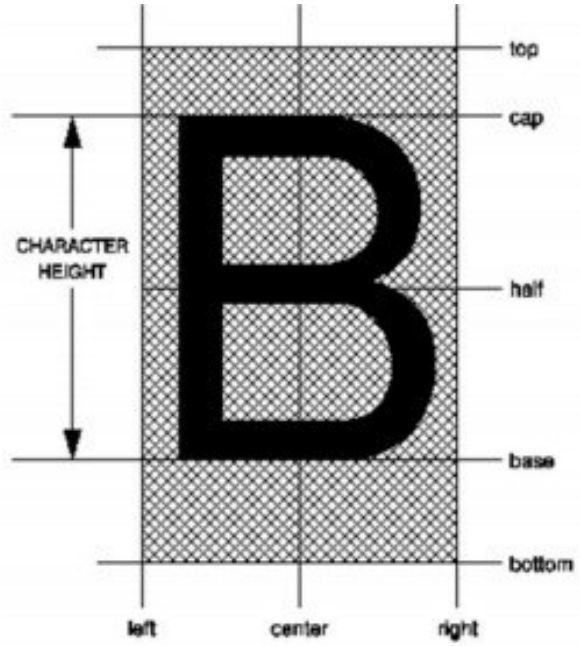


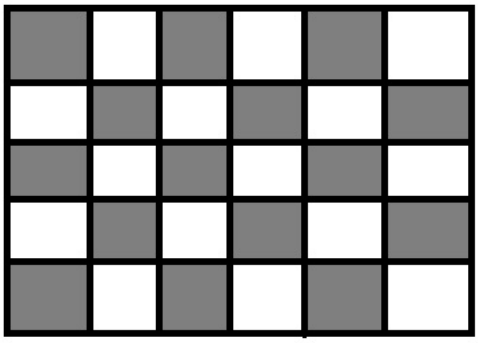
Fig: 3.6

Space for learners:

3.7.5 Cell Array

The CELL ARRAY function shows raster like pictures in a device autonomous manner. The CELL ARRAY function takes two corner points of a rectangle that indicates a number of partitions (M) in the X direction and a number of partitions (N) in the Y direction. Next it partitions the rectangle into MXN sub rectangles noted as cells.

The function cellArray (wcPoints, M, N, colorArray) allows a user to display an arbitrary shape defined as a two- dimensional grid pattern. It maps an M by N color array onto a rectangular area, whose coordinates limits are specified in array wcPoints).



cell array

Fig: 3.7

3.7.6 Curve Functions

The PHIGS standard does not provide explicit functions for the curves, but it includes the this general curve function:

generalizedDrawingPrimitive (n, wcPoints, id, datalist) where wcPoints is a list of n coordinate positions, datalist contains non co-ordinates data values, and parameter id selects the desired function. A circle might be referenced with id=1, an ellipse with id=2, and so on.

CHECK YOUR PROGRESS

8. The use of Polymarker function is to _____
9. The FillArea function _____
10. The Polyline function_____
11. The use of text function is to _____ at a specified coordinate position.

3.8 PROPERTIES OF GRAPHICS PACKAGES

Space for learners:

1. It provides the users with a variety of functions for creating and manipulating images.
2. **Output primitives:** Output primitives are the basic building blocks to create a picture or image.
3. **Attributes:** Attributes are the properties for the output primitives like width line, size, color, etc.
4. **Geometric Transformation:**We can change the size, position and orientation of Graphics object.
5. **Modeling transformations:**we can use this to construct a scene.
6. **Viewing transformations:** We can use this to select a view of scene, the type of projection to be used and the location where the view is to be displayed.
7. **Input functions:** We can use this to control and process the data flow from these input devices like mouse, tablet, joystick, etc.

Space for learners:

3.9 GRAPHICS FUNCTIONS IN C AND C++ PROGRAMMING LANGUAGE

1. **Line( ):** line function is use to draw line on the monitor screen.
   Syntax:line(x1, y1, x2, y2);
   E.g.: line(100,100,300,300);
2. **Circle( ):** circle function is used to draw a circle on the monitor screen.
   Syntax: circle(x, y, radius); where 'x' and 'y' are the x-coordinates and y coordinates.
   E.g.: circle (400, 400, 80);
3. **Rectangle( ):** rectangle function is used to draw the rectangle on the screen.
   Syntax:rectangle(x1, y1, x2, y2); where x1,y1 are the lower left co-ordinates and x2,y2 are the upper right co-ordinates of the rectangle to be drawn on the monitor screen.
   E.g.:rectangle (100,100,300,300);
4. **Ellipse( ):** ellipse function is use to draw an ellipse on the monitor screen.
   Syntax: ellipse (x, y, starting angle, ending angle, xradius, yradius);
   E.g.: ellipse (100, 100, 90, 250, 30, 35);

5. **Putpixel( ):** putpixel function is use to draw the pixel on the monitor screen.
   Syntax: putpixel (x co-ordinate, y co-ordinate, COLOR);
   E.g.: putpixel (200, 300,RED);
6. **Getpixel( ):** getpixel function returns the color of a specified pixel.
   Syntax: getpixel (x, y);
   E.g.: getpixel (200,200);
7. **Setcolor( ):** setcolor is to set color of objects which is to be drawn just after this setcolor line.
   Syntax: setcolor (COLOR);
   E.g.: setcolor (GREEN);
8. **Setbkcolor( ):** setbkcolor function is used to set background color of the screen.
   Syntax: setbkcolor (COLOR);
   E.g.: setbkcolor (GREEN);
9. **Setlinestyle( ):** setlinestyle function is use to set the current line style, width and pattern.
   Syntax: setlinestyle (linestyle, pattern, thickness);
   E.g.: setlinestyle (SOLID_LINE, 1,3);
10. **Initgraph( ):** Initgraph function initializes the graphics system by loading a graphics driver from disk and putting the system into graphics mode.
    Syntax: Initgraph (&gdriver, &gmode,"PATH to the BGI folder");
    E.g.: Initgraph (&gd, &gm,"C:\\turboc\\bgi");
11. **Closegraph( ):** Closegraph function shut down the graphic system.
    Syntax: closegraph ( );
    E.g.: closegraph ( );
12. **Textheight( ):** textheight function returns the height of the string in pixels.
    Syntax: textheight (STRING);
    E.g.:h=textheight ("COMPUTER");
13. **Textwidth( ):** textwidth function returns the width of the string in pixels.
    Syntax: textwidth (STRING);
    E.g.: w= textwidth ("COMPUTER");
14. **Getx( ):** getx function returns the current position of the x-coordinate.
    Syntax: getx ( );

Space for learners:

E.g.: x=getx ( );

15. **Gety( ):** getx function returns the current position of the y-coordinate.
    Syntax: gety ( );
    E.g.: x=gety ( );
16. **Getmaxx( ):**getmaxx function returns the maximum x-cordinate position on the screen.
    Syntax: getmaxx ( );
    E.g.:x=getmaxx ( );
17. **Getmaxy( ):** getmaxx function returns the maximum y-cordinate position on the screen.
    Syntax: getmaxy ( );
    E.g.:y=getmaxy ( );
18. **Settetextstyle( ):** settextstyle function sets the current text characteristics like font- direction, and size;
    Syntax: settextstyle (font, direction, size);
    E.g.: settextstyle (1, 1, 12);
19. **Moveto( ):**moveto function moves current cursor position on the screen.
    Syntax: moveto (x co-ordinate, y co-ordinate);
    E.g.: moveto (100,150);
20. **Outtext( ):** outtext function is used to display the text on the monitor screen, using this function we can display the text in the current position.
    Syntax: outtext (STRING);
    E.g.: outtext ("computer");
21. **Outtextxy( ):** outtext function is used to display the text on the monitor screen in graphics mode, using this function we can display the text in the desired position.
    Syntax: outtextxy (x coordinate, y coordinate,STRING,);
    E.g.: outtextxy (100, 200,"computer");
22. **Sector ( ):** sector function draws and fills an elliptical pie slice.
    Syntax: sector (x co-ordinate, y co-ordinate, starting angle, ending angle, x-radius, y-radius);
    E.g.: sector (200, 100, 50, 120, 100, 50);
23. **Arc ( ):** arc function draws an arc on the screen.
    Syntax:arc(x coordinate, y co-ordinate, starting angle, ending angle, radius);
    E.g.: arc (100,200,45,120,70);

Space for learners:

24. **Setfillstyle ( ):** setfillstyle function is used to set the color and styles to be filled into the objects using the floor fill method.
    Syntax: setfillstyle (STYLE, COLOR);
    E.g.: setfillstyle (2, BLUE);
25. **Floodfill ( ):** floodfill function is used to fill the color in the object; object can be a circle, triangle, octagon, rectangle or any other closed object.
    Syntax: floodfill (x co-ordinate, y co-ordinate, COLOR);
    E.g.: flood fill (100, 200, CYAN);
26. **Getcolor ( ):** getcolor function returns the current drawing color.
    Syntax: getcolor ( );
    E.g.: getcolor ( );

Space for learners:

CHECK YOUR PROGRESS

12. The use of Initgraph( ) function is to _____ the graphics system.
13. The use of closegraph( ) function is to _____ the graphics system.
14. The use of outtextxy ( ) function is to _____ in graphics mode..
15. The syntax for setfillstyle( ) function is to _____ to be filled into the objects.
16. The use of arc ( ) function is _____
17. The line( ) function is used to _____

3.10 COMPUTER AIDED DESIGN

CAD (Computer- aided Design) is the use of computer based software to aid in design processes. It main purpose is to optimize and streamline the designer's workflow, to improve the quality and level of detail in the design process and display the results.

3.10.1 Applications of CAD (Computer- Aided Design)

1. Designing a new aircraft or automobile
2. To design a new building or bridge
3. To design a Printed Circuit Board (PCB)

4. To design a new House (both interior and exterior)

Space for learners:

3.10.2 Advantages of CAD

1. Changes to a design can be made very quickly and easily.
2. The design can be viewed from any angle as we desire.
3. Designs can be tested without the need to build the prototype or model.
4. The design can be sent anywhere using the internet.
5. The design can be easily stored on a storage device to use for the future.
6. The design can be directly used in computer aided manufacturing processes.
7. Complex design can be made in a very convenient way.

3.11 SUMMING UP

- Graphics software are the software which have the capability to create, edit or display a graphic object.
- Graphics software have features like- photo editing, advanced font option freehand drawing, high quality templates, Art brushes, clip art, zooming, 3D capability and exporting & importing files.
- Adobe Photoshop, Illustrator, Paint Shop Pro, CorelDraw, Adobe LightRoom etc. are some popular graphics software.
- Graphics software can be classified as special packages and programming packages.
- Graphics programming packages functions are divided into the following categories: Output primitives, attributes, geometric and modeling transformations, viewing transformation, structure operations, input functions and control operations.
- GKS, PHIGS, PHIGS+ are some common standard graphics package
- PHIGS and GKS use the concept of workstation to specify that are used for input or output operation in a particular application.
- Special purpose application packages are designed mainly for non- programmers, so that the user need not to worry about its internal working mechanism.

- Artist painting program, business CAD system are examples of special purpose application packages.
- Graphics software mainly deals with two kind of files- one is the Raster Files like BMP (bitmap), JPEG, GIF, etc. Here every individual pixels can be modified and it is best for photograph or complex design images.
- Vector files mainly uses mathematical equations to determine how certain elements should be placed within the design. DXF and WMF are the e.g. of vector graphics file format.
- Vector files has advance of using less memory space than bitmap graphics. But vector files doesn't provide the appearance of a realistic image like bitmap.
- GKS or Graphics Kernel System is the first graphics software standard accepted by the international organization like ANSI.
- After GKS, PHIGS and PHIGS+ have been developed with some extensions to its previous.
- PHIGS has the ability to create and edit hierarchical application model.
- PHIGS+ is the extension version of PHIGS having additional functionality of providing three dimensional surface shading that was absent in PHIGS.
- Primitives are the building blocks of a picture or image, like Line, arc, etc.
- Some of the most important standard graphics functions are:- Polyline, Polymarker, FillArea, Text, cellArray.

Space for learners:

3.12 ANSWERS TO CHECK YOUR PROGRESS

1. a. support
2. graphs, charts.
3. Cartesian
4. raster, vector
5. matrix
6. Raster
7. mathematical equations
8. draw symbol of marker
9. fill a polygonal shape
10. draws line segments
11. draw a text string
12. initialize

13. shut down
14. display the text on the monitor screen
15. set the color and styles
16. draw an arc on the screen
17. to draw line on the screen

Space for learners:

3.13 POSSIBLE QUESTIONS

1. What is Graphics software?
2. What are the features of Graphics software?
3. Give some examples of Graphics software.
4. What are the applications of Graphics software?
5. What is the use of art brushes?
6. Explain the architectural design of both general graphics system designed for a programmer and specific application.
7. Write a routine to implement the -

   a. Polyline function b. Polymarker function c. text function d. FillArea function e. CellArray function
8. According to you which raster image or vector image is better to produce a photorealistic image. Give your reasons.
9. What types of Graphics Software can be classified?
10. What are the types of files the Graphics Software deals with?
11. Give some common file formats of Raster files.
12. Write one application of Vector Graphics Software.
13. What are the advantage and disadvantages of Raster Files?
14. What are the advantage and disadvantages of Vector Files?
15. What are the main advantages of PHIGS+ over GKS and PHIGS. Define PHIGS.
16. What do you mean by software standards and what are its needs?

17. What is GKS? What are its objectives?
18. Write the features of GKS.
19. What are the basic parts of GKS?
20. What are the advantages of GKS?
21. What are extra features that are available in PHIGS as compared to GKS?
22. What are the facilities available in PHIGS?
23. What additional functionality does PHIGS+ have?
24. What are the properties of graphics packages?
25. What are the standard Graphics function?
26. Give some examples of standard graphics functions.
27. What are the graphics functions available in c and c ++?
28. What is the difference between getpixel( ) and putpixel ( ) function?
29. What are the advantages of CAD?
30. Write a short notes on the application of CAD?
31. Why software standard are required? Explain.
32. which function will return the current drawing color?
33. what is the main difference between outtext( ) and outtextxy ( ) function?

Space for learners:

3.14 REFERENCES AND SUGGESTED READINGS

- Computer Graphics By Donald Hearn and M. Pauline Baker, Prentice Hall of India , 2004
- Evaluating PHIGS for CAD and general graphics applications T. L. J Howard

UNIT 4: LINE DRAWING ALGORITHMS AND MIDPOINT ALGORITHMS FOR CIRCLE AND ELLIPSE GENERATION

Space for learners:

Unit Structure:

4.1 INTRODUCTION

A straight line can be defined as the collection of some dots. In coordinate geometry a straight line can be defined using the straight-line equation:

Y = mx + c ---------- (1),

Where **m** represents the slope of the line, **c** is the **y** intercept. In computer graphics, line drawing can be defined as a straight line segment by plotting some discrete points between two endpoints. The line equation is used to determine the coordinates of discrete points. In computer graphics, the discrete points can be termed as PIXEL, the smallest element used to represent an object. PIXEL is the abbreviation of picture element. The computer system used binary numbers to control the display of PIXELS. For the bit value 1, the pixel becomes white, and for the value 0, the pixel turns black. By increasing the number of bits, the pixels can feel with different colors. Using two-bit binary numbers, a pixel can have three different colors: 0 0 – black, 0 1 – grey, 1 0 – grey, and 1 1 – white. The pixel position can be represented by the two-dimensional coordinate geometry. If the coordinate of a pixel is (X, Y), then the coordinate of the next pixel in the X direction will be (X + 1, Y), and in Y coordinate, the next position will be (X, Y + 1).

4.2 UNIT OBJECTIVES

After going through this unit, you will be able to:

- Learn about different parameters related to graphics in a computer system.
- Learn about different issues in line drawing algorithms.
- How to deal with the issues in drawing algorithms.
- How to draw a line using DDA line drawing algorithms.
- How to draw a line using Bresenham's line drawing algorithms.
- How to draw Circle using Midpoint algorithm.
- How to draw Ellipse using the Midpoint circle algorithm.

Space for learners:

4.3 POINTS AND LINES

Space for learners:

A point can be defined as a pixel in the computer graphics system. Plotting of points in frame buffer displays becoming essential. In a frame buffer system, the intensity of each pixel is calculated or measured separately. These are based on the Cartesian coordinate system. A point or a pixel is represented in terms of x and y coordinates. The total number of pixels in a display system depends on the coordinate precision and resolution of the system. The resolution of a system or a display device can be defined as the number of pixels or dots that can be displayed in a given area of the screen. It is usually measured in dot per inch or .dpi. Greater the number of pixels, the higher the resolution. Pixels are the smallest addressable screen element. Computer graphics system images are made by controlling the intensity and color of the pixels composing the screen. The following **figure** 4.1, it has been shown a straight line after plotting in coordinate positions. The appearance of the line is a jagged or stair-step pattern. For a system with low resolution, the appearance of straight lines or circles or any picture element will be jagged. But by increasing the number of points or resolution, the effect of stair-step can be avoided or eliminated. This is the main drawback of plotting a point in a computer graphics system. Using line drawing and circle drawing algorithms, it can be reduced.

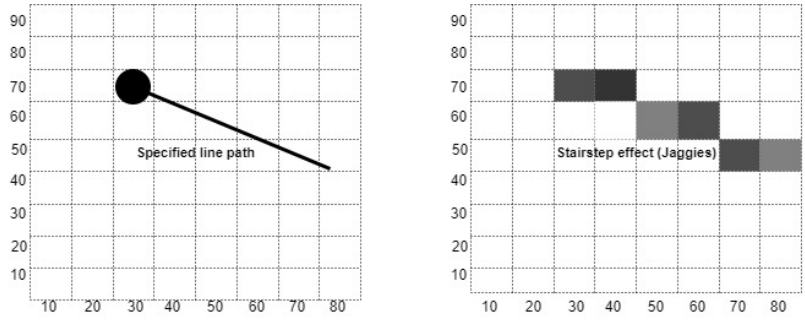


Figure 4.1 A line segment and *jagged* appearance of the line in graphics system

CHECK YOUR PROGRESS

1. A straight line can be defined as the collection of some _____.
2. The smallest element used to represent an object is _____.
3. Pixel becomes white if the bit value is _____ and black if it is _____.
4. The total number of pixels in a display system depends on the _____ _____and _____of the system.
5. The resolution of a system is measured in _____.
6. The smallest addressable screen element is _____.

Space for learners:

4.4. LINE DRAWING ALGORITHMS

The equation of a line is shown in the expression (1). Where the slope (***m***) can be calculated as:

$$m = \frac{y_2 - y_1}{x_2 - x_1} \quad \text{.....(2)}$$

$$c = y_1 - m.x_1 \quad \text{.....(3)}$$

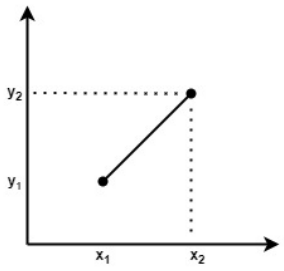


Figure 4.2: Line Segment with Negative Slope

For any given x interval Δx along a line, the corresponding y interval Δy can be calculated as

$$\Delta y = m.\Delta x \quad \text{.....(4)}$$

Similarly, the x interval Δx along a line, corresponding to a specified Δy as

$$\Delta x = \frac{\Delta y}{m} \quad \text{.....(5)}$$

These equations are the basis for determining the deflection voltages in analog devices.

4.5 SCAN CONVERSION OF LINES

Representation of a regular object in terms of discrete pixels is known as scan conversion. In a computer system, each graphics handler has to convert or transform the primitives into a collection of pixels. The method of scan conversion is also known as ***Rasterization***. Each pixel in a graphics system has two states – on or off. The horizontal and vertical frequency can change for different purposes, especially in video processing, which is known as the ***scan converting rate***.

For the scan conversion of a straight line, two endpoint coordinates are needed. In our normal life, we use scale and pencil to draw a line between two endpoints on a copy. But in a computer system, it is not possible to draw the line directly with the help of a roller. A computer can draw a line by calculating all the intermediate points in between the two endpoints. During calculation, it is possible to get fractional numbers for intermediate positions. But a computer system cannot accept a part of any pixel or a fractional number. So the fractions are rounded off to the nearest integers. Then the system fills the computed intermediate coordinate positions. A pixel never fills partially. So a straight line appearance always looks like stair-step format.

There are several algorithms used for this scan conversion. To draw a straight line – algorithms are the direct method, DDA algorithm, and Bresenham's algorithm. Similarly for drawing a circle Midpoint circle algorithm, Bresenham's Circle algorithm. To draw ellipse, midpoint ellipse drawing algorithm, etc.

4.6 DDA LINE DRAWING ALGORITHM

Space for learners:

Digital differential analyzer (DDA) is a scan conversion line algorithm based on calculating Δy or Δx. A line points are sample in unit interval along a coordinate and determine the corresponding integer value of the line path for the other coordinate.

Consider a line with a positive slope, ***(m ≤ 1)***. For a line with a positive slope, the gradient m is always less than or equal to one. If we sample the line path in ***x***-axis with the unit interval ($\Delta x = 1$), then the successive ***y*** value can be calculated using the following equation:

$$y_{k+1} = y_k + m \quad \text{..... } \mathbf{(6)}$$

Where the ***k*** values will start from 1 for the first point, and increase by 1 unit until the sample points reached the endpoint of the line. The value of ***m*** can range In between 0 and 1. The calculated ***y*** value must be rounded to the nearest integer.

For a line with a positive slope greater than 1, the role x and y will be reversed. That is the line path will be a sample in unit y intervals ($\Delta y = 1$), and calculate each successive x value as

$$x_{k+1} = x_k + \frac{1}{m} \text{.....(7)}$$

For the equation numbers (6) and (7), it is assumed that the line path starts from left endpoints to right endpoints. If the sampling is reversed, i.e. the starting endpoint starts from the right end and ended at the left, then the value of $\Delta x = -1$ and

$$y_{k+1} = y_k - m \text{.....(8)}$$

And when the slope is greater than 1, and $\Delta y = -1$, then

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

Equations (6) and (9) can be used to calculate the line path with a negative slope also. If the slope is less than 1, and the start endpoint is at left, set $\Delta x = 1$ and calculate the successive y value with equation (6). If the start endpoint is at right and the slope is positive and the slope value is less than one, set $\Delta x = -1$ and calculate the y value using the equation (8). Similarly if the absolute value of a negative slope is greater than 1, set $\Delta y = -1$ then we can use $\Delta y = -1$ in equation (9) and $\Delta y = 1$ in equation (7). The DDA algorithm is faster than equation (1) in the calculation of pixel position. The accumulation of round-off error in the calculation of successive

Space for learners:

point coordinates can drift away from the true line path from the original one. The performance of the DDA line drawing algorithm can be improved by separating the increments m and 1/m into integer and fractional parts so that all operations are restricted within the integer operation.

DDA Algorithm:

Example 1: Draw a line segment with the endpoint coordinates (0, 0) and (6, 6) using the DDA line drawing algorithm.

Sl. No.	X	Y
1	0	0
2	1	1
3	2	2
4	3	3
5	4	4
6	5	5
7	6	6

Solution: The starting point coordinate of the line is (0, 0) and

The right end point coordinate of the line is (6, 6)

So, $\Delta x = 6 - 0 = 6$ and $\Delta y = 6 - 0 = 6$, where x1=0, x2=6, y1=0 and y2=6

Thus the slope of the line $m = 6/6 = 1$

Since the slope of the line **m = 1**, and the line starts at the left endpoint and is processed to the right endpoint. So the next pixel coordinate along x-axes will be: $X_i = X_i + 1$

And the corresponding y coordinate of the pixel will be calculated $y_i = y_i + m$

The computed values for each pixel point on the line segment are showing in the table.

Example 2: Draw a line between the points (1, 1) and (8, 7) using the DDA line drawing algorithm.

Solution: The starting point is (1, 1), and the right endpoint is (8,7)

Sl. No.	X	Y	Pixel

Space for learners:

			plotted
1	**2**	**2**	**2, 2**
2	**3**	**2+6/7=2.9**	**3, 3**
3	**4**	**2.9+6/7=3.8**	**4, 4**
4	**5**	**3.8+6/7=4.7**	**5, 5**
5	**6**	**4.7+6/7=5.6**	**6, 6**
6	**7**	**5.6+6/7=7.0**	**7, 7**

The slope of the line m = (7 – 1)/(8 – 1) = 6/7 (slope m < 1)

Here it is given that x1=1, x2=8, y1=1 and y2=7

Since the slope m < 1, the x will increment in unit intervals and y has to be determined.

So the next x coordinate position along the line path will be x1 + 1 and

And the next y coordinate of the line path will be **y1 + m =y1 + 6/7** and so on. All the pixels on the line path are shown in the table:

4.7 BRESENHAM'S LINE ALGORITHM

It is the most efficient raster line–generating algorithm developed by Bresenham. In this algorithm, scan conversion is done by calculating the incremental integers which are used to display circles or any other curves. In the following **Figures** 4.3 and 4. 4, the horizontal axes identify the pixel column and the vertical axis shows the scan line positions. While the value of x is increase by one unit, then it h as to determine which of two possible pixel positions is closer to the line path at each sample step. In **figure** 4.3, it has to determine that, in the next pixel.

Space for learners:

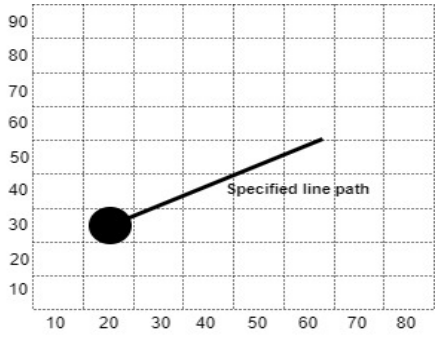


Figure 4.3: Section of a display screen where a positive slope line is plotted

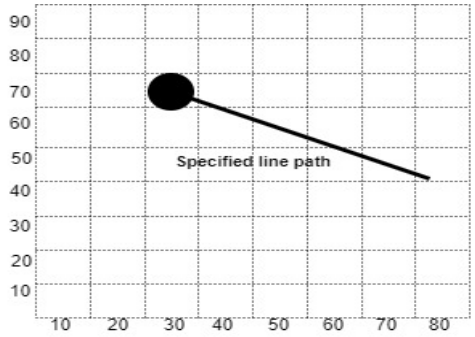


Figure 4.4: Section of a display screen where a negative slope line is plotted

Position whether to plot at the position (30, 30) or (30, 40). Similarly, **figure** 4.4 shows a line with a negative slope from the left endpoint at the coordinate position (30, 70). What will be the next pixel position, whether (40, 70) or (40, 60)? In Bresenham's algorithm, it can be determined by the help of a sign of an integer parameter, which value is proportional to the difference between the separations of the two-pixel positions from the actual line path.

Consider a line with a positive slope less than 1. Now, applying Bresenham's approach to determine the pixel positions along a line path by sampling at unit x intervals. Starting from the left endpoint (x_0, y_0) of a given line, step to each

Space for learners:

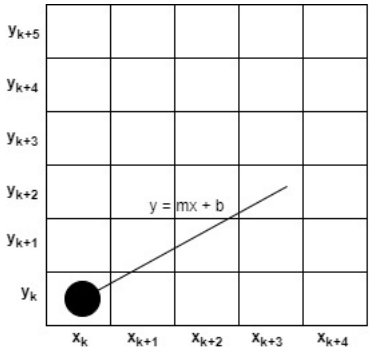


Figure 4.5 Section of a screen grid showing a pixel at (x_k, y_k) Position

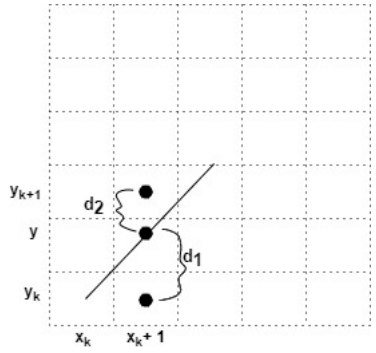


Figure 4.6 Distance between pixel position and the line at x_k+1 position

Successive pixel towards *x*-axis, and calculates the corresponding *y* position which is closer to the line path. **Figure** 4.5 and 4.6 depicts the k^{th} step in the algorithm. Now assume that the current pixel position is determined as (x_k, y_k) as shown in fig. 4.7. Next, need to determine the y coordinate for the pixel position at $x_k + 1$. The choices for the next pixel positions are ($x_k + 1$, y_k) and ($x_k + 1$, $y_k + 1$).

In **figure**: 4.6, at the sampling position $x_k + 1$, the vertical separation from the mathematical or actual line path is considered as d_1 and d_2. Now the ***y*** coordinate on the actual line at sampling position $x_k + 1$ can be calculated as

$$y = m(x_k + 1) + b \text{.....} (10)$$

Then,
$$d1 = y - y_k$$
$$= m(x_k + 1) + b - y_k$$
and

$$d2 = (y_k + 1) - y$$
$$= y_k + 1 - m(x_k + 1) - b$$

The difference between these two separations is

$$d1 - d2 = 2m(x_k + 1) - 2y_k + 2b - 1 \text{.....} (11)$$

Space for learners:

A decision parameter p_k for the k^{th} step in the line algorithm can be obtained from equation (11) by rearranging it. To remove the fractional part or to get an integer value, the ***m*** can be written as m = Δy / Δx, where the Δx and Δy are the vertical and horizontal separations of the endpoint positions. The p_k can be defined as:

$$d1 - d2 = 2\frac{\Delta y}{\Delta x}(x_k + 1) - 2y_k + 2b - 1$$

$$\Delta x(d1 - d2) = 2\Delta y(x_k + 1) - 2\Delta x.y_k + 2b.\Delta x - \Delta x$$

$$\therefore p_k = \Delta x(d1 - d2)$$

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

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

$$= 2\Delta y.x_k - 2\Delta x.y_k + c \quad \text{.....(12)}$$

(where c is a constant, and $c = 2\Delta y + \Delta x(2b - 1)$)

The sign of the $d_1 - d_2$ will be the same as the sign of decision parameter p_k. In our example, Δx > 0, and c is constant and independent of pixel position.

If the position y_k is closer to the actual line path, then the pixel at $y_k + 1$, i.e. $d_1 < d_2$, then the sign of the decision parameter p_k will be – ve, and the lower pixel have to be chosen. The coordinate changes occur in the unit step in either x or y directions. So the successive decision parameter p_{k+1} can have through incremental integer calculations. At step k + 1, from the equation number (12), the decision parameter can be computed as:

$$p_{k+1} = 2\Delta y.x_{k+1} - 2\Delta x.y_{k+1} + c \text{.....(13)}$$

Now subtracting the equation (12) from (13) we have

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

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

Since $x_{k+1} = x_k + 1$, the successive decision parameter for the next pixel position will be:

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

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

$$= p_k + 2\Delta y - 2\Delta x(y_{k+1} - y_k) \text{.....(14)}$$

Space for learners:

The value of the term $y_{k+1} - y_k$ is either 0 or 1, depending on the sign of the decision parameter p_k. If y_k is closer than y_{k+1}, the sign of the decision parameter p_k is negative. At this moment we have to choose the y_k value for the y coordinate y_{k+1}, thus the value for the expression $y_{k+1} - y_k = 0$. If y_{k+1} is closer to the actual line path, the decision parameter will be positive and we have to choose the y_k+1 value for the coordinate position y_{k+1}. Thus the value of $y_{k+1} - y_k = 1$ for a decision parameter with a positive sign. The calculation of decision parameters has to perform recursively to get each integer *x* position starting from the left end of the line path. The value of decision parameter p_k for the initial or the starting coordinates (x_0, y_0) can obtain from the equation (12)

$$p_0 = \Delta x[2m(x_0+1) + 2b - 2y_0 - 1]$$
$$= \Delta x[2(mx_0 + b - y_0) + 2m - 1]$$
$$= \Delta x.0 + 2.\frac{\Delta y}{\Delta x} - 1 \quad \text{Where } mx_0 + b - y_0 = 0 \text{ and } m = \frac{\Delta y}{\Delta x}$$
$$= 2\Delta y - \Delta x$$

STOP TO CONSIDER

i. Slope direction can be positive, negative, zero or undefined.
ii. Positive slope – y value increases as the x values increases. The line slope upwards to the right.
iii. Negative slope – y value decreases as the x values increases. The line slope downwards to the right.
iv. Zero slope – y values does not change with the changing of x value. The line in exactly horizontal.
v. Undefined slope – When the line exactly vertical, it does not have a definition.

4.7.1 Steps Involved in Bresenham's Line Drawing Algorithm

1. Required two pixels coordinates as input. Consider the left endpoint coordinate as (x_0, y_0).
2. (x_0, y_0) will be the first point in the line path

Space for learners:

3. Compute the values for the terms: **Δy, Δx, 2Δy, 2Δx**, and **2Δy - 2Δx** and $\mathbf{p_0}$the decision parameter for the initial point of the line path.
4. For each pixel coordinate x_k along the line path, the following steps will perform recursively until the right endpoint. Initially k=0 or **Δx** times.

   4.1 if $p_k < 0$, the next point to be plot at position $(x_k + 1, y_k)$ and

   $p_k+1 = p_k + \mathbf{2\Delta y}$

   4.2 otherwise, the next point to be plot at position $(x_k + 1, y_k + 1)$ and

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

Example: Draw a line with the endpoints (20, 10) and (30, 18), using Bresenham's line drawing algorithm.

Solution: Here it is given that the starting point coordinate is (20, 10), and the right endpoint is (30, 18).

So the $\Delta x = 30 - 20 = 10$ and $\Delta y = 18 - 10 = 8$

Initial decision parameter $p_0 = 2\Delta y - \Delta x = 16 - 10 = 6$

And $2\Delta y = 16$, $2\Delta x = 20$

Therefore $2\Delta y - 2\Delta x = 16 - 20 = -4$

Initially plotting the pixel at (20, 10) and determines the successive pixel positions along the line path using the decision parameter is shown in the following table.

Space for learners:

K	P_k	(x_{k+1}, y_{k+1})		K	P_k	(x_{k+1}, y_{k+1})
0	6	(20, 11)		5	6	(26, 15)
1	2	(22, 12)		6	2	(27, 16)
2	-2	(23, 12)		7	-2	(28, 16)
3	14	(24, 13)		8	14	(29, 17)
4	10	(25, 14)		9	10	(30, 18)

Example: Draw a line using Bresenham's Line algorithm with the endpoint coordinates (1, 1) and (8, 5).

Solution: Here it is given that the starting point coordinate is (1, 1), and the right endpoint is (8, 5).

So the $\Delta x = 8 - 1 = 7$ and $\Delta y = 5 - 1 = 4$

Initial decision parameter $p_0 = 2\Delta y - \Delta x = 8 - 7 = 1$

And $2\Delta y = 8, 2\Delta x = 2$

Therefore $2\Delta y - 2\Delta x = 8 - 14 = -6$

Initially plotting the pixel at (1, 1) and determines the successive pixel positions along the line path using the decision parameter is shown in the following table.

K	P_k	(x_{k+1}, y_{k+1})		K	P_k	(x_{k+1}, y_{k+1})
0	1	(1, 1)		5	-1	(5, 3)
1	-5	(1, 1)		6	7	(6, 4)
2	3	(2, 2)		7	1	(7, 4)
3	-3	(3, 2)				
4	5	(4, 3)				

For each value of p_k,

Space for learners:

if $p_k < 0$, next point is plotted at (x_k+1, y_k), otherwise, at (x_k+1, y_k+1)

Space for learners:

4.7.2 Advantages and Disadvantages of Bresenham's Line Drawing Algorithm

Bresenham's line drawing algorithm overcomes the drawbacks of the DDA line drawing algorithm. It is implemented using an incremental method for scan conversion lines than the DDA algorithm. It is faster and more efficient than the DDA line drawing algorithm. It uses only integer calculations such as addition, subtraction, and bit shifting operation to give an accurate result. The calculation for the terms $2\Delta y - 2\Delta x$ and $2\Delta y$ are computed once for each line. This algorithm can be used to generate circles and different curves.

The main drawback of this algorithm is that it produces a jagged or stair-step line display.

CHECK YOUR PROGRESS

7. Representation of a regular object in terms of discrete pixels is known as _____ _____.
8. Scan conversion is also known as _____.
9. Each pixel in a graphics system has two states – _____ or _____.
10. Scan converting rate changes the _____ and _____frequency.
11. Bresenham's line drawing algorithm is implemented using an _____ method for scan conversion lines.

4.8 CIRCLE GENERATING ALGORITHMS

The circle is a common picture element used in drawing pictures and graphs. A full circle or arcs of a circle can be generated through a procedure in a computer graphics system.

4.8.1 Properties of a Circle

Space for learners:

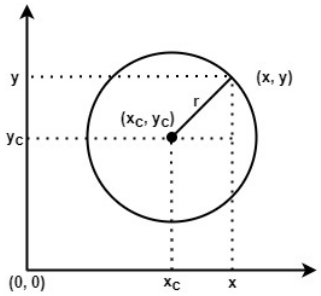


Figure 4.7 Circle with center coordinate (x_c, y_c) and radius r

A circle can be defined as a set of points that are all at a given distance r from a given point (x_c, y_c) as shown in **figure** 4.7. The point is known as the ***center*** of the circle and the equal distance of each point from the center is known as the **radius** (r) of the circle. The complete length of the circle arc or the boundary of the circle is known as the diameter. The angle forms at the center are measured 360^0. A straight line from the circle boundary to the opposite boundary across the center is called the diameter. The diameter of a circle is always measured as two times the circle radius, i.e. diameter d = 2 * radius(r). The relation between radius (r) and the center (c) can be expressed in terms of Pythagorean Theorem in Cartesian coordinate as

$$(x-x_c)^2+(y-y_c)^2=r^2 \quad \text{.....(15)}$$

Using the equation (15), it is possible to calculate the coordinate positions of circle circumference by stepping along the x-axis in unit steps from (x_c - r) to (x_c + r) and calculate the corresponding **y** coordinate for each pixel using the following equation:

$$y = y_c \pm \sqrt{r^2-(x_c-x)^2} \quad \text{.....(16)}$$

This process of circle generation requires considerable computation at each step and the spacing between the plotted pixels is not uniform. This non-uniformity can reduce by calculating the circle boundary points using polar coordinates r and θ. The expression used to calculate the circle boundary pixel position is:

$$\left.\begin{aligned} x &= x_c + r\cos\theta \\ y &= y_c + r\sin\theta \end{aligned}\right\} \quad \text{.....(17)}$$

Another method to reduce the computation steps is the consideration of the symmetry of a circle. The shape of a circle is similar in each quadrant and each octant. We can generate the circle section in the second quadrant of the ***xy*** plane by noting that the two circle sections are symmetric to the y axis. The circle sections in the third and fourth quadrant can obtain from sections in the first and second quadrants considering symmetry to the x-axis. Circle sections in adjacent octants are symmetric about the 45^0 lines dividing the two octants.

Calculation of pixel positions along a circle boundary or circumference using (15) and (17) still requires a remarkable computation time. The Cartesian equation (15) is required to perform multiplication and square root operations, while equation (17) requires multiplications and trigonometric calculations. An efficient algorithm is generally based on the incremental calculation of decision parameters. One efficient algorithm to generate a circle is the Midpoint Circle Algorithm.

Space for learners:

4.8.2 Midpoint Circle Algorithm

As we have seen in the raster line algorithm, the sampling has done at unit intervals and determined the closest pixel position to the specified circle path at each step. Considering the radius r and the center (x_c, y_c), let us set the algorithm to calculate the circle boundary pixel positions centered at the coordinate origin (0, 0). Then each calculated position(x, y) is moved to its proper screen position by adding x_c to x and y_c to y. Along the circle section from x=0 to x=y in the first quadrant, the slope of the curve varies from 0 to -1. Therefore, we can take unit steps in the positive x-direction over this octant and use a decision parameter to determine which of the two possible y positions is closer to the circle path at each step. Positions in the other seven octants can obtain by symmetry.

For applying the midpoint circle algorithm, the definition of circle function can be express as:

$$f_{circle}(x,y) = x^2 + y^2 - r^2 \quad \text{.....(18)}$$

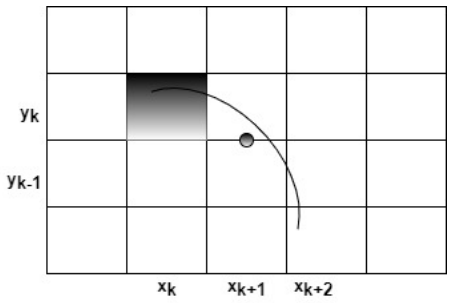


Figure 4.8: Midpoint between candidate pixels at sampling position $x_k + 1$ along the circular path

Any point on the circle circumference with radius r satisfies the equation $f_{circle}(x, y) = \mathbf{0}$. For an interior point, the circle function gives a negative value. If the pixel position is outside of the circle boundary, the circle function gives a positive value. Thus checking the sign of the circle function, the position of (x, y) point can be determined. The following **figure** 4.8 shows the midpoint between two candidate pixels at sampling position $x_k + 1$. Assume that the currently plotted pixel is (x_k, y_k). We need to determine the next pixel position, whether the pixel position $(x_k + 1, y_k)$ or $(x_k + 1, y_k - 1)$ is closer to the circle path. The circle function from the equation (18), we can have the decision parameter to determine the midpoint between these two-pixel positions.

$$p_k = f_{circle}(x_k + \mathbf{1}, y_k - \frac{\mathbf{1}}{\mathbf{2}})$$

$$= (x_k + \mathbf{1})^2 + (y_k - \frac{\mathbf{1}}{\mathbf{2}})^2 - r^2 \quad \text{.....(19)}$$

If $\mathbf{p_k < 0}$, the midpoint is inside the circle. And the pixel on scan line $\mathbf{y_k}$ is closer to the circle boundary.

Otherwise, the mid position is outside or on the circle boundary, and select the pixel position $y_k - 1$.

Successive decision parameter scan be obtained using incremental calculations. We obtain a recursive expression for the next decision parameter by evaluating the circle function at sampling position $x_{k+1} + 1 = x_k + 2$:

Space for learners:

$$p_{k+1} = f_{circle}(x_{k+1}+1, y_{k+1}-\frac{1}{2})$$

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

$$= p_k + 2(x_k+1) + (y^2{}_{k+1} - y^2{}_k) - (y_{k+1} - y_k) + 1 \quad \text{.....(20)}$$

Where y_{k+1} is either y_k or y_k-1 depending on the sign of p_k.

If p_k is negative, then $y_{k+1} = y_k$, **and from** the equation (20), we can calculate the successive p_{k+1}:

$$p_{k+1} = p_k + 2x_{k+1} + 1 \quad \text{.....(21)}$$

Otherwise, $y_{k+1} = y_k - 1$, **and**

$$p_{k+1} = p_k + 2x_{k+1} + 1 - 2y_{k+1} \quad \text{.....(22)}$$

Where
$$2x_{k+1} = 2x_k + 2$$
$$2y_{k+1} = 2y_k - 2$$

Initially, if the starting coordinate is (0, r), then these two terms have the value 0 and 2r respectively. Each successive value is obtained by adding 2 to the previous value of *2x* and subtracting 2 from the previous value of 2y. The initial decision parameter is obtained by evaluating the circle function for the starting coordinate position (x_0, y_0) = (0, r):

$$p_0 = f_{circle}(0+1, r-1/2)$$

$$=1^2 + (r-1/2)^2 - r^2$$

$$= \frac{5}{4} - r \quad \text{.....(23)}$$

If the radius r is specified as an integer, then the value of p_0 can be calculated as:

$$p_0 = 1 - r$$

As in Bresenham's line drawing algorithm, the midpoint circle drawing algorithm calculates the pixel positions on circle boundary using integer addition and subtractions. The summary of the midpoint circle algorithm is as follows:

4.8.3 Steps in Midpoint Circle Algorithm

Space for learners:

1. Consider a circle with radius r, and center coordinate (xc, yc). Let the first point on the circumference of the circle centered at the coordinate origin (0, 0) is

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

2. Calculate the initial decision parameter $p_0 = 5/4 - 1$
3. At each x_k position, k starts from 0, perform the following operations :

   3.1 if $p_k < 0$, the next point on the circle circumference will be (x_{k+1}, y_k) and

   $$p_{k+1} = p_k + 2x_{k+1} + 1$$

   Otherwise, the next point on the circle path is $(x_k + 1, y_k - 1)$ and

   $$p_{k+1} = p_k + 2x_{k+1} + 1 - 2y_{k+1}$$

   Where $2x_{k+1} = 2x_k + 2$, $2y_{k+1} = 2y_k - 2$

4. Need to determine the symmetry points for the other remaining octants.
5. Move each calculated pixel position (x, y) onto the circle boundary centered on (xc, yc) and plot the coordinate values: $x = x + x_c$ and $y = y + y_c$
6. Repeat the steps from 3 to 5 until $x \geq y$.

Example: Show the pixel pints on the circle boundary in the first quadrant, centered at the coordinate origin (0, 0) and the radius r = 10.

Solution: initial decision parameter $p_0 = 1 - r = 1 - 10 = 9$

Let the initial point of the circle is $(x_0, y_0) = (0, 10)$

Therefore the term $2x_0 = 0$ and $2y_0 = 20$

So the successive pixel coordinates along the circle path are:

Space for learners:

k	P_k	(x_{k+1}, y_{k+1})	$2x_{k+1}$	$2y_{k+1}$
0	-9	(1, 10)	2	20
1	-6	(2, 10)	4	20
2	-1	(3, 10)	6	20
3	6	(4, 9)	8	18
4	-3	(5, 9)	10	18
5	8	(6, 8)	12	16
6	5	(7, 7)	14	4

Example: Find the points on a circle on of its octants with the circle centered at (5, 5) and has a radius of 8 units.

Solution: Assume that the circle is centered at the origin and proceed to solve. After finding the points add center (5, 5) to each point.

The initial point $(x_0, y_0) = (0, 8)$

$P_0 = 1 - r = 1 - 8 = -7$

k	P_k	at origin (0, 0) (x_{k+1}, y_{k+1})	At origin (5, 5) (x_{k+1}, y_{k+1})
0	-7	(1, 8)	(6, 13)
1	-4	(2, 8)	(7, 13)
2	1	(3, 7)	(8, 12)
3	-6	(4, 7)	(9, 12)
4	3	(5, 6)	(10, 11)
5	2	(6, 5)	(11, 10)

4.9 ELLIPSE GENERATING ALGORITHM

Ellipse can be defined as a set of points such that the sum of the distances from two fixed positions known as ***foci*** is the same for all points. If the distances between the point p = (x, y) to the foci are d1 and d2 then the general equation of an ellipse can be stated as:

Space for learners:

d1 + d2 =constant ……. (24)

If the coordinates of the foci are f1(x_1, x_2) and f2(x_2, y_2), then the equation becomes

$$\sqrt{(x-x_1)^2+(y-y_1)^2}+\sqrt{x-x_2)^2+(y-y_2)^2}=cons\tan t.....(25)$$

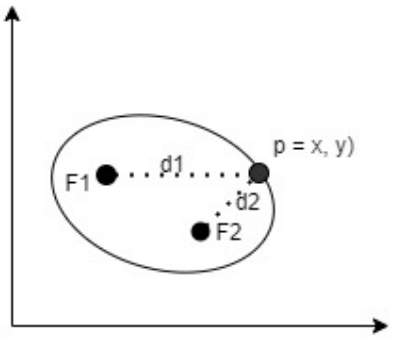


Figure 4.9: Ellipse at foci F1 and F2

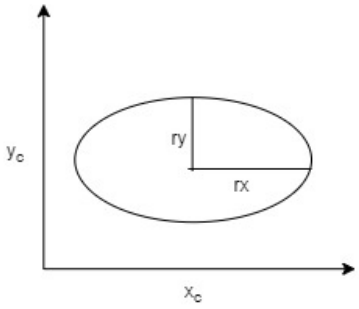


Figure 4.10: Ellipse centered at (x_c, y_c), with semi major axis r_x and semi minor axis r_y.

The major axis of an ellipse is the straight line from one end of the ellipse boundary to the other across the foci. A straight line from one end of the ellipse at the shortest diameter part to the other end, intersecting the major axis at the middle is known as the minor axis. The intersecting point is known as the ellipse center between the two foci.

Space for learners:

The ellipse equation can be expressed in terms of major and minor axes. It becomes more simplified if the axes orientation is aligned with the coordinate axes. The following **figure** 4.10, it has been shown a standard position of an ellipse; where the major and minor axes are aligned parallel to the ***x*** and ***y*** axes. The semi-major and semi-minor axes are represented by the labels $\mathbf{r_x}$ and $\mathbf{r_y}$. The ellipse equation can be written in terms of center coordinates (x_c, y_c), semi-major and minor axes is as follows:

$$\left(\frac{x-x_c}{r_x}\right)^2+\left(\frac{y-y_c}{r_y}\right)^2=\mathbf{1}.....(\mathbf{24})$$

Using polar coordinates r and θ, the equation of an ellipse can be expressed as:

$$\left.\begin{aligned}x&=x_c+r_x\cos\theta\\y&=y_c+r_y\sin\theta\end{aligned}\right\}.....(\mathbf{25})$$

Using symmetry concepts, the steps required in computations can be reduced. An ellipse does not show the symmetric property in between octants. It shows symmetric property only on quadrants. So during the computation of boundary path pixel coordinates, need to calculate the pixel positions along the elliptical arc across one quadrant. Then the position of the pixels in the other quadrant can be calculates using symmetry.

4.9.1 Midpoint Ellipse Algorithm

The goal of this algorithm is to display an ellipse. With the given parameters r_x and r_y and center coordinates (x_c, y_c) we have to determine the pixel coordinates (x, y) for an ellipse in a standard position centered at the coordinate origin and finally shift to the points such that the ellipse centered at (x_c, y_c). To display the ellipse in a non-standard position, the ellipse has to rotate about its center

Space for learners:

coordinates to re-orient the major and minor axes. Here we will consider the only standard position to display the ellipse.

The Midpoint ellipse algorithm is applied throughout the first quadrant in two parts. Following **figure 4.11** shows the division of the first quadrant according to the slope of an ellipse with $r_x < r_y$. We have to process this quadrant by taking unit steps in the x – the direction where the slope of the curve has a magnitude less than 1 and taking unit steps in the y-direction where the slope has a magnitude greater than 1. In **figure 4.12**, we can start our plotting from the point **(0, r_y)** and step clockwise direction along the elliptical path in the first quadrant, shifting from unit steps in **x** to unit steps in **y** when the slope becomes less than **-1**.

Now consider the ellipse function derived from the equation (24) with $(x_c, y_c) = (0, 0)$ as:

$$f_{ellipse}(x,y) = r^2{}_y x^2 + r^2{}_x y^2 - r^2{}_x r^2{}_y \quad \text{.....(26)}$$

This has the following properties:

$$f_{ellipse}(x,y)\begin{cases} < 0, if\ (x,y)\ is\ inside\ the\ ellipse\ boundary \\ = 0, if\ (x,y)\ is\ on\ the\ ellipse\ boundary \\ > 0, if\ (x,y)\ is\ outside\ the\ ellipse\ boundary \end{cases} \quad \text{.....(27)}$$

Thus, the ellipse function can be used as a decision parameter in the midpoint algorithm to generate an ellipse. At each sampling position, the next pixel position along the ellipse boundary can be determined using the sign of the decision parameter or the ellipse function, evaluated at the midpoint between the two candidate pixels.

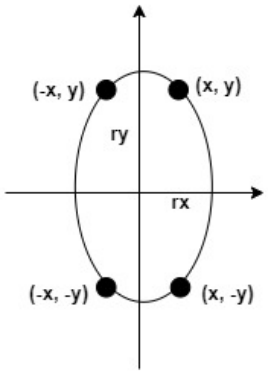


Figure 4.11: Symmetry of an ellipse

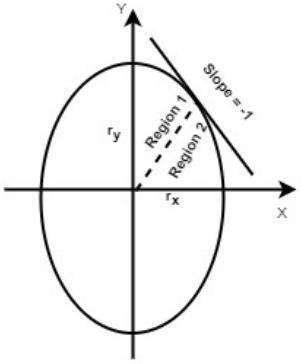


Figure 4.12: Ellipse processing regions

Space for learners:

Space for learners:

Starting at $(0,r_y)$ we take unit steps in the x-direction until we reach the boundary between the region 1 and 2 as shown in the following **figure 4.12**. Then we have to shift the unit steps in the y-direction over the remainder of the curve in the first quadrant. At each step, we need to test the value of the slope of the curve. The ellipse slope can be calculated as:

$$\frac{dy}{dx} = -\frac{2r^2_y x}{2r^2_x y} \quad \text{.....(28)}$$

At the boundary between region 1 and region 2, dy/dx = - 1 and

$$2r^2_y x = 2r^2_x y$$

Therefore, we have to move out from region 1 when, $2r^2_y x \geq 2r^2_x y$

F**igure** 4.13 shows the midpoint between the two candidate pixels at sampling position x_k+1 in the first region. Assume that the point selected in the previous step is (x_k, y_k), and now we have to determine the next position along the ellipse path by evaluating the decision parameter at this midpoint.

$$p1_k = f_{ellipse}(x_k + 1, y_k - \tfrac{1}{2})$$

$$= r^2_y (x_k + 1)^2 + r^2_x (y_k - \tfrac{1}{2})^2 - r^2_x r^2_y \quad \text{.....(29)}$$

If $\mathbf{p1_k < 0}$, the midpoint is inside the ellipse boundary and the pixel on scan line $\mathbf{y_k}$ is closer to the ellipse boundary. Otherwise, the midpoint will remain on or outside the ellipse and the point on scan line $\mathbf{y_k - 1}$ will be closer to the ellipse boundary.

Space for learners:

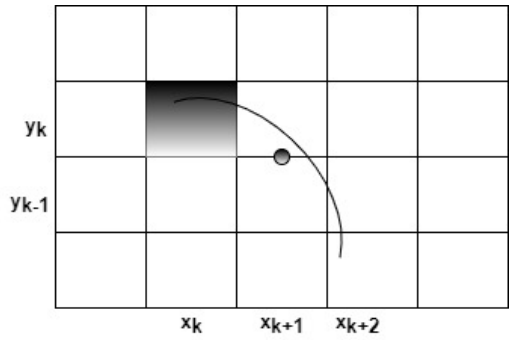


Figure 4.13: Midpoint between candidate pixels at x_k+1 position along an ellipse line path

At the next sampling position ($x_{k+1} + 1 = x_k + 2$), the decision parameter for region 1 is evaluated as:

$$p1_{k+1} = f_{ellipse}(x_{k+1} + 1, y_{k+1} - \tfrac{1}{2})$$

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

$$= p1_k + 2r^2_y(x_k + 1) + r^2_y + r^2_x\left[\left(y_{k+1} - \tfrac{1}{2}\right)^2 - \left(y_k - \tfrac{1}{2}\right)^2\right]$$

Where y_{k+1} is either y_k or $y_k - 1$, depending on the sign of decision parameter $\mathbf{p1_k}$.

If $p1_k < 0$, then y_{k+1} is y_k, so the next point along the ellipse centered on (0, 0) is (x_{k+1}, y_k)

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

Otherwise, the next point along the ellipse is $(x_k + 1, y_k - 1)$

$$p1_{k+1} = p1_k + 2r^2_y x_{k+1} - 2r^2_x y_{k+1} + r^2_y$$

Decision parameters are incremented by the following amounts:

$$increment = \begin{cases} 2r^2_y x_{k+1} + r^2_y, & if\ p1_k < 0 \\ 2r^2_y x_{k+1} + r^2_y - 2r^2_x y_{k+1}, & if\ p1_k \geq 0 \end{cases}$$

In region 1, the initial value of the decision parameter is obtained by evaluating the ellipse function at the start position $(x_0, y_0) = (0, r_y)$;

$$p1_0 = f_{ellipse}(1, r_y - 1/2)$$
$$= r^2_y + r^2_x (r_y - 1/2)^2 - r^2_x r^2_y$$
$$= r^2_y - r^2_x r_y + 1/4\, r^2_x \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (30)$$

In region 2, we sample at unit steps in the negative y-direction, and the midpoint is now taken between horizontal pixels at each step. For this region the decision parameter is evaluated as:

$$p2_k = f_{ellipse}(x_k + 1/2, y_k - 1)$$
$$= r^2_y (x_k + 1/2)^2 + r^2_x (y_k - 1)^2 - r^2_x r^2_y \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (32)$$

If $p2_k > 0$, the mid position is outside the ellipse boundary and we have to select the pixel x_k.

If $p2_k \leq 0$, the mid position is inside or on the ellipse boundary and we have to select the pixel x_{k+1}.

4.9.2 Steps Involved in Midpoint Ellipse Algorithm

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

   $$p1_0 = r^2_y - r^2_x r_y + 1/4\, r^2_x$$

3. At each xk position in region 1, starting at k=0, perform the following test:

   If $p1_k < 0$, then y_{k+1} is y_k, so the next point along the ellipse centered on (0, 0) is (x_{k+1}, y_k)

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

   Otherwise, the next point along the ellipse is $(x_k + 1, y_k - 1)$

Space for learners:

$$p1_{k+1} = p1_k + 2r^2_y x_{k+1} - 2r^2_x y_{k+1} + r^2_y$$

And continue until $2r^2_y x \geq 2r^2_x y$

4. In region 2, calculate the initial value of the decision parameter using the last point (x0, y0) calculated in region 1 as::

$$p2_0 = r^2_y (x_0 + 1/2)^2 + r^2_x (y_0 - 1)^2 - r^2_x r^2_y$$

5. At each yk position in region 2, starting at k=0, perform the following test:

$$p2_{k+1} = p2_k + 2r^2_y x_{k+1} - 2r^2_x y_{k+1} + r^2_x$$

Using the same incremental calculations for x and y as in region 1.

6. Determine the symmetry points in the other three quadrants.
7. Move each calculated pixel position (x, y) onto the elliptical path centered on (x_c, y_c) and plot the coordinate values: x = x + x_c y = y + y_c
8. Repeat the steps for region 1 until $2r^2_y x \geq 2r^2_x y$

4.10 SUMMING UP

1. In a graphics package, the functions which are used to describe the various picture components are known as graphics output primitives or simply primitives.
2. A straight line can express using the expression y = mx + c, where m denotes the slope of the line and c is a constant.
3. The slope of a line can be defined as Δy/Δx.
4. Digital Differential analyzer (DDA) algorithm can be used to draw a straight line in computer graphics.
5. DDA used an incremental approach to calculate the pixel coordinates on the line path.

Space for learners:

6. In the DDA algorithm, the round-off operation and floating-point arithmetic consume time.
7. A highly accurate and efficient line drawing algorithm is Bresenham's line drawing algorithm.
8. The sign of the decision parameter determines the coordinate position of the next pixel on the line path.
9. Bresenham line algorithm gives a jagged or stair-steps appearance to the displayed line.
10. Using the Bresenham algorithm, we can generate a circle also.
11. One of the most used and efficient circles generating algorithms is the Midpoint Circle algorithm.
12. An ellipse can be defined as a set of points located in a distance from two fixed points, where the sum of the distances from each point to the boundary points are equal. The specific points are known as the foci of the ellipse.
13. The major axis of an ellipse connects both the end of the ellipse across the foci.
14. The minor is a straight line segment drawn at the middle of the narrowest part of an ellipse by intersecting the major axes. The intersecting point is known as the center of the ellipse.
15. The circle can divide into eight symmetric segments.
16. An ellipse can divide into four symmetric segments.

Space for learners:

4.11 ANSWERS TO CHECK YOUR PROGRESS

1. dots
2. Pixel
3. 1, 0
4. coordinate precision, resolution
5. dots per inch
6. pixel
7. scan conversion
8. Rasterization
9. on, off
10. horizontal, vertical
11. incremental

4.12 POSSIBLE QUESTIONS

1. What do you mean by graphics output primitives?
2. Explain the DDA line algorithm.
3. Write the merits and demerits of the DDA algorithm.
4. Derive the equations for Bresenham's line algorithm.
5. What are the advantages and disadvantages of Bresenham's line algorithm?
6. Explain the midpoint circle algorithm.
7. How the decision parameter is used in the midpoint circle algorithm, explain.
8. What do you mean by the slope of a line? Explain different slope properties with a suitable diagram.
9. Define ellipse? Write any two properties of an ellipse.
10. Define a circle. Also, write at least four properties of a circle.
11. Explain the midpoint ellipse algorithm.
12. Draw a line segment using DDA line drawing algorithm with the starting coordinate (30, 25) and the endpoint coordinate (10, 20).
13. Draw a line segment using Bresenham's line drawing line drawing algorithm with the starting coordinate (30, 25) and the endpoint coordinate (10, 20).
14. Show the set of points of a circle boundary in first quadrant using Midpoint circle drawing algorithm with the center coordinate (3, 3) and the radius r= 12.
15. Show the set of points of a circle boundary in first quadrant using Midpoint circle drawing algorithm with the center coordinate (0, 0) and the radius r= 7.
16. Using midpoint ellipse algorithm, draw the boundary points of an ellipse with $r_x = 8$ and $r_y = 6$.

Space for learners:

4.13 REFERENCES AND SUGGESTED READINGS

- Introduction To Computer Graphics And Mu, by Anirban Mukhopadhyay and Arup Chatterjee, Vikas Publishing House Pvt. Ltd.
- Computer Graphics: C version by Donald Hearn & M. Pauline Baker.

Space for learners:

UNIT 5: AREA FILLING ALGORITHMS AND CHARACTER GENERATION TECHNIQUES

Space for learners:

Unit Structure:

5.1 INTRODUCTION

Here in this unit, we shall discuss about types of polygons and several algorithms which are used to fill color to the interior pixels of the polygons like scan line, flood fill and boundary fill methods. Non-zero winding method will be applied to find out whether a given point lies inside or outside of a polygon. Here we shall also discuss about various character generation methods.

5.2 UNIT OBJECTIVES

After completion of this unit, you shall be able to

- understand polygons
- know non-zero winding number method
- understand the basics of scan line polygon fill, flood fill, boundary fill
- know how characters are generated.

Space for learners:

5.3 AREA-FILLING ALGORITHMS

Filling an area means coloring an image or region with a specific color. Area can be defined by a set of boundary pixels. Several algorithms are available to fill color to the interior pixels of an image. Points, lines, circles are generated and discussed; here we shall be discussing about the polygons and different ways to fill them.

Polygons- A polygon is a shape which is closed by at least three straight lines connected together forming angles and edges. A polygon may have many edges or sides, these edges are straight lines are connected to each other through vertices. A triangle, square, rectangle, hexagon etc. are all polygons. Any polygon can be represented by number of line segments. Polygon filling means coloring the pixels of the polygon which lies inside the boundary of the polygon. Depending upon the shape of the polygon they are grouped into convex, concave and complex polygons. Some polygon shapes are shown on the figure 5.1 below.

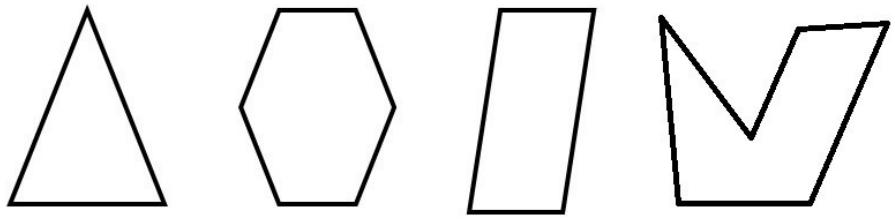


Fig. 5.1: Different shapes of polygons

Convex polygon-If the line connecting two interior points of the polygon lies completely inside the polygon, then it is said to be a convex polygon. In convex polygon all the interior angles are less than 180 degree which makes the vertices pointing towards outside. In the

figure 5.2 given below, shows the line segment AB lies completely inside the polygon.

Space for learners:

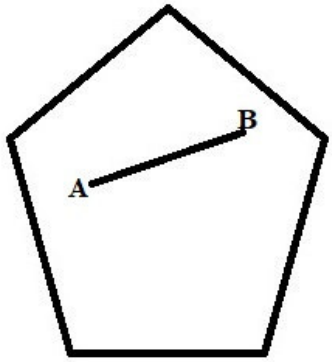


Fig 5.2: Convex Polygon.

Concave polygon-If the line joining any two points of the polygon is not completely inside the polygon is said to be concave polygon. In concave polygons at least one interior angle is more than 180 degree. In the figure 5.3 given below, shows the line segment AB lies partially inside the polygon.

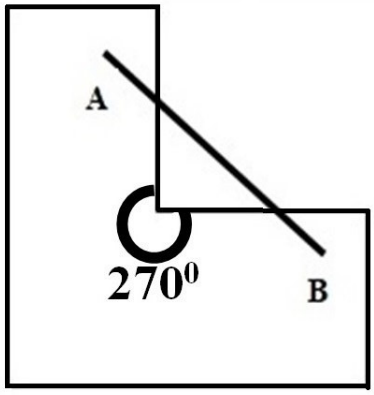


Fig 5.3: Concave Polygon

Complex polygon- Polygons whose edges intersects the other edges one or more time in the same polygon are termed as complex polygons. In the figure 5.4 below, the edges AE and BE, intersects the edge CD.

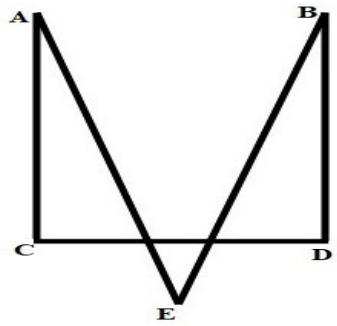


Fig 5.4: Complex Polygon

The polygon can also be categorized based on the direction of visit to their vertices, if the visit is performed in clockwise direction then the polygon is said to be a negative oriented polygon otherwise if the movement is in anticlockwise direction then the polygon is said to be positive oriented. The image of a positive oriented and a negative oriented polygon are shown in figure 5.5(a) and 5.5(b) respectively.

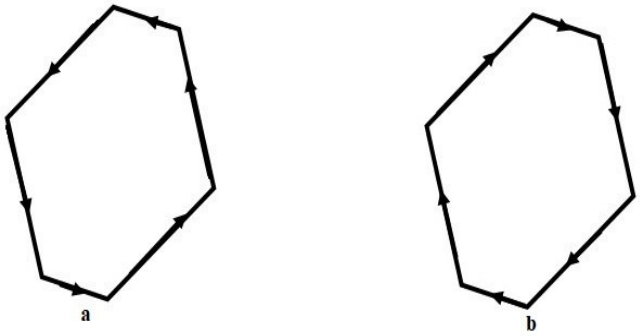


Fig 5.5: (a) Positive oriented, (b) Negative oriented

While filling a polygon with different colors, it is important to differentiate between several areas.

5.3.1 Scan-Line Polygon-Fill

In this algorithm the internal points of the polygon are colored in scan line and these points are turned on or off as needed. When coloring multiple pixels, the polygons are filled with multiple colors. The scan is done using the concept of raster scanning on the display device. The beam scans from the upper left corner of the screen and ends in the

Space for learners:

lower right corner. This algorithm detects the crossing point of the line and the polygon while traversing from left to right and from top to bottom. Each intersection is stored in the frame buffer. This intersection means crossing the boundary line and entering into the polygon. Then the color properties of the interior pixels are changed as specified, then again when beam comes out of the boundary on the other side, the previous properties are assumed. This process is repeated for all the rows until the polygon is filled.

While filling a polygon the following cases may occur:

i. The beam of the scan line may cross and pass through the edges of the polygon without touching any of the vertices. In the figure: 5.5 below, the pixels from point "c" to point "d" will be filled, pixels from point "d" to point "e" will remain as it is, then pixels from point "e" to point "f" will be filled with color.

ii. The beam of the scan line may cross the polygon by touching the any vertex whose nearby vertices lies totally on one side of the scan line. From the figure: 5.5 below, the pixels at vertices V_1, V_2 and V_6 will be filled, i.e., points "a", "b" and "j" will be colored.

iii. The beam of the scan line may cross the polygon by touching some vertices whose nearby vertices lie on both the sides of the scan line. This case is satisfied by the point "g" on the vertex V_3 on the figure: 5.5 below. Here the pixels from point "g" to point "h" and point "h" to point "i" will be filled.

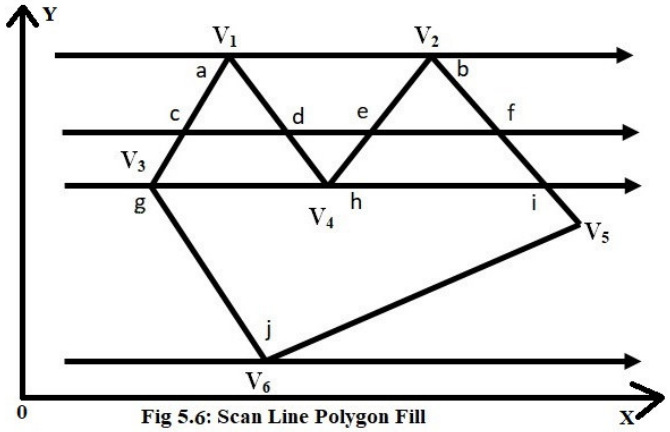


Fig 5.6: Scan Line Polygon Fill

Space for learners:

An example of scan line polygon filling algorithm is given with the following C++ program:

```
#include <stdio.h>

#include <conio.h>

#include <graphics.h>

void main()

{

int n,i,j,k,gd,gm,dy,dx;

int x,y,temp;

int a[10][2],xi[10];

float slope[10];

clrscr();

printf("\n\n\tEnter the number of edges of the polygon :..... ");

scanf("%d",&n);

printf("\n\n\tEnter the cordinates of the polygon in x and y axes
:.....\n\n\n ");

for(i=0;i<n;i++)

{

printf("\tX%d Y%d : ",i,i);

scanf("%d %d",&a[i][0],&a[i][1]);

}

a[n][0]=a[0][0];

a[n][1]=a[0][1];

detectgraph(&gd,&gm);
```

Space for learners:

```
initgraph(&gd,&gm,"C:\\TurboC3\\BGI");
                    //draw the polygon
for(i=0;i<n;i++)
{
line(a[i][0],a[i][1],a[i+1][0],a[i+1][1]);
}
getch();
for(i=0;i<n;i++)
{
dy=a[i+1][1]-a[i][1];
dx=a[i+1][0]-a[i][0];
if(dy==0) slope[i]=1.0;
if(dx==0) slope[i]=0.0;
if((dy!=0)&&(dx!=0)) // inverse slope calculation
{
slope[i]=(float) dx/dy;
}
}
for(y=0;y< 200;y++)
{
k=0;
for(i=0;i<n;i++)
```

Space for learners:

```
 {
if( ((a[i][1]<=y)&&(a[i+1][1]>y))||
((a[i][1]>y)&&(a[i+1][1]<=y)))
{
xi[k]=(int)(a[i][0]+slope[i]*(y-a[i][1]));
k++;
}
}
for(j=0;j<k-1;j++) // Arrange the intersections in x-axis
for(i=0;i<k-1;i++)
{
if(xi[i]>xi[i+1])
{
temp=xi[i];
xi[i]=xi[i+1];
xi[i+1]=temp;
}
}
setcolor(3);
for(i=0;i<k;i+=2)
{
line(xi[i],y,xi[i+1]+1,y);
getch();
```

Space for learners:

```
}
}
}
```

Reference: https://educatech.in/program-for-scan-line-polygon-fill-algorithm/

Space for learners:

5.3.2 Nonzero Winding Number Rule

For two-dimensional images in computer graphics, the nonzero winding number rule is a method used to find out if a given point lies inside or outside of an enclosed geometrical shape. If P is considered to be any point and a straight line is drawn from the point in one direction towards the shape (as shown in the figure 5.7 below) out to the infinity. Then count that how many times the straight line intersects the edge or edges of the shape and in which direction top to bottom or bottom to top or left to right or right to left in any orientation, either clockwise or anticlockwise orientation. It is based on knowing the direction of the stroke of each part of the shape. Here every point has a winding number. For a 2–dimensional object if the winding number value is a non zero, then the point lie inside the object and if the value of the winding number is zero, then the point lies outside the object. To calculate the winding number value the following steps are to be followed:

i. At first, the winding number value is initialized to 0.
ii. Then from any point P, an imaginary line is drawn to a distant point towards the other side of the object/shape, as shown in the figure 5.7(a & b) below.
iii. Then count how many number of times the edges crosses the line in each direction. Subtract 1 each time when an edge of the object crosses the line in the direction from top to bottom and add 1 each time when an edge of the object crosses the line from bottom to top.

iv. If the sum of the winding number is zero then the point is outside the object but if the sum of the winding is any number other than 0, than the point lies inside the object.

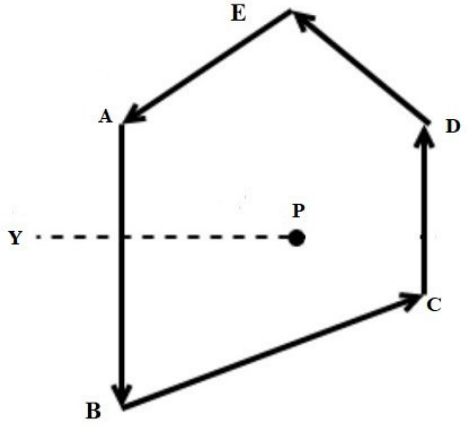


Fig 5.7 (a): Non-zero Winding (Inside).

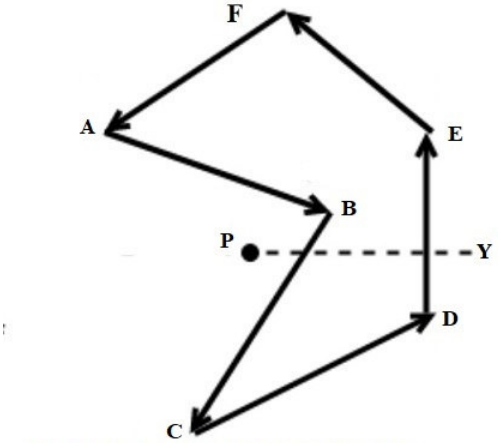


Fig 5.7(b): Non-Zero Winding (Outside)

In the first figure (Fig 5.7(a)) the line from point P towards Y crosses the edge AB directed from top to bottom, so winding number = 0 + (- 1) = - 1, thus the point P is inside the object.

In the second figure (Fig 5.7(b)) the line from point P to Y, crosses the edges BC, directed top to bottom and DE, directed bottom to top, so winding number = 0 + (- 1) + 1 = 0, thus the point P lies outside the object.

There is another method called odd even rule which is also used to determine whether a given point lies inside of the polygon or lies outside the polygon. Similar to non zero winding number method, the odd even method also uses a line segment to be drawn from any point P

Space for learners:

towards the polygon edges, then the number of intersection of the line with the edge or edges of the polygon is counted. If the counted number is odd then the point P lies inside the polygon but if the count is an even number then the point lies outside the polygon.

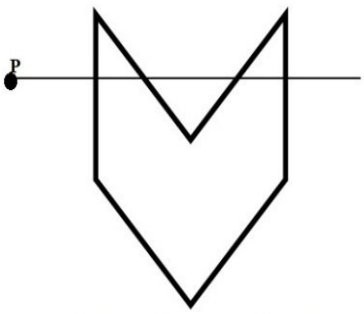


Fig 5.8(a): P lies outside the polygon

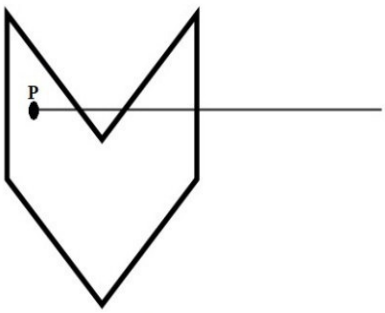


Fig 5.8(b): Point lies inside the polygon

In figure 5.8(a), it can be seen that the line segment from P towards the polygon intersects the edges for 4 times, which is an even number, this means that the point P lies outside the polygon. In the second figure (fig 5.8(b)) the line segment from P intersects the polygon edges for 3 times, which is an odd number, thus the point P lies inside the polygon.

5.3.3 Scan-Line Seed Fill Algorithm

The seed fill algorithm pushes 4 to 8 pixels into the stack that makes the stack large, because the pixels are filled starting from a single seed and then its neighboring pixels are filled. Left, right, top, bottom order in 4-connected or left, right, top, top-left, top-right, bottom, bottom-left, bottom-right order on 8-connected, the stack also holds duplicate pixels. The scan line seed fill algorithm pushes only the left endpoint pixel into the stack in any consecutive empty (non-colored) row of pixels in a single scan line within a boundary specified region which decreases the size of the stack. Scanline seed fill algorithm fills pixels in raster scan pattern, from left to right along each scanline in the region, without moving towards 4-connected or 8-connected directions.

Reference: https://www.csee.umbc.edu/~ebert/435/notes/435_ch5.html

Space for learners:

The steps are:

i. In the beginning a seed pixel is popped out from the stackwhere the seed pixel is present.

ii. Then the pixels on the left and right of the seed pixel and the seed pixel are colored in the row till a boundary color is detected.

iii. The left most and right most unfilled pixel in the scan line are saved as x_{left} and x_{right}, respectively.

iv. Then the rows just above and below the current row are tested from the range x_{left} to $x_{right,}$ to find out whether they areboundary colored pixels or earlier colored pixels. If they are neither boundary colored pixels nor previously filled pixels, then they are treated as seed pixels and are pushed into the stack from the right most pixels in all the non colored spans of line on this row and they are colored accordingly.

Space for learners:

5.3.4 Boundary-Fill Algorithm

This is a seed fill algorithm, here the color of the boundary of the polygon should be predefined with a specific single color. The algorithm will begin from a pixel which is internal to an image and starts filling towards outside. The algorithm fills the pixels one after another moving in outward direction until a boundary colour is detected. The boundary fill algorithm takes an interior pixel coordinate as the input, and starting from that position or pixel it checks the neighboring pixels to find if they are boundary colour or not. If they are boundary color then the algorithm stops moving in that direction but if the pixels are neither boundary coloured nor fill coloured than the pixel or pixels are coloured with the fill coloured and this is repeated until all the pixels on the area inside the boundary are coloured with fill colour. There are two ways to perform these tests: 4 connected and 8 connected.

In 4-connected approach left, right, above and bottom pixels are checked.

Procedure:

```
boundaryfill(x, y, fill, boundary: integer);
var
current:integer;
begin
current=getpixel(x, y);
```

```
        if(current<>boundary) and (current <> fill) then
begin
        setpixel(x, y, fill);
        boundary4(x+1, y, fill, boundary);
        boundary4(x-1, y, fill, boundary);
        boundary4(x, y+1, fill, boundary);
        boundary4(x, y-1, fill, boundary);
end
end; {boundaryfill}
```

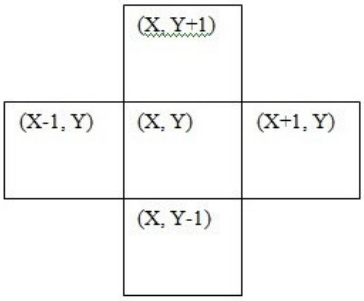


Fig 5.9: Four-connected Boundary Fill.

In 8-connected approach

```
begin
        setpixel(x,y,fill)
        boundary8(x+1, y, fill, boundary);
        boundary8(x+1, y+1, fill, boundary);
        boundary8(x-1, y, fill, boundary);
        boundary8(x-1, y+1, fill, boundary);
        boundary8(x, y+1, fill, boundary);
        boundary8(x-1, y-1, fill, boundary);
        boundary8(x, y-1, fill, boundary);
        boundary8(x+1, y-1, fill, boundary);
        end
end; {boundaryfill}
```

Space for learners:

(X-1, Y+1)	(X, Y+1)	(X+1, Y+1)
(X-1, Y)	(X, Y)	(X+1, Y)
(X-1, Y-1)	(X, Y-1)	(X+1, Y-1)

Fig 5.10: Eight-connected Boundary Fill

5.3.5 Flood-Fill Algorithm

This is also a kind of seed fill algorithm. The algorithm in this method is used to fill the area bounded by different colours. Here without searching for a boundary colour the area is painted by replacing the old colour with a new specified interior colour. The algorithm begins from an interior pixel and it replaces the entire old pixel colours with the new fill colour within the boundary even if the old pixels have different colours. 4-connected or 8-connected approaches can be used to fill the pixels with new colour within the boundary. The algorithm is:

Reference: Computer Graphics: C version by Donald Hearn & M. Pauline Baker.

4-connected approach:

```
Procedure floodFill4(x, y, fillcolor, oldcolor: integer);
begin
if getpixel(x, y)=oldcolor then
begin
setpixel(x, y, fillcolor);
floodFill4(x+1, y, fillcolor, oldcolor);
floodFill4(x-1, y, fillcolor, oldcolor);
floodFill4(x, y+1, fillcolor, oldcolor);
floodFill4(x, y-1, fillcolor, oldcolor);
end
end;
{floodFill}
```

Space for learners:

Space for learners:

8-connected approach:

```
Procedure floodFill8(x, y, fillcolor, oldcolor: integer);
begin
if getpixel(x, y)=oldcolor then
begin
setpixel(x, y, fillcolor);
floodFill8(x+1, y, fillcolor, oldcolor);
floodFill8(x-1, y, fillcolor, oldcolor);
floodFill8(x, y+1, fillcolor, oldcolor);
floodFill8(x, y-1, fillcolor, oldcolor);
floodFill8(x+1, y+1, fillcolor, oldcolor);
floodFill8(x-1, y+1, fillcolor, oldcolor);
floodFill8(x+1, y-1, fillcolor, oldcolor);
floodFill8(x-1, y-1, fillcolor, oldcolor);

end
end;
{floodFill}
```

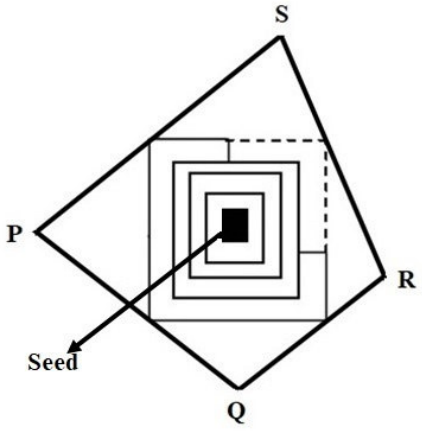


Fig 5.11: Flood Fill Algorithm.

The seed fill algorithm needs large storage space to the store the pixel information, it may not be able to fill all the non-colored pixels inside the area. But the scan line algorithms do not need much space it fills the pixels from left to right in a scan line and then after filling the farthest

pixel which is within the boundary, the algorithm moves to the next row or scan line.

STOP TO CONSIDER

In this sub section we discuss about polygons and its types, found whether a given point lies inside or outside of a polygon, explained various algorithms that are used to fill the area of polygon.

CHECK YOUR PROGRESS

1. For a specific single colored boundary, which algorithm proceeds pixel by pixel until the boundary color is encountered?
   a) Scan-line fill algorithm
   b) Boundary-fill algorithm
   c) Flood-fill algorithm
   d) Parallel curve algorithm
2. For a different colored boundary, which algorithm is used to recolor an area?
   a) Boundary-fill algorithm
   b) Parallel curve algorithm
   c) Flood-fill algorithm
   d) Only b
3. In which type of polygon the line segment joining any two points within the polygon lies completely inside the polygon?
   a) Convex
   b) Concave
   c) Closed
   d) Complete
4. Which of the following approach is used for determining whether a particular point is inside or outside of a polygon?
   a) Even-odd method
   b) Winding number method
   c) Both a & b
   d) None of these

Space for learners:

5.4 CHARACTER GENERATION TECHNIQUES

Space for learners:

Character generator or CG is a hardware or software tool which can produce still or moving graphics and texts, which can be then added to videos for streaming like creating opening credits or closing credits of a television program in television production studios. Computer based character generators can produce both graphical and textual images. In hardware-based character generation some display devices like monitors or printers are used to display the characters of varying sizes and shapes as commanded. Here the logic to generate the characters is built upon the graphic terminal. Although the generation time is relatively short, the sources are limited due to hardware limitations.

Several methods are available for generating characters on a computer screen through software:

(1) Stroke based method

(2) Vector based method and

(3) Starburst method.

Stroke based method is the natural way of writing character by a person. In this method the characters are written line by line. The characters are generated by joining small parts of line segments, just as Digital Differential Analyzer algorithm does.

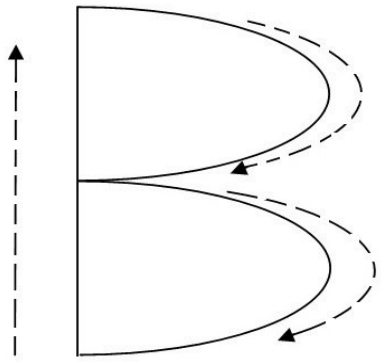


Fig 5.12: Character B using Stroke method

The vector-based methods are basically based on mathematical equations, the character is developed using a set of perfect lines and curves called polylines and splines which determines the outline of the

character. This character representation is completely device independent; memory requirements are lower, because the curve that describes the shape of the character can be manipulated to generate bold, italic, or different sizes; no separate memory block is required for each change. Common vector image files are .pdf, .ai, .eps, .svg, .wmf.

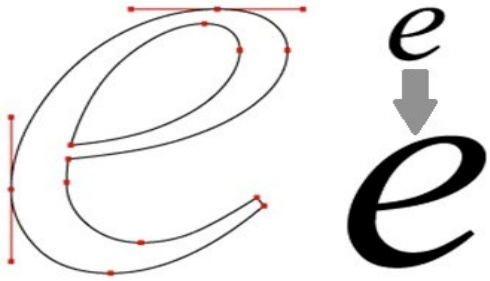


Fig 5.13: Vector based image

Image reference: https://business.oregonstate.edu/student-experience/resources/DAMlab/vector-and-bitmap-image-guide

In Starburst method is defined by a fixed pattern which is formed by joining a number of line segments. A collection of 24 line segments are used where each single line represents a bit. To draw a character out of the pattern on the corresponding line segments the bit values are to be set 0 or 1. To light up a line or turn ON a line then the bit is set to 1 and to make a line disappear or OFF the bit is set to 0. To draw a character we start from line number 1 then move in sequence towards 24, turning the line ON or OFF as required for a particular character. In the figure below the character H is drawn using the starburst method. The line segments are numbered 1 to 24. To draw the character H the line segments which are to be set ON or OFF are:
Line number 1, 2 ON, 3, 4, OFF, 5, 6, 7, 8 ON, 9, 10, OFF, 11, 12, ON, 13, 14, 15, 16, 17, 18, 19, 20 OFF, 21, 22 ON, 23, 24 OFF.
Thus 24 bit code to write character H in Starburst method isgiven in the table below (table 5.1) (from right to left):

Table 5.1

24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
0	0	1	1	0	0	0	0	0	0	0	0	1	1	0	0	1	1	1	1	0	0	1	1

Space for learners:

In table 5.1, the first row represents number of the line segments on the starburst and the second row represents ON/OFF.

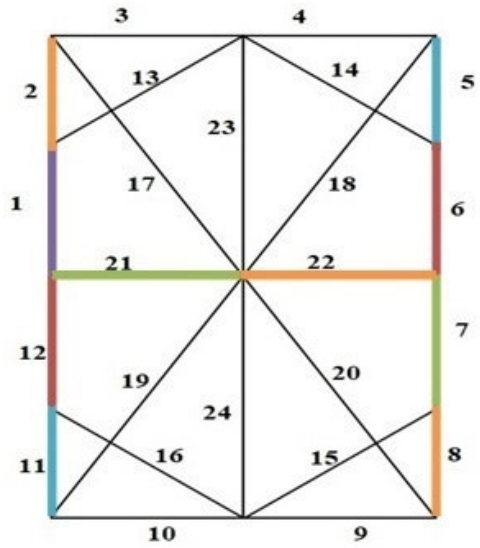


Fig 5.14: Character H using starburst

24 bit code for character H is: 001100000000110011110011

Starburst method has some disadvantages as it requires large memory to store 24-bits code for each character, it needs code convertor software to display the character and it is not suitable for curved shaped characters like, C, D, P etc.

5.4.1 Generation of Bitmap and Outlined Font

In the bitmap-based font method, a small rectangular bitmap called a character mask is used to store the pixel representation in binary 0 and 1 value for each character in a raster buffer called the font buffer. These arrays of bits are used to generate characters. These bit points are the points of a matrix of fixed size. When the point is stored in the array is 1 it will represent a character, that is, the position where the point appears is defined by the value 1 and the value of no point is defined by 0 in the matrix. The relative position of the pixel similar to the character's bitmap is marked according to the size, face, and font of the character.

Space for learners:

The size of each character mask is bounded in between from 5 by 7 to 10 by 12. 40 font caches are required to store a single font in different sizes of fonts and in four faces like normal, bold, italic and bolded italic. In this method, a character can be made bold by putting the corresponding normal character mask into successive frame buffer positions. Normal characters can be twisted or skewed to generate italic characters while writing to the frame buffer. Windows paint and much other software can be used to create new fonts by the designers and developers. The common design style (font and type face) of the character set is known as typeface. The suitable bitmap is selected from the frame buffer and copied to appropriate display screen position to generate and display the characters.

This method is also known as a dot matrix method, because the characters in this method are displayed in a dot matrix manner. It has rows and columns which gives it a two-dimensional orientation. Bitmap based image files are generally of .jpg, .png, .bmp, .gif, .tiff. format. The bitmap images are also called pixel maps or raster graphics.

Reference:
https://annamalaiuniversity.ac.in/studport/download/sci/cis/resources/IT3IITT62CG.pdf

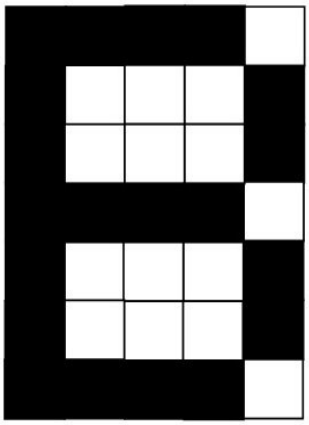


Fig 5.15: Bitmap based character B

Outline fonts use straight and curved parts to describe the shape of characters. When the pattern is copied to a certain area of the frame buffer, 1 bit specifies the position of the pixel to be displayed on the monitor. To display the shape of the character, the scan line fill program

Space for learners:

is used to fill the interior of the character outline. Outline fonts require less storage space because each variant does not require a different font cache. Using outline fonts can generate different shapes and sizes, and is easier to generate. But outline fonts take longer to process because they must be scanned and converted into the frame buffer.

Fig 5.16: Outline based fonts

Aliasing: There are two states of pixels ON and OFF, for this reason the display lines may have jagged or staircase appearance because of sampling process. Sampling is done to convert the analog image into digital. The distortion of information occurs because of under sampling or low frequency sampling and this is known as aliasing. This effect of aliasing occurs because the lines, polygon edges, color boundaries are unbroken (continuous) but the raster device is separate and distinct. Thus to present a line or polygon edges on a raster device it must be sampled at discrete location. Some of the aliasing effects are:

Staircase or zigzags problem: When any signal is sampled, this process utilizes coordinate points on an object to discrete integer pixel positions and jagged appearance easily can be seen. This is staircase problems.

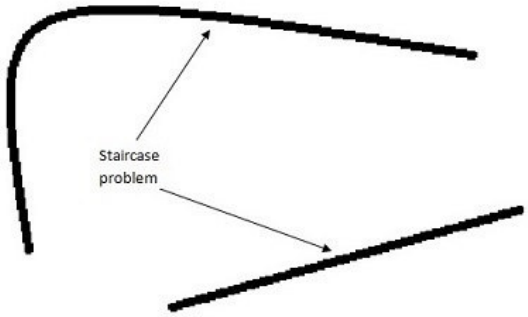


Fig 5.17: Staircase Problem or Jagged Lines.

Unequal intensity: If pixels of equal intensity are placed on the vertical, horizontal and diagonal line, then the intensity of the diagonal line becomes less as compared to the horizontal and vertical lines due to

Space for learners:

some reason (see the figure). As a result, intensity of pixel will be less on the diagonal. Thus, intensity of the line decreases.

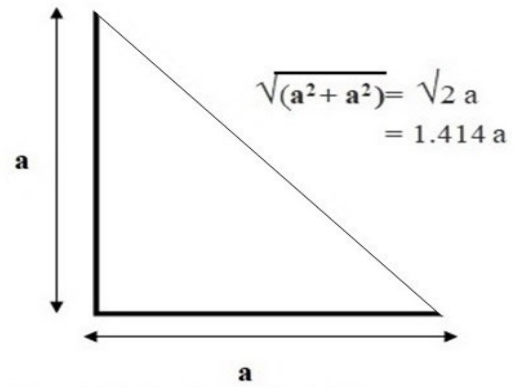


Fig 5.18: Unequal Intensity

Pixels along Horizontal and Vertical lines are just a single unit apart (far from each other), whereas along the Diagonal line they are 1.414 units far from each other.

Antialiasing: It is a method which is used to rectify the issues aroused due to aliasing. It tries to remove the staircase or jagged lines problem occurred due to under sampling. Due to under sampling information of the images are lost. There is a minimum frequency referred to sample an image, which is given by Harry Nyquist, called, "Nyquist sampling frequency" according to Nyquist frequency; 1 cycle in 2 pixels = 0.5 cycles/pixel. There are two methods of antialiasing:

Reference: https://www.imatest.com/docs/nyquist-aliasing/

Post-filtering (i++): The post-filtering method is achieved by increasing the sampling rate, which is performed by increasing the resolution the raster system. The resolution is increased by assuming that the actual pixels are sub divided into smaller pixels and thus increasing the resolution. Then the average intensity of the sub pixels are calculated and displayed at the actual pixels. However, there is some limitations in dividing the pixels into sub pixels. This method is also called super sampling

Pre-filtering (++i): Here the pixels are a fixed area, the colour of the pixels are calculated depending on the overlap of the pixel area with the image to be displayed. It calculates the percentage of overlap of the gray level shade of the image to the area of the pixel covered. This percentage determines the overall intensity of the pixel.

Space for learners:

STOP TO CONSIDER

In this sub section we highlighted on different types of polygons, different types of seed fill and scanline fill algorithms, flood fill and boundary fill polygon filling algorithms. These filling algorithms can be used to fill colours into the object. Non zero winding number rule and odd even rule are used to determine whether a given point lies inside or outside of a polygon. Generation of characters and their different techniques used to generate different typographical fonts has also been discussed. We have also discussed about staircase problems and why the intensity of a diagonal line is lesser than that of horizontal or vertical lines in aliasing and how these problems can solved is also discussed here.

CHECK YOUR PROGRESS

5. With which of the following aliasing is related to?
   a) Generation of characters
   b) Scan conversion of line
   c) Raster based graphics
   d) Conversion of any analog signal to digital signal
6. The video editing tool that produces an animated text which can be inserted into video stream is……..
   a) Character generator
   b) Title generator
   c) Video generator
   d) Animation generator
7. In television production houses the type of character generators used is:
   a) Hardware character generators
   b) Software character generators
   c) Hardware and software character generators both are used
   d) Title generators are used
8. How many character generation methods are there?
   a) 2
   b) 3
   c) 4
   d) 5

Space for learners:

9. Which character generation method is also known as Dot-matrix method?
   a) Stroke method
   b) Bitmap method
   c) Starburst method
   d) None of these
10. In which character generation method, graph is used in form of line by line?
   a) Stroke method
   b) Bitmap method
   c) Starburst method
   d) None of these
11. Which of the following method is used to generate characters in a fixed pattern of line segments??
   a) Stroke method
   b) Bitmap method
   c) Starburst method
   d) None of these
12. Which method has the poorest character quality?
   a) Stroke method
   b) Bitmap method
   c) Starburst method
   d) None of these
13. Which of the following can be generated by character generators?
   a) Different type size but same fonts
   b) Same type size but different fonts
   c) Same type size and fonts
   d) Different type size and fonts
14. Can the font colors be changed using character generators.
   a) True
   b) False

Space for learners:

5.5 SUMMING UP

- Area can be defined by a set of boundary pixels. Several algorithms are available to fill colour to the interior pixels of an image. Filling an area means colouring an image or region with a specific colour.

- The nonzero winding number rule is a method used to find out if a given point lies inside or outside of an enclosed geometrical shape.
- Character generator is a hardware or software tool which can produce still or moving graphics and texts, which can be then added to videos for streaming like creating, opening credits or closing credits of a television program.
- The vector-based methods are basically based on mathematical equations. The character is developed using a set of perfect lines and curves called polylines and splines which determines the outline of the character.
- In the bitmap-based font method, a small rectangular bitmap called a character mask is used to store the pixel representation in binary 0 and 1 value for each character in a raster buffer called the font buffer. These arrays of bits are used to generate characters.
- The bitmap images are also called pixel maps or raster graphics.
- There are two states of pixels ON and OFF, for this reason the display lines may have jagged or staircase appearance because of sampling process.

Space for learners:

5.6 ANSWERS TO CHECK YOUR PROGRESS

Question 1: Answer: b
Explanation: Boundary fill algorithm proceeds outward pixel by pixel until the boundary color is encountered.
Question 2: Answer: c
Explanation: Flood fill algorithm allows to fill color of such areas by replacing a specified interior color.
Question 3: Answer: a
Explanation: In convex polygon the line segment lies completely inside the polygon.
Question 4: Answer c
Explanations: Even odd method and Non-zero winding number method both are used to find out whether a given point lies inside or outside of the polygon.

Question 5: Answer: d

Explanation: Aliasing is related to conversion of any analog signal to digital signal.

Question 6: Answer: a
Explanation: Video editing software or hardware that produces animated text video streams is performed by Character Generator, CG in short.

Question 7: Answer: a
Explanation: Hardware character generators are commonly used in television production houses. They provide a key signal. Translucent areas of the character generator can be determined by compositing vision mixer using an alpha channel.

Question 8: Answer: b
Explanation: There are three methods. Those three methods are Stroke method, Bitmap method and Starburst method. These three methods uses different ways for generating characters on the screen.

Question 9: Answer: b
Explanation: Bitmap method is also known as a dot-matrix method as it uses arrays of dots to generate characters. These dots are the points of an array whose sizes are fixed.

Question 10: Answer: a
Explanation: In Stroke based method, graph is drawn in the form of line by line. This method follows the Digital Differential Analyzer Line drawing algorithm drawing for line.

Question 11: Answer: c
Explanation: In Starburst method, 24 line segments are used and combination to form characters. It is a method in which there is a particular pattern or sequence where only 24 strokes are defined for character generation.

Question 12: Answer: c
Explanation: Character quality is very poor in Starburst method and is even more worst for curved characters like C, D P etc. Whereas, it is very good in Bitmap method and stroke method, as they use new technologies.

Question 13: Answer: d
Explanation: Character generators can generate different type, sizes and

Space for learners:

fonts depending on the requirement. Type, size as well as colour of fonts can be changed using character generators.

Question 14: Answer: a
Explanation: Common systems displays eight colours: black, white, yellow, red, magenta, blue, cyan and green. More sophisticated systems have millions of colours.

Space for learners:

5.7 POSSIBLE QUESTIONS

1. Explain how a polygon can be filled using scan line?
2. What is antialiasing?
3. Define frame buffer or refresh buffer?

a.8 REFERENCES AND SUGGESTED READINGS

- Introduction To Computer Graphics and Mu, by Anirban Mukhopadhyay and Arup Chatterjee, Vikas Publishing House Pvt. Ltd., 2006
- Computer Graphics: C version by Donald Hearn & M. Pauline Baker.
- https://business.oregonstate.edu/student-experience/resources/DAMlab/vector-and-bitmap-image-guide
- https://educatech.in/program-for-scan-line-polygon-fill-algorithm/
- https://www.csee.umbc.edu/~ebert/435/notes/435_ch5.html
- https://www.imatest.com/docs/nyquist-aliasing/
- https://annamalaiuniversity.ac.in/studport/download/sci/cis/resources/IT3IITT62CG.pdf
- Introduction To Computer Graphics And Mu, by Anirban Mukhopadhyay and Arup Chatterjee, Vikas Publishing House Pvt. Ltd.
- Computer Graphics: C version by Donald Hearn & M. Pauline Baker.

- Computer Graphics Principles & Practice by John F. Hughes, Andries Van Dam, Morgan Mcguire, David F. Sklar, James D. Foley, Steven K. Feiner, Kurt Akeley

Space for learners:

BLOCK II:
GEOMETRIC TRANSFORMATIONS AND VIEWING AND CLIPPING

UNIT 1: TWO DIMENSIONAL GEOMETRIC TRANSFORMATIONS

Unit Structure:

Space for learners:

1.1 INTRODUCTION

In computer graphics the output primitives and their attributes can be displayed in various ways, i.e. we can have a variety of images and graphs. In many graphics applications,the images need to alter or manipulate for display. The layout of a picture can be manipulated by arranging the orientation, shape, and sizes of the

component parts of the scene. In computer graphics, geometric transformation can be used to changetheorientation, shape, and size of an object that alters the coordinate positions. Some of the basic transformations we can perform on objects in a graphics system are rotation, shearing, reflection, and scaling. We can perform a particular operation on an object more than once and in a composite way also.

Space for learners:

1.2 UNIT OBJECTIVES

- Learn about the basics of 2D transformation
- Learn different types of transformations and matrix implementation
- Learn about rotation, translation, and scaling.
- Learn about the composite or sequential transformation.
- Learn about the shear and reflection.
- Learn about the generally fixed-point transformation.
- The transformation between two different coordinates.

1.3 BASIC TRANSFORMATIONS

1.3.1 Translation

Translation can be achieved by changing the position of an object along a straight line from its current coordinate position to another.In a two-dimensional coordinate, the translation operation can be performed by adding translation distance t_x and t_y to the original coordinate position (x, y). Let the new coordinate position of the point is (x', y'), then it can be expressed as:

$x' = x + t_{x,}$ and

$y' = y + t_y$ ……..…..(1.1)

Where the t_x and t_y are known as the translation vector or shift-vector. In figure 1.1, it has been shown a translation of a point from position **A** to position **B** in a 2 – dimensional coordinate geometry. Here ***T*** is assumed as the translation vector.

Equations no (1.1) can be expressed in terms of a single matrix equation by using the column vector to represent coordinate positions and the translation vector.

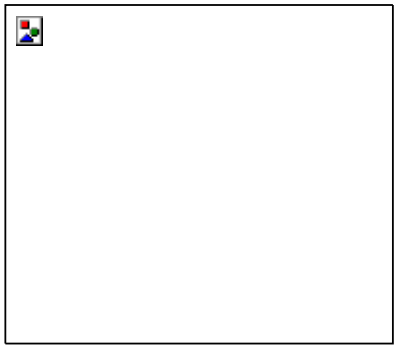


Figure 1.1 Translation of a point from position A to position B with translation vector T

$$A=\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad B=\begin{bmatrix} x'_1 \\ x'_2 \end{bmatrix} \quad T=\begin{bmatrix} t_x \\ t_y \end{bmatrix} \quad \ldots\ldots\ldots\ldots\ldots\ldots (1.2)$$

The matrix form of the translation equation is:

$$B = A + T \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (1.3)$$

The translation matrix can be expressed in terms of row vectors also. In this case, the translation matrix can be expressed as:

$$B = [x \quad y] \quad and \quad T = [t_x \quad t_y]$$

But the column vector representation is the standard mathematical notation for a point and most of the graphics application and system uses the column vector representation. The translation can be defined as the movement of an object or a rigid body without any changes or deformation. Thus each point of the body will transfer by the same amount. A rectangle or a polygon can be translated by adding the translation vector to the coordinate position of each vertex. After getting the new set of vertices, regenerate the rectangle or the polygon without changing the current attributes. Similarly for curved objects such as arc, circle, ellipse, etc. can be translated by transforming the center coordinate, and redrawing the figure in the new coordinate position.

Space for learners:

1.3.2 Rotation

A two-dimensional rotation of an object can be defined as reposition of the object coordinates along a circular path in the xy plane. To rotate an object in a two-dimensional plane, it must specify the rotational angle **θ**and the position of the rotation point (x_r, y_r) as shown in figure 1.2. The rotation point about which the object is to be rotated is known as the ***pivot point***. Counterclockwise rotation about the pivot point always gives a positive value for the rotation angle, and clockwise direction gives a negative value for the rotation angle θ. The rotation operation in computer graphics can be defined as the rotation of an object about a ***rotation axis***, which is perpendicular to the x-y plane and passes through the pivot point.

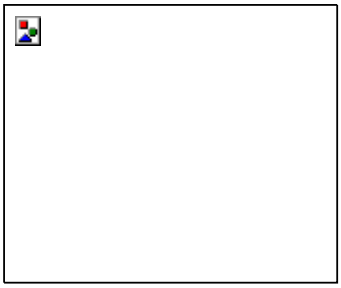


Figure 1.2 Rotation of an object through angle θ about the

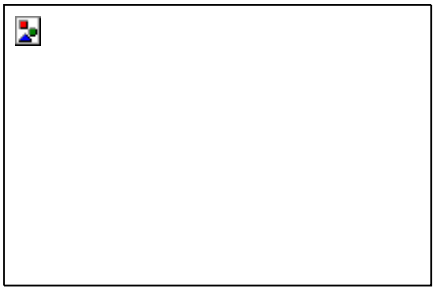


Figure 1.3 Rotation of a point from A to B through an angle θ relative to the

To determine the transformation equation for rotation about of a point position A(x, y), assume that, the coordinate origin is the pivot point as shown in figure 1.3. Figure 1.3 depicts the relationship between the angle and the coordinate position of the original and the transformed point position. In the figure, r is the radius or the constant distance of the point from the coordinate origin. Angle Ø is the original angular position of the point from the x-axis, and the θ is the rotation angle. Using the standard trigonometric equation, we can express the transformed coordinates in terms of θ and Ø as:

$$\left.\begin{aligned} x' &= r\cos(\phi+\theta) = r\cos\phi\cos\theta - r\sin\phi\sin\theta \\ y' &= r\sin(\phi+\theta) = r\cos\phi\sin\theta + r\sin\phi\cos\theta \end{aligned}\right\} \quad \text{.....(1.4)}$$

Space for learners:

STOP TO CONSIDER

since $\cos\theta = \frac{base\ (b)}{hypotenuse\ (h)} = \frac{x'}{r} \therefore x' =$

In our example $\theta = (\theta + \emptyset)$, *so* $x' = r \cos$

Space for learners:

The original coordinates of the point in polar coordinates are:

$$x = r\cos\phi \quad and \quad y = r\sin\phi \text{.....(1.5)}$$

Hence from the equation (1.4), we have

$$\left.\begin{aligned} x' &= x\cos\theta - y\sin\theta \\ y' &= x\sin\theta + y\cos\theta \end{aligned}\right\} \text{.....(1.6)}$$

Now the column vector representation for the coordinate positions, we can express the rotation equation in the matrix form as B = A.R,

Where the rotation matrix R is:

$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \text{.....(1.7)}$$

Figure 1.4 Rotation of a point from A(x, y) to B (x', y') through an angle θ, about the (x_r, y_r)

The rotation of a point A(x, y), about an arbitrary pivot (x_r, y_r), is shown in figure 1.4. Using the standard trigonometric relationships,

from the equation (1.6), we can have the transformation equation for rotation of point A about any arbitrary pivot point position (x_r, y_r) as:

$$\left.\begin{aligned} x' &= x_r + (x - x_r)\cos\theta - (y - y_r)\sin\theta \\ y' &= y_r + (x - x_r)\sin\theta + (y - y_r)\cos\theta \end{aligned}\right\} \quad \text{.....(1.8)}$$

As the translation, the rotation can be defined as the rigid body transformations that make reposition of an object without any deformation. Here each point from an object will be rotated through the same angle about the pivot position.

1.3.3 Scaling

The scaling transformation is used to change or alter the size of an object. This operation can be performed on a rectangle by multiplying the coordinate values of each vertex with the scaling factors to produce the transformed coordinates (x', y'). Let us assume that the coordinate position of a vertex is (x, y), and the scaling factors are s_x along the x-axis and s_y along the y-axis.The transformed coordinates (x', y') can be expressed as:

$$x' = x.s_x, \qquad y' = y.s_x \text{.....(1.9)}$$

The matrix form of the equation (1.9) will be:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} . \begin{bmatrix} x \\ y \end{bmatrix} \text{.....(1.10)}$$

Or

$$P' = \text{S.P} \text{.....(1.11)}$$

The scaling factors can assign with any positive numeric values. The size of the transformed objects will be decreased for the scaling factors less than the value 1, and produce an enlarged object for a value greater than 1. If the scaling factor is equal to 1, there will be no change in the object during transformation. The value of the scaling factors s_x and s_y may be equal or not. For the equal value of the $s_x$and s_y, uniform scaling is produced which will maintain the relative object proportions. Different scaling can be obtained by

Space for learners:

using unequal numeric values for both the scaling factor. Unequal scaling factors are often used to design applications to construct pictures using some of the basic shapes.

Space for learners:

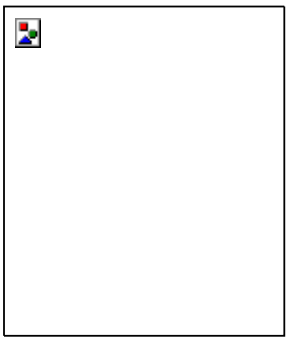


Figure 1.5 turning a square into a rectangle through scaling

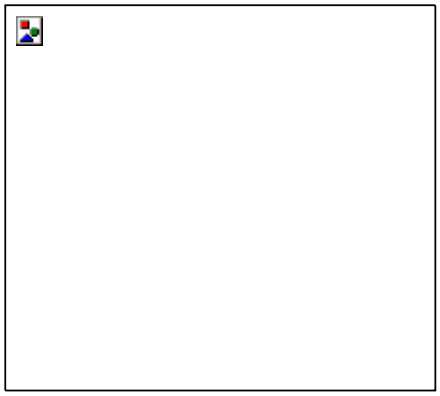


Figure 1.6 A line scaled with reduced size

In figure 1.5, a square has been scaled by the scaling factor $s_x = 2$, and $s_y = 1$, to obtain a rectangle. Similarly in figure 1.6, a straight line is scaled by the scaling factor $s_x = 0.5$, and $s_y = 0.5$. In both, the line, length, and distance from the origin are reduced by a factor of 0.5.

By choosing the position of the fixed point, which remains unchanged during the scaling transformation, we can control the location of the scaled object. The coordinate of the fixed point (x_f, y_f) can be the centroid of the object or any vertex of the object or any other point position. A polygon can then be transformed or scaled to the fixed point by scaling the distance from each vertex to the fixed point. For a vertex coordinate (x, y), the scaled coordinate (x', y') can be calculated as:

$$\left.\begin{aligned} x' &= x_f + (x - x_f)s_x \\ y' &= y_f + (y - y_f)s_y \end{aligned}\right\} \ldots\ldots (1.12)$$

After separating the multiplicative and additive terms the equation (1.12) become:

$$\left.\begin{aligned} x' &= x.\mathrm{s}_x + x_f(1-\mathrm{s}_x) \\ y' &= y.\mathrm{s}_y + y_f(1-\mathrm{s}_y) \end{aligned}\right\} \ldots\ldots(1.13)$$

Where the additive terms $x_f(1-s_x)$ *and* $y_f(1-s_y)$ are constant for all points in the object. The inclusion of coordinates of a fixed point is similar to the addition of a pivot point in the rotation equation. Here we can write a column vector with constant values in equation (1.13), and add this column vector to **S.P** in equation (1.11)

1.4 MATRIX REPRESENTATION AND HOMOGENEOUS COORDINATES

In computer graphics systems, most of the time is needed to perform a sequence of transformations. For example, the case of animation, might require an object to be translated, scaled, and rotated at each movement. To fit the picture components in their appropriate position, graphics applications may perform translation, rotation, and scaling operations. In this section, a discussion has been made on how to formulate and processed efficiently the matrix representation to perform such composite transformation sequences.

From the previous analysis (equation 1.1) it has been observed that the basic formula to represent a transformation in matrix form is:

$$P' = M_1.P + M_2 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(1.14)$$

Where the term P and P' represents coordinate positions as column vectors. M_1 contains the multiplicative factors in a 2 x 2 array form. M_2 contains the translational terms and has been written as a 2 elementscolumn matrix.

For translation, M_1 is an identity matrix, and

For rotation and scaling, M_2 contains the translational terms associated with the pivot point or the scaling fixed point.

To produce a sequence of transformations such as scaling followed by translation then rotation, we must calculate the transformed coordinates, one step at a time. First, perform the scaling operation on the coordinates. Then these scaled coordinates are translated, and finally, the translated coordinates are rotated. Alternatively, we can

Space for learners:

combine all the transformations to obtain the final coordinate positions directly from the initial coordinates. Thus the intermediate calculation steps can be eliminated, and get an optimal way to perform the composite transformation. But to combine and reformulate the equation (1.14) we have to eliminate the matrix addition associated with the translation terms in M_2.

The multiplicative and the translational terms for the two-dimensional geometric transformations matrix can be combined into a single matrix representation by converting the two-dimensional matrix into 3-dimensional matrices. This allows us to express all transformation equations as matrix multiplications, providing that we also expand the matrix representations for coordinate positions. To express any two-dimensional transformation as a matrix multiplication we represent each Cartesian coordinate position (x, y) with the homogeneous coordinate triple (x, y, h), where

$$x = \frac{x_h}{h} \quad and \quad y = \frac{y_h}{h} \text{.....(1.15)}$$

Where the ***h*** is the homogeneous parameter and it can have any non-zero value. Thus for each Cartesian coordinate position (x, y), there is an infinite number of homogeneous representations. A convenient way to select the homogeneous parameter is simply to set ***h = 1***. Thus the homogeneous coordinate for each two-dimensional position is ***(x, y, 1)***.

Homogeneous representation of coordinates allows us to express all geometric transformation equations as matrix multiplications. Here the coordinates are represented by a 3 element column vector. The expression for the translation operation in homogeneous coordinate will be:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \text{.....(1.16)}$$

The abbreviated form of equation (1.16) is:

$$P' = T(t_x, t_y) \cdot P \quad \text{.....(1.17)}$$

Space for learners:

Where, $T(t_x, t_y)$ is the 3 x 3 translational matrix. The inverse of the translational matrix can be obtained by replacing the term t_x *and* t_y *with their negatives* $-t_x$, *and* $-t_y$.

Similarly, the rotation transformation equation about the coordinate origin can express as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \text{.....(1.18)}$$

Or as

$$P' = R(\theta) \, . \, P \quad \text{.....(1.19)}$$

Finally, the scaling transformation equation about a fixed point at the coordinate origin can express as:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \text{.....(1.20)}$$

Or as

$$P' = S(s_x, s_y) \, . \, P \quad \text{.....(1.21)}$$

Equation (1.21), $S(s_x, s_y)$ is a 3 x 3 matrix. We can obtain the inverse scaling matrix by replacing the scaling parameters s_x, *and* s_y with its multiplicative inverse, i.e. $\frac{1}{s_x}$, *and* $\frac{1}{s_y}$.

1.5 COMPOSITE TRANSFORMATIONS

Using the matrices discussed above can obtain any sequence of transformations by calculating the matrix product of the individual transformation which is known as the composite transformation. Forming product of matrices of every individual transformation is often referred to as concatenation or composition of matrices. For column matrix representation of coordinate position, we form

Space for learners:

composite transformation by multiplying matrices in order from right to left. That is each successive transformation matrix pre-multiplies the product of the preceding transformation matrices.

Space for learners:

1.5.1 Translation

The transformed location P' can be calculated for **two successive translation vectors** $T(t_{x1},t_{y1})$ *and* $T(t_{x2},t_{y2})$ by applying to a coordinate position P as:

$$\left.\begin{aligned} P' &= T(t_{x2},t_{y2}).[T(t_{x1},t_{y1}).P] \\ P' &= [T(t_{x2},t_{y2}).T(t_{x1},t_{y1})].P \end{aligned}\right\} \text{.....(1.22)}$$

Where P and P' are calculated as the homogeneous coordinate column vectors. The composite translation matrix for this sequence of translation is:

$$\begin{bmatrix} 1 & 0 & t_{x2} \\ 0 & 1 & t_{y2} \\ 0 & 0 & 1 \end{bmatrix}.\begin{bmatrix} 1 & 0 & t_{x1} \\ 0 & 1 & t_{y1} \\ 0 & 0 & 1 \end{bmatrix}=\begin{bmatrix} 1 & 0 & t_{x1}+t_{x2} \\ 0 & 1 & t_{y1}+t_{y2} \\ 0 & 0 & 1 \end{bmatrix} \text{.....(1.23)}$$

or

$$T(t_{x2},t_{y2}).T(t_{x1},t_{y1}) = T(t_{x1}+t_{x2},t_{y2},t_{y1}+t_{y2}) \text{.....(1.24)}$$

The equation (1.24) shows that two successive translations are additive.

1.5.2 Rotation

Two successive rotations applied to the point P produced the transformed position

$$\left.\begin{aligned} P' &= R(\theta_2)\ .\ [R(\theta_1).P] \\ P' &= [R(\theta_2)\ .\ R(\theta_1)].P \end{aligned}\right\} \text{.....(1.25)}$$

Two successive rotations are additive and it can be verified by multiplying the two rotation matrices, i.e.

$$R(\theta_2)\ .\ R(\theta_1) = R(\theta_1+\theta_2).....(1.26)$$

Thus the final rotated coordinates can be calculated with the composite rotation matrix as

$$P' = R(\theta_1 + \theta_2).P \ .....(1.27)$$

CHECK YOUR PROGRESS

1. Perform a 45^0 rotation of triangle A(0, 0), B(1, 1) and C(5, 2).
   i. About the origin.
   ii. About the point (-1, -1).
2. Translation can be achieved by changing the _____of an object.
3. The rotation point about which the object is to be rotated is known as _____ _____.
4. Counter clockwise rotation about the pivot point always gives a _____ value.
5. Clockwise rotation about the pivot point always gives a _____ value.
6. The scaling transformation is used to change or alter the _____ of an object.

1.5.3 Scaling

By concatenating two successive scaling matrices we can produce the following composite scaling matrix:

$$\begin{bmatrix} s_{x2} & 0 & 0 \\ 0 & s_{y2} & 0 \\ 0 & 0 & 1 \end{bmatrix}.\begin{bmatrix} s_{x1} & 0 & 0 \\ 0 & s_{y1} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_{x1}.s_{x2} & 0 & 0 \\ 0 & s_{y1}.s_{y2} & 0 \\ 0 & 0 & 1 \end{bmatrix} .....(1.28)$$

or

$$S(s_{x2}, s_{y2}).S(s_{x1}, s_{y1}) = S(s_{x1}.s_{x2}, s_{y1}.s_{y2}).....(1.29)$$

The output matrix in the equation (1.28), indicates that the two successive scaling operations are multiplicative.

Space for learners:

1.6 GENERAL PIVOT POINT ROTATION

Space for learners:

Generally, the rotation operation is performed about the coordinate origin. But we can perform rotation about any selected pivot point (x_r, y_r) in the coordinate by performing the following sequences of transformations: **translate – rotate – translate**

1. Translate the object such that the pivot point position lies on the coordinate origin (x_0, y_0).
2. Rotate the object about the coordinate origin.
3. Translate the object so that the pivot point becomes (x_r, y_r), i.e. return to its original position.

The following figure 1.7 depicts the sequences of

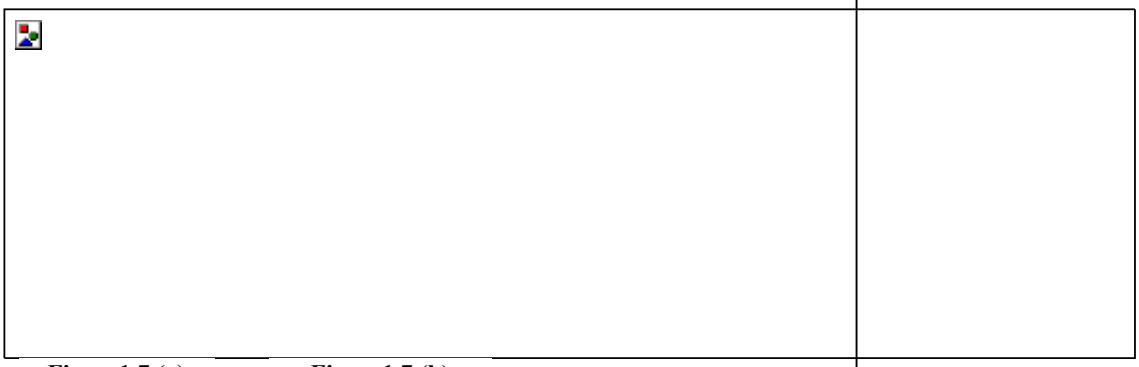


Figure1.7 (a) original position of the object and the pivot point

Figure1.7 (b) Translation of the object so that the pivot point is at origin

Figure1.7 (c) Rotation about origin

Figure1.7 (d) Translation of the object so that the pivot point is returned to its position

transformation.

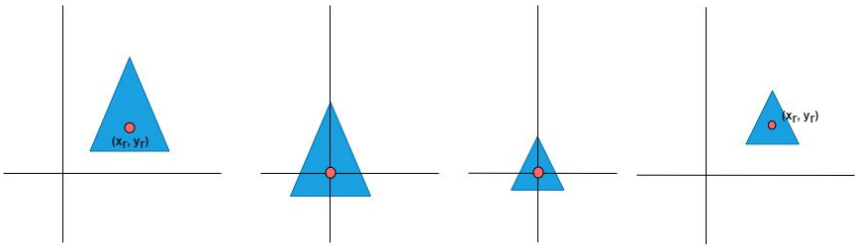
By concatenating the matrices we can obtain the composite transformation matrix for this sequence as

$$\begin{bmatrix} 1 & 0 & x_r \\ 0 & 1 & y_r \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} 1 & 0 & -x_r \\ 0 & 1 & -y_r \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & x_r(1-\cos\theta) + y_r.\sin\theta \\ \sin\theta & \cos\theta & y_r(1-\cos\theta) - x_r.\sin\theta \\ 0 & 0 & 1 \end{bmatrix} \quad \text{.....(1.30)}$$

1.7 GENERAL FIXED-POINT SCALING

Transformation sequences to produce scaling concerning a selected fixed position (x_r, y_r)are depicted in the following figure 1.8, through a scaling function that can scale relative to the coordinate origin. The steps required to perform these sequences of transformation for scaling are:



1. Translate the object such that the fixed point is at coordinate origin.
2. Performed the scaling operation about the fixed point at coordinate origin.
3. Translate the scaled object to the original position, such that the fixed point coordinate lies at ((x_r, y_r).

The set of the concatenation of matrices for these operations produces the required scaling matrix:

$$\begin{bmatrix} 1 & 0 & x_r \\ 0 & 1 & y_r \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_r \\ 0 & 1 & -y_r \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & x_r(1-s_x) \\ 0 & s_y & y_r(1-s_y) \\ 0 & 0 & 1 \end{bmatrix} \quad \text{.....(1.31)}$$

This transformation is automatically generated on systems that provide a scale function that accepts coordinate for fixed point.

1.8 OTHER TRANSFORMATIONS

In most graphics applications, the basic transformations – translation, rotation, and scaling are included. But some additional transformations are provided by some systems, which are useful for certain applications. Two such transformations are: reflection and shear

1.8.1 Reflection

Space for learners:

Reflection transformation is used to generate a mirror image of an object. The mirror image of an object is generated about an axis in a two-dimensional plane by rotating the object 180^0 about the reflection axis. The reflection axis might be any one of the axes in the ***x-y*** plane or may be perpendicular to the ***x-y*** plane. The rotation path about the axis will be perpendicular to the ***x-y*** plane if the reflection axis is in the ***x-y*** plane. If the rotation axis is perpendicular to the ***x-y*** plane, then the rotation path will be in the x-y plane. Some examples of common reflections are:

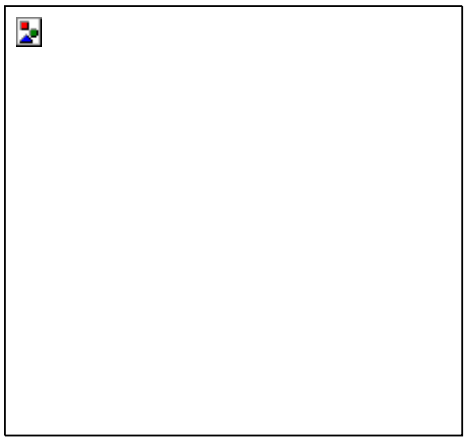


Figure 1.9 Reflection of an object about x-axis

a. Reflection about the line *y = 0*.

This reflection operation is accomplished with the transformation matrix:

$$\begin{bmatrix} 1 & 0 & \mathbf{0} \\ 0 & -1 & \mathbf{0} \\ 0 & 0 & 1 \end{bmatrix} \text{.....(1.32)}$$

This transformation keeps the x values the same but flips the ***y*** values of the coordinate positions. The resulting orientation of an object after it has been reflected about the ***x-axis*** is shown in figure 1.9.

b. A reflection relative to the y axis, i.e. ***x = 0***, keeps the y-coordinate same and flips the x coordinates as shown in figure 1.10andthe matrix for this transformation is:

Space for learners:

$$\begin{bmatrix} -1 & 0 & \mathbf{0} \\ 0 & 1 & \mathbf{0} \\ 0 & 0 & 1 \end{bmatrix} \text{.....(}\mathbf{1.32}\text{)}$$

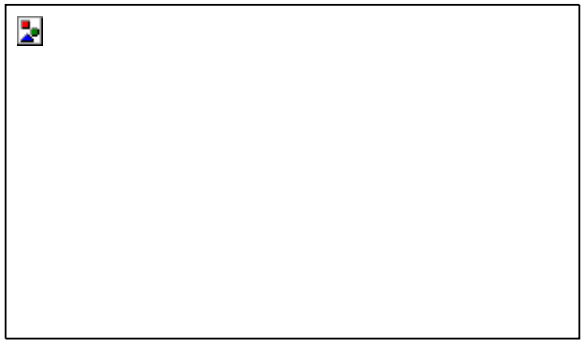


Figure 1.10 Reflection of an object about y-axis

c. Reflection is relative to an axis that is perpendicular to the *x-y* plane and that passes through the coordinate origin. The reflection can be expressed in terms of a matrix as:

$$\begin{bmatrix} -1 & 0 & \mathbf{0} \\ 0 & -1 & \mathbf{0} \\ 0 & 0 & 1 \end{bmatrix} \text{.....(}\mathbf{1.31}\text{)}$$

Figure 1.11 depicts the reflection about an axis that is perpendicular to the x-y plane and passes through the coordinate origin. The matrix (1.31) is the rotation matrix with $\theta = 180^0$. The reflection expressed by the equation (1.1) would be generalized to any reflection point in the ***x-y*** plane.

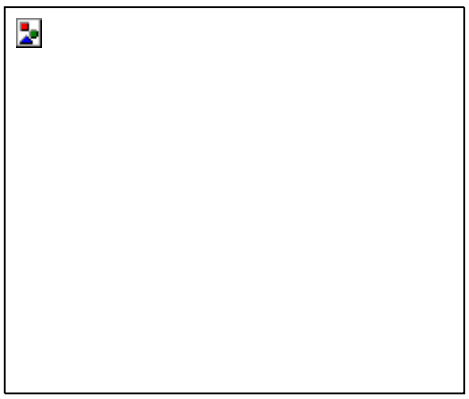


Figure 1.11 Reflection of an object about the coordinate origin

Space for learners:

d. Reflection about a diagonal line $y = x$ is shown in figure 1.12. the reflection matrix is:

$$\begin{bmatrix} 1 & 0 & \mathbf{0} \\ 0 & 1 & \mathbf{0} \\ 0 & 0 & 1 \end{bmatrix} \quad \text{.....(1.32)}$$

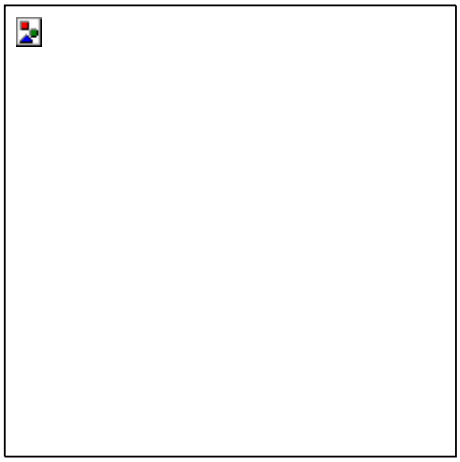


Figure 1.12 Reflection of an object with respect to the line x=y

e. The reflection about any line $y = m.x + c$ in the ***x-y*** plane can be accomplished with **a translate-rotate-reflect** transformation. First, translate the object, such that the reflective axis passes through the coordinate origin. Then perform a rotation operation about the axis, to get a mirror image of the object. Finally,restore the line to its original position with the inverse rotation and translation transformations. The transformation can be expressed in matrix form as:

$$\begin{bmatrix} 0 & -1 & \mathbf{0} \\ -1 & 0 & \mathbf{0} \\ 0 & 0 & 1 \end{bmatrix} \quad \text{.....(1.33)}$$

1.8.2 Shear

A transformation operation that distorts the shape of an object such that the transformed shape appears as the object was composed of internal layers that had been caused to slide over each other is known as ***shear***. Two common shearing transformations are those that shift coordinate ***x values*** and those that shift ***y values***.

Space for learners:

An x-direction shear relative to the x-axis is generated with the transformation matrix

$$\begin{bmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \text{.....(1.34)}$$

Where the term sh_x is used to represent the ***shear parameter***. A coordinate position (x, y) is then shifted horizontally by an amount proportional to its distance y-value from the x-axis.

CHECK YOUR PROGRESS

7. _____ transformation is used to generate a mirror image of an object.

8. _____ transformation operation that distorts the shape of an object

1.8.3 The Transformation Between Coordinate Systems

In a computer graphics system, it is often required to change the object attributes in between the coordinate systems. Sometimes objects from two-dimensional coordinates need to place into three-dimensional coordinate planes. Sometimes objects are described in non-Cartesian reference frames that take advantage of object symmetries. Coordinate description in these systems must then be converted to Cartesian device coordinates for display. Polar coordinates, elliptical coordinates, and parabolic coordinates are examples of two-dimensional non-Cartesian coordinates. In some cases, need to transform between two Cartesian systems. For modeling and design applications, individual objects may be defined in their own local Cartesian references, and then the local coordinates must be transformed to position the objects within the overall scene coordinate system.

In figure 1.13, it has been shown two Cartesian systems, with the coordinate origins at (0, 0) and (x_0, y_0) and with an orientation angle θ between x and x' axis. To transform object descriptions from xy coordinate to x'y' coordinates, we need to set up a transformation that superimposes the x'y' axes onto the xy axes. This is done in two steps:

Space for learners:

1. Translate so that the origin (x_0, y_0) of the x'y' system is moved to the origin of the xy system.
2. Rotate the x' axis onto the x-axis.

Translation of the coordinate origin is expressed with the matrix operation and the orientation of the two systems after the translation operation would appear in figure 1.14.

$$T(-x_0,-y_0)=\begin{bmatrix}1 & 0 & -x_0\\ 0 & 1 & -y_0\\ 0 & 0 & 1\end{bmatrix} \text{.....(1.35)}$$

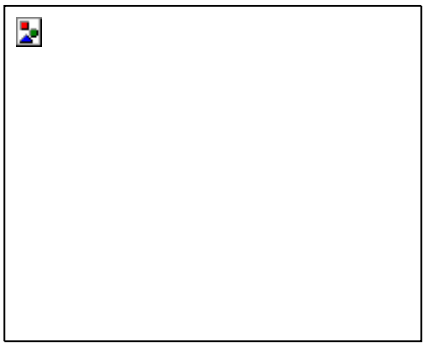


Figure 1.26 A Cartesian x'y' system positioned at (x0, y0) with orientation θ in xy Cartesian system

To get the axes of the two systems into coincidence, then we have to perform a clockwise rotation

$$R(-\theta)=\begin{bmatrix}\cos\theta & \sin\theta & 0\\ -\sin\theta & \cos\theta & 0\\ 0 & 0 & 1\end{bmatrix} \text{....(1.36)}$$

Concatenation of these two transformation matrices gives us the complete composite matrix for transforming object descriptions from the xy system to the x'y' system:

Space for learners:

Space for learners:

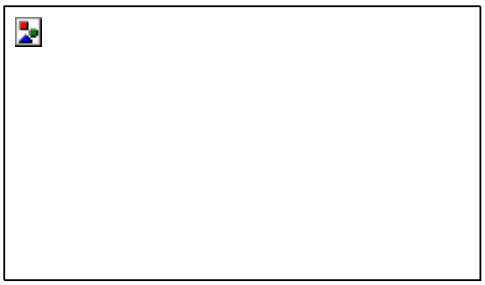


Figure 1.27 Position of the reference frames shown in fig. 1.26after translating the origin of the x'y' system to the coordinate origin of the xy system

1.9 DEFINITION OF AFFINE TRANSFORMATIONS

A coordinate transformation of the form

$$\left.\begin{aligned} x' &= a_{xx}\, x + a_{xy}\, y + b_x \\ y' &= a_{yx}\, x + a_{yy}\, y + b_y \end{aligned}\right\} \quad \text{.....(1.37)}$$

is called a two-dimensional affine transformation. Each of the transformed coordinates x' and y' is the linear function of the original coordinates x and y, and the parameters a_{ij} and b_k are constants determined by the transformation type. Affine transformations have the general properties that parallel lines are transformed into parallel lines and finite points map to finite points.

Translation, rotation, scaling, reflection, and shear are the example of two-dimensional affine transformations.

1.10 SUMMARY

- The layout of a picture can be manipulated by arranging the orientation, shape, and sizes of the component parts of the scene.
- Some of the basic transformations we can perform on objects in a graphics system are rotation, shearing, reflection, and scaling.

- Translation can have by changing the position of an object along a straight line from its current coordinate position to another.
- The translation can be defined as the movement of an object or a rigid body without any changes or deformation.
- A two-dimensional rotation of an object can be defined as reposition of the object coordinates along a circular path in the xy plane.
- The rotation point about which the object is to be rotated is known as the ***pivot point***.
- Counter clock-wise rotation about the pivot point always give a positive value for the rotation angle
- Rotation I clock-wise direction gives a negative value for the rotation angle θ.
- The rotation can be defined as the rotation of an object about a ***rotation axis***, which is perpendicular to the x-y plane and passes through the pivot point.
- The scaling transformation is used to change or alter the size of an object.
- The scaling factors can assign with any positive numeric values.
- By choosing the position of the fixed point, which remains unchanged during the scaling transformation, we can control the location of the scaled object.
- The rotation can be performed about any selected pivot point (x_r, y_r) in the coordinate by performing the following sequences of transformations: **translate – rotate – translate.**
- Reflection transformation is used to generate a mirror image of an object.
- **A reflection relative to the y axis**, i.e. $x = 0$, keeps the y-coordinate same and flips the x coordinates.
- A transformation operation that distorts the shape of an object such that the transformed shape appears as the object was composed of internal layers that had been caused to slide over each other is known as ***shear***.

Space for learners:

1.11 ANSWERS TO THE CHECK YOUR PROGRESS

1. Position
2. pivot point
3. positive
4. negative
5. size
6. Reflection
7. Shear

1.12 MODEL QUESTIONS

1. What do you mean by geometric transformation? How points can be represented in the coordinate system?
2. Explain 2D translation with the help of an example.
3. Calculate the transformation matrix for rotating an object about a specified pivot point.
4. Explain 2D scaling with an example.
5. Explain the term homogeneous coordinate. Why it is necessary for computer graphics?
6. Write the homogeneous coordinate for translation, rotation ad scaling.
7. What is a composite transformation matrix? Explain it with a suitable equation for translation, scaling, and rotation.
8. If a graphic package provides a function that can scale an object concerning the coordinate origin, how can we scale an object concerning any other fixed point?
9. Explain reflection and shearing about the x-axis and y-axis. Represent these transformations with the help of matrices and diagrams.
10. How transformation between two Cartesian frames or systems can be done?
11. Write a short note n 2D affine transformation.
12. Prove that two 2D rotations about the origin commute, i.e. $R_1R_2 = R_2R_1$.
13. Prove that two transformation commute, i.e. $S_1S_2 = S_2S_1$.
14. Applying a 2D rotation followed by a scaling transformation is the same as applying first the scaling transformation and then the rotation. Justify.

Space for learners:

15. Prove that if rotation angle is θ, the information matrix formed when multiplied by the transformation matrix formed when the angle is –θ is equal to the identity matrix.
16. Construct a triangle ABC whose coordinates are A(4, 1), B(5, 2) and C(4, 3).
17. Reflect the given triangle about the x-axis.
18. Reflect the given triangle about the y-axis.
19. Reflect the given triangle about an axis perpendicular to xy plane passing through the origin.
20. Reflect the given triangle about the x = y axis.
    (In each case find the coordinates of the reflected triangle.)

Space for learners:

1.13 REFERENCES AND SUGGESTED READINGS

- Introduction To Computer Graphics And Mu, by Anirban Mukhopadhyay and Arup Chatterjee, Vikas Publishing House Pvt. Ltd.
- Computer Graphics: C version by Donald Hearn & M. Pauline Baker.
- Computer Graphics Principles & Practice by John F. Hughes, Andries Van Dam, Morgan Mcguire, David F. Sklar, James D. Foley, Steven K. Feiner, Kurt Akeley.

UNIT 2: 3-DIMENTIONAL GEOMETRIC TRANSFORMATIONS

Unit Structure:

Space for learners:

2.1 INTRODUCTION

The world consists of three dimensional objects. The objects have height, width and depth. Transformation is a process of modifying and re-positioning the existing graphics. The geometric transformations play an important role in generating images of three dimensional objects with the help of these transformations. 3D-Transformations take place in a three dimensional plane. Transformations are helpful in changing the position, size, orientation, shape etc of the object. The three dimensional transformations are an extension to the two dimensional transformations. In 2D two coordinates are used, i.e., x and y whereas in 3D three co-ordinates x, y, and z are used.

In this unit, you will learn the transformation techniques of the pictures and images. Geometric transformations are mappings from one coordinate system onto itself. The transformations are applied to the objects. By transformation, we mean modifying and re-positioning the

existing graphics. The geometric transformations help to generate the three dimensional objects images. Operations such as moving, reflecting, scaling, rotating an image are called geometric transformations and will be discussed in this unit for three dimensions.

Three-dimensional transformation is similar to 2D transformations, with some minor differences. Here, we have 4 X 4 transformation matrices (in homogeneous coordinate system) instead of 3 X 3 matrices.

In computer graphics, the various transformation techniques are:

i. Translation
ii. Rotation
iii. Scaling
iv. Reflection
v. Shear

Transformation helps to change the size, shape, position etc of an object.

Space for learners:

2.2 UNIT OBJECTIVES

After going through this unit, you will be able to

- Know about the various 3D Geometric transformation techniques
- Learn how the transformations are done in graphics
- Know about the different rotational transformation in computer graphics.

2.3 TRANSLATION

Translation means movement of an object from one position to another. Three dimensional translation is moving an object from one position to another in the three dimensional plane. Translation is done using translation vectors. The translation vectors are also called translation parameters or translation distances. In three dimensional translations, three vectors instead of two are used. Translation in the x-direction is

represented using t_x. The translation is y-direction is represented using t_y. The translation in the z- direction is represented using t_z.

A point (x,y,z) is translated to a new location ($x^/$,$y^/$,$z^/$) by adding the translation parameters t_x , t_y and $t_Z$to the original coordinates x, y and z.

$$x^/ = x + t_x$$

$$y^/ = y + t_y$$

$$z^/ = y + t_Z$$

Three-dimensional transformations are performed by transforming each vertex of the object. If an object has three corners, then the translation will be accomplished by translating all the three points to new locations.

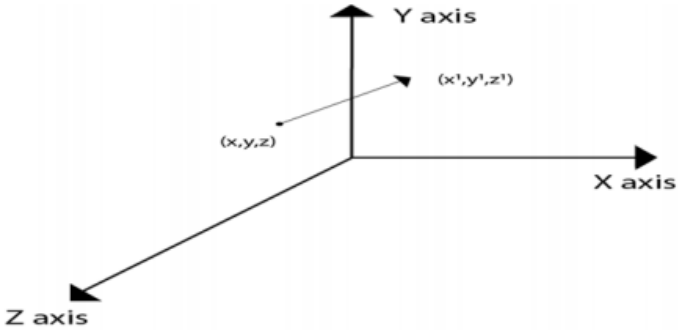


Figure : 3D Translation of a point (x,y,z)

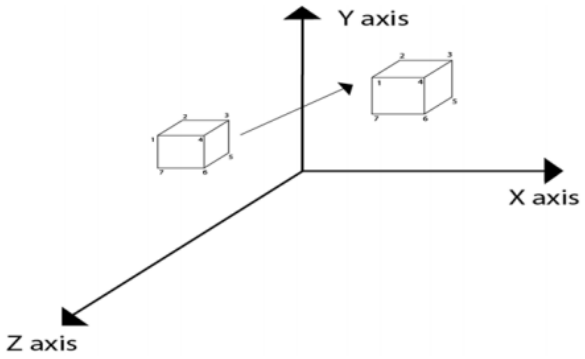


Figure : 3D Translation of an object

Space for learners:

The point (x,y,z) becomes (x',y',z') after translation. t_x, t_y and $t_z$are the translation vectors.

Translation in 3D is a simple extension from that in 2D. In matrix form,

$$T(t_x,t_y,t_z) = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

This can be written as,

$P' = T.P$

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

- Translation is a rigid body transformation which moves the objects without deformation.
- 3D translation is very similar to 2D translation.
- There are now three coordinates to translate (t_x, t_y, t_z).
- A 4X4 matrix is used to perform the three dimensional translation operations.

Example: A Point has coordinates P (1, 2, 3) in x, y, z-direction. Apply the translation with a distance of 2 towards x-axis, 3 towards y-axis, and 4 towards the z-axis. Find the new coordinates of the point?

Solution: We have,
Point P = (x0, y0, z0) = (1,2,3)
Shift Vector = (Tx, Ty, Tz)
Let us assume the new coordinates of P = (x1, y1, z1)
Now we are going to add translation vector and given coordinates, then

Space for learners:

X1 = x0 + Tx = (1 + 2) = 3

Y1 = y0 + Ty = (2 + 3) = 5
Z1 = z0 + Tz = (3 + 4) = 7
Thus, the new coordinates are = (3, 5, 7)

Space for learners:

2.4 ROTATION

3D Transformations take place in a three dimensional plane. In 3D, there are three basic rotations, with respect to each of the principle axes X, Y and Z. The 3-dimensional rotation is much more complex than that of 2D because 3D transformation rotation can be specified around any line in space. The transformations are helpful in changing the position, size, orientation, shape etc of the object.

The transformation matrix for rotation about an arbitrary axis (any axis of rotation other than the principle axis) is more complicated.

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

- X-axis Rotation
- Y-axis Rotation
- Z-axis Rotation

2.4.1 Rotation about the Coordinate Axes

The 2D **z-axis rotation** equations can be extended to 3D as

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

$$y' = x \sin\theta + y \cos\theta \qquad \longrightarrow \qquad (2.1)$$

$$z' = z$$

where θ is the rotation angle. Thus, the 3D ***z-axis rotation*** can be expressed in homogeneous coordinates as

Space for learners:

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

which can be rewritten as

$$P' = R_z(\theta) . P$$

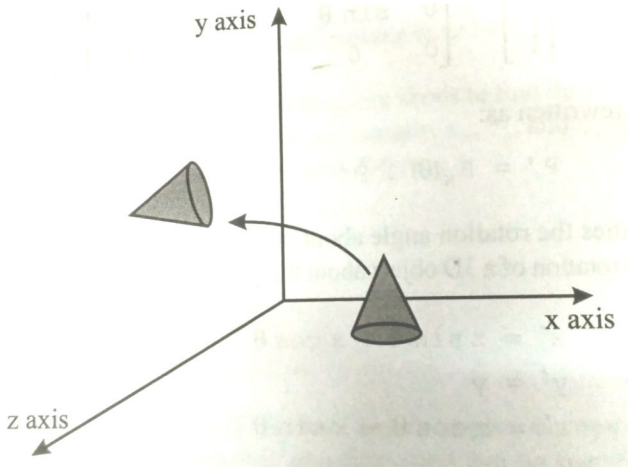


Figure : Rotation of an object about the z-axis

Rotations about the other two axes can be obtained by the following replacement.

X ⟶ y ⟶ z ⟶ x ……. (2.2)

Substituting the (2.2) in the equation 2.1, the equations for the ***x-axis rotation*** are obtained as follows:

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

$$z' = y\sin\theta + z\cos\theta \longrightarrow \quad (2.3)$$

$x^{/} = x$

which can be expressed in homogeneous coordinates as

$$\begin{pmatrix} x^{/} \\ y^{/} \\ z^{/} \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} . \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

This can be rewritten as,

$P^{/} = R_x(\theta) . P$

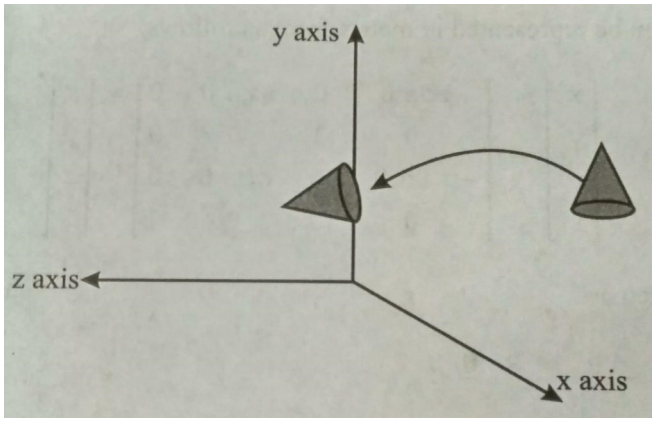


Figure : Rotation of an object about the x-axis

Substituting (2.2) in the equation 2.3, the equations for the ***y-axis rotation*** are obtained as follows:

$z^{/} = z \cos\theta - x \sin\theta$

$x^{/} = z \sin\theta + x \cos\theta$ ⟶ (2.4)

$y^{/} = y$

which can be expressed in homogeneous coordinates as

Space for learners:

Space for learners:

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

This can be rewritten as,

$$P' = R_y(\theta) . P$$

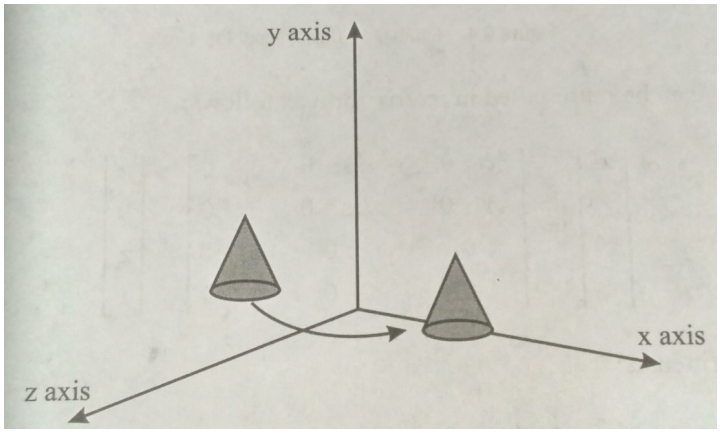


Figure : Rotation of an object about the y-axis

Rotate an object about an axis parallel to one of the coordinate axis:

For rotating an object about an axis parallel to one of the coordinate axis, the following steps needs to be performed.

1. Translate: object is translated in such a way that the rotation axis coincides with the parallel coordinate axis.
2. Rotate : object is then rotated by the specified angle about that axis.
3. Translate: the object is finally translated so that the rotation axis is moved backed to the original position.

Mathematically, it can be expressed as,

$$P' = T^{-1} . R(\theta) . T . P$$

where $R(\theta) = T^{-1}. R_x(\theta) .T$ is the composite transformation.

The steps are shown in the figure below.

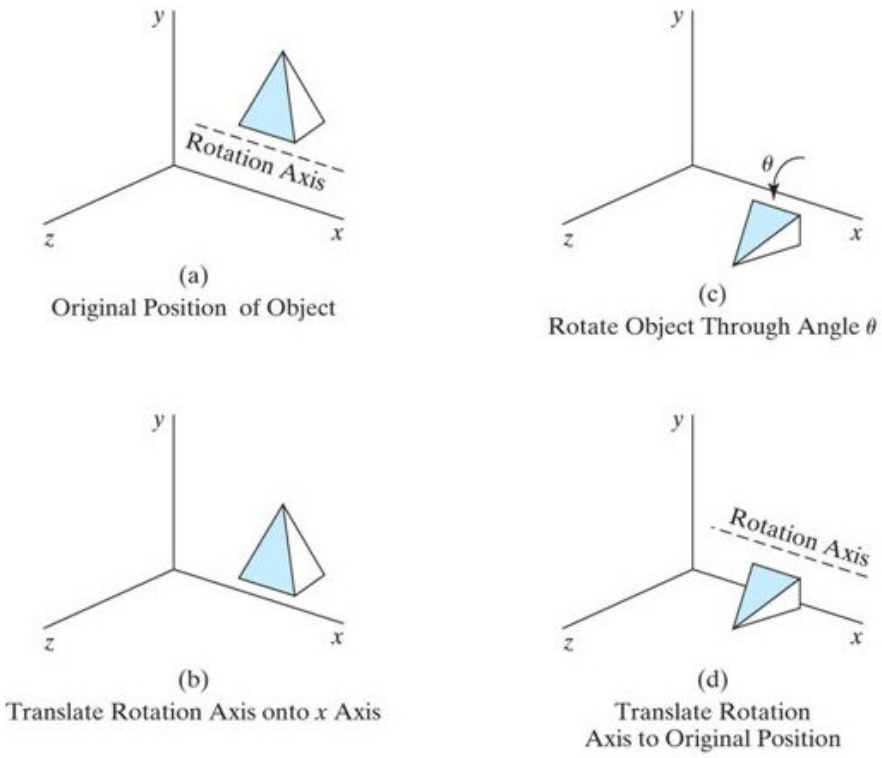


Figure : Rotating an object about an axis parallel to the x-axis

Rotate an object about an axis not parallel to any of the coordinate axis:

For rotating an object about an axis not parallel to any of the coordinate axis, the following steps needs to be performed.

1. Translate: The object is translated such that the rotation axis passes through the coordinate origin.
2. Rotate : Object is rotated such that its axis of rotation coincides with any of the coordinate axes
3. Rotate: It is again rotated by the specified angle about the coordinate axis fro where the axis of rotation has passed.
4. Inverse rotation: This brings the rotation axis to its original position.

Space for learners:

5. Inverse translation: This is done to bring back the rotation axis to its original position.

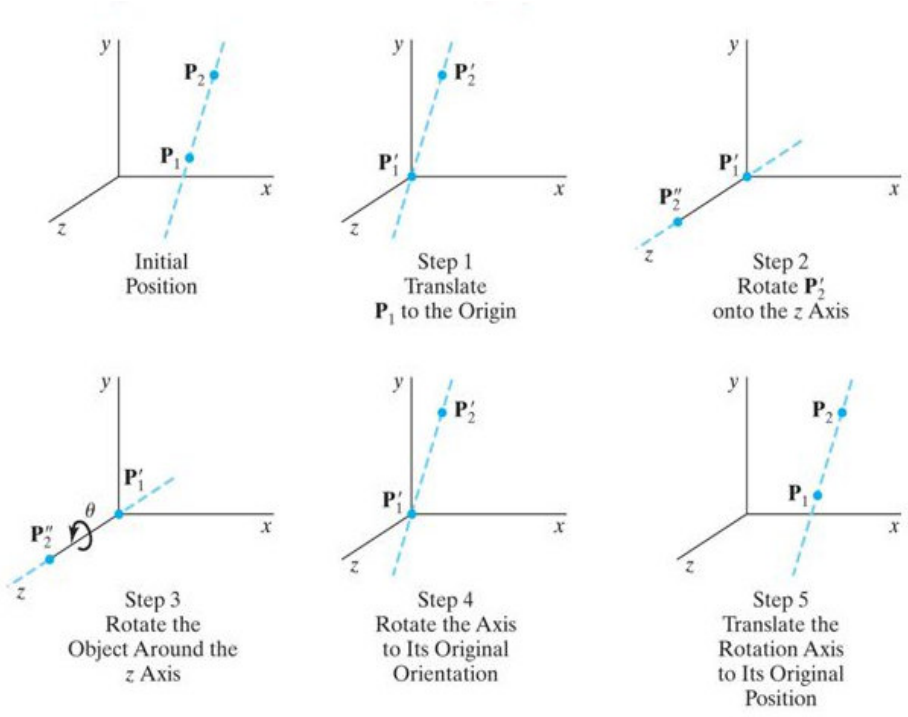


Figure : Rotating an object about an arbitrary axis

Example: A Point has coordinates P (2, 3, 4) in x, y, z-direction. The Rotation angle is 90 degrees. Apply the rotation in x, y, z direction, and find out the new coordinates of the point?

Solution:

The initial coordinates of point = P (x0, y0, z0) = (2, 3, 4)

Rotation angle, $\theta = 90°$

<u>For x-axis</u>

Let the new coordinates = (x1, y1, z1) then,

$x1 = x0 = 2$

$y1 = y0 \cos\theta - z0 \sin\theta = 3 \times \cos90° - 4 \times \sin90° = 3 \times 0 - 4 \times 1 = -4$

$z1 = y0 \sin\theta + z0 \cos\theta = 3 \times \sin90° + 4 \times \cos90° = 3 \times 1 + 4 \times 0 = 3$

The new coordinates of point = (2, -4, 3)

<u>For y-axis</u>

Let the new coordinates = (x1, y1, z1) then,

Space for learners:

X1= z0 sinθ + x0 cosθ = 4 x sin90° + 2 x cos90° = 4 x 1 + 2 x 0 = 4
y1= y0 = 3
z1= y0 cosθ – x0 sinθ = 3 x cos90°– 2 x sin90° = 3 x 0 – 4 x 0 = 0
The new coordinates of point = (4, 3, 0)

For z-axis
Let the new coordinates = (x1, y1, z1) then,
x1= x0 cosθ – y0 sinθ = 2 x cos90° – 3 x sin90° = 2 x 0 + 3 x 1 = 3
y1= x0 sinθ + y0 cosθ = 2 x sin90° + 3 x cos90° = 2 x 1 + 3 x 0 = 2
z1= z0 =4
The New Coordinates of points = (3, 2, 4)

CHECK YOUR PROGRESS – I

1. Repositioning an object along a straight line path from one coordinate location to another coordinate location is called_____.
2. The translation vectors are also called _____.
3. Translation is a transformation which moves the objects without _____.
4. A _____ matrix is used to perform the three dimensional translation operations.
5. In 3D, there are _____ basic rotations, with respect to each of the principle axes X, Y and Z.
6. To rotate an object about an axis parallel to one of the coordinate axis, the steps involved are _____, _____ and _____.
7. Inverse rotation is required to rotate an object about an axis not parallel to any of the coordinate axis so as to brings the rotation axis to its _____ position.
8. The transformation matrix for rotation about an _____ axis is more complicated.

Space for learners:

2.5 SCALING

Space for learners:

Scaling is used to change the size of an object and reposition the object relative to the coordinate origin. The size of an object can be increased or decreased by scaling. The object size is changed by compressing or expanding its dimensions.

For example, the point P will be scaled from its original position P(x,y,z) to a new position $P^{/}$ ($x^{/}$,$y^{/}$,$z^{/}$) relative to the coordinate origin from the scaling matrix expression.

$$\begin{pmatrix} x^{/} \\ y^{/} \\ z^{/} \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} . \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \longrightarrow (2.5)$$

where sx, sy and sz are the scaling factors in the x, y and z directions. The scaling factor determines whether the object size is to be increased or reduced.

- If scaling factor > 1, then the object size is increased.
- If scaling factor < 1, then the object size is reduced.

x', y' and z' can be written as follows:

$$x^{/} = x \,.s_x$$

$$y^{/} = y \,.s_y$$

$$z^{/} = z \,.s_z$$

The equation number 2.5 can also be written as

$$\mathbf{P^{/} = S \,.\, P}$$

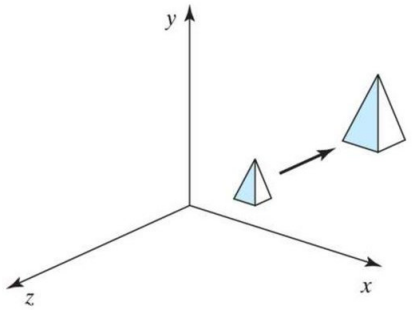


Figure: Scaling an object with the scaling parameters set to 2 (Increase in size)

If the transformation parameters are not all equal, relative dimensions in the object are changed. The original shape of an object is preserved by uniform scaling that is $s_x = s_y = s_z$.

If all scaling factors Sx=Sy=Sz. Then scaling is called as *uniform scaling*. If scaling is done with different scaling vectors, it is called a *differential scaling*.

In the figure above, the scaling parameters $s_x = s_y = s_z = 2$. This has increased both the size and the distance of the object by a factor of 2.

Scaling of an object relative to a fixed point

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

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

Example: A 3D object that have coordinates points P(1, 4, 4), Q(4, 4, 6), R(4, 1, 2), T(1, 1, 1) and the scaling parameters are 3 along with x-axis, 4 along with y-axis and 4 along with z-axis. Apply scaling to find the new coordinates of the object?

Solution: We have,
The initial coordinates of the object = P (1, 4, 4),
Q (4, 4, 6),

Space for learners:

R (4, 1, 2),
S (1, 1, 1)

Scaling factor along with x-axis (Sx) = 3
Scaling factor along with y-axis (Sy) = 4
Scaling factor along with z-axis (Sz) = 4
Let the new coordinates after scaling = **(x1, y1, z1)**

For coordinate P
x1 = x0 x Sx = 1 x 3 = 3
y1 = y0 x Sy = 4 x 4 = 16
z1 = z0 x Sz = 4 x 4 = 16
Therefore, the new coordinates = (3, 16, 16)

For coordinate Q
x1 = x0 x Sx = 4 x 3 = 12
y1 = y0 x Sy = 4 x 4 = 16
z1 = z0 x Sz = 6 x 4 = 24
Therefore, the new coordinates = (12, 16, 24)

For coordinate R
x1 = x0 x Sx = 4 x 3 = 12
y1 = y0 x Sy = 1 x 4 = 4
z1 = z0 x Sz = 2 x 4 = 8
Therefore, the new coordinates = (12, 4, 8)

For coordinate S
x1 = x0 x Sx = 1 x 3 = 3
y1 = y0 x Sy = 1 x 4 = 4
z1 = z0 x Sz = 1 x 4 = 4
Therefore, the new coordinates = (3, 4, 4)

Thus, the new coordinates after scaling = P (3, 16, 16), Q (12, 16, 24), R (12, 4, 8), S (3, 4, 4).

Space for learners:

2.6 REFLECTION

Reflection produces a mirror image of an object. An object is rotated a particular angle about the axis of reflection to get the reflected image.

A 3D reflection can be performed in one of the following ways:

1. With respect to a selected reflection axis
2. With respect to a selected reflection plane.

When reflection is performed with respect to a selected reflection axis, the object of reflection is rotated by 180^0 about that axis. If the reflection is performed with respect to a selected reflection plane then also the object is rotated by 180^0. The plane selected is xy, xz or yz. However, the rotation will be performed in 4D space.

Thus, in 3-dimension, there are three possible types of reflection.

a. Reflection relative to xy-plane
b. Reflection relative to yz-plane
c. Reflection relative to xz-plane

Reflection about the xy- plane

Reflection relative to the xy-plane can be obtained by the following set of instructions.

$x' = x \qquad y' = y \qquad z' = -z$

In homogeneous matrix form,

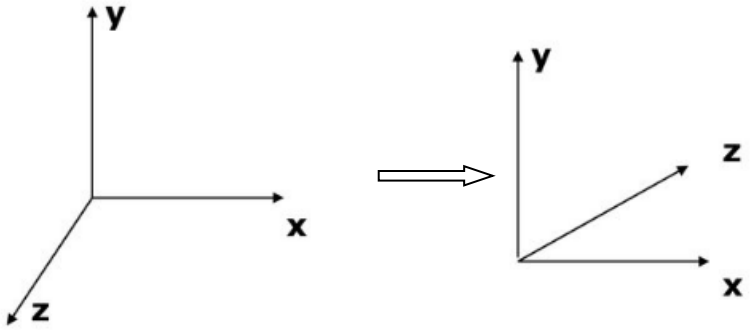
$$RF_{xy} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Reflections about other planes can also be obtained by symmetry.

Reflection about xy-plane is a useful reflection as it converts a right-handed coordinate system into a left-handed coordinate system.

Space for learners:

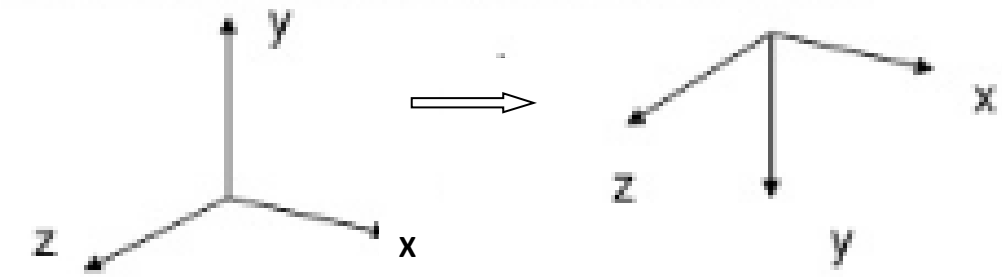
Space for learners:



Similarly, the transformation matrix representation for the reflection of points with respect to the xz-plane (in which only the sign of the y coordinate is changed) is given as follows:

$$RF_{xz} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The figure below shows the reflection transformation with respect to the xz-plane.

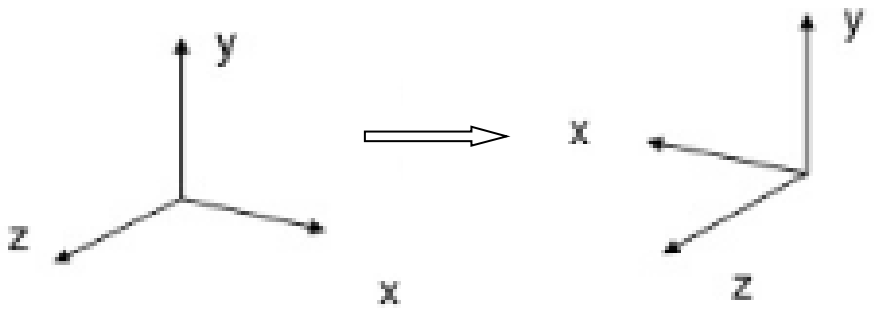


The transformation matrix representation for the reflection in the yz-plane (where only the sign of the x-coordinate value is changed) is given as follows:

Space for learners:

$$RF_{yz} = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The figure below shows the reflection transformation with respect to the yz-plane.



In reflection, the reflected object is always formed on the other side of mirror. Also, size of reflected object is same as the size of original object.

Example: A 3D triangle with coordinates points P (4, 5, 2), Q (7, 5, 3), R (6, 7, 4). Apply reflection on xy plane and find the new coordinates of triangle?

Solution: We have,

The initial coordinates of triangle = P (4, 5, 2),
Q (7, 5, 3),
R (6, 7, 4)

Reflection Plane = xy

Let the new coordinates of triangle = (x1, y1, z1)

For Coordinate P (4, 5, 2)–

X1 = x0 = 4

y1 = y0 = 5

z1 = -z0= -2

Therefore, the new coordinates = (4, 5, -2)

For Coordinate Q (7, 5, 3)–

X1 = x0 = 7

Y1 = y0 = 5

Z1 = -z0= -3

Therefore, thenew coordinates = (7, 5, -3)

For Coordinate P (6, 7, 4)–

X1 = x0 = 6

y1 = y0 = 7

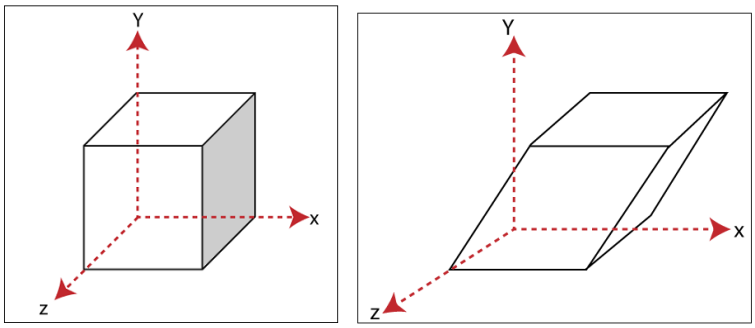
z1 = -z0= -4

Therefore, the new coordinates = (6, 7, 4)

2.7 SHEAR

Shear is a technique that changes the shape of an object. It is also called as deformation. Shearing causes the objects to look as if the internal layers are caused to slide over each other. The object is pulled horizontally by different amounts. In 3-dimension, shear can occur in three directions.

Space for learners:



Original After Shearing

3D shearing transformation is similar to 2D shearing transformation except that in 3D shearing one can shear along the z-axis. Shearing along the x-axis keeps the y-coordinate and the z-coordinate unchanged. Only the x-coordinate gets changed. This makes the object tilt towards the x-direction.

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

- Xnew = Xold
- Ynew = Yold + Sh_y x Xold
- Znew = Zold + Sh_z x Xold

The transformation matrix of 3D shearing along the x-axis is as follows:

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

Where a and b are parameters that can be assigned any real values.

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

Xnew = Xold + Sh_x x Yold

Space for learners:

Ynew = Yold

Znew = Zold + Sh_z x Yold

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

Xnew = Xold + Sh_x x Zold

Ynew = Yold + Sh_y x Zold

Znew = Zold

The transformation matrix of 3D shearing along the y-axis and the z-axis is as follows:

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

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

Shearing in the y-axis and the z-axis tilts the object towards the y- and z- directions respectively.

Example: Given a 3D triangle with points (0, 0, 0), (1, 1, 2) and (1, 1, 3). Apply shear parameter 2 on X axis, 2 on Y axis and 3 on Z axis and find out the new coordinates of the object.

Solution: We have,

Initial coordinates of the triangle = A (0, 0, 0),

Space for learners:

B(1, 1, 2),
C(1, 1, 3)

Shearing parameter along with x-axis (Sh_x) = 3
Shearing parameter along with y-axis (Sh_y) = 4
Shearing parameter along with z-axis (Sh_z) = 4
Let the new coordinates of triangle = (x1, y1, z1)

Shearing in x-axis

For Coordinate A (0, 0, 0)

Let the new coordinates of corner A after shearing = (Xnew, Ynew, Znew).
Applying the shearing equations, we have-
Xnew = Xold = 0
Ynew = Yold + Sh_y x Xold = 0 + 2 x 0 = 0
Znew = Zold + Sh_z x Xold = 0 + 3 x 0 = 0

Thus, the new coordinates of the corner A after shearing = (0, 0, 0).

For Coordinates B (1, 1, 2)

Let the new coordinates of corner B after shearing = (Xnew, Ynew, Znew).

Applying the shearing equations, we have-
Xnew = Xold = 1
Ynew = Yold + Sh_y x Xold = 1 + 2 x 1 = 3
Znew = Zold + Sh_z x Xold = 2 + 3 x 1 = 5

Thus, the new coordinates of the corner B after shearing = (1, 3, 5).

For Coordinates C (1, 1, 3)

Let the new coordinates of corner C after shearing = (Xnew, Ynew, Znew).

Space for learners:

Applying the shearing equations, we have-
Xnew = Xold = 1
Ynew = Yold + Sh_y x Xold = 1 + 2 x 1 = 3
Znew = Zold + Sh_z x Xold = 3 + 3 x 1 = 6

Thus, the new coordinates of the corner C after shearing = (1, 3, 6).

Therefore, new coordinates of the triangle after shearing in X axis = A (0, 0, 0),
B(1, 3, 5),
C(1, 3, 6).

Shearing in Y Axis-

For Coordinates A (0, 0, 0)

Let the new coordinates of corner A after shearing = (Xnew, Ynew, Znew).

Applying the shearing equations, we have-
Xnew = Xold + Sh_x x Yold = 0 + 2 x 0 = 0
Ynew = Yold = 0
Znew = Zold + Sh_z x Yold = 0 + 3 x 0 = 0

Thus, New coordinates of corner A after shearing = (0, 0, 0).

For Coordinates B (1, 1, 2)

Let the new coordinates of corner B after shearing = (Xnew, Ynew, Znew).

Applying the shearing equations, we have-
Xnew = Xold + Sh_x x Yold = 1 + 2 x 1 = 3
Ynew = Yold = 1
Znew = Zold + Sh_z x Yold = 2 + 3 x 1 = 5

Thus, New coordinates of corner B after shearing = (3, 1, 5).

Space for learners:

For Coordinates C(1, 1, 3)

Let the new coordinates of corner C after shearing = (Xnew, Ynew, Znew).

Applying the shearing equations, we have-

$X_{new} = X_{old} + Sh_x \times Y_{old} = 1 + 2 \times 1 = 3$

$Y_{new} = Y_{old} = 1$

$Z_{new} = Z_{old} + Sh_z \times Y_{old} = 3 + 3 \times 1 = 6$

The new coordinates of the corner C after shearing = (3, 1, 6).

Thus, New coordinates of the triangle after shearing in Y axis = A (0, 0, 0), B(3, 1, 5), C(3, 1, 6).

Shearing in Z Axis-

For Coordinates A(0, 0, 0)

Let the new coordinates of corner A after shearing = (Xnew, Ynew, Znew).

Applying the shearing equations, we have-

$X_{new} = X_{old} + Sh_x \times Z_{old} = 0 + 2 \times 0 = 0$

$Y_{new} = Y_{old} + Sh_y \times Z_{old} = 0 + 2 \times 0 = 0$

$Z_{new} = Z_{old} = 0$

The new coordinates of corner A after shearing = (0, 0, 0).

For Coordinates B(1, 1, 2)

Let the new coordinates of corner B after shearing = (Xnew, Ynew, Znew).

Applying the shearing equations, we have-

$X_{new} = X_{old} + Sh_x \times Z_{old} = 1 + 2 \times 2 = 5$

$Y_{new} = Y_{old} + Sh_y \times Z_{old} = 1 + 2 \times 2 = 5$

$Z_{new} = Z_{old} = 2$

The new coordinates of corner B after shearing = (5, 5, 2).

Space for learners:

For Coordinates C(1, 1, 3)

Let the new coordinates of corner C after shearing = (Xnew, Ynew, Znew).

Applying the shearing equations, we have-

$X_{new} = X_{old} + Sh_x \times Z_{old} = 1 + 2 \times 3 = 7$

$Y_{new} = Y_{old} + Sh_y \times Z_{old} = 1 + 2 \times 3 = 7$

$Z_{new} = Z_{old} = 3$

The new coordinates of corner C after shearing = (7, 7, 3).

Hence the new coordinates of the riangle after shearing in Z-axis = A(0,0,0), B (5,5,2), C (7,7,3)

Space for learners:

CHECK YOUR PROGRESS – II

a. _____ is used to change the size of an object and reposition the object relative to the coordinate origin.

b. If scaling factor is greater than 1, then the object size is _____.

c. The original shape of an object is preserved by _____ scaling.

d. Reflection produces a _____ image of an object.

e. If the reflection is performed with respect to a selected reflection plane then the object is rotated by_____.

f. _____ transformation distorts the shape of the objects and they appear to look as if the internal layers are caused to slide over each other.

g. 3D reflection can be performed with respect to reflection axis and reflection _____.

h. Reflection about xy-plane is a useful reflection as it converts a right-handed coordinate system into a _____ coordinate system.

2.8 SUMMING UP

Space for learners:

- Repositioning an object along a straight line path from one coordinate location to another coordinate location is called translation transformation
- A 4X4 matrix is used to perform the 3D transformation operations.
- In 3D, there are three basic rotations, with respect to each of the principle axes X, Y and Z.
- Scaling changes the size of an object and repositions the object relative to the coordinate origin.
- Like 2D scaling, in 3D scaling also the size of an object is changed by compressing or expanding its dimensions.
- The original shape of an object is preserved by uniform scaling that is $s_x = s_y = s_z$
- Reflection produces a mirror image of an object.
- When reflection is performed with respect to a selected reflection axis, the object of reflection is rotated by 180^0 about that axis.
- If the reflection is performed with respect to a selected reflection plane then also the object is rotated by 180^0
- Shear transformation distorts the shape of the objects and they appear to look as if the internal layers are caused to slide over each other.
- 3D shearing transformation is similar to 2D shearing transformation except that in 3D shearing one can shear along the z-axis.
- Complex geometric and coordinate transformations can be done from the basic transformations.

2.9 ANSWER TO CHECK YOUR PROGRESS

CHECK YOUR PROGRESS – I

1. Translation
2. translation parameters or distances
3. deformation
4. 4X4
5. Three
6. Translate, rotate, rotate
7. Original
8. arbitrary

CHECK YOUR PROGRESS – II

a. Scaling
b. Increased
c. Uniform
d. mirror
e. 180^0
f Shear
g Plane
h left-handed

2.10 POSSIBLE QUESTIONS

1. Explain translation of objects in three dimensions.
2. How can you rotate an object about an axis parallel to one of the coordinate axis?
3. Show the steps to rotate an object about an axis not parallel to any of the coordinate axis.
4. Explain 3D Refection and Shearing of objects

2.11 REFERENCES AND SUGGESTED READINGS

- Computer Graphics By Donald Hearn and M. Pauline Baker
- Computer Graphics, Schaum's Outlines, Plastock and Kalley, McGraw-Hill © 1986
- Computer graphics: principles and practice by James D. Foley

Space for learners:

UNIT 3: TWO DIMENSIONAL VIEWING

Space for learners:

Unit Structure:

3.1 INTRODUCTION

In this unit, you will learn about the concept of two-dimensional viewing. The view coordinate system is specified where the view plane displays the view parameter. In the coordinate systems, graphic primitive values are defined. The OpenGL is an Application Programming Interface which specifies graphics, especially three-dimensional graphics. A huge number of applications depend on the use of OpenGL. Before object descriptions, view plane is projected to transfer the viewing coordinates. Conversion of object description from world to viewing coordinates is equivalent to a transformation which superimposes the viewing reference frame into the world frame using basic geometric translate and rotate operations. By defining a closed boundary or window, the enclosed portion of a world coordinate scene is clipped against the window boundary and the data of the clipped

portion is extracted for mapping to a separately defined region known as viewport. Normalization transformation (N) maps world coordinates (or viewing coordinates) to normalized device coordinates and Workstation transformation (W) maps normalized device coordinates to physical device coordinates. You will learn about the various clipping operations. The purpose of clipping procedure is to determine which part of a scene or specifically which points, lines (or curves) or portions of the lines (or curves) of a scene lie inside the clipping window. You will also learn about text as well as exterior clipping. There are several methods which can be used to provide the text clipping in a graphics package. The clipping techniques used depend on the methods used to generate characters and the requirements of a particular application.

Space for learners:

3.2 UNIT OBJECTIVES

After going through this unit, you will be able to:

- Understand the dimensional viewing transformation pipeline of real world objects.
- Know the window to viewport coordinate transformation
- Know the functions related to two-dimensional viewing

3.3 DEFINITION

The two dimensional viewing is a transformation process of real world object into position point which is relative to the viewing volume, especially, the points behind the viewer.

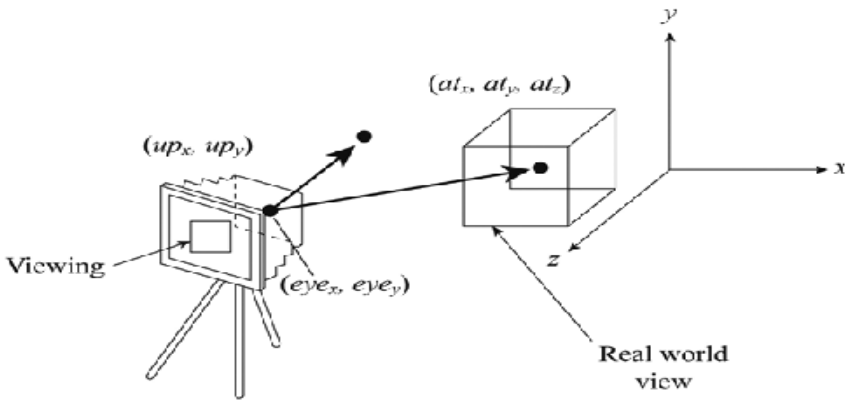


Figure to represent the viewing

3.4 VIEWING TRANSFORMATION PIPELINE

Space for learners:

The viewing pipeline-

Visualization of an object is achieved by a sequence of operations called the viewing pipeline. It is a series of clip operations and transformations that are applied to graphics primitive before capturing the image. It basically transforms the geometric coordinates. A view position in a scene is represented by camera which is associated with a coordinate system. The view coordinate system is specified where we are looking and the view plane where we are displaying sets the view parameter. In the coordinate systems, graphic primitive values are defined. The application of viewing pipeline controls the world coordinates, virtual device coordinates, etc. The world coordinates system performs lighting in the space of window, if illumination requires. The Virtual Device Coordinates (VDC) isolates the applications to view clipping if application requires it. Device coordinates are the actual coordinates of the device (refer Figure 3.1). If the device is a raster, the units of device coordinates are in pixels. In device coordinates, (0,0,0) is the upper left corner, where the axes are oriented right for positive x, down for positive y, and away for positive z.

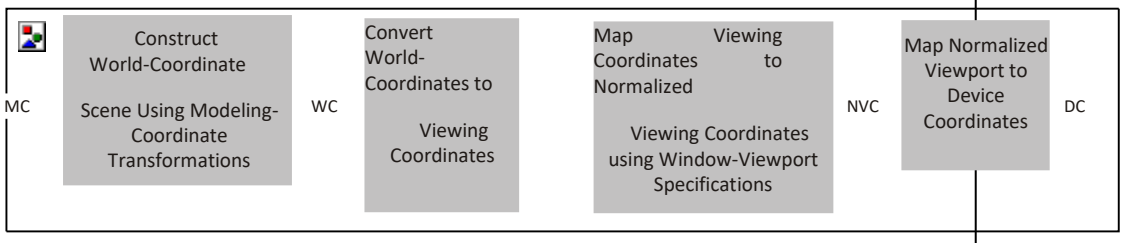


Fig. 3.1 Viewing Pipeline

The following mechanisms are involved in viewing a pipeline

Viewing Transformations

Viewing a pipeline gives mapping view volume to output device, such as screen derive transformation matrix (refer Figure 3.2).

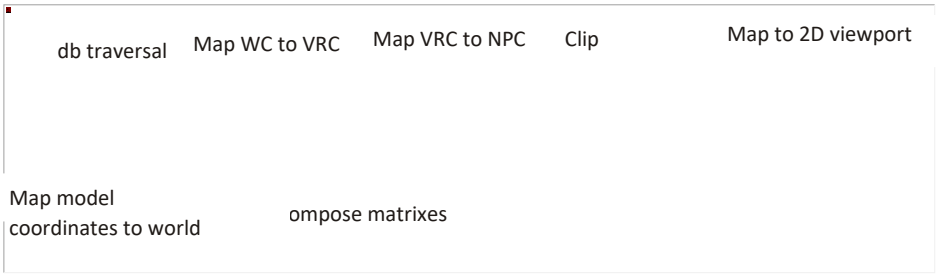
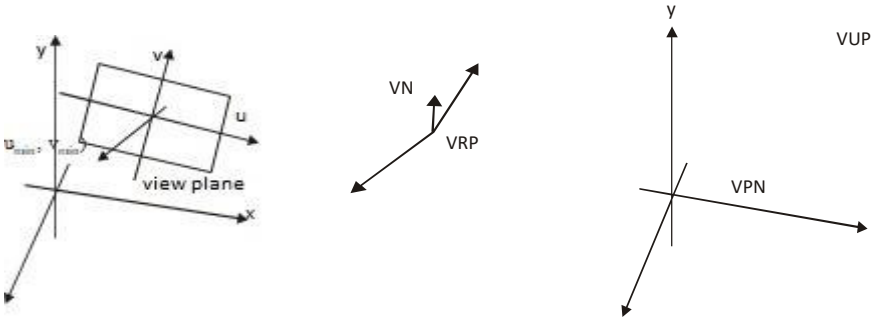


Fig. 3.2 Viewing Transformations

WC represents world coordinates, VRC represents view reference coordinates and NPC represents normalized projection coordinates that are used in the canonical View Volume (CVV). Each mapping corresponds to matrix that is combined to accelerate the process.

VRC

To determine the position of the view plane in the world coordinate system, the first step is to locate the origin, known as View Reference Point (VRP), of the VRC system. After positioning the origin, the n-axis corresponding to the z-axis of the world coordinate system is defined by a vector called the view plane normal. The direction of the view plane normal n, which is perpendicular to the view plane, is defined as a displacement of x units along the x-axis, y units along the y-axis and z units along the z-axis.



View Plane in World Coordinates World Coordinates

Fig. 3.3: View Reference in World Coordinates and View Plane in World Coordinates

Space for learners:

In Figure 3.3, VRP represents view reference point, VN represents view normal, VUP represents view up vector and VPN represents view plane normal. The equation u = v × n gives the value of u where n || VPN and v || VN. A system of world coordinates is given in which VRP, VN and VUP vector are defined. In VRC, the projection reference point, the location of the window in the view plane and the distance of the front and back are given. Clipping planes are related to the projection reference point. Finding transformations that yield a coordinate system so that the front bottom left is at (–1, –1, –1) and the back top right is at (1, 1, 1) is called a canonical view volume. Figure 3.4 shows the parallel projection and perspective projection.

Space for learners:

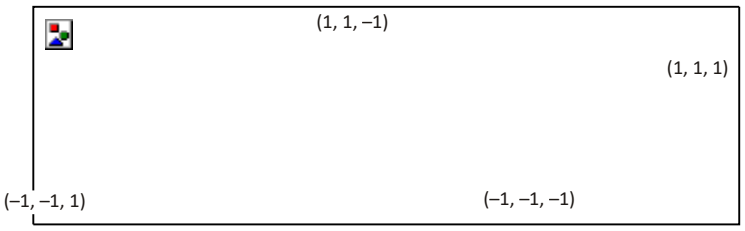


Fig. 3.4 Parallel Projection (Right-handed Coordinates) and Perspective Projection (Left-handed Coordinates)

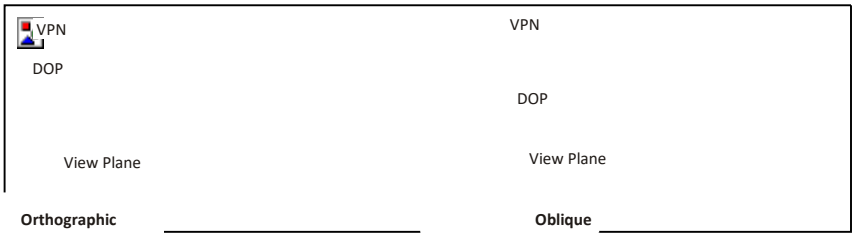


Fig. 3.5: Orthographic and Oblique View Plane

Figure 3.5 shows that in orthographic projection the view plane normal is parallel to the direction of projection and in oblique projection the view plane normal is not parallel to the direction of projection. To transform the world coordinate system from world coordinates to view reference coordinates, the required steps are as follows:

Step 1: Translate VRP to Origin

The first stage of transforming the world coordinate system from world coordinates to view reference coordinates is a translation of the VRP to the origin of the world system. The translation can then be applied to

each model.

Step 2: Rotate (u, v, n) into (x, y, z)

The second stage of the transformation is to rotate the axes of the VRC

$$T = \begin{bmatrix} 1 & 0 & 0 & -VRP_x \\ 0 & 1 & 0 & -VRP_y \\ 0 & 0 & 1 & -VRP_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

system into the axes of the world coordinate system. Applying this transformation after the translation in the previous slide allows the models to be transformed into VRC. First the u, v and n in world coordinates are determined.

$n = VPN / ||VPN||$ $u = VUP \times VPN / ||VUP \times VPN||$ $n = n \times u$

Then the rows of the rotation matrix out of the components of the three vectors are formed.

OpenGL commands used in viewing pipeline

The OpenGL is an application programming interface (API) that specializes to bring graphics, especially three-dimensional graphics. A huge number of applications depend on the use of OpenGL. It is an efficient way to save money and time when creating software. In OpenGL, the view reference coordinates are the parts of the 'model view' matrix. There are two ways of setting up the coordinates. In GL, the commands for translation and rotation can be used:

```
void glTranslatef(GLfloat x, GLfloat y, GLfloat z)
\
void glRotatef(GLfloat angle, GLfloat x, GLfloat y, GLfloatz)
```

Alternatively, in the utility toolkit GLU, the ommands are as follows:

```
void gluLookAt(GLdouble eyex, GLdouble eyey, GLdouble eyez,
 GLdouble centerx, GLdouble centery, GLdouble centerz,
GLdouble upx, GLdouble upy, GLdouble upz)
```

This eye coordinates specify a look-from point, the centre coordinates a look-at point and the up coordinates the head-up vector. The matrix stack currently used is chosen with the following command:

```
void glMatrixMode()
```

OpenGL only implements orthographic projections. If you need other oblique projections, you have to form the M0 matrix yourself and use

Space for learners:

glMultMatrix() to modify the projection matrix. GL provides the following function:

> void glOrtho(GLdouble left, GLdouble right, GLdouble bottom, GLdouble top, GLdouble near, GLdouble far)

By the above command, the matrix produces as follows:

$$M = ST = \begin{bmatrix} \frac{2}{u_{max}-u_{min}} & 0 & 0 & \frac{-(u_{max}+u_{min})}{u_{max}-u_{min}} \\ 0 & \frac{2}{v_{max}-v_{min}} & 0 & \frac{-(v_{max}+v_{min})}{v_{max}-v_{min}} \\ 0 & 0 & \frac{-2}{n_{max}-n_{min}} & \frac{-(n_{max}+n_{min})}{n_{max}-n_{min}} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Parallel Projection OpenGL

The following code sets up an orthogonal projection and then returns to the model view mode. The glLoadIdentity() function is needed to replace the existing matrix with an identity matrix as the glOrtho() function pre-multiplies whatever is at the top of the matrix stack.

```
glMatrixMode(GL_PROJECTION);
glPushMatrix(); //optional, preserves a previous matrix glLoadIdentity();
//sets top of stack to id matrix glOrtho(Xmin, Xmax, Ymin, Ymax, nearClip,
farClip);

//Defined Functions along with coordinates
glMatrixMode(GL_MODELVIEW);
```

A shear must be applied to bring the centre line of the view volume parallel to the z axis, the x and y coordinates scaled so that the sides of the frustrum lie on the lines x = +–z and y = +– z (refer Figure 3.6)

Space for learners:

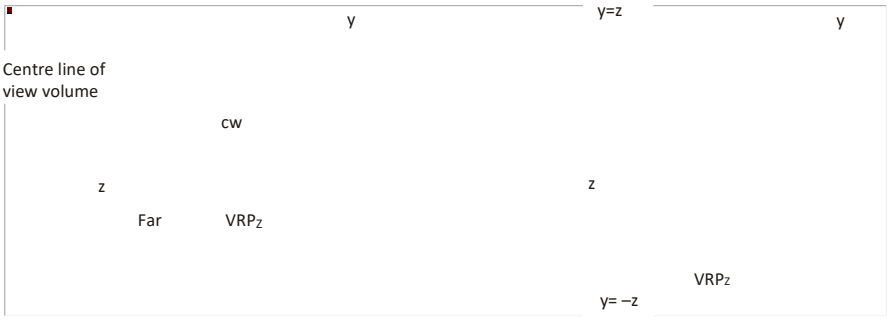


Fig. 3.6 Centre Line of View Volume

The OpenGL perspective matrix can be generated by following GL command:

void glFrustum(GLdouble left, GLdouble right, GLdouble bottom,

GLdouble top, GLdouble near, GLdouble far);

In the above command, left and right are the max and min of x, whereas top and bottom are max and min of y. The simplified version for symmetric viewports in the utility library GLU is as follows:

void gluPerspective(GLdouble fovy, GLdouble aspect, GLdouble near, GLdouble far)

In the above command, fovy represents the vertical field of view and aspect tells the aspect ratio of (width / height) (refer Figure 3.7)

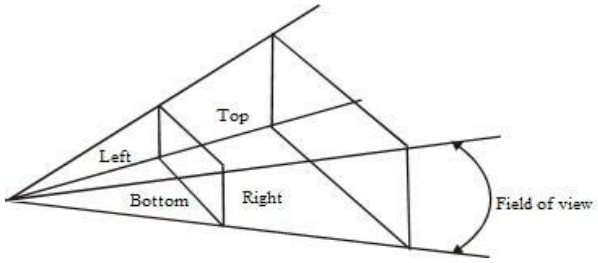


Fig. 3.7 Field of View

The final stage is to map the window on the view plane into the view port on the screen. To do this, a transformation is applied that maps the

Space for learners:

bottom left corner of the window into the pixel location of the bottom left of the viewport. The height and width of the window are also transformed to match the viewports pixel height and width. GL provides the following command:

void glViewport(GLint x, GLint y, GLsizei width, GLsizei height)

Note: If the aspect ratio of the viewport is different from the window on the view plane, then distortions will occur. The following code shows the produced output as cone in the red colour on the screen.

```
#include <Inventor/Xt/SoXt.h > #include <Inventor/Xt/SoXtRender.h >

//Camera

#include <iostream.h > #include <stdio.h >

#include <Inventor/nodes/SoCone.h>

#include <Inventor/nodes/SoDirectionalLight.h> #include
<Inventor/nodes/SoPerspectiveCamera.h> #include
<Inventor/nodes/SoTranslation.h> #include <Inventor/nodes/SoSeparator.h>
#include <Inventor/nodes/SoMaterial.h>

#include <Inventor/nodes/SoWriteAction.h>

//This header file writes a scene graph to memory buffer.


//All header files are declared reside in /Inventor/ nodes system directory


int main( int, char** argv )

//Main function is of integer data type having two arguments

{

// Initialize Inventor


Widget myWindow = SoXt::init(argv[0]);

//Define my window using scope resolution operator
```

Space for learners:

```
if(myWindow == NULL)
//Checking condition if value NULL comes

exit(1);
//Exit after checking the condition

// Make a scene containing a red cone by defining constructor

SoSeparator *root = new SoSeparator; SoPerspectiveCamera *myCamera = new
SoPerspectiveCamera; SoMaterial *myMaterial = new SoMaterial;

root->ref();
root->addChild(myCamera);
root->addChild(new SoDirectionalLight); myMaterial-
>diffuseColor.setValue(1.0, 0.0, 0.0);
//Red color is set for cone root->addChild(myMaterial); root->addChild(new
soCone);
//Create a render Area from cone graph
SoXRenderArea *myRenderArea = new SoXRenderArea(myWindow);
  //The render area will appear within the main window. m y C a m e r a -> v i ew Al l( ro ot , my Re nd er
>getViewportRegion();
//Writing inventor data to ASCII file “myCone.iv” SoWriteAction *myAction =
new SoWriteAction; myAction->getOutput()->openFile(“myCone.iv”);
//Openening the file ‘myCone.iv’ myAction->getOutput()->setBinary(FALSE);
//Set the binary statement myAction->apply();
myAction->getOutput()->closeFile ();
//Closing the file delete myAction;
//Destructor
myRender Area->setSceneGraph(root); myRender Area->setTitle(“Hello Cone”);
//Title is set the for cone graphics myRenderArea->show();
SoXt::show(myWindow);
```

Space for learners:

```
//Display main window SoXt::mainLoop();

//Main Inventor event loop

}
```

Figure 3.8 shows how the result comes on the screen.

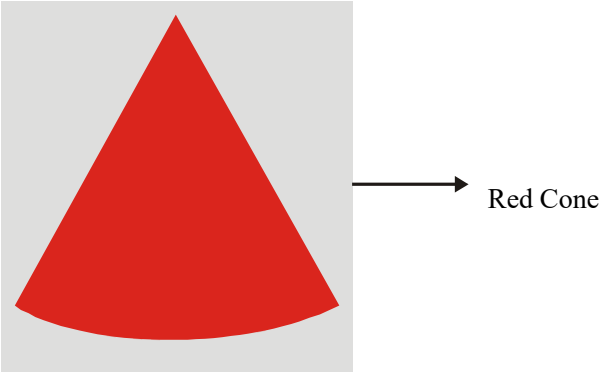


Fig 3.8: Cone

3.4.1 Viewing Coordinate Reference Frame

Before object a description, view plane is projected to transfer the viewing coordinates (refer Figure 5.9). Conversion of object description from world to viewing coordinates is equivalent to a transformation which superimposes the viewing reference frame into the world frame using basic geometric translate and rotate operations. Following are the transformation sequence:

- Translate the view reference point to the origin of the world coordinate system.
- Apply rotations to align the x, and z axes with the world x , y and z axes respectively.

If the view reference point is specified at world position (x_0, y_0 and z_0) then this point is translated to the world origin with the matrix transformation.

$$T = \begin{bmatrix} 1 & 0 & 0 & -x_0 \\ 0 & 1 & 0 & -y \\ & & & 0 \\ 0 & 0 & 1 & -z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Space for learners:

The rotation sequence requires up to three coordinate axis rotations depending on the direction which is chosen for N. In fact, N is not aligned with world coordinate axis. The viewing and world systems are superimposed with transformation sequence

$R_z.R_y.R_x$. The world x_w axis is brought z_v into the $x_w z_w$ plane. Then, the world is rotated around y_w axis to align the z_w and z_v axes. The final rotation is about z_w axis to align the y_w and y_v axes. The composite transformation matrix is then applied to world coordinate descriptions to transfer them to viewing coordinates. Then given

vectors N and V are calculated as a set of unit vectors to define the viewing coordinate system which is obtained by the following equations.

$$\mathbf{n} = \frac{\mathbf{N}}{|\mathbf{N}|} = (n_x, n_y, n_z)$$

$$\mathbf{u} = \frac{\mathbf{V} \times \mathbf{n}}{|\mathbf{V}|} = (u_x, u_y, u_z)$$

$$\mathbf{v} = \mathbf{n} \times \mathbf{u} = (v_x, v_y, v_z)$$

The complete world to viewing coordinate transformation matrix is obtained as the matrix product in the following way:

$$M_{wc.vc} = R.T$$

This transformation is then applied to coordinate descriptions of objects in which u is transformed into the world x_w axis, v is transformed onto the y_w axis and n is transformed onto z_w axis. The complete world to viewing coordinate transformation
matrix is obtained as the matrix product.

$$M_{wc.vc} = R.T$$

This transformation is then applied to coordinate descriptions of objects scene to transfer them with reference to the viewing reference frame.

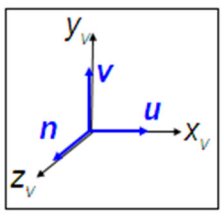


Fig. 3.9 *Viewing Coordinate Reference Frame*

Space for learners:

3.5 WINDOW TO VIEWPORT COORDINATE TRANSFORMATION

Space for learners:

Normalization transformation (N) maps world coordinates (or viewing coordinates) to normalized device coordinates and workstation transformation (W) maps normalized device coordinates to physical device coordinates. In general, the mapping of a part of a world coordinate scene to device coordinates is referred as viewing transformation (V), mathematically expressed as:

$$V = W.N$$

Sometimes, the two-dimensional viewing transformation is simply called window-to-viewport transformation.

By defining a closed boundary or window, the enclosed portion of a world coordinate scene is clipped against the window boundary and the data of the clipped portion is extracted for mapping to a separately defined region known as viewport. While window selects a part of the scene, viewport displays the selected part at desired location on the display area. When the window is changed, we see a different part of the scene at the same portion (viewport) on the display. If we change the viewport only, we see the same part of the scene drawn at, different scale or at, different place on the display. By successively increasing or

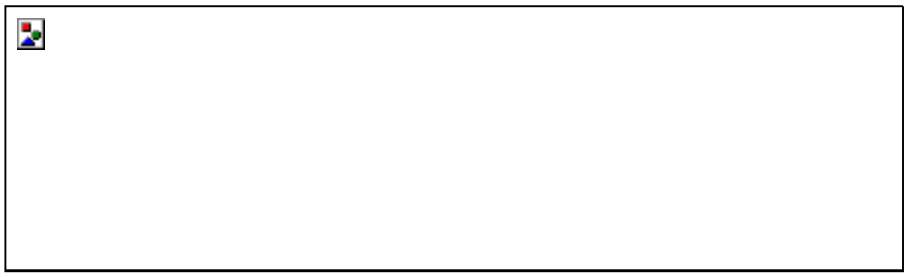


Fig. 3.10 *Window and Viewport*

decreasing the size of the window around a part of the scene the viewport remaining fixed, we can get the effect of zoom out or zoom in respectively on the displayed part.

Window or viewport can be general polygon shaped or circular. For simplicity, we will consider a rectangular window and a rectangular viewport, with edges of the rectangles being parallel to the coordinate axes.

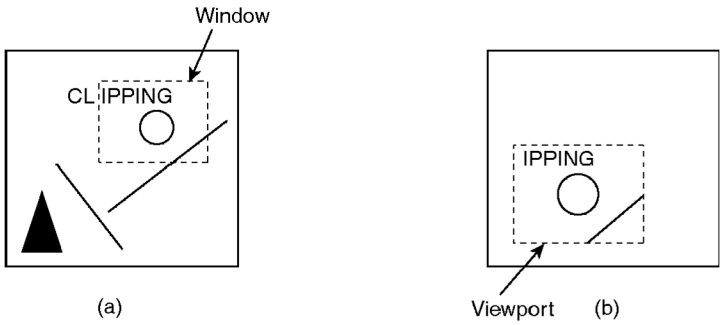


Fig. 3.11 *Out of Several Objects drawn in the Object Space, only those*

Clipped by the Window are Displayed in the Viewport in Image Space; note that the Viewport Objects are Larger than the Window Objects, though the Object Shapes are not Distorted because the Viewport is a Uniformly Scaled Version of the Window.

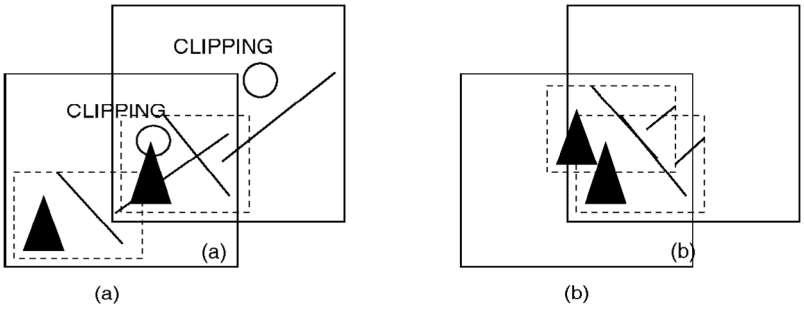


Fig. 3.12 When the Window is changed we see different Objects of the Scene Displayed in the Same Viewport

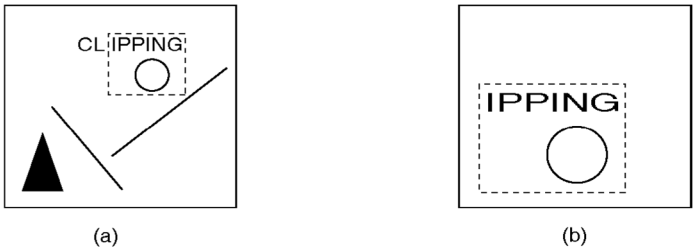


Fig. 3.13 When only the Viewport is Moved and the Window remains the Same, we see Same Objects Displayed through the Viewport at a different Position

Fig. 3.14 Zooming In (refer Figure 3.11)

Space for learners:

Let us consider a point (xw, yw) within the window enclosure as shown in Figure. 3.10. The window is mapped to the viewport such that (xw, yw) transform to (xv, yv) in the device space.

(a) (b)

Fig. 3.15 The Viewport Remaining fixed, the Window size is Gradually Reduced, Maintaining the Scale factors Same (refer Figures 3.11 and 3.14); The Zooming effect is Obtained through the Viewports.

To maintain the same relative placement of the point in the viewport as in the window,

$$\frac{xv - xv_{min}}{xv_{max} - xv_{min}} = \frac{xw - xw_{min}}{xw_{max} - xw_{min}}$$

And also

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

This implies,

$$xv = xv_{min} + (xw - xw_{min}) \frac{(xv_{max} - xv_{min})}{(xw_{max} - xw_{min})}$$

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

If $\frac{xv_{max} - xv_{min}}{xw_{max} - xw_{min}}$ be termed as s_x (the x scale factor for scaling the window to the size of the viewport) and $\frac{yv_{max} - yv_{min}}{yw_{max} - yw_{min}}$ be termed as s y (the y scale factor for window-to-viewport scaling), then:

$xv = xv_{min} + (xw - xw_{min})S_x$ and $yv = yv_{min} + (yw - yw_{min})S_y$

Space for learners:

In terms of two-step geometric transformation, the above relation can be interpreted as

Step 1: Scaling the window area to the size of the viewport with scale factors s_x and s_y with respect to a fixed point (xw_{min}, yw_{min}).

$$=> [T_1]= \begin{bmatrix} s_x & 0 & xw_{min}(1- s_x) \\ 0 & s_y & yw_{min}(-s_y) \\ 0 & 0 & 1 \end{bmatrix}$$

Step 2: Translating the scaled window to the position of the viewport so that

$\Delta x = xvmin - xwmin$ and

$\Delta y = yvmin - ywmin$

$$=> [T2]= \begin{bmatrix} 1 & 0 & \Delta x \\ 0 & 1 & \Delta y \\ 0 & 0 & 1 \end{bmatrix}$$

Concatenating [T1] and [T2] we get:

$$[T]=[T1][T2]= \begin{bmatrix} s_x & 0 & \Delta x + xw_{min}(1-s_x) \\ 0 & s_y & \Delta y + yw_{min}(1- s_y) \\ 0 & 0 & 1 \end{bmatrix}$$

Replacing the values of Δx and Δy in the above transformation matrix, we finally get:

$$[T] = \begin{bmatrix} S_x & 0 & -s_x xw_{min} + xv_{min} \\ 0 & s_y & -s_y yw_{min} + yv_{min} \\ 0 & 0 & 1 \end{bmatrix}$$

The aspect ratio of a rectangular window or viewport is defined by:

$$a = {}^{x}\text{max} - {}^{x}\text{min} \; {}^{y}\text{max} - {}^{y}\text{min}$$

Space for learners:

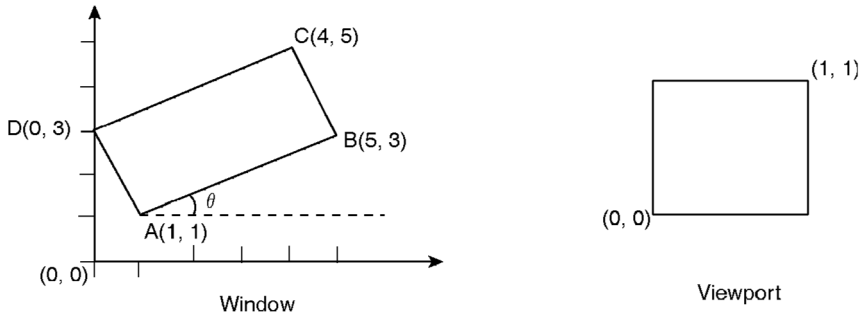
Space for learners:

$$\text{if } s_x = s_y \text{ then } \frac{xv_{max} - xv_{min}}{xw_{max} - xw_{min}} = \frac{yv_{max} - yv_{min}}{yw_{max} - yw_{min}}$$

$$\Rightarrow \frac{xv_{max} - xv_{min}}{yv_{max} - yv_{min}} = \frac{xw_{max} - xw_{min}}{yw_{max} - yw_{min}} \Rightarrow a_v = a_w$$

So, it can be said that if the aspect ratio a_v of the viewport equals the aspect ratio aw of the window, then $s_x = s_y$ and no distortion of displayed scene occurs other than uniform magnification or compression. If v w a ≠ a then the displayed scene in the viewport gets somewhat distorted with respect to the scene captured by the window.

Example 3.1: Find the normalization transformation N which uses the rectangle A (1, 1), B (5, 3), C (4, 5) and D (0, 3) as a window and the normalized device screen as the viewport



Window and Viewport

Here, we see that the window edges are not parallel to the coordinate axes. So, we will first rotate the window about *A* so that it is aligned with the axes.

$$= 3 - 1 = 1 \quad 5 - 1 \quad 2$$

Now, $\tan\theta$

$$\Rightarrow \sin\theta = \quad , \cos\theta = \frac{2}{5} \quad 1$$

Here, we are rotating the rectangle in clockwise direction. So, θ is (–)ve, i.e., – θ . The rotation matrix about A(1, 1) is:

Space for learners:

$$[T_{R,\theta}]_A = \begin{bmatrix} 2/\sqrt{5} & 1/\sqrt{5} & (1-3)/\sqrt{5} \\ -1/\sqrt{5} & 2/\sqrt{5} & (1-1)/\sqrt{5} \\ 0 & 0 & 1 \end{bmatrix}$$

The x extent of the rotatedwindow is the length of $\sqrt{(4^2 \square 2^2} = \sqrt{}$

—

AB which is

Space for learners:

Similarly, the y extent is length of AD which is:

For scaling the rotated window to the normalized viewport, we calculate s_x

and s_y: as,

$$s_x = \frac{\text{Viewport } x \text{ extent}}{\text{Window } x \text{ extent}} = \frac{1}{2\sqrt{5}}$$

$$s_y = \frac{\text{Viewport } y \text{ extent}}{\text{Window } y \text{ extent}} = \frac{1}{\sqrt{5}}$$

As in Equation (5.1), the general form of transformation matrix representing mapping of a window to a viewport is:

$$[T] = \begin{bmatrix} s_x & 0 & -s_x xw_{min} + xv_{min} \\ 0 & s_y & -s_y yw_{min} + yv_{min} \\ 0 & 0 & 1 \end{bmatrix}$$

In this problem, [T] may be termed as N as this is a case of normalization transformation with

$xw_{min}=1 \quad xv_{min}=0$

$yw_{min}=1 \quad yv_{min}=0$

$s_x=1/2\sqrt{5} \quad s_y=1/\sqrt{5}$

By substituting the above values in [T], i.e., N,

$$\begin{bmatrix} 1/2\sqrt{5} & 0 & (-1/2)1/\sqrt{5} + 0 \end{bmatrix}$$

$$N = \begin{matrix} 0 & 1/\sqrt{5} & (-1/\sqrt{5})\ 1+0 \\ 0 & 0 & 1 \end{matrix}$$

Now, we compose the rotation and transformation N to find the required viewing transformation N

$$N_R = N[T_R, \theta]_A = \begin{bmatrix} 1/2\sqrt{5} & 0 & -1/2\sqrt{5} \\ 0 & 1/\sqrt{5} & -1/\sqrt{5} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2/\sqrt{5} & 1/5 & 1-3/\sqrt{5} \\ -1/\sqrt{5} & 2/\sqrt{5} & 1-1/\sqrt{5} \\ 0 & 0 & 1 \end{bmatrix}$$

3.6 TWO-DIMENSIONAL VIEWING FUNCTIONS-

Two-dimensional viewing functions include world coordinates to viewing coordinates in which Window to viewport process is used. Window is a region of the scene selected for viewing which is also called clipping window. Viewport is a region on display device for mapping to Window. Figure 5.16 shows the world coordinates and viewing coordinates.

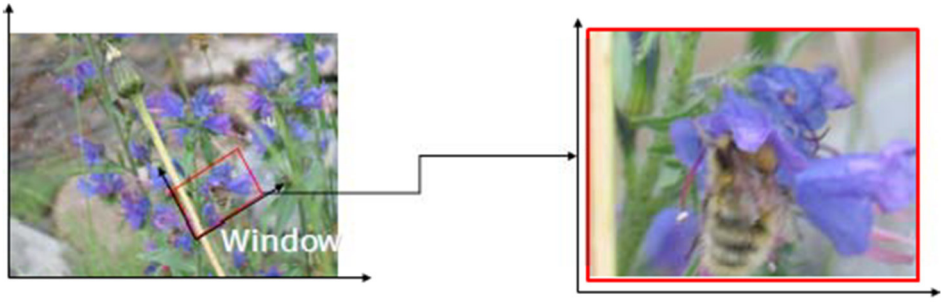


Fig. 3.16 World Coordinates and Viewing Coordinates

Space for learners:

Figure (3.16) displays the world coordinates and viewing coordinates area in which the clipping Window selects what we want to see in our virtual 2-D world. The viewport indicates where it is to be viewed on the output device or within the display Window.

3.7 EXAMPLES

Some OpenGL 2D Viewing Functions

OpenGL Projection Mode

glMatrixMode (GL_PROJECTION); //projection matrix

glLoadIdentity ();

GLU Clipping-Window Function

gluOrtho2D (xwmin, xwmax, ywmin, ywmax);

2D parallel projection.

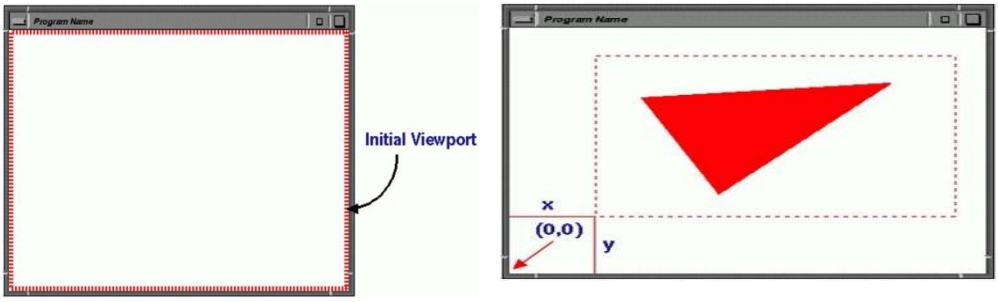
OpenGL Viewport Function

glViewport (xvmin, yvmin, vpWidth, vpHeight);

glGetIntegerv (GL_VIEWPORT, vpArray); To obtain the parameters for the currently active viewport: xvmin, yvmin, vpWidth, vpHeight

OpenGL Viewport Function

glViewport (GLint x, GLint y, GLsizei width, Glsizei height);



Space for learners:

Space for learners:

You can change it by glViewport(). Always rectangle; x, y are in window coordinates

OpenGL Viewport Function

The rectangle area has an aspect ratio: width / height

Windows

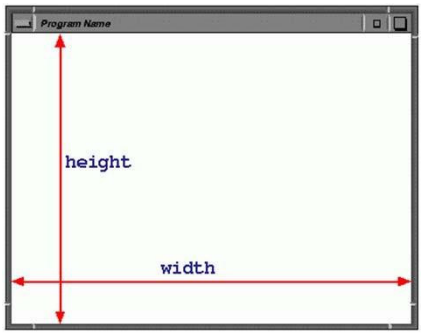
glutInitWindowSize (width, height);

2D clipping window

gluOrtho2D (left, right, bottom, top);

Viewport

glViewport (x, y, width, height)

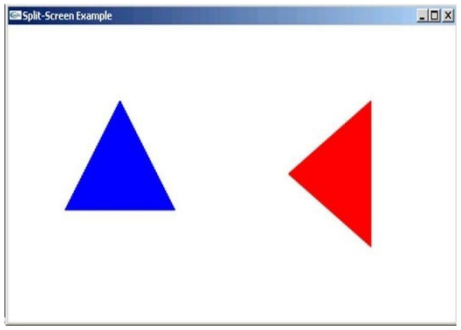


OpenGL 2D Viewing Program Example

Two views of a triangle in the xy plane shown in a split screen, with its centroid at the world-coordinate origin.

```
#include<GL/glut.h>
class wcPt2D
{
   public:
  GLfloat x, y;
};
```

Space for learners:



OpenGL 2D Viewing Program Example

```
void init (void)
{
    /*  Set color of display window to white. */
    glClearColor (1.0, 1.0, 1.0, 0.0);

    /*  Set parameters for world-coordinate
    clipping window. */ glMatrixMode
    (GL_PROJECTION);
    gluOrtho2D (-100.0, 100.0, -100.0, 100.0);

    /*  Set mode for construction geometric
    transformation matrix. */ glMatrixMode
    (GL_MODELVIEW);
}
```

OpenGL 2D Viewing Program Example

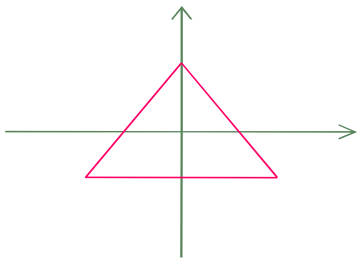
```
void triangle (wcPt2D *verts)
{
    GLint k;
    glBegin (GL_TRIANGLES);
    for (k =0; k < 3; k++)
    glVertex2f (verts [k].x, verts [k].y);
```

glEnd ();

}

OpenGL 2D Viewing Progra Example-

```
void displayFcn (void)
{
    /* Define initial position for triangle. */
    wcPt2D verts [3] = {{-50.0, -25.0}, {50.0, -25.0},
    {0.0, 50.0}};
    glClear (GL_COLOR_BUFFER_BIT);     // Clear
    display window. glColor3f (0.0, 0.0, 1.0); // Set fill
    color to blue.
    glViewport (0, 0, 300, 300);                    // Set
    left viewport.
    triangle (verts);                          // Display red
    rotated triangle.
    /* Rotate triangle and display in right half of display
    window. */ glColor3f (1.0, 0.0, 0.0);          // Set fill
    color to red. glViewport (300, 0, 300, 300);       //
    Set right viewport.
    glRotatef (90.0, 0.0, 0.0, 1.0);          // Rotate about
    z axis. triangle (verts);                  // Display red
    rotated triangle.
     glFlush ();
```



Space for learners:

OpenGL 2D Viewing Program Example

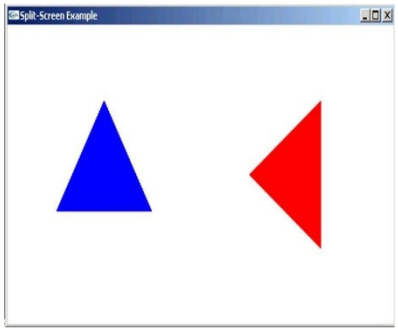
```
void main (int argc, char **argv)
        {

            glutInit (&argc, argv);

            glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
            glutInitWindowPosition (50, 50);
            glutInitWindowSize (600, 300); glutCreateWindow
            ("Split-Screen Example");


            init();

            glutDisplayFunc (displayFcn); glutMainLoop ();
        }
```



CHECK YOUR PROGRESS

1. Visualization of an object is achieved by a sequence of operations called the _____ _____.

2. A view position in a scene is represented by camera which is associated with a _____ _____.

3. Two-dimensional viewing functions include world coordinates to viewing coordinates in which _____ _____ _____is used.

4. What is Normalization transformation?

Space for learners:

3.8 LET US SUM UP

- The world coordinates system performs lighting in the space of window, if illumination requires. The virtual device coordinates isolates the applications to view clipping if application requires it. Device coordinates are the actual coordinates of the device.
- While window selects a part of the scene, viewport displays the selected part at desired location on the display area. When the window is changed, we see a different part of the scene at the same portion (viewport) on the display.
- Window is a region of the scene selected for viewing which is also called clipping window. Viewport is a region on display device for mapping to Window. The viewport indicates where it is to be viewed on the output device or within the display Window.

3.9 ANSWERS TO CHECK YOUR PROGRESS

1. viewing pipeline.
2. coordinate system
3. Window to viewport process
4. Normalization transformation (N) maps world coordinates (or viewing coordinates) to normalized device coordinates and workstation transformation (W) maps normalized device coordinates to physical device coordinates

3.10 POSSIBLE QUESTIONS

Short-Answer Questions

1. What is the usage of VRC?
2. What is the difference between point clipping and line clipping?
3. List the coordinates of a rectangle.

Space for learners:

Long-Answer Questions

1. Describe the steps for transforming the world coordinate system.
2. Explain the pseudocode for Sutherland-Hedgman.

3.11 REFERENCES AND SUGGESTED READINGS

- Computer Graphics by Donald Hearn and M Pauline Baker.
- Basic Computer Graphics by Darshan Institute of Engineering and Technology.
- Introduction Computer graphics by Dr John T. Bell.

Space for learners:

UNIT 4: TWO-DIMENSIONAL CLIPPING OPERATIONS

Space for learners:

Unit Structure:

4.1 INTRODUCTION

Any techniques that identify pieces of a picture that are either inside or outside a specific region of space are known as clipping algorithms, or simply clipping. A clip window is the area against which an object will be clipped.Internal clipping eliminates sections of a picture that are outside of a viewing window, whereas external clipping removes elements of a picture that are within a viewing window. Two basic

clipping techniques are line clipping and polygon clipping.Line clipping algorithms take two endpoints of a line segment as input and produce one or more line segments, whereas polygon clipping algorithms take the vertices of a polygon as input and output one or more polygons.Clipping can be done in a variety of methods, some of which are listed below:

- Point Clipping
- Line Clipping
- Polygon Clipping'
- Text Clipping
- Curve Clipping

There are also numerous algorithms for performing various types of clipping operations, some of which will be discussed in this unit.

Line Clipping Algorithms:

- Cohen Sutherland Line Clipping
- Midpoint subdivision Line Clipping
- Liang–Barsky Line Clipping and
- Cyrus–Beck Line Clipping

Polygon Clipping Algorithm:

- Sutherland–Hodgeman Algorithm

Space for learners:

4.2 UNIT OBJECTIVES

After going through this unit, you should be able to:

- Explain the concept of clipping;
- Explain how to point clipping is performed;
- Examine how to line clipping is performed;
- It is really important to understand and use the algorithms that make up the line clipping concept;
- Explain how polygon clipping is performed;
- It is really important to understand and use the algorithms that make up the polygon clipping concept;
- Explain how to curve clipping is performed, and
- Explain how text clipping is performed.

STOP TO CONSIDER

WINDOW may be defined as the world coordinates are selected for display. VIEWPORT refers to the area to the area where the window is mapped. Thus, WINDOW specifies what is to be shown or displayed whereas the VIEWPORT specifies where it is to be shown or displayed. In terms of Clipping, it is necessary to mention these two coordinates namely WINDOW and VIEWPORT coordinates, as well as the transformation from WINDOW to VIEWPORT coordinates.

4.3 TWO DIMENSIONAL CLIPPING

As we know, viewing requires us to define a viewing window, often referred to as a clipping window. This window may assume that we have a rectangular window and have a clipping algorithm to clip line segments against it. This window may be rectangular, circular, or in any arbitrary shape. But for computer processing, rectangular and circular windows have been studied extensively. Most of the developed techniques have been based on a rectangular window that is defined by a set of four lines: x=Left, x=Right, y=Top, y=Bottom. The visible portion of a window is defined by the boundary region between these four lines segments given in **Figure1.**

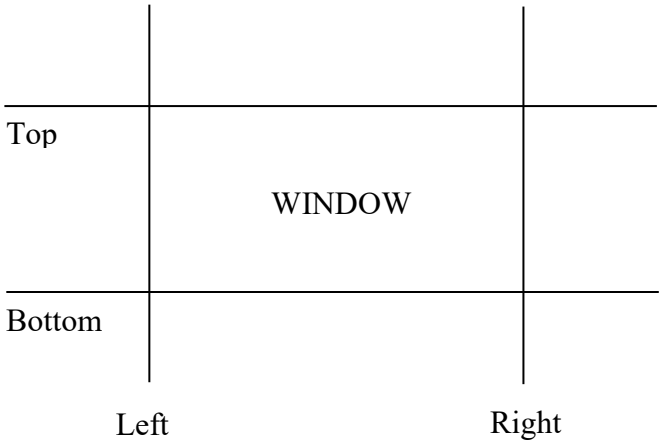


Figure 1: A rectangular Window

Space for learners:

The portion of the picture that lies inside the window should be displayed, and the other areas of the picture should not be displayed. A clipping algorithm determines whether points, lines or portions of the lines are within the clipping window. Points and lines within the window are retained for display, and all other points and lines are discarded.

Space for learners:

4.4 POINT CLIPPING

Point clipping is a technique of removing those points that lie outside the window region. The points lying inside the window or at its boundary are not clipped. Consider a rectangular point clipping window; the lowest and highest coordinate values, i.e. *(*XW*max,* YW*max) and (XWmin,* YW*min),* are sufficient to establish window size, and any point P = (X, Y) that can be displayed must satisfy the inequalities below.Otherwise, the point will not be visible. Thus, the point will be clipped or not can be decided based on the following inequalities.

$$\left.\begin{aligned} XWmin \le X \le XWmax \text{ and} \\ YWmin \le Y \le YWmax \end{aligned}\right\} \quad \text{----------} \quad (1)$$

Where XWmin and XWmax are the edges of the clipping window parallel to the y-axis; YWmin and YWmax are the edges of the clipping window parallel to the x-axis.

It is to be noted that P (x, y) must satisfy all inequalities of Equation (1) to display P=(x, y) on an output device. If the point does not satisfy any of the inequalities, it will be clipped (not saved for display). Though point clipping is less frequently used as compared to other clipping techniques, in some situations, such as the scenes involving particle movements such as explosion, dust, seafoam etc., it is quite useful.

4.5 LINE CLIPPING

A line clipping is a technique of removing lines or portions of lines outside the clipping window region. This technique tests each line

segment in a scene to determine whether it completely lies inside the clipping window, or outside the clipping window, or intersects one or more clipping boundaries. The line segments are processed by applying the inside-outside tests. The line segments with both endpoints inside the clipping window are saved for display. The line segments with both endpoints outside the clipping window are clipped. For all those lines which intersect one or more clipping boundaries, some more calculations need to be performed.

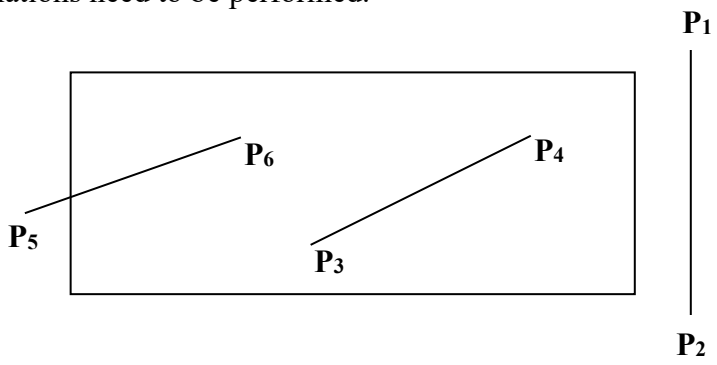


Figure 3: Line Clipping

In **Figure 3**, since the line segment P_3P_4 lies completely inside the clipping boundaries, it is saved for display; whereas the line segment P_1P_2 lies completely outside the clipping boundaries, hence it is discarded. However, the line segmentP_5P_6 intersects the left clipping boundary, and some more calculations are to be performed. To reduce such intersection calculations, we can apply a more efficient clipping algorithm.For any line segment whose one or both endpoints lie outside the clipping window, we can use the following parametric representation:

$$\mathbf{X = X_1 + u\,(X_2 - X_1)\ and}$$

$$\mathbf{Y = Y_1 + u\,(Y_2 - Y_1),\ 0 \leq u \leq 1}$$

Where (X_1, Y_1) and (X_2, Y_2) is the endpoints of a line segment and u is the parameter ranging from 0 to 1.

From this representation, we could find the value of parameter u, which further helps us in determining the intersections with the clipping boundaries. If the value of u does not lie between 0 and 1, the line does not enter the interior of the window at that boundary and thus can be

Space for learners:

discarded. Otherwise, the line segment intersects the clipping area. This procedure is iteratively applied to each clipping boundary edge to determine which part of the line segment is to be displayed. However, such tests require a good deal of computation and are slower as well. A lot of line clipping algorithms is available in the area of computer graphics, but we will focus on the following Line clipping methods, which are named after their creators:

1) Cohen–Sutherland algorithm
2) Midpoint subdivision method
3) Liang–Barsky algorithm
4) Cyrus–Beck algorithm

Space for learners:

4.5.1. Cohen-Sutherland Line Clipping Algorithm

It is one of the oldest and popular algorithms used for clipping line segments. The basic idea behind this algorithm is to perform some initial tests of line segments to reduce the number of intersections to be calculated and to speed up the processing of line segments. The algorithm determines whether a line segment lies completely inside the clipping window, or completely outside the clipping window, a line falls in one of the following three defined clipping categories:

- **Visible:** In this, both the line endpoints lie inside the clipping window. For example, in **Figure 4**, lines PQ and ST are visible lines.
- **Not visible:** In this, both the line endpoints lie outside the clipping window. For example, in **Figure 4**, line WX is not visible.
- **Partially visible:** In this, the line endpoints neither lie inside nor outside the clipping window completely. That is, the line intersects one or more clipping boundaries. For example, in **Figure 4**, lines UV and YZ are partially visible.

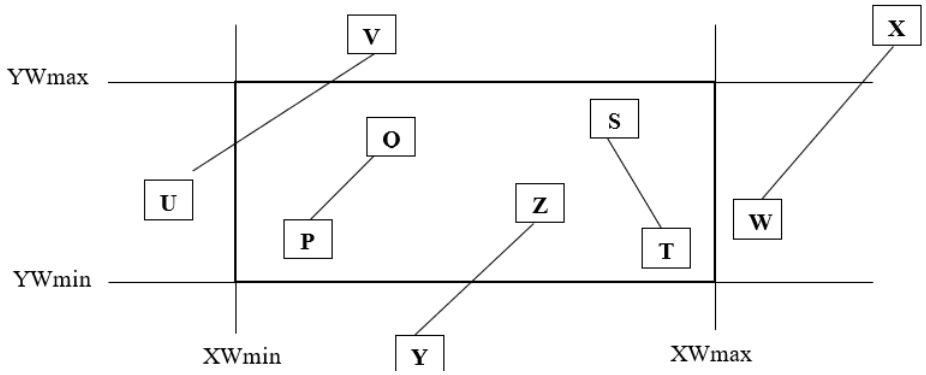


Figure 4: Visible, Not Visible and Partially Visible

Cohen-Sutherland algorithm requires a window in a rectangular shape, and thus, it is not a general procedure. The four parameters that define a rectangular window are (XL, XR, YB, and YT), where the parameters determine the left, right, bottom, and top boundaries, respectively.

Inside the window, any point must satisfy the following relationship:

$$\left.\begin{array}{l} X_L < X < X_R \text{ and} \\ Y_B < Y < Y_T \end{array}\right\} \quad \text{---------- (2)}$$

where the coordinates of the point are (x, y). The above concept is used to determine the visibility line segments also. Cohen and Sutherland simplified the method by assigning a four-bit code to each line's endpoint. These four-bit codes are referred to as "region codes" (shown below).

1001	1000	1010
0001	0000	0010
0101	0100	0110

Here, each bit in a code corresponds to one of the four regions outside the window boundaries, i.e. the first bit is on the left, the second on the right, the third on the bottom, and the fourth on the top. After obtaining area codes for all line segments, we can simply distinguish between visible and invisible lines.The line is completely visible if both endpoint codes are 0000.The line is completely invisible if the logical AND of

Space for learners:

the two endpoint codes is not 0000.The intersection of the remaining lines with the window boundaries will be tested.The interiors of the windows may or may not be intersected by these lines.As a result, these lines may be completely outside the window zone or a portion of them may fall inside the window.

For a line joining (x_1,y_1) and (x_2, y_2),

$$m = (Y_2-Y_1) / (X_2-X_1),$$

and if the line passes through (x, y) then,

$$y= m(x – x_1) + y_1.$$

Let (x, y) be the point of intersection of the line with a window edge. Then either $x = X_L$ or X_R to calculate y or $y = Y_B$ or Y_T to calculate x.

For left edge: $y = m (X_L – X_1) + Y_1;$ $m \neq \infty$

For right edge: $y = m (X_R – x_1) + Y_1;$ $m \neq \infty$

For bottom edge: $x = 1/m (Y_B – Y_1) + X_1;$ $m \neq 0$

For top edge: $x = 1/m (Y_T – Y_1) + X_1;$ $m \neq 0$

For horizontal lines: $y_2 – y_1= 0$ => if $x_1<x_L$, then $x = x_L$ and if $x_1>x_R$, then $x = x_R$.

For vertical lines: $x_2 – x_1= 0$ => if $y_1<y_B$, then $y = x_B$ and if $y_1>y_R$, then $y = y_T$.

Finally, we are to check whether $y_B\leq y \leq y_T$ and $x_L\leq x \leq x_R$

Example 1: Consider the line P_1 (-1, 1) to P_2 (9, 3). Determine its visibility against the clipping window (rectangular) defined by vertices (0, 0), (8, 0), (8, 4) and (0, 4) using Cohen-Sutherland algorithm (see **Figure 5**).

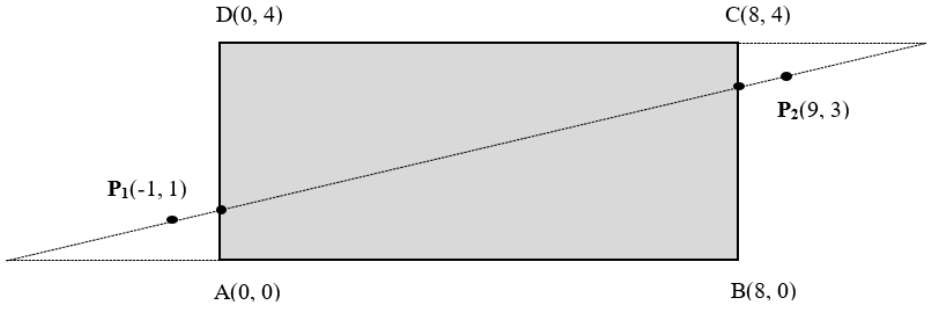


Figure 5:Cohen–Sutherland Clipping(*Example 1*)

Space for learners:

***Solution*:**The slope of the line is m = $(y_2–y_1) / (x_2–x_1)$ = (3–1) / (9+1) = 2/10 = 1/5

For left edge: $x = x_L = 0$, $y = m (x_L – x_1) + y_1 = 1/5(0+1) +1 = 6/5$

For right edge: $x = x_L = 0$, $y = m (x_R – x_1) + y_1 = 1/5(8+1) +1 = 14/5$

For bottom edge: $y = y_B = 0$, $x = 1/m (y_B – y_1) + x_1 = 5(0–1) –1 = –6$

For top edge: $y = y_B = 0$, $x = 1/m (y_T – y_1) + x_1 = 5(4–1) –1 = 14$

Now, we are to check whether $y_B \leq y \leq y_T$ and $x_L \leq x \leq x_R$ or not. The intersections with the left and right edges are the only ones that satisfy the aforesaid requirements. Hence, we get the intersecting points as (0, 6/5) and (8, 14/5) respectively.

Space for learners:

4.5.2. Midpoint Subdivision Method

The midpoint subdivision method is a simple recursive method based on binary search and divide-and-conquer strategy for clipping of lines. The algorithm performs the test using endpoint codes and finds if the line is visible or invisible. If the test fails, the algorithm divides (bisects) the line segment in two halves using the midpoint formula, and the two halves are tested for visibility separately by a similar process of test and divide to find the intersection endpoints defining the visible portion inside the window. It is a recursive algorithm wherein the base criterion is reached when the line is within the boundary of the window.

The midpoint (X_{mid}, Y_{mid}) of the line segment with endpoints $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ is

$$X_{mid} = (x_1 + x_2) / 2 \text{ and}$$
$$Y_{mid} = (y_1 + y_2) / 2 \quad \text{---------- (3)}$$

The steps of the algorithm may be summarized as follows:

***Step 1*:** [Test visibility]

If the endpoints $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ are visible, then the line lies inside the clipping window. Return P_1P_2 as a visible segment. Else go to Step 2.

***Step 2*:** [Test invisibility]

If the line segment with endpoints $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ is trivially invisible, then no output is generated in the clipping window. Else go to Step 3.

***Step 3*:** [Repeat Step 4 until line reduces to point satisfying the visibility test]

***Step 4*:** [Perform bisection and obtain midpoint]

Obtain the midpoint Pm (x_{mid}, y_{mid}) by dividing the line P_1P_2 using equation (3). Apply visibility tests to two segments P_1P_m and P_mP_2 one by one. Continue with P_1P_m and obtain midpoint by overestimation of the farthest visible point if P_mP_2 is trivially rejected. Otherwise, the midpoint is an underestimation of the farthest visible point and continues with P_2P_m. If the segment becomes

too small, and the midpoint corresponds to a point or specified endpoint, evaluate it for visibility of point and the process is complete. **Figure 6** illustrates the concept of midpoint selection and evaluation of the visible portion of the line segment P_1P_2.

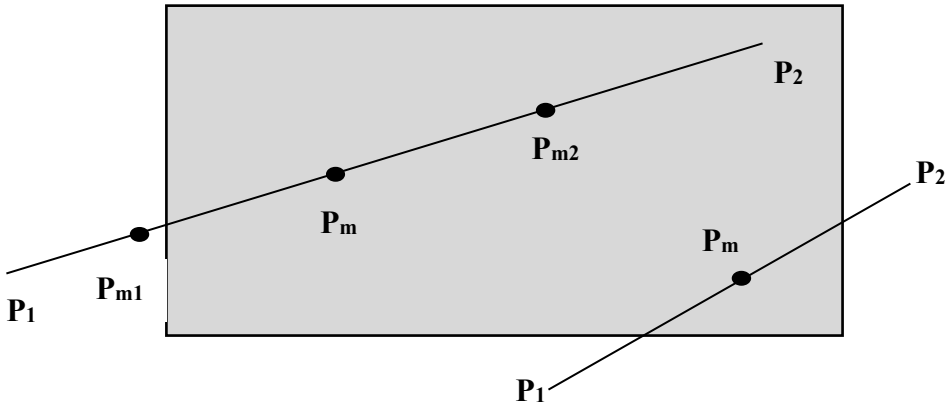


Figure 6: Examples of midpoint subdivision

4.5.3 Concept of Parametric Clipping

A line can be expressed parametrically as

$\mathbf{P}(t) = \mathbf{P_1} + (\mathbf{P_2} - \mathbf{P_1})\, t;\ 0 \le t \le 1$ ---------- (4)

P(t) is the correct position vector, and t is the parameter. Restriction of value *t*between 0 and 1 makes it a finite line segment. We have a pair of parametric equations in a two-dimensional Cartesian coordinate system:

---------- (5)

Space for learners:

$x(t) = x_1 + (x_2 - x_1)\, t;\ 0 \le t \le 1$

$y(t) = y_1 + (y_2 - y_1)\, t;\ 0 \le t \le 1$ ---------- (6)

The following formula can be used to calculate the value of the parameter t:

$$t = (\mathbf{P}(t) - \mathbf{P_1}) / (\mathbf{P_2} - \mathbf{P_1}) \quad \text{---------- (7)}$$

For a regular rectangular window, one of the coordinates for each intersection with the window edges is known. Only the other is to be calculated. Let us denote the left, right, top and bottom window edges by lines: $x = x_L$, $x = x_R$, $y = y_T$, and $y = y_B$

For intersections with the window boundaries, the following t-values can be obtained.

Left edge: $t = (x_L - x_1) / (x_2 - x_1);\ 0 \le t \le 1$

Right edge: $t = (x_R - x_1) / (x_2 - x_1);\ 0 \le t \le 1$

Top edge: $t = (y_T - y_1) / (y_2 - y_1);\ 0 \le t \le 1$

Bottom edge: $t = (y_B - y_1) / (y_2 - y_1);\ 0 \le t \le 1$

In any case, it t is outside the range of $0 \le t \le 1$, and then those solutions are discarded since they represent points beyond the end of the line segment.

Example 2:

The window $(x_L, x_R, y_T, y_B) \equiv (-1, 1, -1, 1)$. For the line: $\mathbf{P_1}$ (–3/2, –3/4) to $\mathbf{P_2}$ (3/2, 1/2). Calculate the intersecting points with the window edges.

Solution: The first two t-values are within a reasonable range.As a result, the line only intersects the left and right boundaries.

Table 1: Computation of parametric intersections (Example 2)

Edge	$\mathbf{P}(t) - \mathbf{P_1}$	$\mathbf{P_2} - \mathbf{P_1}$	t
Left	$x_L - x_1 = 1/2$	$x_2 - x_1 = 3$	1/6
Right	$x_R - x_1 = 5/2$	$x_2 - x_1 = 3$	5/6
Bottom	$y_B - y_1 = -1/4$	$y_2 - y_1 = 5/4$	–1/5
Top	$y_T - y_1 = 7/4$	$y_2 - y_1 = 5/4$	7/5

4.5.4. Liang-Barsky Clipping Algorithm

The Liang-Barsky clipping algorithm is another clipping algorithm that can be applied to regular rectangular clipping windows only. This algorithm deals with a line in its parametric form. The parametric

Space for learners:

equation of a line between points (x1, y1) and (x2, y2), in terms of parameter t, can be expressed as

$$\mathbf{x = x + \Delta x.\ t \text{ and } y = y + \Delta y.\ t} \quad \text{---------- (8)}$$

where $\Delta x = x2 - x1$ and $\Delta y = y2 - y1$. The parametric line is defined in the range of $0 \leq t \leq 1$.

For the extended line, this range is $-\infty < t < +\infty$. The extended parametric line cuts the extended clipping window boundaries at parameter values t_L, t_R, t_B, and t_T, where the suffixes left, right, bottom and top respectively.

$t_{min} = \max(0, t_L, t_B)$ and $t_{max} = (1, t_R, t_T)$ ---------- (9)

Now, for any point (x, y) inside the clipping window [((x, y) is a point on the line between points (x1, y1) and (x2, y2)], the following inequalities hold.

$$\left.\begin{aligned} x_{min} &\leq x1 + \Delta x.\ t \\ x_{max} &\geq x1 + \Delta x.\ t \\ y_{min} &\leq y1 + \Delta y.\ t \\ y_{max} &\geq y1 + \Delta y.\ t \end{aligned}\right\} \quad \text{---------- (10)}$$

where t is the value of the parameter at point (x, y). The above four inequalities can be expressed commonly ~~as $p_i^L \leq q_i$; for I =~~ (11) , 4 where

$$p_1 = -\Delta x;\ q_1 = x_1 - x_{min}$$
$$p_2 = \Delta x;\ q_2 = x_{max} - x_1$$
$$p_3 = -\Delta y;\ q_3 = y_1 - y_{min}$$
$$p_4 = \Delta y;\ q_4 = y_{max} - y_1$$

In the above equations, the indices i = 1, 2, 3 and 4 stands for four window boundaries, i.e., left, right, bottom and top boundaries respectively. Now, we can observe the following facts:

1. $p_i = 0 \Rightarrow$ the line is parallel to the ith boundary.
   $-q_i < 0 \Rightarrow$ the line is completely on the invisible side
   $-q_i \geq 0 \Rightarrow$ the line is completely on the visible side
2. $p_i < 0 \Rightarrow$ the line comes from outside to inside the window intersecting the ith boundary.

Space for learners:

3. $p_i > 0 \Rightarrow$ the line goes from inside to outside the window intersecting the ith boundary.
4. $p_i \neq 0 \Rightarrow$ the value of the parameter t at the intersection with the ith boundary is found to be $t = q_i/p_i$

Example 3: Window is: (–1, 1, –1, 1). The line is P_1(–5/2, –1) to P_2(3/2, 2). Find intersection points using the Liang-Barsky clipping algorithm.

Solution:

The general equation of a parametric line is $\mathbf{P}(t) = \mathbf{P_1} + (\mathbf{P_2} - \mathbf{P_1})\, t$; $0 \leq t \leq 1$

From the equation we get, $t = (\mathbf{P}(t) - \mathbf{P_1}) / (\mathbf{P_2} - \mathbf{P_1})$

The computation of parametric intersections is given in **Table 2**.

Table 2: Computation of parametric intersections (*Example 3*)

Edge	$\mathbf{P}(t) - \mathbf{P_1}$	$\mathbf{P_2} - \mathbf{P_1}$	t
Left	$x_L - x_1 = -3/2$	$x_2 - x_1 = -8/2$	3/8
Right	$x_R - x_1 = 7/2$	$x_2 - x_1 = -8/2$	–7/8
Bottom	$y_B - y_1 = 0$	$y_2 - y_1 = 3$	0
Top	$y_T - y_1 = 2$	$y_2 - y_1 = 3$	2/3

We observed from table 2 that all the t-values are outside the range of $0 \leq t \leq 1$. If we plot the values of t in the clipping window, then we can say that the line is partially visible.

Example 4: Window is: (–1, 1, –1, 1). The line is P_1(–1/2, 1/2) to P_2(1/2, –1/2). Find intersection points using the Liang-Barksy clipping algorithm.

Solution:

The general equation of a parametric line is $\mathbf{P}(t) = \mathbf{P_1} + (\mathbf{P_2} - \mathbf{P_1})\, t$; $0 \leq t \leq 1$

From the equation we get,$t = (\mathbf{P}(t) - \mathbf{P_1}) / (\mathbf{P_2} - \mathbf{P_1})$

The computation of parametric intersections is given in Table 3.

Space for learners:

Space for learners:

Table 3: Computation of parametric intersections (*Example 4*)

Edge	$\mathbf{P}(t) - \mathbf{P}_1$	$\mathbf{P}_2 - \mathbf{P}_1$	t
Left	$x_L - x_1 = -1/2$	$x_2 - x_1 = 1$	$-1/2$
Right	$x_R - x_1 = 3/2$	$x_2 - x_1 = 1$	$3/2$
Bottom	$y_B - y_1 = -3/2$	$y_2 - y_1 = -1$	$3/2$
Top	$y_T - y_1 = 1/2$	$y_2 - y_1 = -1$	$-1/2$

4.5.5. Cyrus-Beck Clipping Algorithm

The general equation of a parametric line is $\mathbf{P}(t) = \mathbf{P}_1 + (\mathbf{P}_2 - \mathbf{P}_1)\, t$; $0 \leq t \leq 1$

For every value of t, if Q is a boundary point of the convex area R and n is an inward normal vector for one of the boundaries, i.e., at any particular point on **P1P2**.

$\mathbf{n}.|\mathbf{P}(t) - \mathbf{Q}| < 0$ implies that the vector $|\mathbf{P}(t) - \mathbf{Q}|$ is pointed away from the interior of *R*.

$\mathbf{n}.|\mathbf{P}(t) - \mathbf{Q}| = 0$ implies thatthe vector $|\mathbf{P}(t) - \mathbf{Q}|$ is parallel to the plane containing **Q** and perpendicular to the normal.

$\mathbf{n}.|\mathbf{P}(t) - \mathbf{Q}| > 0$implies thatthe vector $|\mathbf{P}(t) - \mathbf{Q}|$ is pointed towards the interior of *R*.

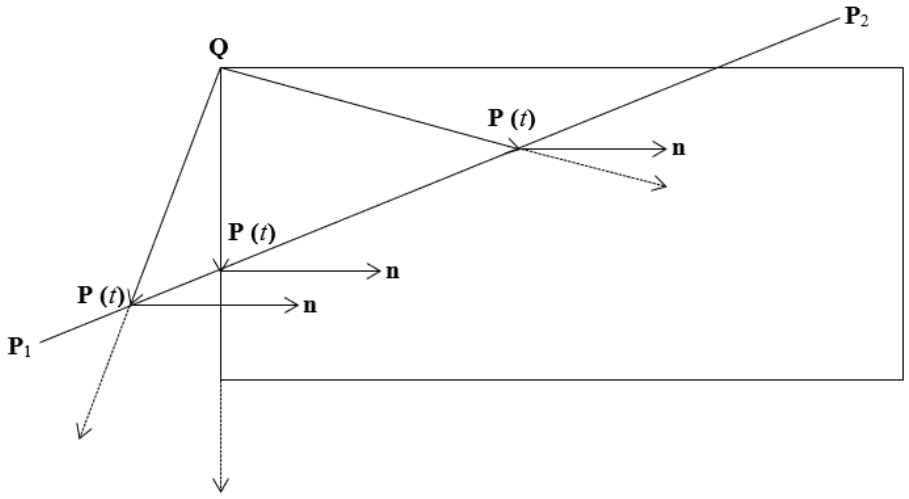


Figure 7: Cyrus-Beck clipping principle

If we consider a closed convex polygon, an infinite line intersects the window at exactly two points that do not lie on the same boundary. Thus, $\mathbf{n}.|\mathbf{P}(t)-\mathbf{Q}| = 0$ has only one solution. If the point Q lies in the boundary edge for which **n** is normal, then the point $\mathbf{P}(t)$ on the line satisfies the above condition at the intersection of the line and the boundary edge. **Figure 7** depicts the scenario as a whole.

Space for learners:

Example 5

Check the line is completely visible or not. The line to clip: from P1 (1, 1) to P2 (7, 2). Window: rectangular defined by vertices (0, 0), (8, 0), (8, 4), (0, 4) (see **Figure 8**)

Solution: The parametric equation is: $\mathbf{P}(t) = \mathbf{P}_1 + (\mathbf{P}_2 - \mathbf{P}_1)\, t = (6t + 1)\,\mathbf{i} + (t + 1)\,\mathbf{j}$; $0 \le t \le 1$

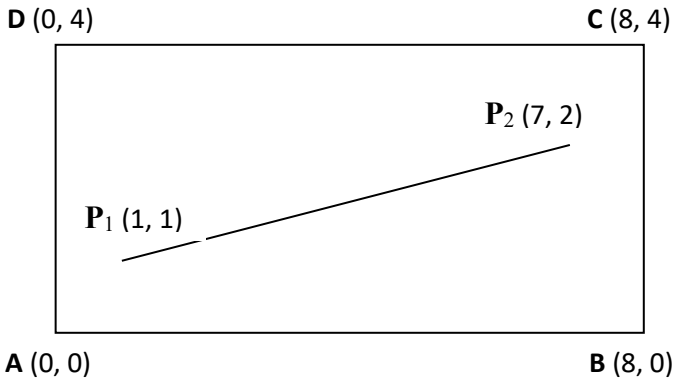


Figure 8: Cyrus-Beck clipping (*Example 5*)

Table 4: **Computation of parametric intersections(*Example 5*)**

Edge	**n**	**Q**	$\mathbf{P}(t)-\mathbf{Q}$	$\mathbf{n}.\lvert\mathbf{P}(t)-\mathbf{Q}\rvert$	t
Left	**i**	(0, 0)	$(6t+1)\,\mathbf{i} + (t+1)\,\mathbf{j}$	$6t+1$	–1/6
Right	**–i**	(8, 4)	$(6t-7)\,\mathbf{i} + (t-3)\,\mathbf{j}$	$7-6t$	7/6
Bottom	**j**	(0, 0)	$6t+1)\,\mathbf{i} + (t+1)\,\mathbf{j}$	$t+1$	–1
Top	**–j**	(8, 4)	$(6t-7)\,\mathbf{i} + (t-3)\,\mathbf{j}$	$3-t$	3

From Table 4, we observe that for intersections with all the window boundaries, values of t lie outside the range $0 \le t \le 1$. The entire line is visible because it is completely within the viewport.

Example 6: Consider the line from P1 (–1, 1) to P2 (9, 3). Determine its visibility against the clipping window (rectangular) defined by vertices (0, 0), (8, 0), (8, 4), (0, 4) using the Cyrus-Beck algorithm.

Solution: The parametric equation for the given line is:

$\mathbf{P}(t) = \mathbf{P}_1 + (\mathbf{P}_2 - \mathbf{P}_1)\, t = (10t - 1)\,\mathbf{i} + (2t + 1)\,\mathbf{j};\ 0 \le t \le 1$

Here, $d = \mathbf{P}_2 - \mathbf{P}_1 = 10i + 2j$. Calculations of parametric intersections are shown in **Table 5**.

Table 5: **Computation of parametric intersections(Example 6)**

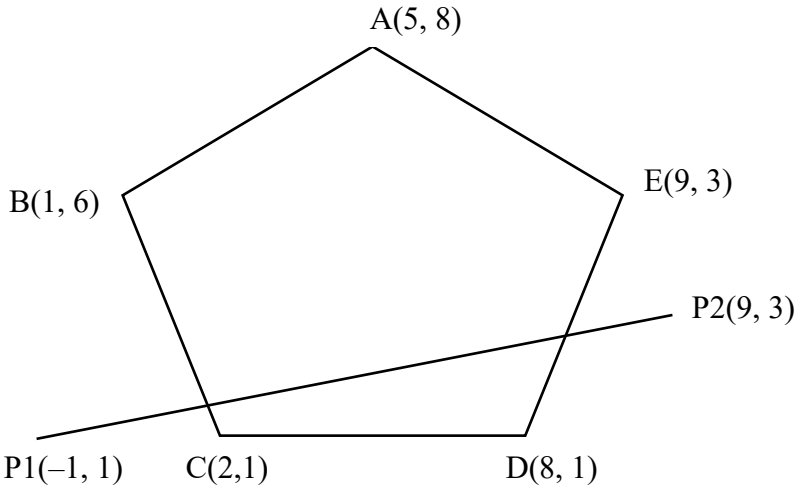
Edge	**n**	**Q**	**w**	**n.w**	**n.d**	t_{min}	t_{max}
Left	**i**	(0, 0)	**–i + j**	–1	10	1/10	—
Right	**–i**	(8, 4)	**–9i – 3j**	9	–10	—	9/10
Bottom	**j**	(0, 0)	**–i + j**	1	2	–1/2	—
Top	**–j**	(8, 4)	**–9i – 3j**	3	–2	—	3/2

In the earlier example, the clipping window was a conventional rectangular clipping window. However, the significance of the Cyrus–Beck algorithm is its applicability to windows of arbitrary (convex). In the following example, we consider one such non-rectangular window.

Example 7: Consider the line P1 (–1, 1) to P2 (9, 3). Determine its visibility against the clipping window (pentagon) defined by vertices A(5, 8), B(1, 6), C(2, 1), D(9, 6) using the Cyrus-Beck algorithm.

Solution:The parametric equation for the given line is:

$$\mathbf{P}(t) = \mathbf{P}_1 + (\mathbf{P}_2 - \mathbf{P}_1)\, t = (10t - 1)\,\mathbf{i} + (2t + 1)\,\mathbf{j};\ 0 \le t \le 1$$



Space for learners:

Space for learners:

Figure 9: Convex window corresponding (*Example 7*)

Here, d = $\mathbf{P}_2 - \mathbf{P}_1$=10i + 2j.
Windows boundaries are: AB = –4i –2j, BC = i –5j, CD = 6i + 0j, DE = i + 5j, and EA = –4i + 2j
Table 6 shows the normal vectors to the window edges. Now, following the logic of the Cyrus-Beck algorithm, appropriate values of t_{min} and t_{max} come out to be 5/16 and 15/16 corresponding to edge BC and DE. The intersection with BC is (2.12, 1.62), while the intersection with DE is (8.4, 2.9).Tables 6 and 7 present the result.

Table 6: Computation of normal vectors (*Example 7*)

Edge	*Edge vector*	*Inward normal*
AB	–4i –2j	i –2j
BC	i –5j	5i –j
CD	6i + 0j	0i –j
DE	i + 5j	–5i + j
EA	–4i + 2j	–i –2j

Table 7: Computation of normal vectors (*Example 7*)

Edge	**n**	**Q**	**w**	**n.w**	**n.d**	t_{min}	t_{max}
AB	i –2j	(5, 8)	–6i – 7j	8	6	–4/3	—
BC	5i –j	(2, 1)	–3i	–15	48	5/10	—
CD	j	(2, 1)	–3i	0	2	0	—
DE	–5i + j	(8, 1)	–9i	45	–48	—	15/16
EA	–i – 2j	(5, 8)	–6i – 7j	20	–14	—	10/7

STOP TO CONSIDER

The **straight line** is composed of an unlimited number of points that are linked on both sides of a point.**Straight lines** can be horizontal, vertical, or inclined in any direction. The standard equation of a straight line is, **y = mx +** c where m is the slope of the line.

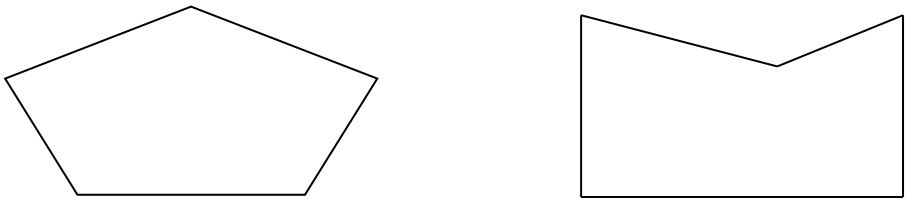
Space for learners:

4.6 POLYGON CLIPPING

A polygon is a closed figure bounded by straight lines with one bounded component and one unbounded component. Computer graphics deals with both convex and concave polygons. If we traverse a polygon in one direction, we fix the orientation of the polygon. That is, we can identify two sets of points: a set of points inside the polygon and another set of points outside the polygon.

A convex polygon has no extended sides that cut other sides or vertices; it can only be split in two points by a straight line. A line segment connecting any two points of the interior of the convex polygon does not go outside the polygon; none of the interior angles is greater than 180 degrees [**Figure 9(a)**]. This is the most studied polygon variant in graphics. Concave polygons do not satisfy this property [**Figure 9(b**)]. Many a time, problems involving concave polygons are solved by portioning them into more than one convex ones.

A polygon is represented as a set of vertices as P_1, P_2, …, Pn with the assumptions that P_iP_{i+1} is an edge for i=1, 2,..., n–1 and P_nP_1 is also an edge. In such a case, the polygon can be represented as a circular link list. With the help of this presentation, a polygon traverse is equivalent to traversing a circular link list.



(a) Convex polygon (b)Concave polygon

Figure 9: (a) A convex polygon; (b) a concave polygon

The suppression of graphics outside a predefined area is called clipping. Because a polygon is often kept as a collection of vertices, any clipping technique will generate a new collection.After all, a cropped polygon is still a polygon.The clipped polygon usually has more vertices than the

unclipped polygon, although it can alternatively have the same number or fewer.The clipped polygon has zero vertices if the unclipped polygon is totally outside the clipping boundary.

Space for learners:

4.6.1 Sutherland – Hodgeman Polygon Clipping

There are a number of well-known polygon clipping algorithms, each with its own set of advantages and disadvantages. The oldest one is called the Sutherland-Hodgeman algorithm. It is relatively straightforward in its most basic form. It can also be used in two situations: when the polygon is completely within the borders and when it is completely outside the boundaries. This algorithm compares each window edge to the polygon as a whole. It starts with a set of polygon vertices and creates a new set of vertices by clipping against the left window edge. The new set of vertices is now clipped against the bottom edge, the right edge, and the top edge respectively. Sutherland-Hodge approaches the challenge using a divide-and-conquer strategy. The polygon is initially clipped using the right clipping boundary. The (partially clipped) polygon is then clipped against the top boundary, followed by the other two boundaries. It works in any order to make sure that all the edges of the clipping polygon are taken sequentially (**Figure 10**).

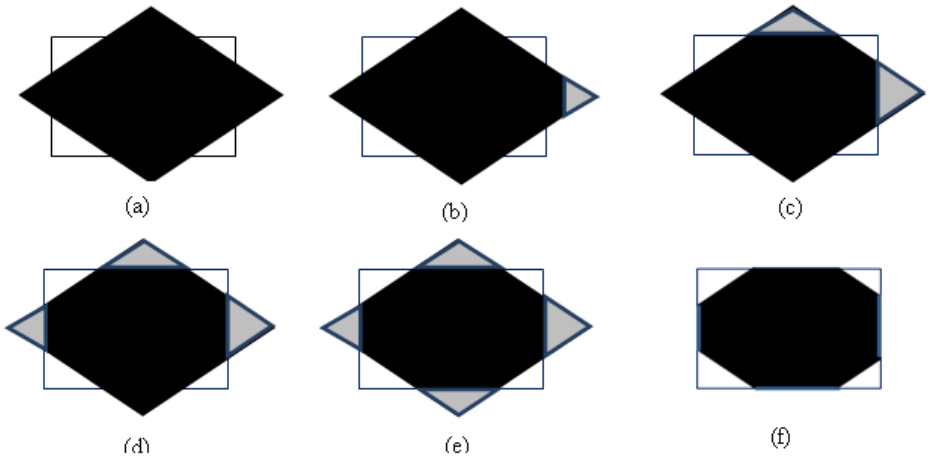


Figure 10: Clipping of polygon using Sutherland–Hodgeman technique

A new set of vertices is created and handed to the next window boundary clipper at each step. Possible causes for the vertices to be clipped are as follows:

***Case 1*:**The intersection point with the window edge and the current vertex are both added to the new vertex list if the previous vertex was outside the window boundary and the current vertex was inside.

***Case 2*:** If both the vertices are inside the window, then only the current vertex will be added to the list.

***Case 3*:** If the previous vertex is inside the window boundary, and the current vertex outside the window boundary, then only the intersection point with the window edge is added to the list.

***Case 4*:** If both the vertices are outside the window boundary, then the intersection points are tested for visibility and then added to the vertex list.

This way, we get a new polygon, clipped against one boundary and ready to be clipped against the next boundary. The method works, but it has one drawback: in order to clip against all four boundaries, you must save the intermediate (partially clipped) polygons. This is a costly operation (**Figure 11**).

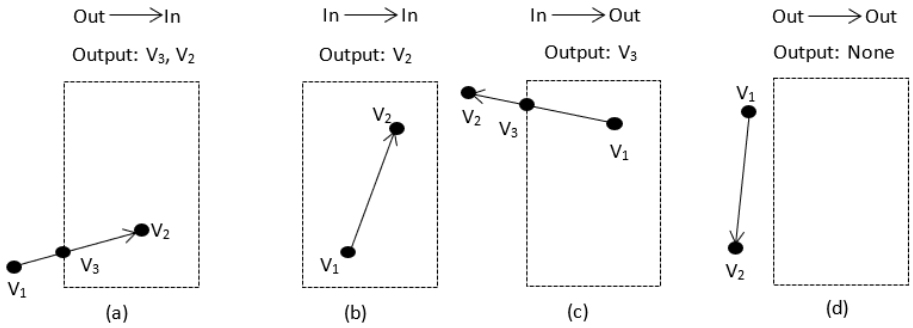


Figure 11: Evaluating polygon edges against the left boundary of the clipping window

To reconcile, Sutherland-Hodgeman comprises two key processes in the algorithm:

1) Perform the inside-outside test to determine the visibility of a point or vertex.
2) Find the point where the polygon edge and the clipping plane intersect.

Space for learners:

Let us consider a method to determine the visibility of a vertex or a point. Let V be the vertex under consideration and AB the window boundary. Vectors AB and AV line in the plane defined by three points A, B, and V. If this plane is considered in the *xy* plane, then the vector cross product AV X AB has a z component given by

$$(X_V - X_A)(Y_B - Y_A) - (Y_V - Y_A)(X_B - X_A).$$

The sign of the z-component decides the position of point V concerning the window boundary. This is how the user interface is described:

z>0: Point is on the right side of the window boundary.

z=0: point lies on the window edge

z<0: point lies on the left of the window edge.

The intersection of the polygon edge and the clipping plane is the second critical process in the Sutherland–Hodgeman method. For this purpose, any of the well-known techniques can be applied. Formally, the Sutherland-Hodgeman polygon clipping algorithm can be stated as:

Algorithm

***Step1*:** Read the coordinates of vertices of the subject polygon and clipping polygon.

***Step2*:** Consider an edge of the clipping window and compare the vertices of each edge of the subject polygon with the clipping window plane or the edge and record the intersections.

***Step3*:** Store the new intersection and vertices in the new list of vertices as per the cases discussed above.

***Step4*:** Perform Steps 2 and 3 for the remaining edges of the clipping polygon each time, the resulting list of polygon vertices are successively passed to process the next edge of the clipping polygon.

***Step5*:** Finish.

Example 6: Consider a clip polygon defined by A (2, 2), B (2, 6), C (8, 6) and D (8, 2). Find if the points V_1 (1, 4) and V_2 (4, 4) lie inside or outside of the clip polygon, using polygon edge AB as reference.

Solution:

Consider the point V_1 (1, 4). Using the cross product V_1, we have

$$(X_V - X_A)(Y_B - Y_A) - (Y_V - Y_A)(X_B - X_A) = (1 - 2)(6 - 2) - (4 - 2)(2 - 2) = -4$$

Space for learners:

Since the cross product of $V_1$is negative, V_1lies left of the window boundary AB (outside window). Similarly, using the cross product of V_2 (4, 4), we get

$$(4 – 2) (6 – 2) – (4 – 2) (2 – 2) = 8$$

Since the cross product for V_2 is positive, the point V_2 lies on the right side of the polygon boundary (inside the clip polygon).

Space for learners:

4.7 CURVE CLIPPING

Curve clipping is a bit complex as compared to polygon clipping because curve-clipping procedures involve nonlinear equations. The bounding rectangle of a circle or any other curved object is used to evaluate the overlap with the rectangular clip window, just like polygon clipping. There exist three cases as follows:

***Case 1*:** If the bounding rectangle for the object is completely inside the clip window, the object is saved for display [see **Figure 12(a)**].

***Case 2*:** If the bounding rectangle for the object is completely outside the clip window, the object is discarded [see **Figure 12(b)**].

***Case 3*:** If the two regions overlap, more computations need to be performed to determine the clipping intersection points. **Figure 12(c)** shows a circle clipping against a rectangular window.

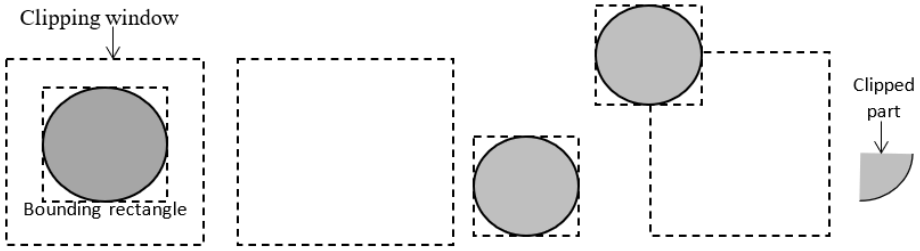


Figure 12: Curve Clipping

4.8 TEXT CLIPPING

Text clipping refers to the process of either removing the entire string if it does not lie completely inside the clipping window or removing that character from the string which overlaps with a window boundary. There are three ways of text clipping, which are as follows:

- ***All-or-none string clipping:*** in this approach, if the string completely lies inside the clipping window, it is displayed; otherwise, it is discarded. It is the simplest and fastest text clipping technique. To implement this technique, a bounding rectangle is taken around the string. The boundary positions of the rectangle are then compared to the boundaries of the window. The string is discarded if it overlaps the window or lies completely outside the window. **Figure 13 (a)shows** the all-or-none string clipping.
- ***All-or-none character clipping:*** In this technique, only those characters are considered which completely lie inside the window. To implement this technique, the boundary limits of individual characters are compared to the boundaries of the window. Any character that overlaps the window or lies outside the window is discarded. **Figure 13 (b)** shows the all-or-none character clipping.
- ***Clipping character components:*** In this approach, components of individual characters are considered for clipping. To implement this technique, the characters are treated like lines. We discard only that part of the character which overlaps a window boundary. **Figure 13 (c)** shows clipping character components.

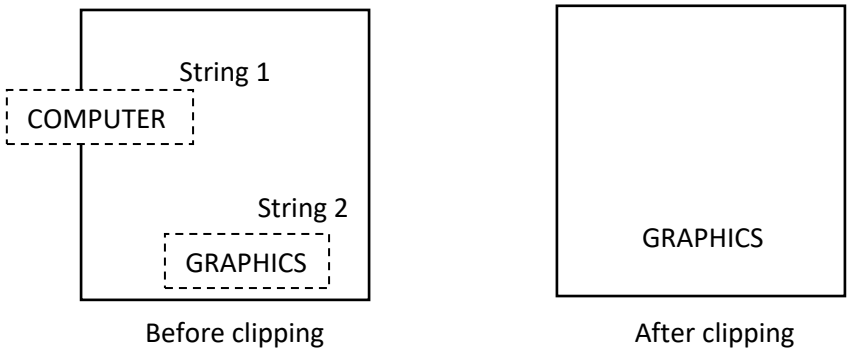
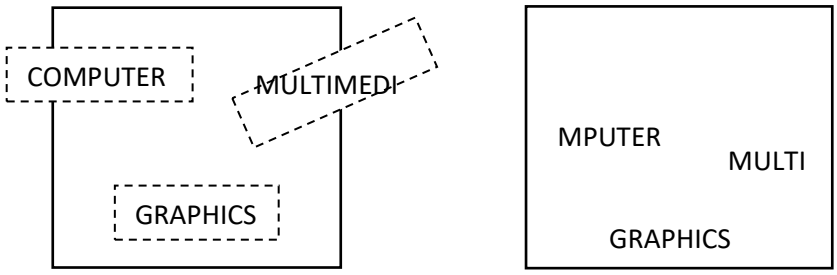


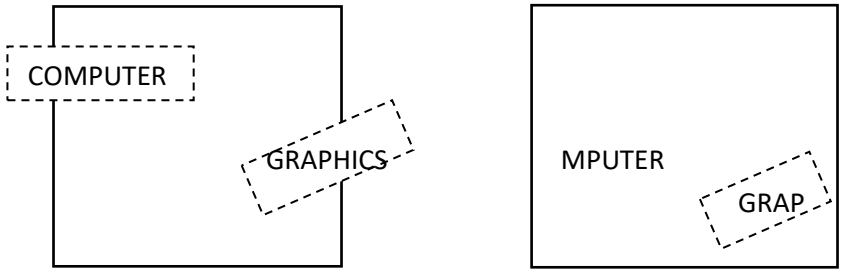
Figure 13(a): All-or-none string clipping



Space for learners:

Before clipping After clipping

Figure 13(b): All-or-none character clipping



Before clipping After clipping

Figure 13(c): Clipping character components

CHECK YOUR PROGRESS

(a) MC stands for _____

i. Modelling coordinates ii. Measuring coordinates

iii. Measuring clipper iv. None of these

(b) WC stands for _____

i. Window clipper ii. Window coordinates

iii. World coordinates iv. None of these

(c) Clipping window is also known as _____

i. Coordinate clipper ii. Clipping coordinates

iii. Clipping ratio iv. Clipping region

(d) _____ is based on parametric equations of a line segment.

i. Midpoint subdivision method

ii. Liang – Barsky algorithm

iii. Cohen – Sutherland algorithm iv. All of these

(e) _____ the process of removing the portion of a polygon which does not lie within the clipping window.

i. Polygon removal ii. Polygon cutting

iii. Polygon clipping iv. None of these.

Space for learners:

4.9. SUMMING UP

In this unit, we have discussed the concept of clipping along with various types of clipping. Also, we have discussed the entire important clipping algorithm in this unit. From the point of achieving realism through computer graphics, the unit is important. This unit contains an algorithm, which is easy to implement in any programming language.

Space for learners:

4.10 ANSWER TO CHECK YOUR PROGRESS

(a) ii. Modelling coordinates
(b) iii. World Coordinates
(c) iv. Clipping region
(d) ii. Liang–Barksy algorithm
(e) iii. Polygon clipping

4.11 POSSIBLE QUESTIONS

1. What is clipping? Discuss its two types.
2. Among Cohen–Sutherland clipping algorithm and Liang–Barsky algorithm, which one is more efficient? Why?
3. Define convex and concave polygon.
4. Write the limitations of the Cohen–Sutherland Line algorithm?
5. Suppose a rectangular window ABCD is defined such that A= (–1, –2) and C (3, 1) using the generalized geometrical approach, clip the line segment joining the points P(–20, 0) and Q(20, 30).
6. Given a clipping window with an upper left corner at (0, 20) and lower right corner at (30, 0). The line has its endpoints at (10, 30) and (40, 0) is to be clipped against the given window. Apply the Cohen-Sutherland method to clip this line and show all the steps clearly.
7. A rectangular window is described by the lower-left corner at (–3, 1) and the upper right corner at (2, 6). Apply the Cohen-Sutherland

method to determine whether the line from A(–4, 2) to B(–1, 7) is subject to clipping.

8. A rectangular window is described by the upper left corner at (3, 8) and lower right corner at (10, 2). The boundaries of the window are parallel to the X-axis and Y-axis. The line has its endpoints at (1, 2) and (12, 7) is to be clipped against the given window. Apply Liang–Barsky method is to clip this line and show all the steps clearly. Intersecting points may be rounded to the nearest integral values.
9. Consider a rectangular window whose vertices are at A(0, 0), B(7, 0), C(7, 5), D(0, 5). A line segment is drawn from P_1(6, 10) to P_2(10, 4). Apply the Cyrus–Beck method to clip this line.
10. The line segment is drawn between P_1(10, 8) and P_2(2, 3) whose direction is from P1 to P2. Determine whether the point P (5, 6) is on the left or the right side of the line.
11. A rectangular window is described by the following corners: (0, 3), (1, 1), (5, 3), (4, 5). Use the Sutherland-Hodgeman clipping method to clip the line segment joining (–1, 2) to (6, 4).

Space for learners:

4.12 REFERENCES AND SUGGESTED READINGS

- Hearn D., Baker M.P., "Computer Graphics", Pearson, 2nd Edition 2011.
- Bhattacharya S, "Computer Graphics", Oxford higher education, 1st edition 2015.
- Malay K. Pakhira, "Computer Graphics, Multimedia and Animation", Phi Learning, 2nd Edition 2010.

UNIT 5: THREE DIMENSIONAL VIEWING AND CLIPPING OPERATIONS

Unit Structure:

Space for learners:

1.1 INTRODUCTION

In 2-dimensional graphics operations, the positions in the world-coordinate plane are transferred to the plane of the output device as pixel positions using viewing operations. Using a 2-dimensional package, we map the world scene to the device coordinates with the help of rectangular boundaries in the world-coordinate window and the device viewport. We clip the scene inside the rectangular boundaries and ignore everything outside.

For 3-dimensional graphics operations, we can generate more views based on our choices. The work involved is more as we can view an object from any spatial position:

- From the front.
- From the above.
- From the back.

We can also generate a view of choice based on how we perceive a given scene, e.g. how we see an object when we are in between a group of objects or when we are inside a single object say a box.

3-dimensional viewing of objects requires us to specify the following:

- A projection plane (called the view plane).
- A center of projection (called the viewpoint) or the direction of projection.
- A view volume in world coordinates.

So, a 3-dimensional description of objects is to be projected on a 2-dimensional flat viewing surface (e.g. output device). The clipping operation defines a shape with a volume enclosure of space depending on the type of projection we would select.

Space for learners:

1.2 UNIT OBJECTIVES

After going through this unit, you will be able to:

- *understand* the fundamental concepts of 3-dimensional viewing.
- *define* the viewing pipeline and viewing coordinates.
- *classify* different types of projections.
- *describe* view volume and clipping operations.

1.3 VIEWING PIPELINE

Generate a 3-dimensional view of a scene is similar to the steps involved in taking a photograph with a camera. To click a photograph, we position our camera from a point in space and then decide on its orientation. Finally, we close the shutter with a click and the world scene is cropped based on the size of the aperture of the camera. The camera film has a projection of the light from the visible surfaces.

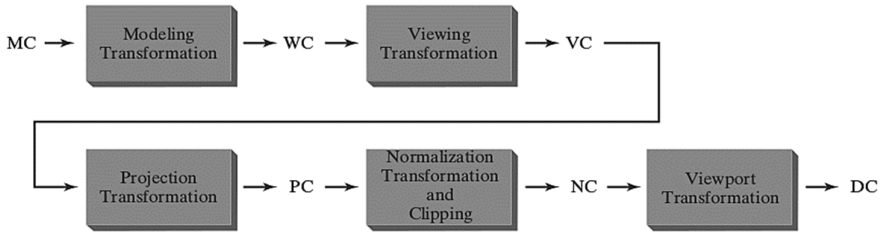


Fig 1[1]: 3-dimensional transformation pipeline, from modeling coordinates (MC) to world coordinates (WC) to viewing coordinates (VC) to projection coordinates (PC) to normalized coordinates (NC) and finally to device coordinates (DC)

The diagram describes the steps involved in converting a 3-dimensional view of a scene into device coordinates. Firstly, the world coordinate coordinates are converted into viewing coordinates once the view has been modeled. The graphics packages use the viewing-coordinate system as a reference to specify the observer's viewing position and the position of the projection plane. The viewing coordinate's description of the scene is converted to coordinate positions on the projection plane using various projection operations. These positions are then mapped to the output device. Clipping operations are done by selecting the objects only inside the viewing boundaries and these objects are processed through visible surface identification. Finally, surface-rendering procedures are done to produce the display within the device viewport.

1.4 VIEWING COORDINATES

When we take a picture of a scene or an object, we can walk around to take its picture from any angle, at various distances, and with varying camera orientations. Whatever we see through the camera lens is projected on the camera film. The size and type of the camera lens determine how much information from the real world will be projected on the film. Using a 3-dimensional graphics package, the view of the scene will be generated based on the spatial position, orientation, and aperture size of the camera lens.

Space for learners:

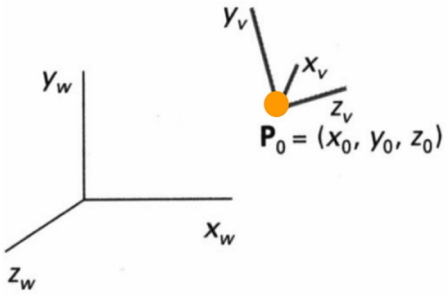


Fig 2[1]: A right-handed viewing coordinate system

1.4.1 Specifying the View Plane, z_v

We first establish a viewing-coordinate system, also called the view reference coordinate system. A view plane or projection plane is then set up for the perpendicular viewing $\boldsymbol{z_v}$ axis. World-coordinate positions from the scene are transformed into viewing coordinates. Then these viewing coordinates are projected onto the view plane.

For establishing the viewing-coordinate reference frame, a world-coordinate position called the view reference point is picked. This point acts as the origin of our viewing-coordinate system. Often, we select the point close to the object or on the surface of the object. Also, we could choose it to be at the center of an object, the center of a group of objects, or somewhere in front of the scene as a camera position.

Then, we select the positive direction of the viewing $\boldsymbol{z_v}$ axis, and the orientation of the view plane, by specifying the view-plane normal vector, **N**. We have to choose a world-coordinate position, and this point gives the direction of **N** relative to the world origin or the viewing-coordinate origin.

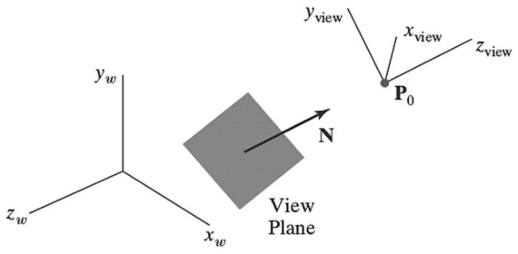


Fig 3[1]: View-plane orientation and view-plane normal vector, **N**

Space for learners:

We also choose the up direction of the view by specifying a vector **V**, called the view-up vector. This vector establishes the positive direction of the y_v axis. We can keep the view-up vector, **V** in any convenient direction which is not parallel to **N**. With vectors **N** and **V**, a third vector **U** is defined for the direction for xv axis , which is perpendicular to both **N** and **V**. Graphics packages do allow users to choose the position of the view plane. The view plane is kept along the z_v axis by specifying the view-plane distance from the viewing origin. The view plane is always parallel to the $x_v y_v$ plane. The corresponding view of the scene is the projection of the objects on the view plane that will be displayed on the output device.

Sometimes, left-handed viewing systems are used in the graphics packages as the viewing direction is in the positive z_v direction. But right-handed viewing systems are also used as they have the same orientation as the world-reference frame. To obtain a series of views of a scene, we keep the view-reference point fixed and then change the direction of **N**.

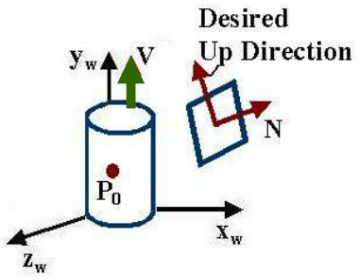


Fig 4[1]: View-up vector, **V**and the view-plane normal vector, **N**

1.4.2 Transformation From World to Viewing Coordinates

Before object descriptions can be projected to the view plane, they must be transferred to viewing coordinates. World coordinate positions are converted to viewing coordinates using the following transformation sequence:

We translate the view reference point to the origin of the world coordinate system.

Then we apply rotations to align thex_v, y_v, and z_v axes with the world x_w, y_w and z_w axes respectively.

Space for learners:

If the view reference point is given at world position (x_0, y_0, z_0) then this point is translated to the world origin with the matrix transformation

$$T = \begin{bmatrix} 1 & 0 & 0 & -x_0 \\ 0 & 1 & 0 & -y_0 \\ 0 & 0 & 1 & -z_0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$M_{w_c,v_c} = R_z * R_y * R_x * T$$

Another method for generating the rotation transformation matrix is to calculate the unit ***on*** vectors and form the composite rotation matrix directly:

$$n = \frac{N}{|N|} = (n_1, n_2, n_3)$$

$$u = \frac{V \times N}{|V \times N|} = (u_1, u_2, u_3)$$

$$v = n \times u = (v_1, v_2, v_3)$$

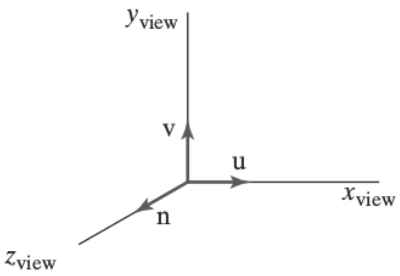


Fig 5[1]: Unit vectors **u**, **v,** and **n** defined right-handed viewing system

Automatically, this method adjusts the direction for **V** so that **v** is perpendicular to **n**. The composite rotational matrix for the viewing transformation is given by:

$$R = \begin{bmatrix} u_1 & u_2 & u_3 & 0 \\ v_1 & v_2 & v_3 & 0 \\ n_1 & n_2 & n_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

The complete world-to-viewing coordinate transformation matrix is obtained by:

$$M_{w_c,v_c} = R * T$$

Space for learners:

This transformation is then applied to coordinate the description of objects in the scene to transfer them to the viewing reference frame.

STOP TO CONSIDER

The 3-dimensional Cartesian (rectangular) coordinate system mainly consists of a reference point, called the origin, and three mutually perpendicular lines passing through the origin. These lines are arbitrarily labeled as x, y, and z axes. The coordinate systems are classified based on orientation as right-handed orientation and left-handed orientation.

Space for learners:

1.5 PROJECTIONS

There are two basic projection methods for projecting 3-dimensional objects onto the 2-dimensional view plane.

Parallel projection:

Coordinate positions are transformed to the view plane along the parallel lines. It preserves the relative proportion of objects and is mainly used in drafting to produce a scale drawing of 3-dimensional objects. It does not give us a realistic representation of the object, but we can get accurate views from all sides of the object.

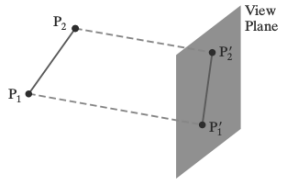


Fig 6[1]: Parallel projection on the view plane

Perspective projection:

Object positions are transformed to the view plane along lines that converge to a point called the projection reference point (or centre of projection). We can determine the projected view of an object by calculating the intersection of the projected lines with the view plane. It gives a realistic view of the object but does not preserve relative proportions. Distant objects appear smaller in projection

compared to objects of the same size at a closer distance from the projection plane.

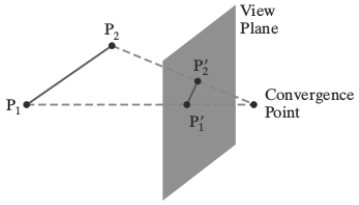


Fig 7[1]: Perspective projection on the view plane

1.5.1 Parallel Projection

A parallel projection is specified with a projection vector. An orthographic parallel projection is obtained when the projection is perpendicular to the view plane otherwise we have an oblique parallel projection.

Orthographic projection gives the front, side, and top views of an object. Front, side, and rear orthographic projections of an object are called elevations and a top orthographic projection is called a plan view. Lengths and angles are accurately shown in orthographic projections and are hence commonly used in engineering and architectural drawings.

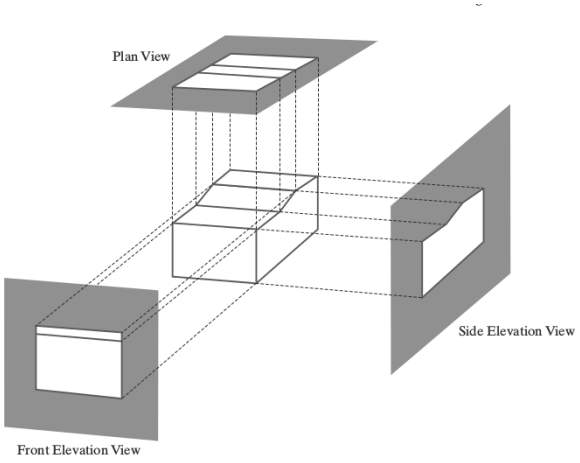


Fig 8[1]: Orthographic projections of an object

Axonometric orthographic projections display more than one face of an object and the most commonly used such projection is the

Space for learners:

isometric projection. An isometric projection is generated by aligning the projection plane such that it intersects each coordinate axis in which the object is defined (called the principal axes) at the same distance from the origin. Whereas in a general axonometric projection, the distance from the origin may be different depending on the scaling factors.

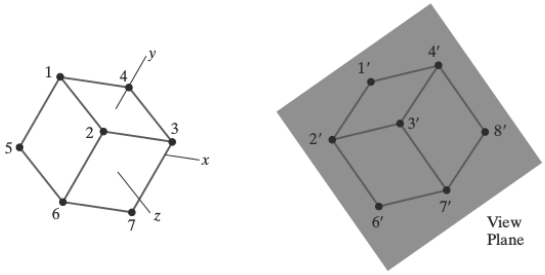


Fig 9[1]: Isometric projection of a cube

An oblique projection can be obtained by projecting points along parallel lines that are not perpendicular to the projection plane. An oblique projection vector can be specified with 2 angles, **α** and **φ**. We can express the projection coordinates in terms of x, y, L and **φ** as:

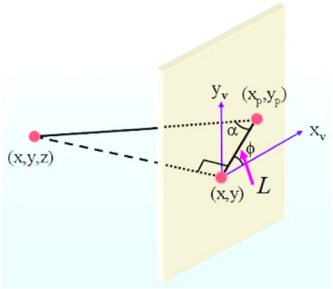


Fig 10[1]: Oblique projection of a coordinate point onto the view plane

$$x_p = x + L\cos\varphi$$

$$y_p = y + L\sin\varphi$$

$$\tan\alpha = \frac{z}{L}$$

$$L = \frac{z}{\tan\alpha} = zL_1$$

Space for learners:

$$x_p = x + z(L_1 cos\ \varphi)$$

$$y_p = y + z(L_1 sin\ \varphi)$$

$$M_{parallel} = \begin{bmatrix} 1 & 0 & L_1 cos\ \varphi & 0 \\ 0 & 1 & L_1 sin\ \varphi & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

When L_1 = 0, an orthographic projection can be obtained whereas oblique projections are generated for non-zero values forL_1. For angles $\varphi = 30^o$ and $\varphi = 45^o$, we get a combined view of the front, side and top (or front, side and bottom) of an object. For $\alpha = 45^o$ i.e. tan α = 1, the views obtained are called ***cavalier projections*** where all perpendicular lines to the projection plane are projected as it is without any change in length.

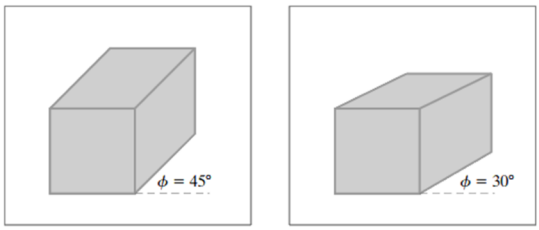


Fig 11[1]: Cavalier projections of a cube

When $\alpha = 63.4^o$ i.e. tan α = 2, the views obtained are called ***cabinet projections*** where lines perpendicular to the viewing plane are projected at half of the original length. As the length of the perpendiculars to the viewing plane is reduced, cabinet projections appear more realistic than cavalier projections.

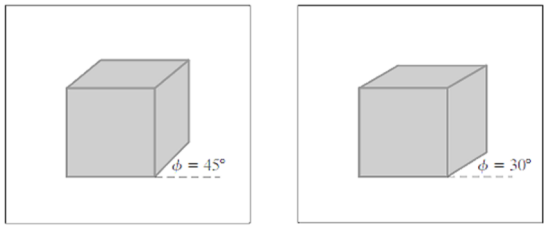


Fig 12[1]: Cabinet projections of a cube

Space for learners:

1.5.2 Perspective Projection

Space for learners:

When we transform points along the projection lines that are meeting at the projection reference point we obtain the perspective projection of a 3-dimensional object.

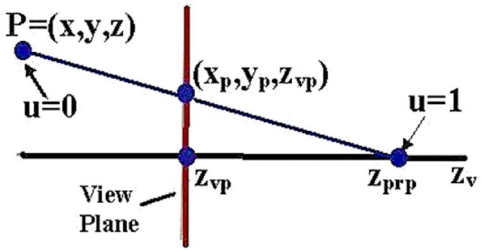


Fig 13[1]: Perspective projections of a point on the view plane

$$x' = x - xu$$

$$y' = y - yu$$

$$z' = z - (z - z_{prp})u$$

$$u = \frac{z_{vp} - z}{z_{prp} - z}$$

$$x_p = x\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right) = x\left(\frac{d_p}{z_{prp} - z}\right)$$

$$y_p = y\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right) = y\left(\frac{d_p}{z_{prp} - z}\right)$$

$$\begin{bmatrix} x_h \\ y_h \\ z_h \\ h \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-z_{vp}}{d_p} & z_{vp}\left(\frac{z_{prp}}{d_p}\right) \\ 0 & 0 & \frac{-1}{d_p} & \frac{z_{prp}}{d_p} \end{bmatrix} * \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Parallel lines parallel to the view plane will be projected as parallel lines, whereas nonparallel lines are projected into converging lines when a 3-dimensional object is projected on a view plane. The point at which a set of projected parallel lines appears to converge is called a vanishing point. A given scene can have any number of vanishing points, a vanishing point of each set of parallel lines. A principal vanishing point is a vanishing point for a set of lines that is parallel to one of the principal axes of an object. We can have one-

point, two-point, or three-point perspective projections depending on the number of principal vanishing points with the orientation of the projection plane.

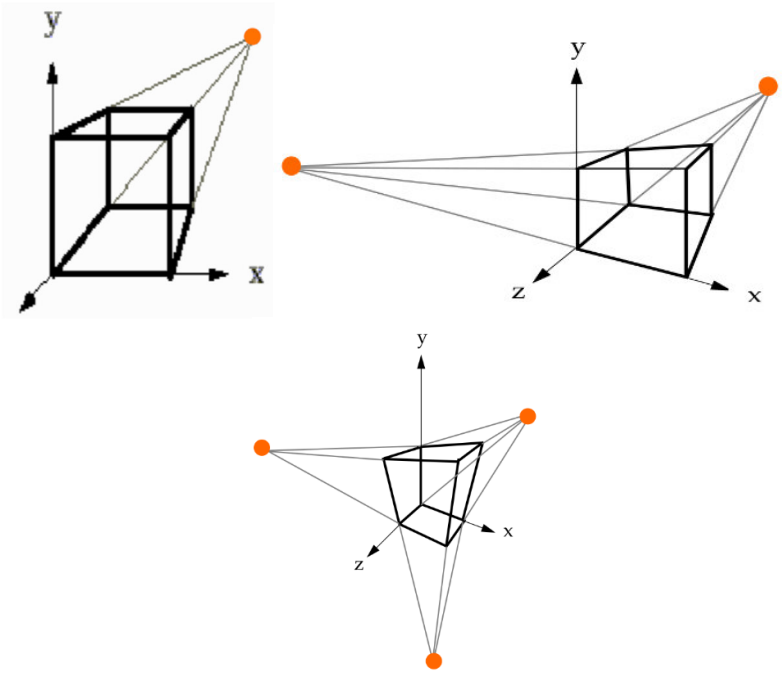


Fig 14[1]: 1-point, 2-point, and 3-point vanishing point views of a cube

STOP TO CONSIDER

Orthographic parallel projections are further sub-divided as isometric, dimetric, and trimetric based on the direction of projection making no. of equal angles with the principal axes respectively.

Perspective projection introduces certain anomalies like perspective foreshortening, vanishing points, view confusion, and topological distortion.

1.6 VIEW VOLUMES

In 3-dimensional viewing, a rectangular view window or projection window in the view plane determines how much information is projected on the view plane. A view volume is set up with the given specifications of the view window within the window boundaries.

Space for learners:

The four sides of the view volume form an infinite parallelepiped for a parallel projection and it forms a pyramid for a perspective projection.

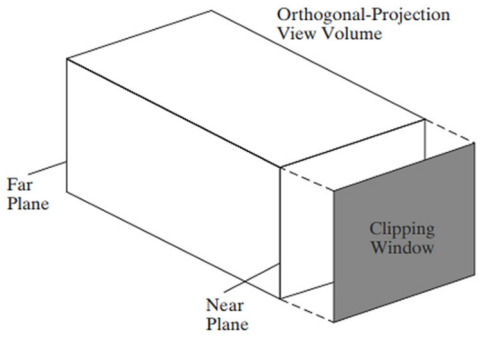


Fig 15[1]: Parallelepiped view volume (Orthogonal parallel projection)

For an orthographic parallel projection, the six planes form a rectangular parallelepiped while an oblique parallel projection forms an oblique parallelepiped view volume, whereas, for a perspective projection, front and back clipping planes truncate the infinite pyramid view volume to form a frustum. Based on depth, these two planes allow us to discard parts of the scene from the viewing operations.

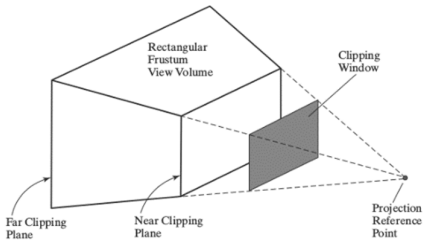


Fig 16[1]: Frustum view volume (Perspective projection)

The view-plane positioning does not affect orthographic parallel projections as the projection lines are perpendicular to the view plane irrespective of its location. The view-plane positioning may affect the oblique projections depending on how the projection direction is specified. In a perspective view, the projected size of an object is also affected by the relative position of the object and the view plane. When the view plane is kept in front of the object, the

Space for learners:

projected object appears smaller, and when it is kept in the back of the object, the size increases. Also, depending on the view-plane positioning, we can generate a static view of an animation sequence of a perspective projection.

Space for learners:

1.7 CLIPPING OPERATIONS

The 3-dimensional clipping algorithm identifies and saves all surface segments within the view volume and discards everything outside of it. We can now clip objects against the boundary planes of the view volume. Clipping view volume boundaries are planes whose orientations is depended on the type of projection, the projection window, and the position of the position reference point. Before clipping, if we convert the view volume into a rectangular parallelepiped, the whole process is simplified. Clipping a rectangular parallelepiped is much simpler as each surface is perpendicular to one of the coordinate axes. For an orthographic parallel projection, view volume is a rectangular parallelepiped whereas for an oblique projection view volumes are converted into a rectangular parallelepiped by a combination of shearing and scaling transformations.

Two different clipping strategies have been devised to perform 3-dimensional clipping:

1. **Direct clipping**: In this method, clipping is done directly against the view volume.

2. **Canonical clipping**: Normalized transformations are applied to transform the original view volume into a canonical view volume. After that, clipping is done against the canonical view volume.

1.7.1 Normalized View Volumes

In the 3-dimensional transformational pipeline, at the projection stage, the viewing coordinates are converted into projection coordinates converting the view volume into a rectangular parallelepiped. This is mapped into a unit cube (normalized view volume) called a normalized projection coordinate system. Finally, normalized projection coordinates are converted into device coordinates for display in the workstation stage.

The normalized view volume is a region defined by the planes:

$$x = 0, x = 1, y = 0, y = 1, z = 0, z = 1$$

In the projection coordinate system, clipping against the unit cube gives several advantages:

1. The normalized view volume gives a standard shape for representing any sized view volume, i.e. unit cube can be mapped to a workstation of any size.

2. Clipping procedures are being simplified and standardized with unit clipping planes or viewport planes.

3. Since, the z-axis is pointing towards the user so depth cueing and visible surface determination are simplified.

Space for learners:

1.7.2 Viewport Clipping

In 2-dimensional clipping, a four-digit binary region code is used to identify the position of a line endpoint relative to the viewport boundaries. Now, for 3-dimensional points, a six-bit region code is used to identify the region of the point relative to the viewport. We can assign bit positions in the region code from left to right for a line endpoint at (x, y, z).

$$bit\ 1 = 1, \quad if\ x < xv_{min}(left)$$

$$bit\ 2 = 1, \quad if\ x > xv_{max}(right)$$

$$bit\ 3 = 1, \quad if\ y < yv_{min}(bottom)$$

$$bit\ 4 = 1, \quad if\ y > yv_{max}(top)$$

$$bit\ 5 = 1, \quad if\ z < zv_{min}(near)$$

$$bit\ 6 = 1, \quad if\ z > zv_{max}(far)$$

The region code of **000 000** indicates the point is within the volume, whereas the code of **010 100** indicates a point being below and in front of the viewport. We test for intersections with the bounding planes of the volume if we could not identify the region of the endpoints of a line segment inside or outside the view volume.

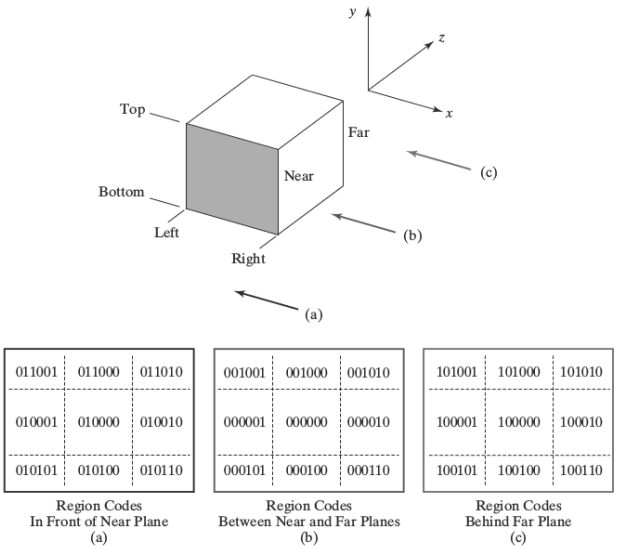


Fig 17[1]: Six-bit region codes that identify boundaries of a view volume

1.7.3 Clipping in Homogeneous Coordinates

Various graphics packages represent the coordinate positions in homogeneous coordinates. When any coordinate position enters the transformation pipeline, it is converted into a homogeneous-coordinate representation:

$$(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) \rightarrow (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}, \boldsymbol{1})$$

Various transformations can be applied to obtain the final homogeneous point:

$$\begin{bmatrix} \boldsymbol{x_h} \\ \boldsymbol{y_h} \\ \boldsymbol{z_h} \\ \boldsymbol{h} \end{bmatrix} = \begin{bmatrix} \boldsymbol{a_{11}} & \boldsymbol{a_{12}} & \boldsymbol{a_{13}} & \boldsymbol{a_{14}} \\ \boldsymbol{a_{21}} & \boldsymbol{a_{22}} & \boldsymbol{a_{23}} & \boldsymbol{a_{24}} \\ \boldsymbol{a_{31}} & \boldsymbol{a_{32}} & \boldsymbol{a_{33}} & \boldsymbol{a_{34}} \\ \boldsymbol{a_{41}} & \boldsymbol{a_{42}} & \boldsymbol{a_{43}} & \boldsymbol{a_{44}} \end{bmatrix} * \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ \boldsymbol{z} \\ \boldsymbol{1} \end{bmatrix}$$

where $\boldsymbol{h}$ can be any real value.

Clipping operations are done in homogeneous coordinates and then they are converted to nonhomogeneous coordinates in 3-dimensional normalized-projection coordinates:

$$\boldsymbol{x' = \frac{x_h}{h}} \ , \boldsymbol{y' = \frac{y_h}{h}} \ , \boldsymbol{z' = \frac{z_h}{h}}$$

Space for learners:

Any homogeneous coordinate position (x_h, y_h, z_h, h) is inside the viewport if it satisfies the following inequalities:

$$xv_{min} \leq \frac{x_h}{h} \leq xv_{max}\ ,\ yv_{min} \leq \frac{y_h}{h} \leq yv_{max}\ , zv_{min} \leq \frac{z_h}{h} \leq zv_{max}$$

Thus, the homogeneous clipping limits are:

$$hxv_{min} \leq x_h \leq hxv_{max}\ ,\ hyv_{min} \leq y_h \leq hyv_{max}\ , hzv_{min} \leq z_h \leq hzv_{max} \quad \text{if h} > 0$$

$$hxv_{max} \leq x_h \leq hxv_{min}\ ,\ hyv_{max} \leq y_h \leq hyv_{min}\ , hzv_{max} \leq z_h \leq hzv_{min} \quad \text{if h} < 0$$

CHECK YOUR PROGRESS

1. State whether true or false :

a. The view volume formed by a parallel projection is a frustum.

b. In 3-dimensional clipping, every information of the scene inside the view volume is kept intact.

c. The normalized projection coordinates are converted into device coordinates for display.

d. A six-digit region code is used in 3-dimensional clipping to identify the region of the endpoints of a line.

e. The homogeneous are directly converted into output device coordinates for display.

1.8 SUMMING UP

- The general approach for viewing 3-dimensional objects/scenes is the same as in 2-dimensional viewing.
- More spatial parameters are involved in 3-dimensional viewing operations.

Space for learners:

- To describe the various 3-dimensional viewing parameters, we can use the camera analogy.
- A viewing-coordinate reference frame is established with a view reference point, a view-plane normal vector **N**, and a view-up vector, **V**.
- We can either use perspective or parallel projections to transfer 3-dimensional object descriptions to the view plane.
- Parallel projections can be either orthographic or oblique, which can be specified with a projection vector.
- Perspective projections are obtained with projection lines that converse to a projection reference point.
- Clipping objects from a 3-dimensional scene is done against a view volume. It is done by testing the object coordinates against the bounding planes of the view volume.
- Various graphics packages perform clipping in homogeneous coordinates after all viewings and other transformations are complete.
- Finally, these homogeneous coordinates are then converted into 3-dimensional Cartesian coordinates to be displayed on the output screen.

Space for learners:

1.9 ANSWERS TO CHECK YOUR PROGRESS

1. a. FALSE

1. b. TRUE

1. c. TRUE

1. d. TRUE

1. e. FALSE

1.10 POSSIBLE QUESTIONS

1. How do we determine whether a point P is inside or outside the view volume?

2. Give the equations of the planes forming the view volume for the general parallel projection?

3. Let $P(x_v, y_v)$be the view plane coordinates of a point on the view plane. Find the world coordinates $P(x_w, y_w, z_w)$ of the point?

4. What are the principal vanishing points for the standard perspective transformation?

5. Describe the following:

(a) One-principal vanishing point perspective.

(b) Two-principal vanishing point perspective.

(c) Three-principal vanishing point perspective.

6. Derive the equation for parallel projection onto the XY plane in the direction of the projection, $P = a\boldsymbol{I} + b\boldsymbol{J} + c\boldsymbol{K}$.

7. Find the general form of an oblique projection onto the XY plane.

Space for learners:

1.11 REFERENCES AND SUGGESTED READINGS

1. Hearn, D. & Baker, M. P., *Computer Graphics C Version*, Pearson.

2. Zhigang, X.&Roy, A. P., *Theory and Problems of Computer Graphics,* McGraw-Hill.

BLOCK III: 3D GRAPHICS AND MULTIMEDIA SYSTEMS

UNIT 1: 3D CONCEPT

Space for learners:

Unit Structure:

1.1 INTRODUCTION

In this unit, you will learn about three-dimensional concepts and object representation. Coordinates are used for the purpose of representation of three- dimensional scene. In this regard, the first step is to establish a coordinate reference system for the camera from which the scene is to be captured. The three-dimensional system has three axes x, y and z. The orientation of a coordinate system is determined by two systems. In the right-handed system, you have the thumb of the right hand pointing in the positive z direction as you curl the fingers of the right hand from *x* into *y.* In the left-handed system, the thumb points in the negative *z* direction. In drafting terminology, a spline is a flexible strip that is used to produce a smooth curve by using a designated set of points. To keep it in position on the drafting table (as the curve is drawn), many small weights are distributed along the length of the strip. In industries like shipbuilding, automobile and aircraft, it is a common practice to layout the model's shape curves in full or near full scale on a large drafting floor. This is done using a long, narrow strip of wood or plastic known as Loftman's spline. Unlike the piecewise spline curve, a Bezier spline curve can be fitted to any number of control points. Without

necessitating tangent vector specification at any of the control points, a set of characteristic polynomial approximating functions, called Bezier blending functions blend the control points to produce a Bezier curve segment. You will also learn about B-spline curves and surfaces.

In 3-D viewing we specify a view volume in the world, a projection on to a projection plane, and a viewport on the view surface. Conceptually, objects in the 3D world are clipped against the 3D view volume and are then projected. The contents of the projection of the view volume on to the projection plane, called the window and then transformed into the viewport for display. The following below fig-1.1- shows the conceptual model of 3D viewing process, which is the model presented to the users of numerous 3D graphics subroutine packages.

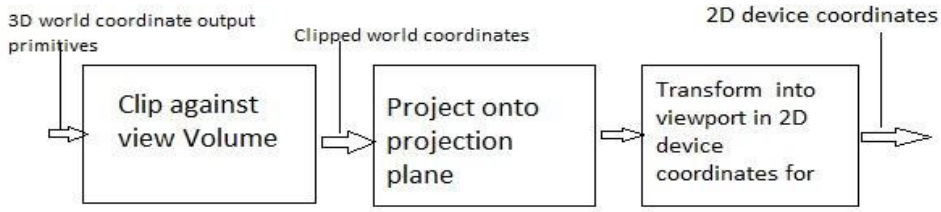


Fig-1.1- Conceptual model of the 3D viewing processs.

1.2 UNIT OBJECTIVES

After going through this unit, you will be able to:

- Discuss the various methods of 3D display
- Understand the concept of 3D graphics
- Describe the various 3D representations

1.3 3D DISPLAY METHODS

Coordinates are used for the purpose of representation of three-dimensional scene. In this regard, the first step is to establish a coordinate reference system for the camera from which the scene is to be captured. By using this coordinate reference system, you can define the position and the orientation of the plane in relation to the camera. The

Space for learners:

object descriptions can be transferred to the coordinates of camera reference system and can be projected onto the specified plane. After this, the objects can be transferred to a wire frame model. Figure 1.2 shows how the lightening and surface rendering techniques can be applied to the shaded visible surface.

Space for learners:

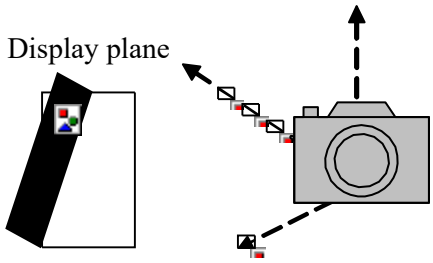


Fig. 1.2 Reference of Coordinates Obtaining a Particular Three-Dimensional Viewing Scene

Figure 1.3 shows the wire frame model of the object in the previous figure.

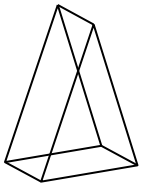


Fig. 1.3- The Wire Frame View of a Three-Dimensional Scene

Viewing a Three Dimensional Scene

- To obtain a display of a three-dimensional world-coordinate scene, we first set up a coordinate reference for the viewing, or "camera," parameters.
- This coordinate reference defines the position and orientation for a view plane (or projection plane) that corresponds to a camera film plane as shown in the above figure1.1.
- We can generate a view of an object on the output device in wireframe (outline) form, or we can apply lighting and surface-rendering techniques to obtain a realistic shading of the visible surfaces Projection.

1.4 DIFFERENT 3D DISPLAY METHOD

Space for learners:

1.4.1 Projection

In general, projection transform points in a coordinates system of dimension n into points in a coordinate system of dimension less than n. In fact, computer graphics has long been used for studying n-dimensional objects by projecting them into 2D for viewing.

Planar geometric projection referred to simply as projections, can be divided into two basic classes: perspective and parallel. The distinction is in the relation of the center of projection to the projection plane. If the distance from the one to the other is finite, then the projection is perspective, if the distance is infinite, the projection is parallel.

Projection is the process of representing a 3D object onto a 2D screen. It is basically a mapping of any point P (x, y, z) to its image P (x', y', z') onto a plane called as projection plane.

1.4.1.1 Parallel Projection

In a parallel projection, parallel lines in the world-coordinate scene projected into parallel lines on the two-dimensional display plane. Another way we can also say that Parallel projection is a method for generating a view of a solid object is to project points on the object surface along parallel lines onto the display plane. This technique is used in engineering and architectural drawings to represent an object with a set of views that maintain relative proportions of the object.

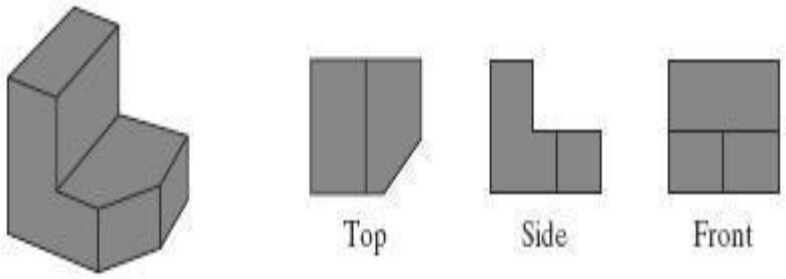


Fig1.4-Three parallel-projection views of an object, showing relative proportions from different viewing positions

Parallel projection are categorized into two types, depending on the relation between the direction of projection and the normal to the projection plane. In orthographic parallel projections, directions are same, so the direction of projection is normal to the projection plane. For the oblique parallel projection, they are not.

The most common types of orthographic projections are the front-elevation, top evaluation and side elevation projections.

Orthographic projection: Orthographic projection utilizes perpendicular projectors from the object to a plane of projection to generate a system of drawing views.

These projections are used to describe the design and features of an object.

It is one of the parallel projection form, all the projection lines are orthogonal to the projection plane

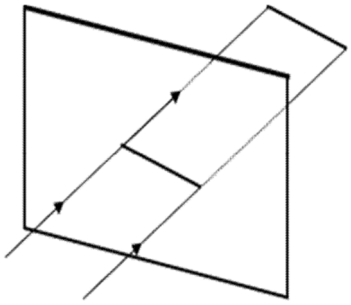


Figure 1.5: Projection plane and projection lines in orthogonal projection

- It is often used to generate the front, top and side views of an object.

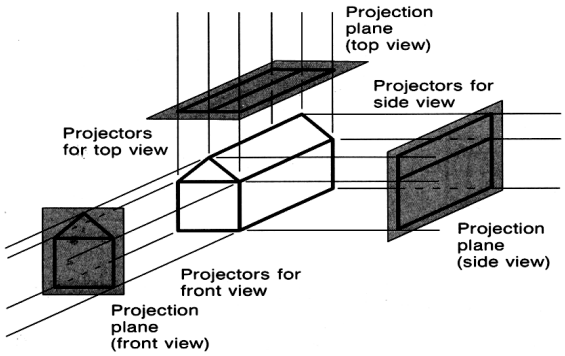


Figure 1.6: Views in orthographic projection

Space for learners:

- It is widely used in engineering and architectural drawings.
- Orthographic projection that displays more than one face of an object is known as axonometric orthographic projections.
- Axonometric projections use projection planes that are not normal to a principal axis. On the basis of projection plane normal N = (dx, dy, dz) subclasses are o Isometric: | dx | = | dy | = | dz | i.e. N makes equal angles with all principal axes.

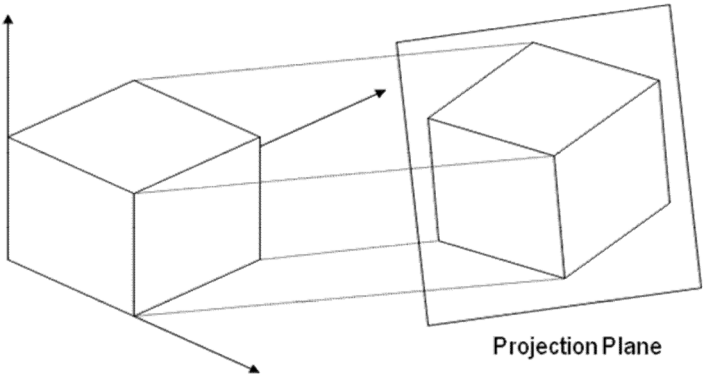


Figure 1.7: Axonometric projection

- Dimetric : | dx | = | dy |
- Trimetric : | dx | ≠ | dy | ≠ | dz |

1.4.1.2 Perspective Projection

The perspective projection of any set of parralle lines that are not parallel to the projection plane coverage to a vanishing point. In 3D, the parallel lines meet only at infinity, so the vanishing point can be thought of as the projection of a point at infinity of directions in which a line can be oriented. This projection is categorised by their number of principal vanishing points and therefore by the number of axes the projection plane cuts.

For generating a view of a three-dimensional scene is to project points to the display plane along converging paths. This causes objects farther from the viewing position to be displayed smaller than objects of the same size that are nearer to the viewing position. In a perspective projection, parallel lines in a scene that are not parallel to the display plane are projected into converging lines.

Space for learners:

This projection method borrows idea from the artists who uses the principle of perspective drawing of three dimensional objects and scenes. The center of projection can be said analogous to the eye of the artist and the plane containing the canvas can be considered as view plane. Perspective projection is used to model 3D objects on 2D plane. It is done by projecting 3D points along the lines that pass through the single viewpoint until they strike an image plane.

Space for learners:

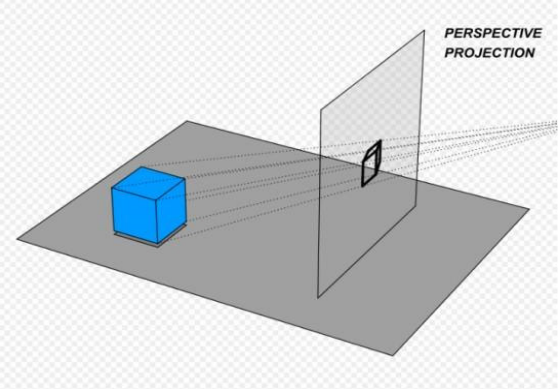


Fig 1.8 - Perspective Projection

- **Frustum view volume:** It specifies everything that can be seen with the camera or eye. It is defined by left plane, right plane, top plane, bottom plane, front (near) plane and back (far) plane.

The following figure illustrates perspective projection

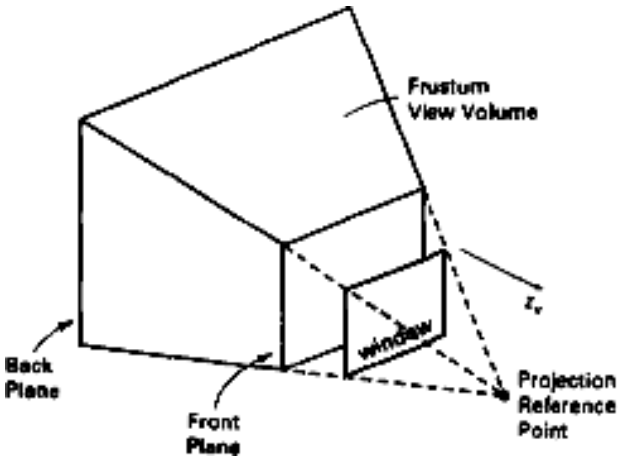


Figure 1.9: Perspective projection

- **Center of projection**: When a 3D scene is projected towards a single point, the point is called as center of projection. Vanishing

points parallel to one of the principal axis is known as principal vanishing point. Projection from 3D to 2D is defined by straight projection rays (projectors) emanating from the center of projection, passing through each point of the object, and intersecting the projection plane to form a projection

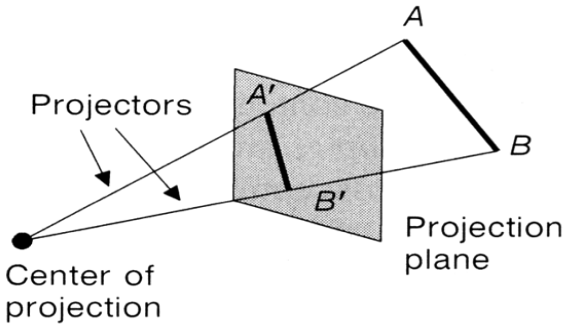
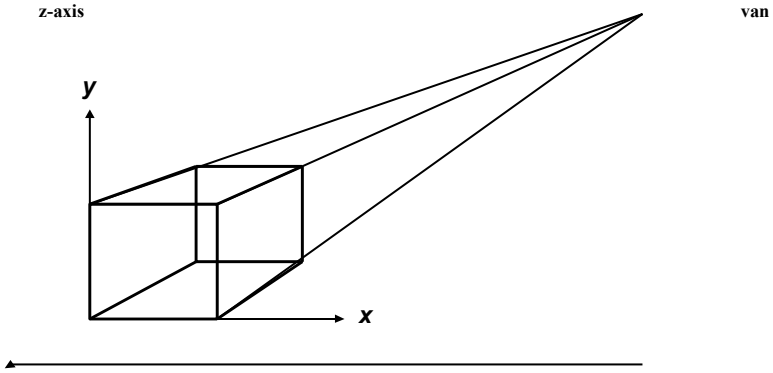


Figure 1.10: Perspective projection illustrating center of projection , projectors and projection plane

- **Perspective foreshortening**: It is the term used for the illusion in which the object or length appears smaller as the distance from the center of projection increases.

- **Vanishing points**: One more feature of perspective drawing is that sometimes a certain set of parallel lines appear to meet at a point. These points are known as vanishing points.



Space for learners:

Figure 1.11: Vanishing point

Space for learners:

1.4.2 Depth Cueing

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

Conceptually we also describe depth cueing in the following way

- Depth information is important in a three-dimensional scene so that we can easily identify, for a particular viewing direction, which is the front and which is the back of each displayed object.
- There are several ways in which we can include depth information in the two- dimensional representation of solid objects.
- A simple method for indicating depth with wire-frame displays is to vary the brightness of line segments according to their distances from the viewing position which is termed as depth cueing.
- The lines closest to the viewing position are displayed with the highest intensity, and lines farther away are displayed with decreasing intensities.
- Depth cueing is applied by choosing a maximum and a minimum intensity value and a range of distances over which the intensity is to vary.
- Another application of depth cuing is modeling the effect of the atmosphere on the perceived intensity of objects

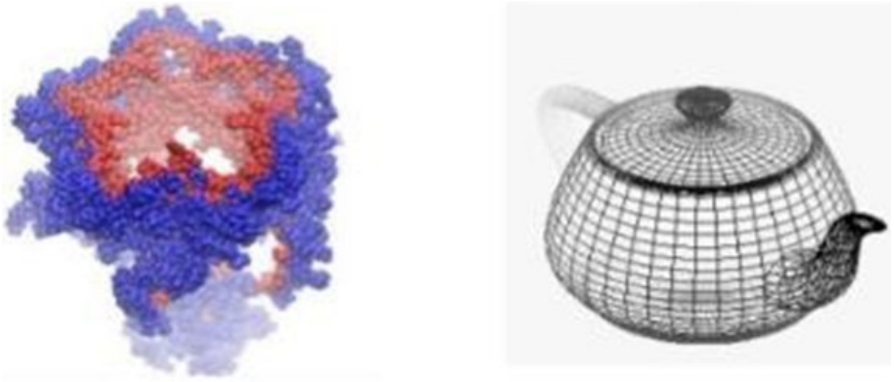
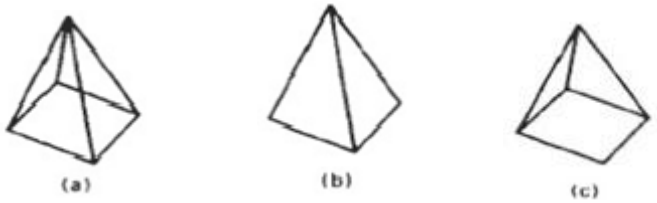


Fig 1.12– Objects displayed with depth cueing, so that the brightness of lines decreases from the front of the object to the back

1.4.3 Visible Line and Surface Identification

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

The wireframe representation of the pyramid in



a) Contains no depth information to indicate whether the viewing direction is
b) Downward from a position above the apex or
c) upward from a position below the base

1.4.4 Surface Rendering

Surface rendering involves the careful collection of data on a given object in order to create a three-dimensional image of that object on a computer. It is an important technique used in a variety of industries.

We can also function of the Surface rendering in the following way -

- Surface rendering method is used to generate a degree of realism in a displayed scene.
- Realism is attained in displays by setting the surface intensity of objects according to the lighting conditions in the scene and surface characteristics.
- Lighting conditions include the intensity and positions of light sources and the background illumination.
- Surface characteristics include degree of transparency and how rough or smooth the surfaces are to be.

One of the techniques to construct an image using surface rendering is with illumination. An illumination model shines a light on a surface and makes calculations based on the intensity of the light reflected back. This collects enough specific data to create a 3D image later. On the computer, a cutting plane is used to give renderers a geometric space with all the data

Space for learners:

collected represented by a color map. This data is then used to recreate an image of the rendered surface.

Fig1.13 – Surface rendering view

Surface rendering is used in a number of industries, such as in health care. There, parts of the body are rendered so doctors can closely examine specific areas of a patient or wounds they may have incurred. Archaeologists also use rendering to make an image of very fragile objects in order to examine them without harming them.

1.4.5 Exploded and Cutaway View

An exploded view drawing is a type of drawing that shows the intended assembly of mechanical or other parts. It shows all parts of the assembly and how they fit together. In mechanical systems usually the component closest to the center are assembled first, or is the main part in which the other parts get assembled. This drawing can also help to represent the disassembly of parts, where the parts on the outside normally get removed first.

Exploded diagrams are common in descriptive manuals showing parts placement, or parts contained in an assembly or sub-assembly. Usually such diagrams have the part identification number and a label indicating which part fills the particular position in the diagram. Many spreadsheet applications can automatically create exploded diagrams, such as exploded pie charts.

A cutaway drawing, also called a cutaway diagram is a 3D graphics, drawing, diagram and or illustration, in which surface elements of a three-dimensional model are selectively removed, to make internal features visible, but without sacrificing the outer context entirely.

Both of them we can define in the following way

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

Space for learners:

- An alternative to exploding an object into its component parts is the cut away view which removes part of the visible surfaces to show internal structure

Fig1.14- Exploded and Cutaway views

1.4.6 Three-Dimensional and Stereoscopic View

Three-dimensional views and Stereoscopic can be obtained by

- ❖ Three-dimensional views can be obtained by reflecting a raster image from a vibrating, flexible mirror.
- ❖ The vibrations of the mirror are synchronized with the display of the scene on the CRT.
- ❖ Stereoscopic devices present two views of a scene; one for the left eye and the other for the right eye
- ❖ As the mirror vibrates, the focal length varies so that each point in the scene is reflected to a spatial position corresponding to its depth.
- ❖ The viewing positions correspond to the eye positions of the viewer. These two views are typically displayed on alternate refresh cycles of a raster monitor

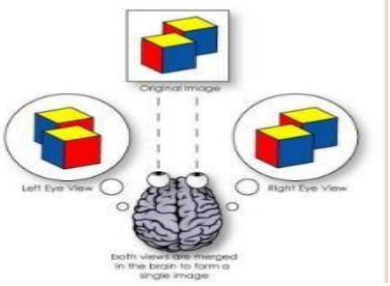


Fig- 1.15-Three dimensional view

For a stereoscopic view, two images are rendered from the point of view of each eye. These images will be stored in a color buffer temporal for display. The resultant images will be displayed on computer screens or

Space for learners:

other output devices, such as project wall, head mounted display Stereoscopic devices present two views of a scene: one for the left eye and the other for the right eye.

Computer technology uses stereoscopic viewing to recreate the way we naturally see depth - stereoscopically. Stereoscopic viewing describes how we use both eyes, each with a slightly different perspective, to perceive depth in a physical environment. It delivers the most realistic visual representation possible of complex digital models, giving engineers, architects and scientists the best possible understanding of three-dimensional information, and yielding levels of technical proficiency not available using a typical 3D view.

The below image can be perceived by a user wearing special glasses which continuously transmit separate images to the left and right eyes, creating a view of computer or video-based objects that have depth, perspective and presence in three-dimensional space.

Fig-1.16 Stereoscopic view

CHECK YOUR PROGRESS

1. Define axonometric projections.
2. Differentiate between isometric projections.
3. Define center of projection.

1.5 SUMMING UP

In this chapter we learnt about world coordinate system and view coordinates. We then learnt the fundamental definition of projection. Orthographic projection with its application was discussed in short. We then learnt perspective projection and terms associated with it and also learn different other technique of 3D viewing.

Space for learners:

1.6 ANSWER TO CHECK YOUR PROGRESS

1. Axonometric projection is a type of orthographic projection used for creating a pictorial drawing of an object, where the object is rotated around one or more of its axes to reveal multiple sides.

2. Axonometric projection is a type of orthographic projection used for creating a pictorial drawing of an object, where the object is rotated around one or more of its axes to reveal multiple sides.

3. Center of Projection: It is the location of the eye on which projected light rays converge. Projectors: It is also called a projection vector. These are rays start from the object scene and are used to create an image of the object on viewing or view plane

Space for learners:

1.7 POSSIBLE QUESTIONS

1. What is the term used for the illusion in which the object or length appears smaller as the distance from the center of projection increases?
2. Define viewing coordinates.
3. Explain orthographic projection with its applications.
4. What is topological distortion?
5. Describe perspective projection and explain perspective foreshortening and vanishing points
6. Differentiate between isometric, dimetric and trimetric projections.

1.8 REFERENCES AND SUGGESTED READINGS

- Computer Graphics, Donald Hearn, M P. Baker, PHI.
- Procedural elements of Computer Graphics, David F. Rogers, Tata McGraw Hill.
- Computer Graphics, Rajesh K. Maurya, Wiley – India.

UNIT 2: PROJECTIONS

Unit Structure:

Space for learners:

2.1. INTRODUCTION

Projection is a process of transforming a three-dimensional (3D) object into a two-dimensional (2D) object for viewing. To display a complex object for viewing on a simpler plane, these projections use visual perspective and aspect analysis.3D projections establish a map of points from the primary features of an object's basic geometry, which are then joined to create a visual element. The outcome is a graphic with conceptual features that perceive the figure or picture as a solid object (3D) being viewed on a 2D display rather than a flat (2D) figure.Many 3D items are shown on two-dimensional surfaces (i.e., paper and computer monitors). As a result, graphical projections are a popular design feature in many fields, including engineering drawing, drafting, and computer graphics. The use of mathematical analysis and equations, as well as numerous geometric and optical techniques, can be used to calculate projections.

In this unit, we shall elaborate various types of projections and their transformation equations. At the end of the unit, a summary of the entire unit is presented along with some exercises.

Space for learners:

2.2. UNIT OBJECTIVES

Students will be able to

- Explain various aspects of projections
- Differentiate different categories of projection
- Explain the mathematical foundations behind projection
- Analyze different types of projections and their applications

2.3. TAXONOMY OF PROJECTION

Projection is obtained by the virtue of imaginary lines called as projectors. When the projectors or line of sights emanating from an object intersects with the view plane an image is formed. These projectors are either parallel to each other or converge together at a specific point. Thus, 3D projection can be broadly categorized into ***parallel*** and ***perspective*** projection. In parallel projection, the 2D view of a 3D object is created using parallel projectors as shown in Fig. 2.1. Such projections are used basically for the actual measurement of objects; however, it fails to project realistic view. On the other hand, in perspective projections, the 2D view of a 3D object is created using projectors which meet at a specific point called as projection reference point. Fig. 2.2 demonstrates an example of perspective projection. Perspective projection creates realistic images for viewing; it works the same way the human visual system works. The size of the projected image of an object is inversely proportionate to the distance between the physical object and the viewpoint.

Space for learners:

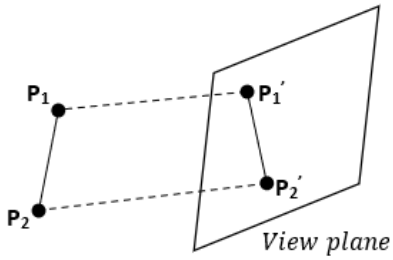


Fig. 2.1: Projection of an object onto view plane using parallel projection

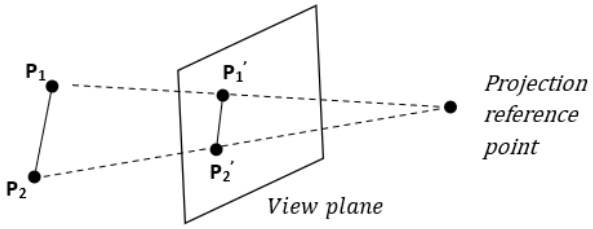


Fig. 2.2 Projection of an object onto view plane using perspective projection

Both parallel and perspective projections can be further classified into multiple categories as shown in Fig. 2.3. These categories are discussed in detail in section 2.4 and 2.5.

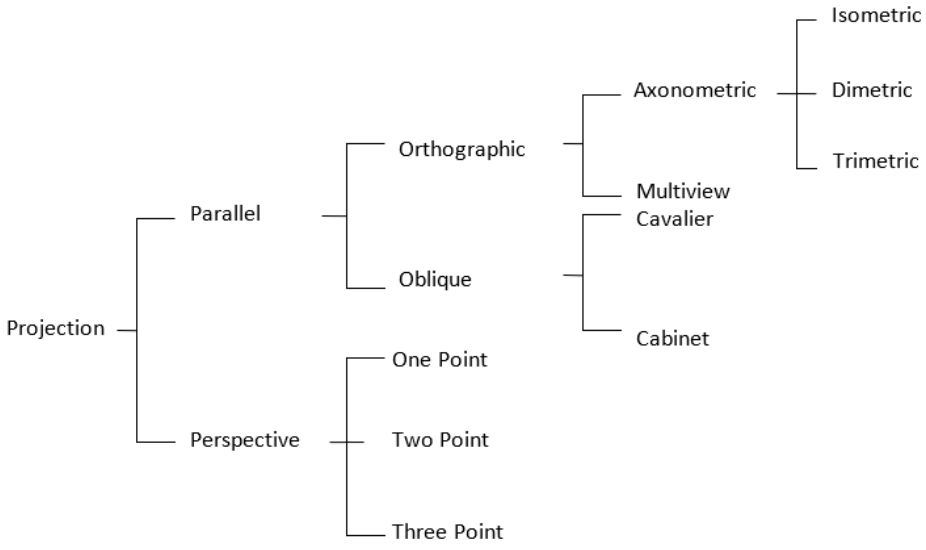


Fig. 2.3: Taxonomy of Projection

2.4. PARALLEL PROJECTION

Space for learners:

Projecting points on the object surface along parallel lines (projectors) onto the display plane is one approach for obtaining a view of a solid object. Multiple views of an object can be displayed by selecting multiple viewing positions. The parallel lines of the object in the world coordinate scene remains parallel in the projected views. In engineering and architectural drawings, this approach is used to show an object with a series of views that maintain the object's relative proportions. From the major views, the structure of the solid object can then be recreated.

Parallel projection can be classified into two categories depending on the direction of the projectors. When the projectors are perpendicular to the view plane then the projection is called as ***orthographic parallel projection***. Otherwise, it is called as ***oblique parallel projection***. Fig. 2.3 shows the two types of parallel projections.

2.4.3. Orthographic Projection

Orthographic projections are mostly used to create the front, side, and top views of an item. Elevations are the front, side, and back orthographic projections of an object, while a plane view is the top orthographic projection. These orthographic projections are frequently used in engineering and architectural designs because lengths and angles are properly shown and may be measured from the drawings.

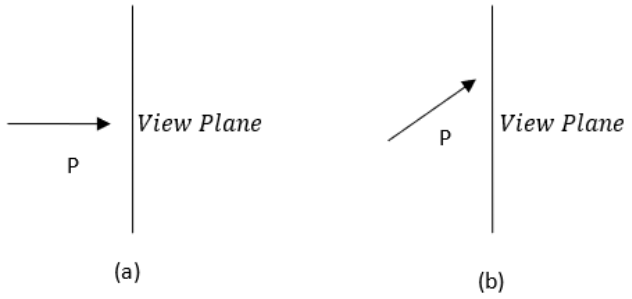


Fig. 2.3: Direction of the projector P to generate (a) orthographic projection and (b) oblique projection

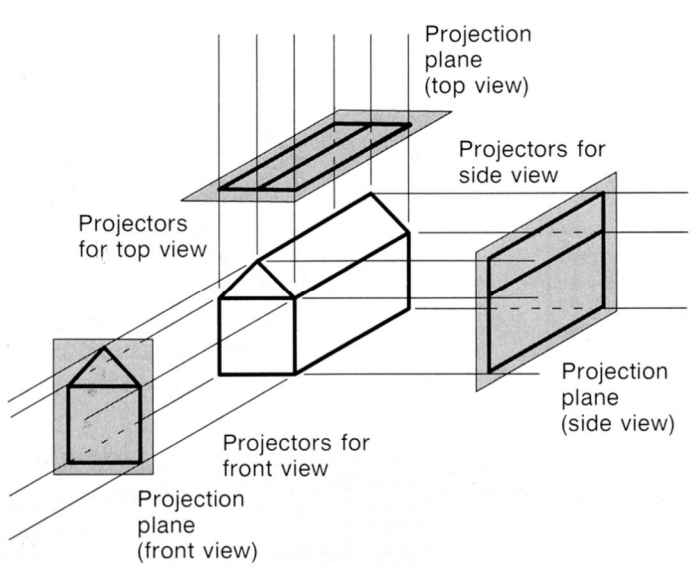


Fig. 2.4 Orthographic projection of an object displaying front, top and side views

Orthographic projections can also be used to display more than one faces of an object as shown in Fig. 2.4. Such type of projections is called as ***axonometric projection***. Axonometric projections have three types: *isometric*, *dimetric* and *trimetric*. The ***isometric projection*** is the most frequent axonometric projection. Isometric projections are created by aligning the projection plane so that it meets each of the object's coordinate axes (known as the primary axes) at the same distance from the origin. Different scaling factors are applied on different principal axes in general axonometric projections. However, in isometric projection same scaling factors are applied for all the principal axes to maintain the relative proportion. An isometric projection of a cube is shown in Fig.2.5. By aligning the projection vector with the cube diagonal, the isometric projection is achieved. For an isometric view, there are eight locations, one in each octant.

Space for learners:

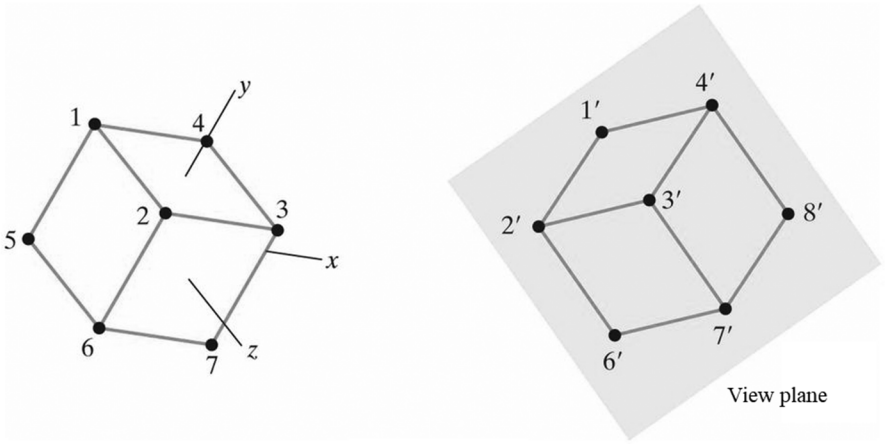


Fig. 2.5 Isometric projection of a cube

In ***dimetric projection*** two of the three principal axes uses same amount of scaling factor while the third one uses a different one resulting uniform foreshortening for two axes. On the other hand, in ***trimetric projection*** all the axes use different amount of scaling factors. Thus, different foreshortenings are seen in this case. Dimensional approximations are common in case of both dimetric and trimetric projections.

An orthographic parallel projection's transformation equations are simple. If the view plane is positioned at z_{prp} along the z-axis (Fig. 2.6), any point in viewing coordinates (x, y, z) is translated to projection coordinates as follows:

$$x_p = x, y_p = y (2.1)$$

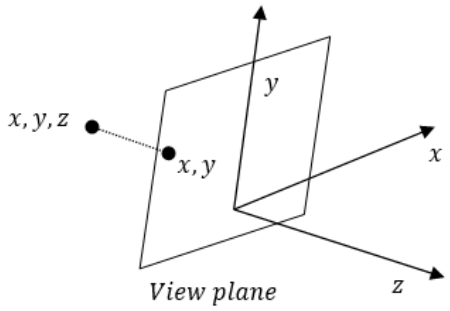


Fig. 2.6 Orthographic projection of a point (x, y, z) onto a view plane

Space for learners:

2.4.4. Oblique Projection

Space for learners:

The projection points along parallel lines that are not perpendicular to the projection plane produce an oblique projection. An oblique projection vector is defined with two angles, α andϕ, in some software packages, as shown in Fig. 2.7. On the view plane, the point (x, y, z) is projected to position (x_p, y_p). On the plane, orthographic projection coordinates are (x, y). The oblique projection line joining the points (x, y, z) to $(x_{p,}y_p)$ forms an angle α with the line on projection plane that connects the points $(x_{p,}y_p)$ and (x, y). In the projection plane, this line of length L is at an angle $\phi$with the horizontal axis. The projection coordinates can be expressed in terms of x, y, L, and ϕ as:

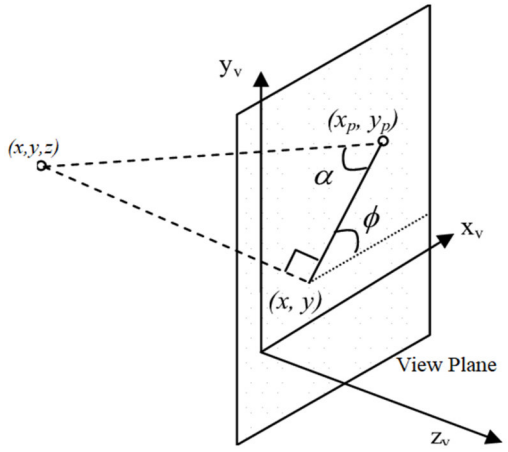


Fig. 2.7 Oblique projection of a point at coordinate position (x, y, z) to coordinate position (x_p, y_p) at view plane.

$$x_p = x + \mathrm{L}\cos\emptyset \quad (2.2)$$

$$y_p = y + \mathrm{L}\sin\emptyset (2.3)$$

We know that,

$$\tan\alpha = \frac{z}{L} (2.4)$$

Where z is the distance from the point (x, y, z) on 3D space to the 2D point (x_p, y_p) on the view plane along the z-axis.

Thus, from equation (2.4) we obtain the length L as:

$$L = \frac{z}{\tan \alpha} (2.5)$$

Let us take, $(\tan \propto)^{-1} = L'$, then we can redefine the value of L as:

$$L = zL' (2.6)$$

Hence, equation (2.4) and (2.5) can be redefined as:

$$x_p = x + z(L' \cos \emptyset) \quad (2.7)$$

$$y_p = y + z(L' \sin \emptyset) \quad (2.8)$$

The transformation matrix for parallel projection can be defined as:

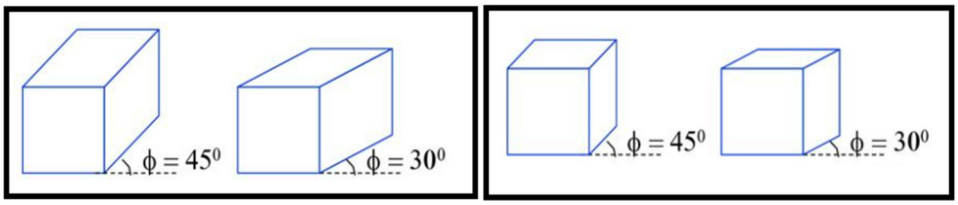
$$T_{parallel} = \begin{bmatrix} 1 & 0 & L' \cos \emptyset & 0 \\ 0 & 1 & L' \sin \emptyset & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} (2.9)$$

The transformation matrix $T_{parallel}$ is applicable to both orthographic and oblique projection, orthographic projection being a special case of oblique projection in which the angle of projection $\alpha = 90$ degree. L becomes 0 in case of orthographic projection ($(\tan 90^0)^{-1} = 0$). Thus, equation (2.1) holds true. In case of oblique projection, the value of L is always non-zero.

For the angleϕ, two values, 30^0 and 45^0 are commonly chosen to project a combination of the object's front, side, and top (or bottom) faces. Two common values chosen for α are 45^0 and 63.4^0.When the value of αis chosen as 45^0, $tan\, \alpha = 1$and the projection is called as ***cavalier projection***. The lengths of all lines perpendicular to the projection plane remain same in their projected images.

On the other hand, when the projection angle α is chosen as 63.4^0so that $tan\, \alpha = 2$, the projection is called as ***cabinet projection***. Lines perpendicular to the viewing surface are projected half their original length for this angle. Because the length of the perpendiculars has been reduced, cabinet projections appear more realistic than cavalier projections. Fig. 2.8

Space for learners:



(a) (b)

Fig. 2.8 Examples of (a) cavalier and (b) cabinet projections for ϕ=45^0 and 30^0

2.5. PERSPECTIVE PROJECTION

Perspective projection of a 3D object is obtained by transforming the points along the projectors that converges at projection reference point. Suppose the view plane is placed at position $z_{vp}$and the projection reference point is set at (0,0,z_{prp}) as illustrated in Fig. 2.9. The coordinate position of the points along the projection line can be expressed using equation in parametric form:

$$x' = x - tx (2.10)$$

$$y' = y - ty (2.11)$$

$$z' = z - t(z - z_{prp}) (2.12)$$

Where$t$ is the parameter which takes the values in the range of 0 to 1 and (x', y', z') represents any point on the projection line. When we set$t = 0$, we get the point (x, y, z) and when we set $t = 1$ we get the projection reference point (0,0,z_{prp}). The projected point ($x_{p,}y_p$) can be calculated by placing appropriate values of t in the equations (2.10-2.12). The z-value on the view plane is z_{vp}, so placing this in equation (2.12) we get:

$$t = \frac{z_{vp} - z}{z_{prp} - z} (2.13)$$

The equations for the perspective projection are obtained by substituting this value of t in the equation (2.10) and (2.11).

Space for learners:

$$x_p = x\left(\frac{z_{prp}-z_{vp}}{z_{prp}-z}\right) (2.14)$$

$$y' = y\left(\frac{z_{prp}-z_{vp}}{z_{prp}-z}\right) (2.15)$$

The matrix representation of perspective projection equations can be written as:

$$T_{perspective} = \begin{bmatrix} \frac{z_{prp}-z_{vp}}{z_{prp}-z} & 0 & 0 \\ 0 & \frac{z_{prp}-z_{vp}}{z_{prp}-z} & 0 \\ 0 & 0 & 1 \end{bmatrix} (2.16)$$

In both parallel and perspective projections, the original z-values of points are saved for any technique that require depth analysis of the surfaces. Although we set the projection reference point along the z-axis, but it can be placed at any arbitrary position.

In perspective projection, any set of parallel lines of a 3D object that are parallel to the plane are projected as parallel lines. However, the lines not parallel to the projection plane do not appear parallel; rather they tend to converge at a point. Such a point is called as a vanishing point. Each set of projected parallel lines will have its own vanishing point, and a scene can have an unlimited number of vanishing points, depending on the number of sets of parallel lines in the scene.

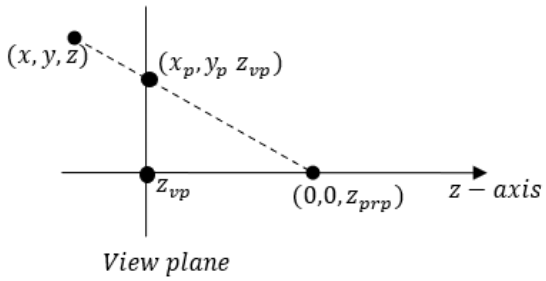
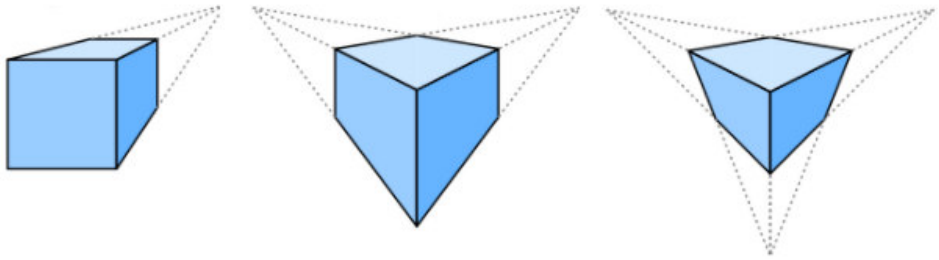


Fig. 2.9 Perspective projection of a point at position (x, y, z) to a position (x_p, y_p) on the view plane.

The vanishing points resulting from any set of lines parallel to one of the principal axes of the object are called as principal vanishing points.

Space for learners:

The number of principal vanishing points (1, 2, or 3) can be controlled based on the orientation of the projection plane. Thus, perspective projections are classified as one-point, two-point, or three-point projections. In a projection, the number of principal axes intersecting the view plane determines the number of principal vanishing points. Fig. 2.10 illustrates one-point, two-point and three-point vanishing points.



(a) (b) (c)

Fig. 2.10 Perspective projections of objects showing (a) one-point, (b) two-point and (c) three-point vanishing points

Space for learners:

CHECK YOUR PROGRESS

i. Straight lines drawn from different positions on an object's contour to intersect a view plane are referred as _____.

a. connecting lines **c.** perpendicular lines

b. projectors **d.** hidden lines

ii. The projection is termed as _____ when the projectors are parallel to one other and perpendicular to the view plane.

a. Perspective projection **c.** Isometric projection

b. Oblique projection **d.** Orthographic projection

iii. Which projection did the objects we perceive in our environment with naked eyes come from?

a. Perspective projection **c.** Isometric projection

b. Oblique projection **d.** Orthographic projection

iv. All lines of sight in perspective projection begin at a _____ point.

a. double **c.** multiple

b. triple **d.** single

v. The appearance of an object will vary depending on how it is seen from different angles and distances. This projection is known as_____ projection.

a. Perspective
b. Oblique
c. Isometric
vi. Orthographic

vii. The point at which the projectors meet is termed as _____.

a. vanishing point
b. projection point
c. projection reference point
d. none of the above

viii. In perspective projection, the point at which the parallel lines meet is called as

a. vanishing point
b. projection point
c. projection reference point
d. none of the above

ix. The angle α is chosen as ___ in cavalier projection.

a. 30^0
b. 45^0
c. 63.4^0
d. 90^0

x. The angle α is chosen as ___ in cabinet projection.

a. 30^0
b. 45^0
c. 63.4^0
d. 90^0

xi. The axonometric projection which uses same scale for all three principal axes is called as_____.

a. Isometric
b. Dimetric
c. Trimetric
d. Uniform

Space for learners:

2.6. SUMMING UP

- Projection is a process of transforming a three-dimensional (3D) object into a two-dimensional (2D) object for viewing.
- graphical projections are a popular design feature in many fields, including engineering drawing, drafting, and computer graphics.
- Projection is obtained by the virtue of imaginary lines called as projectors.
- Projection can be broadly categorized into ***parallel*** and ***perspective*** projection.
- In parallel projection, the 2D view of a 3D object is created using parallel projectors.

- Such projections are used basically for the actual measurement of objects; however, it fails to project realistic view.
- In perspective projections, the 2D view of a 3D object is created using projectors which meet at a specific point called as projection reference point.
- Parallel projection can be classified into two categories depending on the direction of the projectors. When the projectors are perpendicular to the view plane then the projection is called as ***orthographic parallel projection***. Otherwise, it is called as ***oblique parallel projection***.
- In perspective projection, any set of parallel lines of a 3D object that are parallel to the plane are projected as parallel lines. However, the lines not parallel to the projection plane do not appear parallel; rather they tend to converge at a point. Such a point is called as a vanishing point.

Space for learners:

2.7. ANSWERS TO CHECK YOUR PROGRESS

(i) b	(ii) d	(iii) a	(iv) d	(v) a
(vi) c	(vii) a	(viii) b	(ix) c	(x) a

2.8. POSSIBLE QUESTIONS

1. Define projection. What are the different types of projections available?
2. Differentiate between parallel and perspective projection.
3. What are the different types of parallel projection?
4. What do you mean by a vanishing point? What is the maximum number of vanishing points that an image can have?
5. Parallel projections are not suitable for creating realistic images. Justify your answer.

6. What is the difference between orthographic and oblique projection? What are the different types of orthographic projections available?
7. Construct a transformation matrix for general parallel projection.

Space for learners:

2.9.REFERENCES AND SUGGESTED READINGS

- Hearn and Baker, Computer Graphics (C version 2nd Ed.), Pearson.
- Mukherjee, Fundamentals of Computer graphics and Multimedia, PHI.
- D. F. Rogers, J. A. Adams, Mathematical Elements for Computer Graphics, TMH.

UNIT 3: VISIBLE SURFACE DETECTION

Unit Structure:

Space for learners:

3.1 INTRODUCTION

In the computer graphic scenes, the view of an object can be fully or partially blocked from the viewer by some other object. In the realistic image generation, the blocked portion of the image should be eliminated before the scene is rendered. Clipping algorithms cannot be used for this purpose. Instead, another set of algorithms are used to do it. These algorithms are collectively known as visible surface detection methods (or hidden surface removal methods). In this unit, we shall learn about these methods.

In this unit, a right-handed coordinate system would be assumed with the viewer looking at the scene along the negative z-direction. An important thing to remember in this unit is that in visible surface detection, a specific viewing direction is being assumed. This is because a scene hidden from one direction may not be so from

another direction. Also, we would assume objects with polygonal surfaces only as the curved surfaces are converted to polygonal meshes.

Space for learners:

3.2 UNIT OBJECTIVES

After going through this unit you will be able to

- Get an overview of the concept of visible surface detection in computer graphics
- Understand the two broad categories of visible surface detection techniques - object space method and image space method
- Get an idea about the object space method known as back face elimination
- Learn about the well-known image space method- Z-buffer algorithm and A-buffer algorithm

3.3 DEFINITIONS

We see different realistic objects in the real world. In computer graphics, when these objects are transferred, there are parts of the objects that are hidden. In a 3-dimensional scene, all sides of the objects may not be visible to the human eyes. The objects in the scene may be obstructed by their own parts or may be by some other objects in that viewing direction. It is important to identify those parts of the objects that are visible from a viewing position and to remove those parts of the objects that are not visible to the viewer. This would help in reducing the computations required for creating the realistic scene in computer graphics. To remove all those parts such as the lines and surfaces, which are not to be displayed in a 3D scene, the visible surface detection methods, also known as, the hidden surface removal methods are used.

3.4 CLASSIFICATION OF ALGORITHMS

Hidden surfaces of objects are those surfaces that are not directly visible to the viewer. Normally when we see objects, the parts at the back or rear side are not visible to us. Those parts are called the hidden parts or hidden edges.T he algorithms or methods are chosen

on different aspects such as constraints, complexity of scene, available instruments etc. Hidden surface removal algorithms are also known as visible surface detection methods.

Depending on whether these methods deal with object definition directly or with their projected images, they are classified into two categories, namely object-space methods and image-space methods.

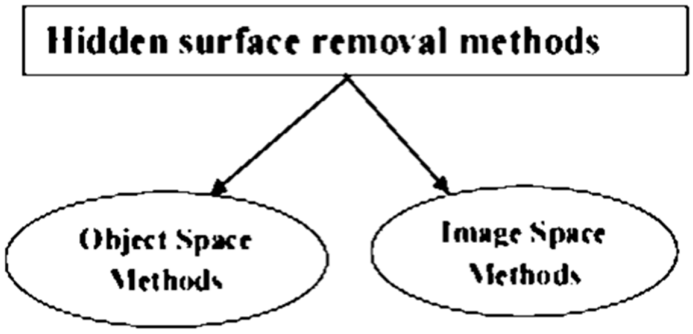


Figure: Types of hidden surface removal methods

3.4.1 Object-Space Method

The objects in the real world need to be projected in computer screen in some coordinate system. The object space algorithms are the algorithms that determine which part of the objects are to be rendered in 3D coordinates. These types of algorithms are implemented in physical coordinate system in which the object is defined. In object space methods the main focus on comparisons between those parts which are visible from specific position and which are not.

Object space algorithms are based on comparison of objects for their 3-dimensional positions and dimensions with respect to a viewing position. The object space algorithms compare the objects and their parts to each other, within the graphic three dimensional scene, so as to determine the visible surfaces. That is, determine those parts of the object whose view is unobstructed by its other parts or any other surrounding object. The 3 dimensional coordinate system is used for such algorithms.

Examples of object space algorithms are Back-face detection, Octree method, ray-casting method, BSP tree method etc.

These algorithms work directly on the objects before they are converted into pixels. Thus, these methods work before projection.

Space for learners:

The object space algorithms are suitable for simple scenes with a small number of objects. For N objects, they may require N*N operations. These algorithms work before projection and have both advantage and disadvantage.

The advantage of these methods is that they are device independent and can work for any resolution.

The Object-space method is implemented in physical coordinate system. Algorithms used in image space for hidden surface removal are much more efficient than object space algorithms. But object space algorithms for hidden surface removal are much more functional than image space algorithms for the same. The combination of these two algorithms gives the best output.

Space for learners:

3.4.2 Image-Space Method

The image-space methods determine the visibility point-by-point at each pixel position on the projection plane. These algorithms work while the objects are being converted into pixels in the frame/refresh buffer. The calculations in these methods are done on a pixel-by-pixel basis and so the resolution of the display device plays an important role. This is a type of discrete method.

The image space algorithms work after the surfaces are projected and rasterized (that is, mapped to pixels). The computations involved in these methods are usually less. A change in resolution requires re-computation of pixel colors. Display resolution effects the accuracy calculation. If the display resolution is changed then re-calculation is required. Implement aspects which are used in hidden surface removal methods are sorting and coherence.

a. Coherence: It is the result of local similarity. Efficiency of sorting is increased using coherence.

Advantage of regularity in the scene is taken by coherence.

b. Sorting: To compare the distance between different surface from the view plane

Image-space method is implemented in screen coordinate system. Most visible surface detection methods make use of the image space methods. Some examples are A-buffer method, Z-buffer or Depth-buffer method, Scan line, Painter's algorithm.

CHECK YOUR PROGRESS - I

Fill up the blanks:

1. Hidden surfaces of objects are those surfaces that are not _____ to the viewer.
2. Hidden surface removal algorithms are also known as _____ methods.
3. Visible surface detection methods are classified into two categories, namely _____ and _____.
4. Object space algorithms are based on _____ of objects for their 3-dimensional positions and dimensions with respect to a _____.
5. The _____ algorithms work directly on the objects before they are converted into pixels.
6. The image space algorithms work after the surfaces are projected and _____.
7. Advantage of regularity in a scene is taken by _____.
8. Image-space method is implemented in _____ coordinate system.

Space for learners:

3.5 ALGORITHMS FOR VISIBLE SURFACE DETECTION

3.5.1 Back-Face Detection

Back-Face detection, also known as Plane Equation method, is an object space method in which objects and parts of objects are compared to find out the visible surfaces.

This algorithm would be plotting the objects parts which are visible to the viewer from his viewing point. The back surfaces of the object that are not visible would be removed. This would be removing about half of the polygons from the scene.

The back face algorithm can be represented geometrically. Each polygon has several vertices. Each surface has a normal vector. If this normal vector is pointing in the direction of the center of projection, then it is a front face and can be seen by the viewer. If

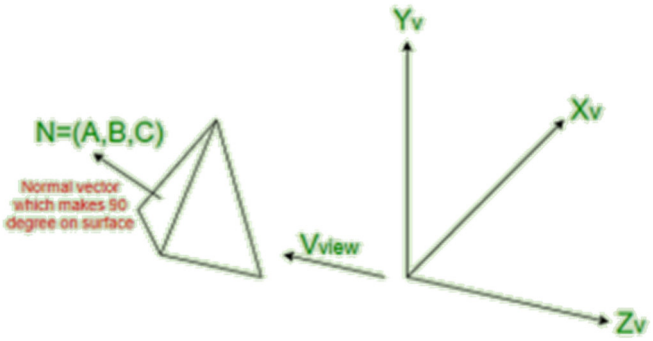
this normal vector is pointing away from the center of projection, then it is a back face and can not be seen by the viewer.

It is based on the "inside-outside" tests.A point (x, y, z) is "inside" a polygon surface with plane parameters A, B, C, and D if

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

When an inside point is along the line of sight to the surface, the polygon must be a back face (we are inside that face and cannot see the front of it from our viewing position). The normal vector N to a polygon surface, has Cartesian components (A, B, C). In general, if V is a vector in the viewing direction from the eye (or "camera") position, then this polygon is a back face if

$$V.N > 0$$



If Object description has been converted to projection co-ordinates and viewing direction is parallel to viewing Zv axis, then

$$V = (0, 0, Vz) \text{ and } V.N = Vz.C$$

The sign of C, the Z-component of the Normal Vector N, needs to be considered.

Thus, any polygon face is back-face if its Normal Vector has Z-component value

$$C <= 0.$$

3.5.2 Depth Buffer Algorithm

Depth Buffer algorithm is also called Z- Buffer algorithm and it is developed by "Edwin Catmull " in 1947. A commonly used image-space approach to detecting visible surfaces is the depth-buffer method, which compares surface depths at each pixel position on the projection Plane. Each surface of a scene is processed separately,

Space for learners:

one point at a time across the surface. The method is usually applied to scenes containing only polygon surfaces, because depth values can be computed very quickly and the method is easy to implement. This algorithm can be implemented on both hardware and software.

For both types of surface, that is, planer and non-planer this algorithm is suitable but it is best for planer surface than non planer, because depth value can be calculated easily. The depth value is measured along z- axis; due to this name of the algorithm is Z-buffer algorithm. The value of z coordinate is zero on screen coordinate system 2D (x, y) while views as 3D (x, y, z).

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

In Depth Buffer, the Z- Value or depth of each pixel is stored in this buffer. Initially value is set to be zero for each. Refresh Buffer is used to store the intensity value of each pixel (x, y).

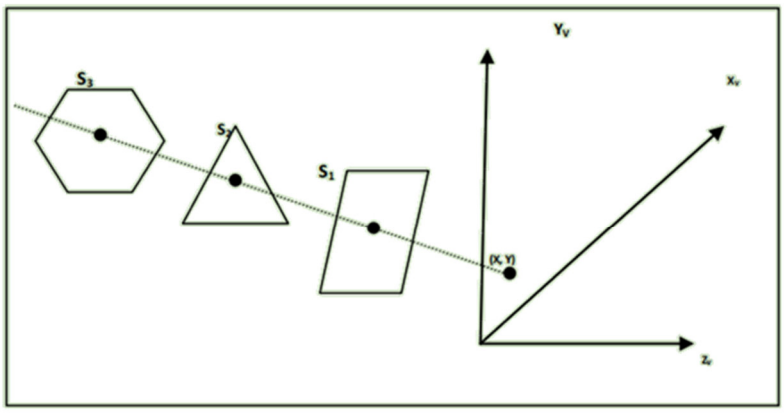


Fig. Plane at different depth along a same project line

Steps in Algorithm: Steps which are followed in Z-Buffer algorithm are given below

1. Initialize the depth buffer for all position

   Depth (x, y) = 0

2. Initialize the refresh buffer for all pixel

   Refresh (x, y) =Intensity of background.

3. Calculate the depth value (Z- Value) for each pixel (x, y)

Space for learners:

4. Compare the depth value of all objects with already stored value. If newly calculated (Z value) is greater than previously stored value then update both buffer otherwise do nothing

If Z-value > depth (x, y) then

Set depth (x, y) = Z-value

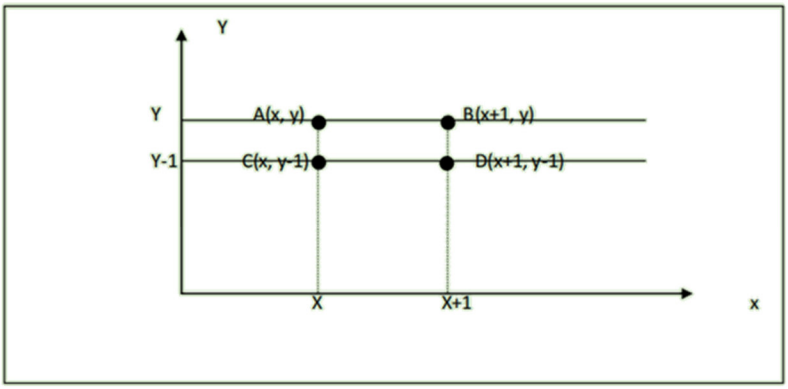
Set Refresh (x, y) = Intensity (x, y)

End if

Depth Calculation: The equation which is used for depth calculation is

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

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



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

This algorithm has some advantages like

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

It has some disadvantages also like

1. It requires large memory.

Space for learners:

2. It is time consuming process.

The Depth-buffer method performs best with the scenes that contains many surfaces. This method has nearly constant processing time independent of the number of surfaces in a scene. This is because the size of the surface areas decreases as the number of surfaces in the scene increases. Thus, this method shows relatively low performance with simple scenes and a relatively high performance with complex scenes.

Space for learners:

3.5.3 A-Buffer Algorithm

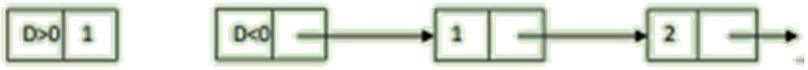
A- Buffer algorithm is an extended version of the Z-Buffer algorithm with some additional information. A drawback of the depth-buffer method is that it can only find one visible surface at each pixel position. In A-Buffer Method more than one surface intensity can be taken into consideration at each pixel position.

This algorithm stores the intensity for each pixel of multiple surfaces using link list. A-Buffer algorithm is also known as area – averaged, anti-aliased, and accumulation buffer method.

In this algorithm, each position in the A- Buffer has two fields.

a. Depth Field : Field of depth stores the +ve / -ve values.

b. Intensity Field : Field of Intensity of each pixel for all surfaces.



If the value of depth is greater than zero (d>0) means there is no link list but if there is negative value of depth (d <0) means multiple surface intensity is exists.

Data for each surface in the linked list includes □

- ✓ RGB intensity components
- ✓ opacity parameter (percent of transparency)
- ✓ depth
- ✓ percent of area coverage
- ✓ surface identifier
- ✓ other surface-rendering parameters

- ✓ pointer to next surface

Space for learners:

3.6 CURVED SURFACE DETECTION

A curved surface can be approximated as a set of polygon surfaces. Each of the surfaces is replaced by a polygon mesh. On this polygon mesh, any of the hidden surface detection, discussed in the above sections, can be used.

In case of curved surfaces, the visible surface detection algorithms like ray-casting and octree methods are being generally used. In ray-casting method, the ray surface intersections are calculated. Then the smallest intersection distance along the pixel ray is located. In the octree method, the visible surfaces are identified once the representations have been established from the input definition of the objects. There is no need to give any special considerations to the different kinds of curved surfaces.

A surface can be represented with an implicit equation of the form,

$$f(x,y,z) = 0$$

In some cases, it is useful to obtain an explicit surface equation. For example, a height function over any xy plane.

$$z= f(x,y)$$

Objects such as spheres, cylinders, cones etc have quadratic representations. There are different techniques such as parallel calculations and hardware implementations have been used for solving the curved surface equations for the commonly used objects.

With curved surfaces, the contour plots could be displayed.

3.7 WIREFRAME DISPLAYS

The objects consist of edges, vertices and polygons. The vertices are joined by the edges and the polygons are simply sequences of edges or vertices. The edges may be straight lines or curved. When only the outline of an object is to be displayed, the visibility tests are applied only to the surface edges. There are two ways to do so. Either the visible sections are being displayed, or else the hidden edges can be displayed differently from the visible edges. For example, the visible edges can be displayed as the outline and the

hidden edges can be displayed as some dotted or dashed lines. These kinds of displays are called wireframe displays.

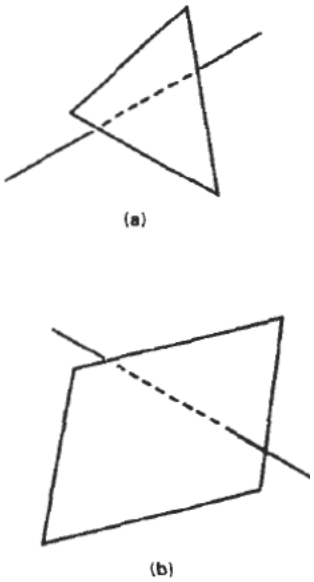
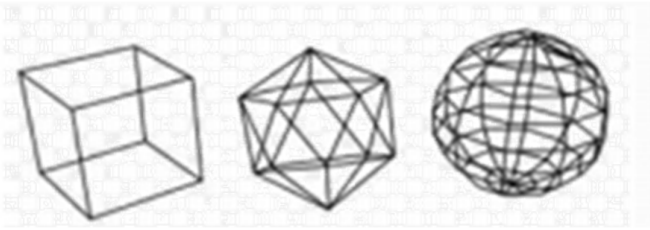


Figure : Dashed line for a line that (a) Passes behind a surface, (b) Passes through a surface

The methods that are used for determining the visible edges of the objects in a scene are called the ***wireframe-visibility methods***. They are also known as *visible-line detection methods* or *hidden-line detection methods*.

A wireframe model is a visual representation of a 3-dimensional object used in 3-dimensional computer graphics. The object is projected on the display screen by some lines for each of the edge. The term '*wireframe*' has come from the designers using metal wire to represent the 3-dimensional shape of the solid objects.



The visible-surface methods discussed above are readily adapted to wireframe visibility testing. Using a back-face method, all the back surfaces of an object could be identified and display only the boundaries for the visible surfaces. With depth sorting, surfaces can

Space for learners:

be painted into the refresh buffer so that surface interiors are in the background color, while boundaries are in the foreground color.

CHECK YOUR PROGRESS II

1. Back-Face detection method is also known as _____ method.
2. In _____ method, the objects and parts of objects are compared to determine the hidden surfaces.
3. When normal vector is pointing in the direction of the center of projection, then it is a _____ and can be seen by the viewer.
4. The Depth Buffer algorithm is also called _____ algorithm.
5. The surface depth at each pixel position on the projection plane is compared in the _____.
6. The depth value is measured along _____ of the coordinate plane.
7. _____ is used to store the intensity value of each pixel (x, y).
8. In Depth Buffer the _____ of each pixel is stored.
9. In _____Method more than one surface intensity can be taken into consideration at each pixel position.
10. A- Buffer has two fields namely _____ and _____.
11. The visible surface detection algorithms generally used in curved surfaces are _____ and _____.
12. The methods that are used for determining the visible edges of the objects in a scene are called the _____.

Space for learners:

3.8 SUMMING UP

- In computer graphics, when the real-world objects are transferred to the coordinate systems, there are parts of the objects that are hidden or not visible to the human eyes.
- The objects in the scene may be obstructed by their own parts or may be by some other objects in that viewing direction.
- It is important to remove those parts of the objects that are not visible to the viewer.

- The visible surface detection methods, also know as, the hidden surface removal methods are used to remove such hidden or not visible parts.
- 5. Depending on whether these methods deal with object definition directly or with their projected images, they are classified into two categories, namely object-space methods and image-space methods.
- Object space algorithms are based on comparison of objects for their 3- dimensional positions and dimensions with respect to a viewing position.
- Examples of object space algorithms are Back-face detection, Octree method, ray-casting method, BSP tree method etc.
- The object space algorithms are suitable for simple scenes with a small number of objects
- The advantage of these methods is that they are device independent and can work for any resolution.
- The image-space methods determine the visibility point-by-point at each pixel position on the projection plane.
- These algorithms work while the objects are being converted into pixels in the frame/refresh buffer.
- The calculations in image-space methods are done on a pixel-by-pixel basis and so the resolution of the display device plays an important role.
- Some examples are A-buffer method, Z-buffer or Depth-buffer method, Scan line, Painter's algorithm.
- Back-Face detection, also known as Plane Equation method, is an object space method in which objects and parts of objects are compared to find out the visible surfaces.
- The back face algorithm can be represented geometrically. Each polygon has several vertices. Each surface has a normal vector.
- It is based on the "inside-outside" tests.
- When an inside point is along the line of sight to the surface, the polygon must be a back face (we are inside that face and cannot see the front of it from our viewing position).
- Depth Buffer algorithm is also called Z- Buffer algorithm

Space for learners:

- It compares surface depths at each pixel position on the projection Plane. Each surface of a scene is processed separately, one point at a time across the surface.
- The depth value is measured along z- axis; due to this name of the algorithm is Z- buffer algorithm.
- A- Buffer algorithm is an extended version of the Z-Buffer algorithm with some additional information.
- A drawback of the depth-buffer method is that it can only find one visible surface at each pixel position. In A-Buffer Method more than one surface intensity can be taken into consideration at each pixel position.

Space for learners:

3.9 ANSWERS TO CHECK YOUR PROGRESS

Check your Progress I

1. directly visible
2. visible surface detection
3. object-space methods, image-space methods
4. Comparison, viewing position
5. Object space algorithms
6. Rasterized
7. coherence
8. Screen

Check your Progress II

1. Plane Equation
2. object space method
3. front face
4. Z- Buffer
5. depth-buffer method
6. z- axis
7. Refresh Buffer
8. Z- Value or depth
9. A-Buffer
10. Depth Field, Intensity Field
11. ray-casting, octree methods
12. wireframe-visibility methods

3.10 POSSIBLE QUESTIONS

1. What do you mean by visible surface detection? Why is it needed to detect the visible surfaces in computer graphics?
2. What are the different types of algorithms that are used for hidden surface removal? Explain them.
3. Explain the Back face detection algorithm.
4. Explain the Z-buffer algorithm. Write down the steps of the algorithm.
5. How is A-buffer algorithm different from the Z-buffer algorithm?

Space for learners:

3.11 REFERENCES AND SUGGESTED READINGS

- Computer Graphics By Donald Hearn and M. Pauline Baker
- Computer Graphics, Schaum's Outlines, Plastock and Kalley, McGraw-Hill © 1986
- Computer graphics: principles and practice by James D. Foley

UNIT 4: ILLUMINATION AND SURFACE RENDERING

Unit Structure:

Space for learners:

4.1 INTRODUCTION

In the previous unit, we had learned about visible surface detection. In any given scene, the colour of a point is based on different factors like its position, orientation to the light sources, surface characteristics and others. Once visible surface has been identified by the hidden surface algorithm, an illumination model is used to compute the intensities and color to display for the surface. For realistic displaying a three dimensional scene the appropriate color or intensity for that scene needs to be calculated.

In this unit, we would discuss how the object depends on the lighting that illuminates the scene, and on the interaction of light with the objects in the scene. The lighting in a scene depends on the surface properties. The different surface rendering methods would also be discussed in this unit. The shading methods make the images look more realistic.

4.2 UNIT OBJECTIVES

After going through this unit, you will be able to

- Get an overview of the fundamental activities performed in illumination
- Understand the point light source and spot light source
- Learn about ambient, diffuse and specular reflection types
- Learn about the illumination models

Space for learners:

4.3 DEFINITION AND IMPORTANCE

The human eye is like a camera. It has sensors at the back of the eye that captures the amount of light coming from the different directions. When we see an object, how it appears to us is based on the position of the object the eye is looking at, the position of the light source which emits the light and also the colour and intensity of the light that falls on that object. The emitted light may fall directly on the object or sometimes the emitted light falls upon the object after getting reflected from the surrounding surfaces.

The transport of the light from the light source to the point or the object is called illumination.

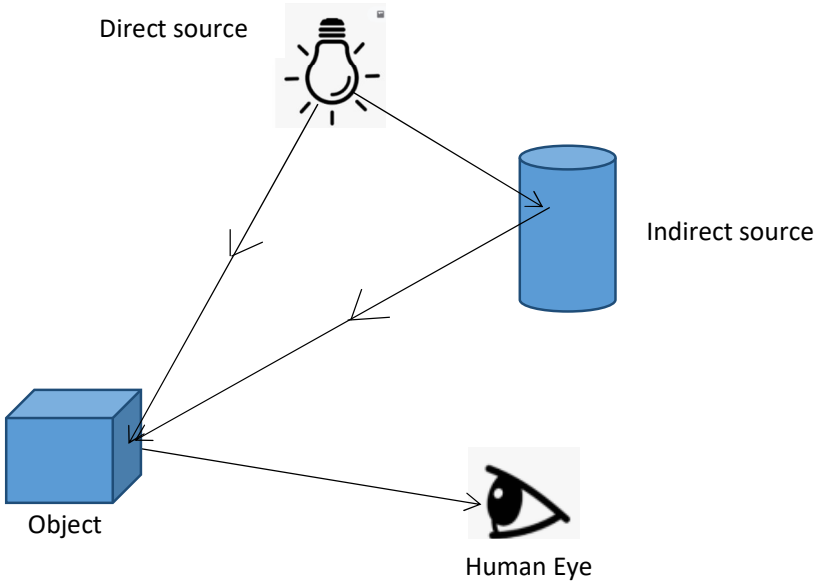


Figure: Perception of an object by the human eye

The light, as shown in the figure above, can fall directly or indirectly on the object and the incident light then gets reflected from the object and reaches the human eye. The light intensity that enters the human eye is the colour of that object.

The process of computing the luminous intensity or the colour at a point is called lightning. It is one of the most important considerations for three-dimensional graphics. The process of assigning colour to the points in an object is called shading, also known as surface rendering. In other words, the illumination model is invoked by the surface rendering method.

An illumination model is also known as a lightning model. It calculates the colour of a given point on the object's surface. The colour depends on both the source of the light as well as its surface.

Space for learners:

4.4 LIGHT SOURCES

We can see an object when light falls on it. Also, an object that is not directly exposed to a light source is visible to us if the nearby objects are illuminated. The reflected light from an object is always the sum total of the contributions from the light sources and the other reflecting surfaces in the scene. The light sources are referred to as light-emitting sources and the reflecting surfaces such as the walls of a room etc are called as the light-reflecting sources.

Light source is an object that radiated light energy. Examples are sun, lamp, bulb, fluorescent tube etc. They are also known as luminous objects. The objects that cannot emit light energy by themselves are known as Non-luminous objects. Objects like the moon that do not give out or emit light of their own are non-luminous objects. Moon reflects light from the sun. Other examples of Non luminous bodies are pen, pencil, chair, wood etc. The non-luminous object reflects the light that falls on them.

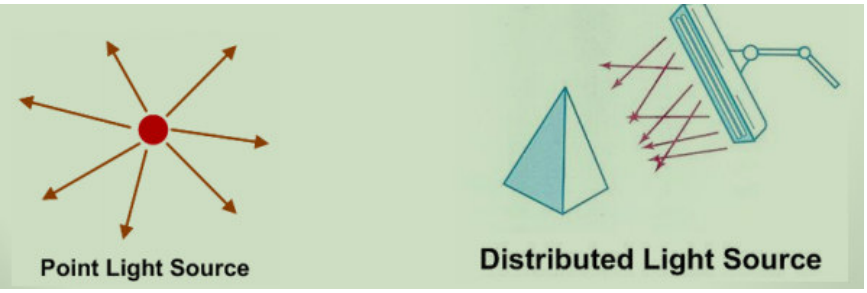
The laptop screen, mobile screen are luminous but the page of the book is a non-luminous object, and that is why we need a light on to read it.

There are some objects that behave both as light emitter as well as light reflector. For example, a coloured semitransparent plastic film covering an electric bulb reflects as well as emits light.

In graphics, light source can be specified by several properties like the colour of light emitted, shape and its position etc. There are some light sources that emits light in a certain direction whereas some light sources emits light in different directions.

The simplest light emitter is a point source. ***Point light source*** emits light in all directions as shown in the figure below. It is a light source which is smaller than the objects in the scene. An example of a point light source is a small electric bulb illuminating objects in a scene. When a point light source is used in a scene, the light on the object falls from single direction only. The light sources which are sufficiently far away from the object in the scene such as the sun can also be considered as a point light source.

Distributed light source is the one whose light emitting surface area is wide and the light source is placed near the objects in the scene. Example of distributed light source is the fluorescent tube as shown in the figure below. When distributed light source is used in a scene, light on an object falls from multiple directions and multiple points.



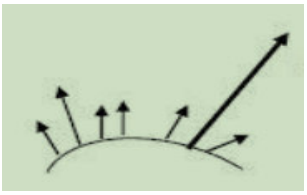
Objects are visible to us due to reflection of light. When light falls on the opaque objects, part of that light gets reflected and a part of it gets absorbed. The amount of light reflected depends on the surface material. The shiny surfaces reflect more of the incident light and the dull surfaces absorb more of the incident light. The reflection of light can be roughly categorized into two types of reflection.

Reflection from rough or grainy surfaces such as clothing, paper etc leads to a type of reflection known as ***diffuse reflection***. Whether the surface is rough or smooth has a tremendous impact upon the subsequent reflection of a beam of light. Diffuse reflection tends to reflect light in all directions as illustrated in the figure below. In diffuse reflection, the light is reflected with equal intensity in all directions. Since the intensity of the reflected light is equal in all

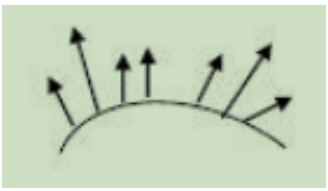
Space for learners:

directions, the objects appear equally bright from all viewing points. The amount of the radiant energy coming from any point on the surface can be calculated using the Lambert's cosine law. That is why ideal diffuse reflectors are also known as Lambertian reflectors.

Light reflected from a smooth surface at a definite angle is called ***specular reflection***. The light sources create highlights or bright spots in specular reflection. The highlighting effect is more on shiny surfaces than on dull surfaces.



Diffuse reflection



Specular reflection

4.5 BASIC ILLUMINATION MODELS

In this section, the simplified methods for calculating light intensities would be discussed. An illumination model, also known as lighting model, takes into account the various factors like surface characteristics, its position and orientation relative to the light sources illuminating it and calculates the colour of a given point on the object's surface. It basically calculates the intensity of the light reflecting from the given point that contributes ti its colour. Different optical parameters such as glossy, matte, opaque and transparent are used to set the surface properties. The illumination models follow the laws of physics for the surface lighting effects.

Illumination or lighting models are needed to care of the light, its intensity and effects on the different textures.

Ambient light

An object on which the light does not fall directly from the light source may still be visible because of the light reflections from the nearby objects. This light is known as ambient or background light. Ambient light spreads uniformly in all directions.

Space for learners:

To simulate the ambient light in a scene, the brightness level can be specified for it. In this way, all the surfaces in the scene, in all the directions, are illuminated with a constant amount of ambient light. The light reflected from the surfaces in the scene, due to this, is a diffuse reflection.

Diffuse Reflection

The diffuse light reflections are constant over the surfaces in a scene, independent of the viewing direction. The reflected light intensity, in a scene, is a sum of the intensities of the ambient light reflection and direct light reflection. The intensity of light of a point in an object, reaching the eye of a viewer can be modeled as the sum of three intensities:

1. Intensity of the ambient light after diffuse reflection from the point (I_{amb})
2. Intensity of the light from the light source after diffuse reflection from the point (I_{diff})
3. Intensity of the light from the light source after specular reflection from the point (I_{spec})

Thus, the intensity (colour) of a surface point (I_p) can be represented as:

$$I_p = I_{amb} + I_{diff} + I_{spec}$$

The incident light rays gets reflected. This reflected light intensity is a fraction of the incident light intensity. This fraction is determined by a surface property known as ***reflection coefficient*** or ***reflectivity***.

To control the light in the synthesized images, the following reflection coefficients can be defined:

i. Diffuse reflection coefficient for ambient light, k_a

ii. Diffuse reflection coefficient for direct light, k_d

iii. Specular reflection coefficient for direct light, k_s

The values of the coefficients vary from 0 to 1. The value 0 represents a dull surface with no reflection and the value 1 represents a shiny surface that reflects all of the incident light.

Space for learners:

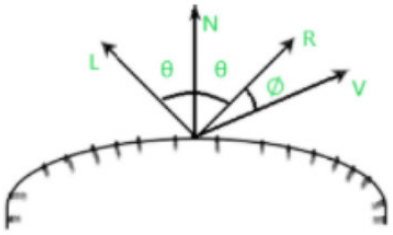
Ambient light has no spatial or directional characteristics and amount on each object is a constant for all surfaces and all directions. If a surface is exposed only to ambient light, having intensity I_a, the intensities of the diffuse reflection at any point on the surface can be calculated as:

$$I_{amb} = k_a.I_a$$

Specular Reflection and Phong Model

When light falls on shiny opaque objects, a major portion of the light gets reflected in a particular direction. This results in a bright spot, highlight or specular reflection on a portion of the object's surface. The reflection from the rest part remains diffuse. Specular reflection can be observed on exposing an object with a shiny surface, for example, a polished metal or an apple to white light. The bright spot appearing on the apple's surface is the effect of specular reflection. It is to be noticed that the colour of the bright spot on the surface of the apple appears to be white rather than the red colour of the apple.

An empirical model for calculating the specular reflection range, invented by the Phong Bui Tuong is also known as Phong specular reflection model. The figure below illustrates the specular reflection direction R from a point on the object's surface.



The vector L represents the point light source illuminating the object's surface. The vectors L and R forms the same angle θ with the surface normal N. R is representing the unit vector in the direction of ideal specular reflection. V is the viewing direction. Ø is the viewing angle relative to the specular reflection direction R.

In case of ideal reflector surfaces, example, perfect mirror, the incident light is reflected only in the specular-reflection direction. Thus, the reflected light could be seen when the vectors V & R coincides, that is, viewing angle Ø=0.

Space for learners:

The Phong specular reflection model sets the intensity of specular reflection directly proportional to the $\cos^{n}s(\emptyset)$. The range of angle Ø can lie between $0 \leq \emptyset \leq 1$.

^{n}s is a specular reflection parameter whose value is determined by the type of surface to be displayed. The value of ^{n}s for shiny surfaces could be 100 or more whereas for dull surfaces its value is 1 or less than 1.

The intensity of the specular reflection depends on the angle of incidence θ and the surface material properties. To model the specular intensity variation, specular reflection coefficient W(θ) can be used. This coefficient tends to increase as the angle of incidence increases. Using W(θ), the Phong specular reflection model can be expressed by the following equation:

$$I_{spec} = w(\theta) I_l \cos^{n_s} \phi$$

Where I_l is the intensity of the light source and Ø is the viewing angle relative to specular reflection direction R. At $\theta = 90^0$, $W(\theta) = 1$, thus all incident light is reflected.

CHECK YOUR PROGRESS - I

1. The transport of the light from the light source to the point or the object is called _____.

2. The light intensity that enters the human eye is the _____ of that object.

3. The process of computing the luminous intensity or the colour at a point is called _____.

4. The process of assigning colour to the points in an object is called _____.

5. The reflected light from an object is always the sum total of the contributions from the _____ and the other reflecting surfaces_____ in the scene.

6. Light source is an object that radiated _____ energy.

7. Point light source emits light in _____ directions.

Space for learners:

8. Reflection from rough or grainy surfaces such as clothing, paper etc leads to a type of reflection known as _____.

9. The light sources create highlights or bright spots in _____ reflection.

10. Illumination models are needed to care of the light, its _____ and effects on the different _____.

11. The reflected light intensity is a sum of the intensities of the _____ light reflection and _____- light reflection.

12. When light falls on shiny opaque objects, a major portion of the light gets reflected in a particular direction in _____ reflection.

Space for learners:

4.6 SURFACE RENDERING METHODS

Surface rendering is applying a lighting model to obtain pixel intensities for all the surface positions in a scene. Surface rendering method is used to generate a degree of realism in a displayed scene. Realism is attained in displays by setting the surface intensity of objects according to the lighting conditions in the scene and surface characteristics. Lighting conditions include the intensity and positions of light sources and the background illumination. An illumination model can be applied to graphics formed with polygon surfaces. An illumination model shines a light on a surface and makes calculations based on the intensity of the light reflected back. Shading models are needed to take care of the type of surface - its shape, texture etc and rendering the intensity at every point of the surface.

Illumination models are used to calculate the amount of light reflected from a certain position on the surface. They thus provide ways to compute the colour on a pixel of the surface.

Applying the illumination model at each surface point is computationally expensive. It involves a lot of operations. As such, generating an image using computer graphics is expensive in terms of resources like processor, memory, time etc. In computer animations, games etc, the screen image requires to be changed frequently. This makes the process even more complicated as the

computations have to be carried out repeatedly and within a short span of time.

The surface rendering methods or the shading models can greatly reduce the number of computations in image generation. The shading models allows to assign colour to the screen pixels without performing the lighting calculations for all the points to be rendered. The colour of few points is only calculated using the illumination models. Those colour are then used to interpolate colour at the other surface points on the screen pixels.

The differences between the coloring of an image using the illumination or lighting models and the shading or rendering models.

1. The lighting models use a large number of operations and are thus expensive. On the other hand, shading model uses interpolation approach.

2. Lighting models are applied on the scene. In contrast, the rendering/shading models work at the pixel -level.

Every point on the surface is unique. The question in shading arises that should we calculate the intensity of light at all points of the object? If the surface is defined as a mesh of polygons, which point should be used?

There are different answers to this question, each with different implications for the visual quality of the result.

FLAT SHADING

Flat shading is the simplest of the shading models. It involves the least number of computations. The model computes the colour of any one point on the surface using the lighting model. The colour obtained is then assigned to all the points on the surface which are mapped to the screen pixels.

This method gives an accurate rendering provided the following assumptions are true.

1. All the light sources are sufficiently far from the surface.

2. The viewer is at infinity so that V.R is constant across the polygon surface. Here, V is the view-point vector looking at the surface and R is the specular-reflection vector.

Space for learners:

3. The polygon represents the actual surface being modelled and is not an approximation of any object with a curved surface.

However, even if these assumptions are wrong, flat shading method gives an approximate surface lighting effect. The main advantage of this method is that it is quick and simple. However, it cannot produce the variations in shade across the polygon. It can lead to unrealistic images.

GOURAUD SHADING

Gouraud shading is an intensity interpolation method which was developed by Henri Gouraud in 1971. In this method, with a little increase in computation, the surfaces can be coloured more realistically. The method renders the polygon surface by first applying the illumination model on a subset of surface points and then interpolating the intensity of the remaining points on that surface. It is used to produce continuous shading of surfaces.

The Gouraud shading woks as follows:

1. The unit normal vector at each vertex of the polygonal surface is determined by averaging the surface normal vectors of all polygons meeting at that vertex.

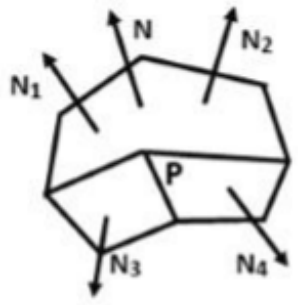


Figure : Surface normals of polygons

For any vertex position V, the unit vertex normal can be obtained with the calculation

$$N_v = \frac{N_1 + N_2 + N_3 + N_4}{|N_1 + N_2 + N_3 + N_4|}$$

Space for learners:

More generally, it can be written as,

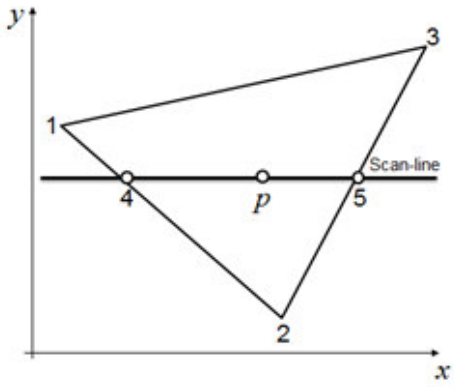
$$N_v = \frac{\sum_{i=1}^{n} N_i}{\sum_{i=1}^{n} N_i}$$

2. Apply illumination model at each vertex to compute the colour at that position.

3. The colour intensities of the vertices are linearly interpolated over the projected area of the polygon.

Intensity values for each polygon are matched with the values of adjacent polygons along the common edges thereby eliminating the intensity discontinuities that can occur in flat shading.

The illumination values are linearly interpolated across each scan line as shown in the figure below:



- The intensities at point 4 can be interpolated from the intensities of points 1 and 2
- The intensities at point 5 can be interpolated from the intensities of points 2 and 3
- Thus, the intensities of the points 4 and 5 can be calculated from the scan line.

The advantage of Gouraud shading is that it removes the intensity discontinuities that exists in the flat shading by matching the intensity values of each polygon with the values of the adjacent polygons along the common edges. This adds a curved feel to a polygon that would otherwise appear to be flat. However, it has a problem with specular reflection. The highlights are not rendered

Space for learners:

correctly. It can introduce anomalies known as *Mach bands*, in which the bright and dark intensity streaks appear on the surface due to linear intensity interpolation. Objects shaded with Gouraud shading often appears to be dull and chalky as it lacks the accurate specular component.

PHONG SHADING

Phong shading is also known as normal vector interpolation shading. This is a more accurate interpolation based approach for rendering a polygon that was developed by Phong Bui Tuong. Unlike Gouraud shading, it interpolates normal vectors instead of interpolating the intensity values. Here the illumination model is applied to each surface point. Phong shading accepts the same input as Gouraud shading but it produces very smooth looking results.

To render a polygon Phong surface rendering proceeds as the following steps:

1. Determine the average unit normal vector at each vertex of the polygon, as done in the Gouraud shading method.
2. The vertex normal is then linearly interpolated over the surface of the polygon.
3. Apply an illumination model at the positions along the scan lines to calculate the pixel intensities using the interpolated normal vectors.

For example, for the polygon surface shown in the figure below, the normal vector N at point P can be obtained by interpolating the normal vectors N_1 and N_2 respectively as follows.

$$N = \frac{y - y_2}{y_1 - y_2} N_1 + \frac{y_1 - y}{y_1 - y_2} N_2$$

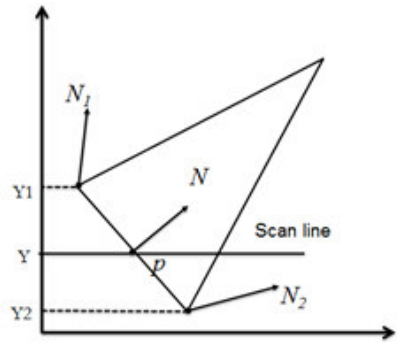


Figure: Interpolation of surface normal along a polygon edge

Space for learners:

The advantages of Phong shading are as follows:

i. It displays more realistic highlights on a surface.

ii. It greatly reduces the Mach band effect.

iii. It gives more accurate results.

However, there is a disadvantage too:

It requires more calculations and greatly increases the cost of shading steeply. Thus, the processing time is more as compared to the other techniques.

The three methods of surface rendering can be explained in the figure shown below:

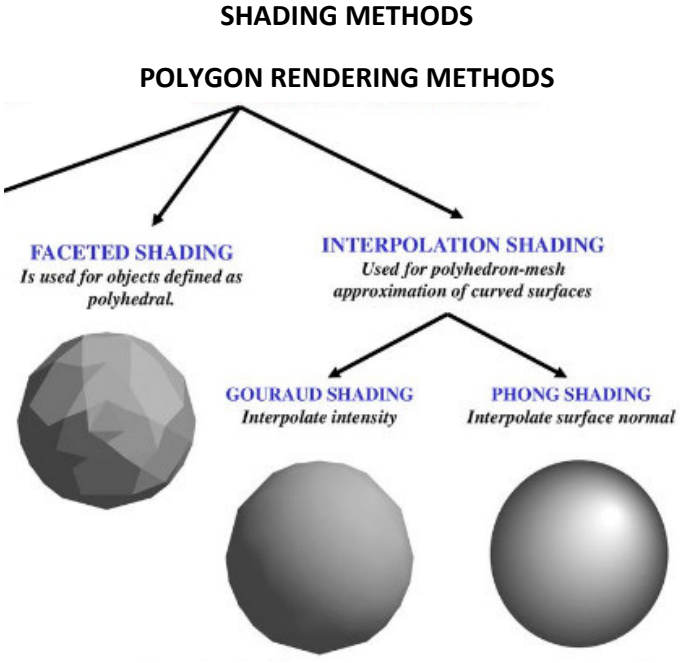


Figure: Different types of shading methods

The shading models can be summarized as follows:

- For realistic rendering of polygons, interpolation methods are needed to determine lighting positions.
- Flat shading is fast but it is unrealistic.
- Gouraud shading is better but it does not handle the specular reflections very well.
- Phong shading is better still, but it can be slow.

Space for learners:

CHECK YOUR PROGRESS - II

1. Surface rendering method is used to _____ a degree of realism in a displayed scene.
2. Lighting conditions include the intensity and positions of _____ and the _____ illumination.
3. An illumination model can be applied to graphics formed with _____ surfaces.
4. The surface rendering methods or the shading models reduces the number of computations in image _____.
5. The lighting models uses a large number of operations and is thus _____.
6. The shading models work at the _____ level.
7. Flat shading model assumes that all the light sources are sufficiently _____ from the surface.
8. Gouraud shading can introduce anomalies known as _____.
9. Phong shading is also known as_____ interpolation shading.
10. Phong shading interpolates normal vectors instead of interpolating the _____ values.

Space for learners:

4.7 SUMMING UP

- The human eye has sensors that capture the amount of light coming from the different directions.
- The transport of the light from the light source to the point or the object is called illumination.
- The light, as shown in the figure above, can fall directly or indirectly on the object and the incident light then gets reflected from the object and reaches the human eye.
- The light intensity that enters the human eye is the colour of that object.
- The process of computing the luminous intensity or the colour at a point is called lightning.

- The process of assigning colour to the points in an object is called shading, also known as surface rendering.
- Light source is an object that radiated light energy. Examples are sun, lamp, bulb, fluorescent tube etc. They are also known as luminous objects.
- The objects that cannot emit light energy by themselves are known as Non- luminous objects. The non luminous object reflects the light that falls on them.
- The simplest light emitter is a point source. ***Point light source*** emits light in all directions.
- Distributed light source is the one whose light emitting surface area is wide and the light source is placed near the objects in the scene.
- Objects are visible to us due to reflection of light
- When light falls on the opaque objects, part of that light gets reflected and a part of it gets absorbed. The amount of light reflected depends on the surface material.
- Reflection from rough or grainy surfaces such as clothing, paper etc leads to a type of reflection known as diffuse reflection.
- In diffuse reflection, the light is reflected with equal intensity in all directions.
- Light reflected from a smooth surface at a definite angle is calledspecular reflection. The light sources create highlights or bright spots in specular reflection. The highlighting effect is more on shiny surfaces than on dull surfaces.
- An illumination model, also known as lighting model, takes into account the various factors like surface characteristics, its position and orientation relative to the light sources illuminating it and calculates the colour of a given point on the object's surface.
- An object on which the light does not fall directly from the light source may still be visible because of the light reflections from the nearby objects. This light is known as ambient or background light.

Space for learners:

- The diffuse light reflections are constant over the surfaces in a scene, independent of the viewing direction.
- When light falls on shiny opaque objects, a major portion of the light gets reflected in a particular direction. This results in a bright spot, highlight or specular reflection on a portion of the object's surface.
- In case of ideal reflector surfaces, example, such as a perfect mirror, the incident light is reflected only in the specular-reflection direction
- The Phong specular reflection model sets the intensity of specular reflection directly proportional to the $\cos^n s(Ø)$. The range of angle Ø can lie between $0 \leq Ø \leq 1$.
- Surface rendering is applying a lighting model to obtain pixel intensities for all the surface positions in a scene.
- Realism is attained in displays by setting the surface intensity of objects according to the lighting conditions in the scene and surface characteristics.
- Applying the illumination model at each surface point is computationally expensive as it involves a lot of operations. The surface rendering methods or the shading models can greatly reduce the number of computations in image generation
- Flat shading is the simplest of the shading models. It involves the least number of computations
- Flat shading computes the colour of any one point on the surface using the lighting model. The colour obtained is then assigned to all the points on the surface which are mapped to the screen pixels.
- In Gouraud shading, the surfaces can be coloured more realistically. It renders the polygon surface by first applying the illumination model on a subset of surface points and then interpolating the intensity of the remaining points on that surface.
- Phong shading is also known as normal vector interpolation shading. It interpolates normal vectors instead of interpolating the intensity values. It greatly reduces the Mach band effect.

Space for learners:

4.8 ANSWERS TO CHECK YOUR PROGRESS

Check your Progress I

1. Illumination
2. Colour
3. Lightning
4. Shading
5. light sources, light sources
6. Light
7. All
8. diffuse reflection
9. Specular
10. Intensity, textures
11. Ambient, direct
12. Specular

Check your Progress II

1. generate
2. light sources, background
3. polygon
4. Generation
5. Expensive
6. Pixel
7. Far
8. Mach bands
9. normal vector
10. intensity

4.9 POSSIBLE QUESTIONS

1. What are light sources?
2. What is diffuse and specular reflection?
3. What do you mean by illumination? Discuss the different illumination models.
4. Explain the surface rendering techniques that are used to produce images?

Space for learners:

4.10 REFERENCES AND SUGGESTED READINGS

- Computer Graphics By Donald Hearn and M. Pauline Baker
- Computer Graphics, Schaum's Outlines, Plastock and Kalley, McGraw-Hill © 1986
- Computer graphics: principles and practice by James D. Foley

Space for learners:

UNIT 5: COLOR MODELS AND APPLICATIONS

Unit Structure:

Space for learners:

5.1 INTRODUCTION

The way of describing a color with numerical value is termed a color model. Color models are used for processing images. Image properties like hue, brightness, saturation, etc. are discussed in color models. We can generate a different range of colors from the set of primary colors like Red, Green, Blue, Cyan, Yellow, etc. RGB color model is one of the most popular color models used in TV, Computer monitors.

5.2 UNIT OBJECTIVES

This unit is an attempt to give an idea of different color models. After going through this unit you will be able to-

- *discuss* the various light properties
- *explain* the application of color models
- *discuss* the various color models and conversion between color models

Space for learners:

5.3 PROPERTIES OF LIGHT

Light consists of multiple components like diffuse, ambient, and specular. It is an electromagnetic wave. The human eye responds to wavelengths from about 380 to 750 nanometers. And the frequency value from 4.3x 10^{14} hertz (red) to 7.5x10^{14} hertz (violet) is the visible range.

A few of the light properties are given below:

i) ***Brightness***: It is the total light energy.

ii) ***Saturation***: It defines a range from pure color to gray at a constant lightness level. A pure color is fully saturated.

iii) ***Chromaticity***: It combines purity and dominant frequency.

iv) ***Hue***: It defines pure color in terms of "green", "red" or "magenta". It also defines mixtures of two pure colors like "red-yellow" or "yellow-green".

5.4 STANDARD PRELIMINARIES-XYZ MODEL, CIE CHROMATICITY DIAGRAM

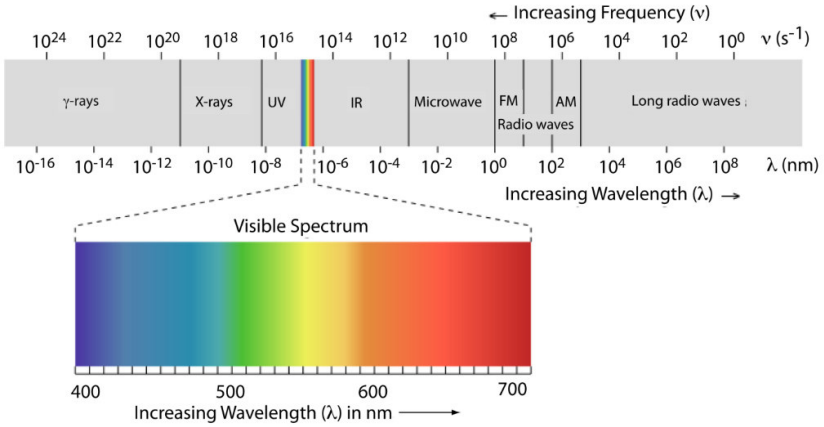


Fig. 5.1: Electromagnetic Spectrum

What we perceive as a color seems to depend on characteristics of brightness, hue, and saturation. We generally regard the basic colors like Red, Green, and Blue, and define other colors as a mix of these three, the amount of each basic color being specified by the values of X, Y, and Z, respectively. Color is then specified by its trichromatic coefficients:

$$x = \frac{X}{X+Y+Z}$$
$$y = \frac{Y}{X+Y+Z}$$
$$z = \frac{Z}{X+Y+Z}$$

And, $x + y + z = 1$

Another approach to specifying color is to use the CIE chromaticity diagram

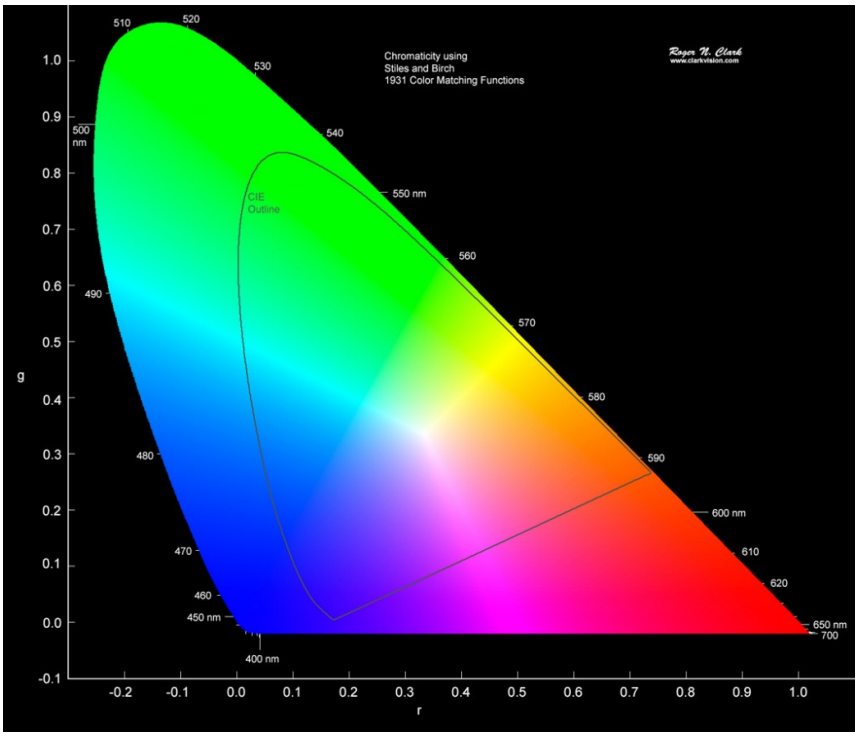


Fig. 5.2: CIE chromaticity diagram

This gives color composition as a function of x (red) and y (green), with the amount of z being determined from z = 1 - (x + y). Points around the border of the diagram are considered fully saturated,

Space for learners:

while the saturation goes to zero as one moves on a straight line from a boundary point to the equal energy point which represents white. A straight line joining any two points represents all the colors that are possible by combining those two colors in varying amounts, while the triangle enclosing three non-collinear points represents all the colors that can be produced by combinations of the three colors represented by the triangle corners.

STOP TO CONSIDER

Basic colors as red, green, and blue, and define other colors as a mix of these three.

5.5 COLOR MODELS

The color model is a 3D color coordinate system to produce all ranges of color through the primary color set. It is also known as color space. The light displays color. A Color model is a hierarchical system in which we can create every color by using RGB (Red, Green, Blue, and CMYK (Cyan, Magenta, Yellow, Black) models. The color model is divided into the Additive (RGB) and Subtractive (CMYK) color models.

a) Additive color model:

The Additive color model uses a mixture of light to display colors. The perceived color depends on the transmission of light. It is used in digital media.

For Example– Computer Monitor, Television, etc.

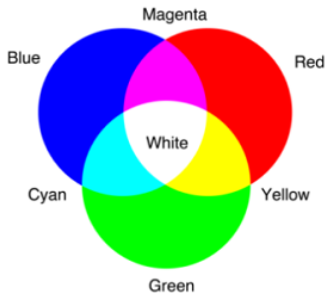


Fig. 5.3: Additive color

b) **Subtractive Color Model:**

Space for learners:

The Subtractive model uses a reflection of light to display the colors. The perceived color depends on the reflection of light.

For Example– Paint, Pigments, color filter, etc.

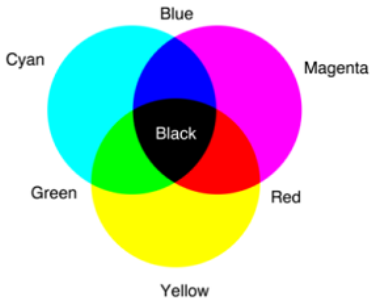


Fig. 5.4: Subtractive color

CHECK YOUR PROGRESS

1. _____ is the total light energy.
2. A range from pure color to gray at a constant lightness level is _____.
3. _____ combines purity and dominant frequency.
4. Color model is also known as_____.
5. The color model is divided into the _____and _____color models.
6. Additive color model uses a _____ to display colors.
7. The Subtractive model uses a _____to display the colors.

The RGB color model is one of the most widely used color representation models. It uses a color coordinate system with R(red), G(green), and B(blue). These are the three primary colors of the RGB model.

Space for learners:

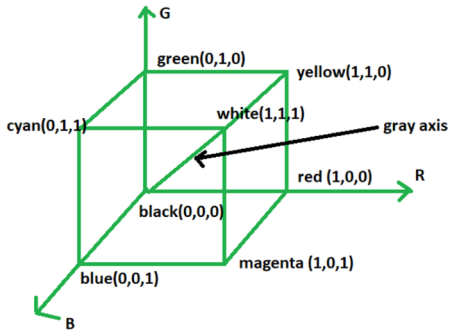


Fig. 5.5: RGB Color Model[1]

The corner of the RGB color cube that is at the origin of the coordinate system corresponds to black, whereas the corner of the cube that is diagonally opposite to the origin represents white. The diagonal line connecting black and white corresponds to all the gray colors between black and white, which is also known as the gray axis.

In the RGB color model, an arbitrary color within the cubic color space can be specified by its color coordinates: (r, g, b).

For example:

(0, 0, 0) for black, (1, 1, 1) for white,

(1, 1, 0) for yellow, (0.7, 0.7, 0.7) for gray

Color specification using the RGB model is an additive process. We begin with black and add on the appropriate primary components to yield the desired color. The concept RGB color model is used in the Display monitor.

Properties:

- This is an additive color model.
- The colors are added to the black.
- 3 main colors: Red, Green, and Blue.

5.5.2 YIQ

This model was designed to separate chrominance from luminance. This was a requirement in the early days of color television when black-and-white sets still were expected to pick up and display what

Space for learners:

were originally color pictures. Y stands for luminance part and IQ stands for chrominance part. A color television set would take these three channels, Y, I, and Q, and map the information back to R, G, and B levels for display on a screen. This is used for color TV. Here **Y** is the luminance.

The advantage of this model is that more bandwidth can be assigned to the Y-component (luminance) to which the human eye is more sensible than to color information. So for NTSC TV there are 4MHz assigned to ***Y*** , **1.5**MHz to ***I*** , and **0.6**MHz to ***Q***.

Space for learners:

5.5.3 CMY

The CMY color model uses a **subtraction process** and this concept is used in the **printer**. In CMY model, we begin with white and take away the appropriate primary components to yield the desired color.'

For example:
If we subtract red from white, what remains consists of green and blue which is cyan. The coordinate system of the CMY model uses the three primaries' complementary colors: C(cyan), M(magenta), and Y(yellow)

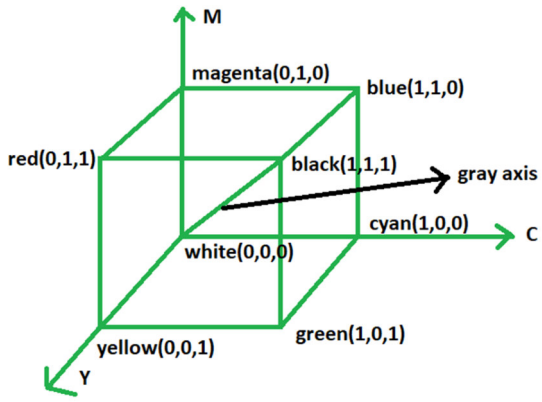


Fig. 5.6: CMY Color Model[1]

The corner of the CMY color cube that is at (0, 0, 0) corresponds to white, whereas the corner of the cube that is at (1, 1, 1) represents black. The following formulas summarize the conversion between the two color models:

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

Each primary color can take an intensity value ranging from 0(lowest) to 1(highest). Mixing these three primary colors at different intensity levels produces a variety of colors. The collection of all the colors obtained by such a linear combination of red, green, and blue forms the cube-shaped RGB color space.

As the RGB color model is additive, the CMY and the CMYK color model are based on the subtractive mixture of color. The characters stand for the components cyan (C), magenta (M), yellow (Y), and key (K, black).

5.5.4 HSV

All color models treated so far are hardware-oriented. The Hue-Saturation-Value model is oriented towards the user/artist. The allowed coordinates fill a six-sided pyramid with the 3 top faces of the color cube as a base. Note that at the same height colors of different perceived brightness are positioned. Value is given by the height, saturation is coded in the distance from the axes, and hue by the position on the boundary.

In the HSV color model, a color is defined by its hue (H), its saturation (S), and its lightness or blackness value (V) and so, it resembles the human color perception more than the additive and the subtractive color models. It is easy to adjust the color by its saturation and brightness.

Because of these advantages, the color selection using the HSV color space is used, for example, in many common graphics programs. The standard color selection dialog, for example from the Windows operating system, is also based on the HSV color model: There is a color field in which the color can be selected and arranged according to hue and saturation, as well as an additional controller for the brightness from white to black, with which the selected color can be adjusted.

Space for learners:

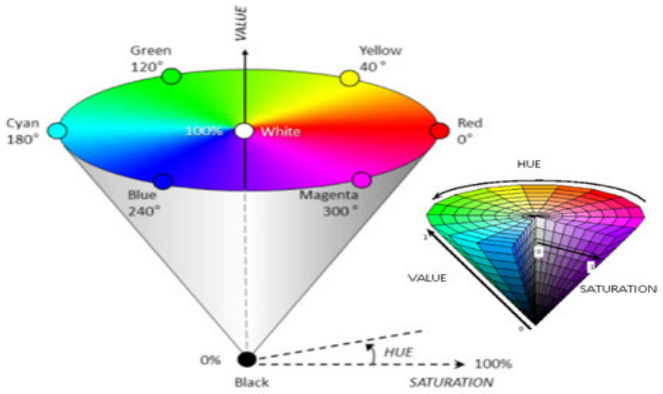


Fig. 5.7: HSV Color Model

5.5.5 HLS

HLS color model defines colors by the three parameters hue (H), lightness (L), and saturation (S). It was introduced by Tektronix Inc. Here the RGB cube is deformed in such a way that a six-sided double pyramid results with the same base as in the HSV model, but with two tips in black and white.

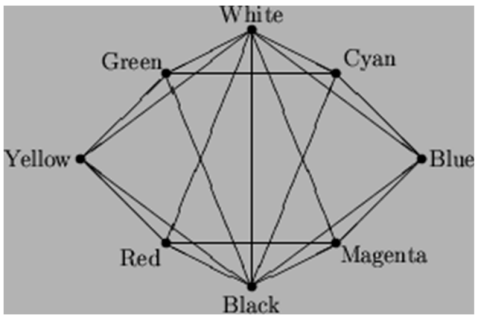


Fig. 5.8: HLS Color Model

CHECK YOUR PROGRESS

8. RGB is an_____.
9. In YIQ model, Y stands for _____ part and IQ stands for _____ part.
10. The CMY color model uses a _____ process.

Space for learners:

5.6 CONVERSION BETWEEN COLOR MODELS

Color models can be converted into one another. Different formulae are used to make conversions among the color model.

i) RGB and YIQ

a) Conversion of RGB to YIQ:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

b) Conversion of YIQ to RGB:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0.956 & 0.621 \\ 1 & -0.272 & -0.647 \\ 1 & -1.106 & 1.703 \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix}$$

ii) RGB and CMYK

a) Conversion of RGB to CMYK:

The R, G, and B values are divided by 255 to change the range from 0...255 to 0...1:

R = R/255

G = G/255

B = B/255

The black key (K) color is calculated from the red (R), green (G), and blue (B) colors:

K = 1-max(R, G, B)

The cyan color (C) is calculated from the red (R) and black (K) colors:

C = (1-R-K) / (1-K)

The magenta color (M) is calculated from the green (G) and black (K) colors:

M = (1-G-K) / (1-K)

Space for learners:

The yellow color (Y) is calculated from the blue (B) and black (K) colors:

Y = (1-B-K) / (1-K)

b) Conversion of CMYK to RGB:

The R, G, and B values are given in the range of 0….255.

The red (R) color is calculated from the cyan (C) and black (K) colors:

R = 255 × (1-C) × (1-K)

The green color (G) is calculated from the magenta (M) and black (K) colors:

G = 255 × (1-M) × (1-K)

The blue color (B) is calculated from the yellow (Y) and black (K) colors:

B = 255 × (1-Y) × (1-K)

iii) RGB and HSV

a) Conversion of RGB to HSV

The R,G,B values are divided by 255 to change the range from 0..255 to 0..1:

R = R/255
G = G/255
B = B/255
Cmax = max(R, G, B)
Cmin = min(R, G, B)
Δ = Cmax - Cmin

Hue calculation:

$$H = \begin{cases} 0^\circ & \Delta = 0 \\ 60^\circ \times \left(\frac{G-B}{\Delta} mod 6\right) & , C_{max} = R \\ 60^\circ \times \left(\frac{B-R}{\Delta} + 2\right) & , C_{max} = G \\ 60^\circ \times \left(\frac{R-G}{\Delta} + 4\right) & , C_{max} = B \end{cases}$$

Saturation calculation:

Space for learners:

$$S = \begin{cases} 0 & , C_{max} = 0 \\ \frac{\Delta}{C_{max}} & , C_{max} \neq 0 \end{cases}$$

Value calculation:

$$V = Cmax$$

b) Conversion of HSV to RGB:

When $0 \leq H < 360$, $0 \leq S \leq 1$ and $0 \leq V \leq 1$:

C = V × S

X = C × (1 - |(H / 60°) mod 2 - 1|)

m = V - C

$$(R, G, B) = \begin{cases} (C, X, 0) & , 0° \leq H < 60° \\ (X, C, 0) & , 60° \leq H < 120° \\ (0, C, X) & , 120° \leq H < 180° \\ (0, X, C) & , 180° \leq H < 240° \\ (X, 0, C) & , 240° \leq H < 300° \\ (C, 0, X) & , 300° \leq H < 360° \end{cases}$$

(R,G,B) = ((R+m)×255, (G+m)×255, (B+m)×255)

iv) RGB and HLS

a) Conversion of RGB to HLS

The R, G, and B values are divided by 255 to change the range from 0..255 to 0..1:

R = R/255

G = G/255

B = B/255

Cmax = max(R, G, B)

Cmin = min(R, G, B)

Δ = Cmax – Cmin

Space for learners:

Space for learners:

Hue calculation:

$$H = \begin{cases} 0^\circ & \Delta = 0 \\ 60^\circ \times \left(\frac{G-B}{\Delta} mod 6\right) & , C_{max} = R \\ 60^\circ \times \left(\frac{B-R}{\Delta} + 2\right) & , C_{max} = G \\ 60^\circ \times \left(\frac{R-G}{\Delta} + 4\right) & , C_{max} = B \end{cases}$$

Lightness calculation:

L = (Cmax + Cmin) / 2

Saturation calculation:

$$S = \begin{cases} 0 & , \Delta = 0 \\ \frac{\Delta}{1-|2L-1|} & , \Delta <> 0 \end{cases}$$

b) Conversion of HLS to RGB

When $0 \leq H < 360$, $0 \leq S \leq 1$ and $0 \leq L \leq 1$:

C = (1 - |2L - 1|) × S

X = C × (1 - |(H / 60°) mod 2 - 1|)

m = L - C/2

$$(R, G, B) = \begin{cases} (C, X, 0) & , 0^\circ \leq H < 60^\circ \\ (X, C, 0) & , 60^\circ \leq H < 120^\circ \\ (0, C, X) & , 120^\circ \leq H < 180^\circ \\ (0, X, C) & , 180^\circ \leq H < 240^\circ \\ (X, 0, C) & , 240^\circ \leq H < 300^\circ \\ (C, 0, X) & , 300^\circ \leq H < 360^\circ \end{cases}$$

(R,G,B) = ((R+m)×255, (G+m)×255,(B+m)×255)

5.7SUMMING UP

- A color model is a mathematical model describing the way colors can be represented as tuples of numbers

- The RGB color model is an additive color model in which the red, green, and blue primary colors of light are added together in various ways to reproduce a broad array of colors.
- The CMY color model is a subtractive color model in which cyan, magenta, and yellow pigments or dyes are added together in various ways to reproduce a broad array of colors.
- The CMYK color model (also known as process color, or four colors) is a subtractive color model, based on the CMY color model, used in color printing, and is also used to describe the printing process itself. CMYK refers to the four ink plates used in some color printing: cyan, magenta, yellow, and key (black).
- HSL (for hue, saturation, lightness) and HSV (for hue, saturation, value; also known as HSB, for hue, saturation, brightness) are alternative representations of the RGB color model.
- In the YIQ model, Y stands for luminance part and IQ stands for chrominance part.

Space for learners:

5.8 ANSWER TO CHECK YOUR PROGRESS

1. Brightness
2. Saturation
3. Chromaticity
4. color space
5. Additive (RGB), Subtractive (CMYK)
6. mixture of light
7. reflection of light
8. additive color model
9. luminance, chrominance
10. subtraction

5.9 MODEL QUESTIONS

1. Define light.
2. What is chromaticity

3. Explain the properties of light.
4. What is a color model?
5. Write the difference between RGB and HSV color models.
6. Write the conversion formula of RGB to HSV.
7. What are the color models in computer graphics?
8. What is the use of a color model?
9. Why RGB model is an additive color model?
10. How can YIQ color model be converted to RGB color model?
11. How can HLS color model be converted to RGB color model?
12. How can CMYK color model is converted to RGB color model?

Space for learners:

5.10 REFERENCES AND SUGGESTED READINGS

- www.geeksforgeeks.org
- https://academic.mu.edu/phys/matthysd/web226/L0221.htm
- https://www.mat.univie.ac.at/~kriegl/Skripten/CG/node14.html
- https://www.mat.univie.ac.at/~kriegl/Skripten/CG/node15.html
- https://www.geeksforgeeks.org/difference-between-rgb-cmyk-hsv-and-yiq-color-models/
- https://www.rapidtables.com/convert/color/rgb-to-cmyk.html
- https://www.mat.univie.ac.at/~kriegl/Skripten/CG/node16.html
- www.easyengineeringclasses.com

UNIT 6: MULTIMEDIA SYSTEM

Unit Structure:

Space for learners:

6.1 INTRODUCTION

Multimedia is a technology where different components like audio, video, text, animation and graphics are combined together to form multimedia data or information. **Multi** means more than one; **Medium** means intermediary and **Media** means conveying the information. That means if a system presents information using more than one media then it is termed as multimedia. A multimedia system is capable of processing multimedia data.

6.2 UNIT OBJECTIVES

This unit is an attempt to give an idea of multimedia system. After going through this unit you will be able to-

- *explain* the basic concepts of multimedia and its properties
- *discuss* the various multimedia technologies
- *explain* the concept of different multimedia systems
- *discuss* about multimedia technology

6.3 REVIEW OF TYPICAL INTERACTIVE MULTIMEDIA SYSTEM

Interactive multimedia system refers to the technology through which user can easily navigate any site/ application for acquiring information or services. Here the user can directly communicate with the system. For example websites, mobile telephony, e-learning, gaming etc. Due to the COVID pandemic, interactive multimedia is very helpful for e-learning. For example, now days teachers take classes online through various video conferencing apps like Google Meet, Skype, Zoom etc. basically through interactive multimedia system users can share information with one another or system virtually.

6.3.1 Application of Interactive Multimedia

i) Software:

There is lots of video editing software that use large amount of data to make the video with effect, text etc. This basically leads to interaction of the user with the software.

ii) Medical:

In medical, for management purpose, purchasing medicine different software is used. Also machines used for medical purpose are also interactive multimedia system.

iii) Games:

Computer games, web games or mobile games are very popular among the young generation. Basically gaming allows the user to play in a virtual reality with the help of multimedia content.

Space for learners:

Space for learners:

iv) Industry:

In industry software is used to record data. Machines used in industry follow artificial intelligence.

v) Entertainment:

Online web series and movies are easily available in YouTube, digital platform, where we can easily enjoy by sitting in our home. This also includes multimedia interaction.

vi) Online training/ teaching :

Online training is very much efficient using the interactive multimedia systems. There are many online sites available from which one can lean so many things according to his / her desire. Due to the COVID pandemic, education system depends on the online teaching learning approach. Here both the educators and learners are allowed to take part in e-learning processing using different software available online easily. The users can give input to the online courses through mouse click, touching an input screen, live interaction etc. online evaluation is also possible due to the multimedia software.

vii) Engineering:

In today's technological world, in every aspect of engineering depends on the software. This involves interactive multimedia

6.3.2 Types of Interactive Multimedia System

1. **Menu driven programs/ presentations:**

   Sub menus are there to navigate through a particular app. For example, e-commerce site. Here different sections are available through which we can navigate.

2. **Hypermedia:** there are multiple links that lead us through the app.

3. **Non-linear, quick access to information**: these involve quick navigation.

4. **Simulations**: Virtual environment is present here. For example game, flight simulator

Space for learners:

6.3.3 Advantages and Disadvantages of Interactive Multimedia System

Advantages:

i) Make user active, communicate virtually and share information in secure manner
ii) Provides easy accessibility of daily resources
iii) Allows user to combine, manipulate, control various media types like text, audio, video etc.

Disadvantages:

i) Intrusive: may mislead the data or you will be redirected to situation which you do not want.
ii) Practical field knowledge is necessary in learning. It is missing in e-learning which is a very good example of interactive learning system.

6.4 MULTIMEDIA SYSTEM

As we already know that multimedia is a combination of 5 major components or elements like audio, video, text, graphics and animation. And multimedia system is responsible for developing multimedia application which uses multiple media like text, audio, video, animation and graphics. Multimedia system can create, manipulate, delete, store more than one media digitally. For example a video is one of the major elements of multimedia which is a collection of multiple images.

6.4.1 Characteristics of Multimedia System

i) High processing power:
As multimedia system uses very large amount of data so processing speed needs to be very high.

ii) Multimedia capable file system:

The file system should satisfy the requirements for multimedia data. The multimedia data requires high disk bandwidth rates so the disk schedulers must reduce latency time to increase the disk bandwidth.

iii) File formats that support multimedia:
Multimedia data can have variety of file formats like JPEG, MPEG, AVI, GIF, PNG, DOC etc.

iv) Input/Output:
As multimedia system deals with large amount of multimedia data so the input and output should be fast enough to do real time recordings.

v) Operating system:
As multimedia applications consists interactive environment so the operating system should provide quick response time.

vi) Storage and Memory:
Multimedia systems require large storage and memory capacity to store it consists different elements.

vii) Network support:
Now a days from streaming video in internet to playing games in virtual reality, there is a tremendous need of internet, intranet, LAN and WAN, ATM and other.

viii) Software Tools:
For developing multimedia systems we need various graphics software, multimedia editing software etc. including drivers for multimedia peripherals.

Space for learners:

6.4.2 Multimedia System Components

Multimedia system needs multimedia application for user interaction with the system. The multimedia application must be capable of handling the multimedia files. Also a good graphical user interface is required. The operating system helps the multimedia application to provide quick response. As the multimedia system deals with very large amount of media files so the processing speed needs to be very high for quick input / output response and also needs large amount

of storage capacity for the media data. To support the other necessary multimedia devices, drivers need to be installed.

The following figure shows the different components of multimedia system.

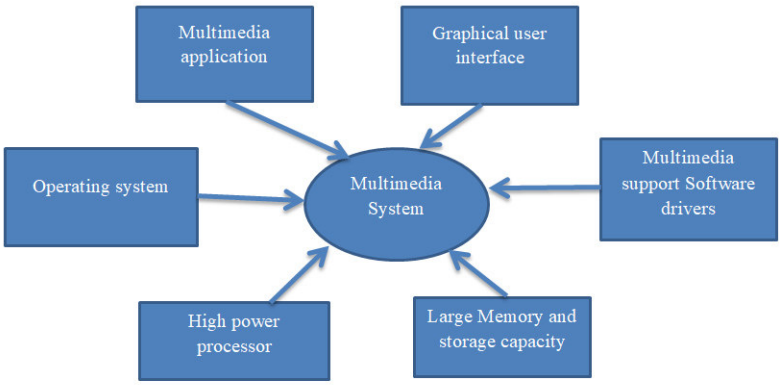


Fig. 1.1: Components of Multimedia System

6.4.3 Components of Multimedia

Following are the common components of multimedia:

Text: All multimedia productions contain some amount of text. The text can have various types of fonts and sizes to suit the profession presentation of the multimedia software.

Graphics: Graphics makes the multimedia application attractive. In many cases people do not like reading large amount of textual matter on the screen. Therefore, graphics are used more often than text to explain a concept, present background information etc. There are two types of Graphics:

Bitmap images: Bitmap images are real images that can be captured from devices such as digital cameras or scanners. Generally bitmap images are not editable. Bitmap images require a large amount of memory.

Vector Graphics: Vector graphics are drawn on the computer and only require a small amount of memory. These graphics are editable.

Audio: A multimedia application may require the use of speech, music and sound effects. These are called audio or sound element of multimedia. Speech is also a perfect way for teaching. Audio are of

Space for learners:

analog and digital types. Analog audio or sound refers to the original sound signal. Computer stores the sound in digital form. Therefore, the sound used in multimedia application is digital audio.

Video: The term video refers to the moving picture, accompanied by sound such as a picture in television. Video element of multimedia application gives a lot of information in small duration of time. Digital video is useful in multimedia application for showing real life objects. Video have highest performance demand on the computer memory and on the bandwidth if placed on the internet. Digital video files can be stored like any other files in the computer and the quality of the video can still be maintained. The digital video files can be transferred within a computer network. The digital video clips can be edited easily.

Animation: Animation is a process of making a static image look like it is moving. An animation is just a continuous series of still images that are displayed in a sequence. The animation can be used effectively for attracting attention. Animation also makes a presentation light and attractive. Animation is very popular in multimedia application.

6.5 MULTIMEDIA DESIGN TECHNIQUES

Following are the several design techniques used in multimedia:

i) **Semi structured systems development life cycle:**
Here the development of the multimedia design follows a partially structured approach.

ii) **Prototyping:**
Here a prototype is designed first to develop the multimedia design. Then extra features are added to the designed prototype.

iii) **Production oriented approach:**
Here the focus mainly depends on the production. Based on the availability of the technologies, the design of multimedia is done.

iv) **Structured systems development life cycle:**
Here a series of steps followed in design of multimedia.

v) **Graphics Design**:
Here, the graphics designer creates interactive multimedia systems. This in the most effective way to design multimedia.

Space for learners:

Basically different forms of media like text, audio, video, animation are combined together to create interactive multimedia systems.

STOP TO CONSIDER

Multimedia design techniques are used to create multimedia applications for better user interaction to access multimedia information for entertainment and education.

6.6 MULTIMEDIA TECHNOLOGY

The technology that uses multimedia elements such as text, audio, video, images, animation etc. to deliver information is termed as multimedia technology. Multimedia technologies contain systems that use the multimedia elements to design interactive multimedia. Multimedia technologies use more than one medium in order to communicate Audio devices include radios, CDs MP3 players etc. video devices include TVs, DVD players etc.

Some of the important evolving technologies for multimedia systems are shown in the following diagram.

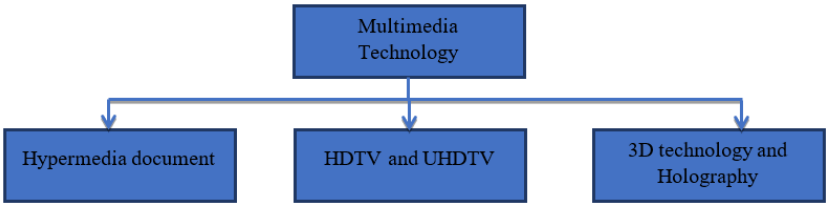


Fig. 1.2 Multimedia technologies

i) **Hypermedia Document**:

This includes hypermedia text and speech. Hypermedia includes multiple forms of media such as text, graphics, audio, images, and video. Hypermedia refers to connecting hypertext with other elements of multimedia. It provides better user experience. World Wide Web is an example of hypermedia. And hypermedia speech is a technology where audio feature is used to navigate through pages.

ii) **HDTV and UHDTV**:

Space for learners:

HDTV stands for High Definition Television. It produces high resolution images with more color and finer way. Also it provides higher quality audio signals.

UDTV stand for Ultra High Definition Television. It is a digital television display format in which the horizontal screen resolution is on the order of 4000 pixels or 8000 pixels

iii) **3D technology and Holography**:

The technology which uses 3D visual appearances to create real life experiences virtually is termed as 3D technology. And the technique where wave front of object is recorded and later reconstructed and presents it in a way that appears 3D.

CHECK YOUR PROGRESS

1. _____ means intermediary.
2. _____ means conveying the information.
3. _____ application must be capable of handling the multimedia files.
4. Graphics makes the multimedia application _____.
5. Bitmap images require a large amount of_____.
6. _____ _____ are editable.
7. _____ _____require a small amount of memory.
8. Hypermedia Document includes hypermedia _____ and _____.
9. World Wide Web is an example of _____.

6.7 NETWORK BASED MULTIMEDIA SYSTEM

Multimedia is a combination of different media. Especially it is useful for education and entertainment. Network based multimedia is to build the multimedia on network so different users can share the multimedia information containing image, sound, video, voice etc. Multimedia has showed its value as a powerful set of technologies used for training, business presentations, and digital marketing. With the help of network based multimedia, people can share the multimedia resource and communicate, work collaboratively, so they may produce more effective results and save more time and money.

Space for learners:

6.8 SUMMING UP

- Multimedia is a technology where different components like audio, video, text, animation and graphics are combined together to form multimedia data or information Security mainly focuses on the confidentiality, integrity and availability of the system resources.
- Interactive multimedia system refers to the technology through which user can easily navigate any site/ application for acquiring information or services.
- As we already know that multimedia is a combination of 5 major components or elements like audio, video, text, graphics and animation.
- Multimedia system needs multimedia application for user interaction with the system.
- Multimedia technologies contain systems that use the multimedia elements to design interactive multimedia. Multimedia technologies use more than one medium in order to communicate Audio devices include radios, CDs MP3 players etc. video devices include TVs, DVD players etc.
- Network based multimedia is to build the multimedia on network so different users can share the multimedia information containing image, sound, video, voice etc.

Space for learners:

6.9 ANSWERS TO CHECK YOUR PROGRESS

1. Medium
2. Media
3. Multimedia
4. attractive
5. memory
6. Vector graphics
7. Vector graphics
8. Text, speech
9. hypermedia

6.10 POSSIBLE QUESTIONS

1. What is interactive multimedia system?
2. What are the different types of interactive multimedia systems?
3. Write few applications of interactive multimedia system.
4. What do you mean by multimedia?
5. What are the components of multimedia?
6. Write some of the characteristics of multimedia system.
7. What is the difference between interactive multimedia system and multimedia?
8. Explain the multimedia technologies.
9. How a multimedia system can be made?

Space for learners:

6.11 REFERENCES AND SUGGESTED READINGS

- https://en.wikipedia.org/wiki/Multimedia
- https://www.tutorialspoint.com/
- https://www.scribd.com/presentation/25931497/Design-Techniques-for-Multimedia
- https://www.indiastudychannel.com/resources/151966-Multimedia-Components-Applications.aspx
- https://www.computerhope.com/

UNIT 7: ANIMATION

Unit Structure:

Space for learners:

7.1 INTRODUCTION

Animation is basically an illusion of movement of pictures. This word has been derived from Greek and Roman. In Greek, it is animous and in Roman, it is anima. From these the word animation has been derived. Both terms have meaning "TO BRING TO ALIVE". Suppose we have some static things, those static object are need to bring alive is animation. Animation can be produced by variation in following:

1. Camera parameters: Change in location with respect to the object, distance from the object, focal length.
2. Lighting condition: Change in direction of light, color of light, number of light and so on.
3. Transforming the object itself or/and the background scenery.

7.2 UNIT OBJECTIVES

Space for learners:

After going through this unit student will be able to learn

- Basic concept of animation
- Types of animation
- How to design animation
- Languages use for animation
- File format and software use for animation etc.

7.3 BASIC CONCEPT AND USAGE OF SOFTWARE MAINTENANCE

Animation is defined as the act of making something come alive. In animation a series of images rapidly changed to create an illusion of movement. Animation includes motion verification, editing and production or rendering, synchronization of sound track.

Usage of animation: It is used in artistic purpose, storytelling, displaying data in scientific visualization, Instructional purpose and games, Entertainment (e.g., cartoons), Advertising (e.g., car converted to a tiger), training and education.

7.4 TYPES OF ANIMATION

1. Cel Animation: A traditional form of animation used in the production of cartoons of animated movies where each frames of the scene is drawn by hand. A full length feature film produced using cel animation would often require a million of more drawings to complete.

2. Computer Animation: Subset of both computer graphics and animation technologies. It is the creation of moving images using computer technology.

3. Kinematics: It is the study of the movement and motion of structures that have joints such as walking man. Such type of animation is usually used in the areas like mechanics etc.

4. Morphing: It is popular effect in which one image transforms into another. The morphed images are build at a rate of 8 frames per seconds, with each transition taking a total of 4 seconds.

STOP TO CONSIDER

A virtual change in a scene with respect to time is known as animation. It is generally achieved by a series of transformation, scaling, translation, rotation or any other mathematical techniques, to produce sequence of scenes.

CHECK YOUR PROGRESS

1. How many types of animation are there?
2. What is the purpose of animation?

7.5 FILE FORMATS OF ANIMATION AND SOFTWARE USED

1. Directorn*.dir
2. Animation Pro *.fli
3. 3D Studio Max *.max
4. Super card and Director *.pics
5. CompuServe *.gif
6. Flash *.fla *.swf

Software used are 3D Studio Max, Flash and Animation Pro

7.6 DESIGNING AN ANIMATION

There are four steps in designing an animation sequence:

1. StoryBoard Layout
2. Object Definition

Space for learners:

3. Keyframe Specification
4. Generation of in-between frames

1. Story Board layout
   - It is the outline of an action. It defines the motion sequences as a set of basic that are to take place.
   - Depending on the type of animation to be produced the story board could be consist of a set of rough sketches of it ,could be a list of basic ideas for motion.
2. Object Definitions
   - Each object participating in the action is given object definition,such as terms of basic shapes such as polygon of splines. It is specifying the characteristics of an object present in the scene.
   - For each object present in the scene, object definition has to be described.
3. Key frame Specification
   - A key frame in animation is a drawing and film making is a drawing that defines the starting and ending points of any smooth transition.
   - A sequence of key frames which movement the spectator will see, but the position of the key frames on the film, defines the timing of the movement 2 of 3 can be present for a span of a second.
4. Generation of in-between frames
   - It is a process of generating intermediate frames between 2 images to give appearance that the first image evolves smoothly into the second image.In between are the drawing between the key frames which help to create the illusion of motion.
   - Film requires 24 frames per second and graphic terminals are refreshed at a rate of 30 to 60 frames per second.

STOP TO CONSIDER

The story board within the workspace is the frame sequence or ordering of the clips of the complete project which defines the series of actions or basic events.

Space for learners:

7.7 COMPUTER ANIMATION LANGUAGES

1. Key frame System : By using this types of languages, whenever key frames are specified we can generate in between. Key frames can be represented as a set of splines or can be represented using physical description how much force applied on an object.
2. Parameterized System: It allows motion characterized by specifying the object definition system. Motion characters are specified as part of object definition.
3. Scripting System: Here the motion characteristics are specified as a scripting input which will be given by user itself.

CHECK YOUR PROGRESS

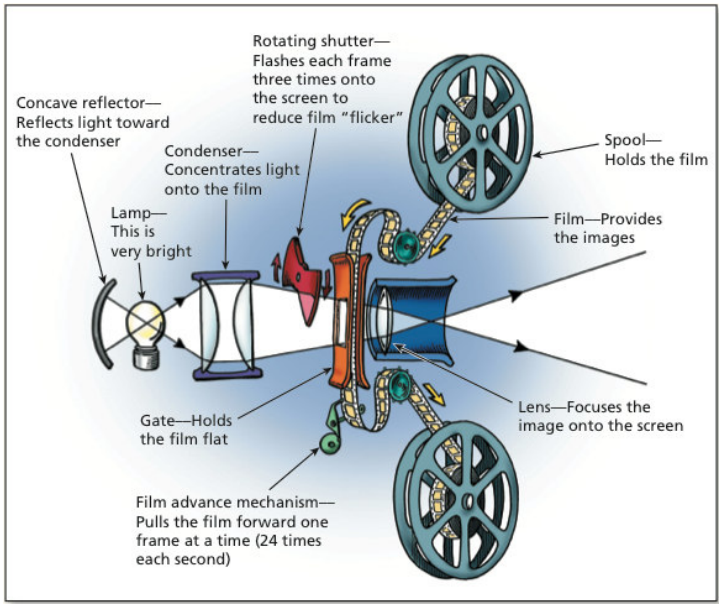
3. How key frame system is different from parameterized system?

Space for learners:

7.8 DEVICES FOR PRODUCING ANIMATION

7.8.1 Film Projector

A film projector is an optical device from which light shines through a microfilm and projects the magnified image in front of it over a white wall or board. The light is made to pass through a series of lenses to focus the image marked on the microfilm properly. The projector is placed at any suitable distance from the wall. The microfilm reel is made to pass through the film gate through sprocket wheels. The film is illuminated by a shuttered light, the shutter of which opens and closes as the film passes by. The lower sprocket roller pulls the film down, one frame at a time. It is also fitted with an optional sound head that reads the sound track which runs parallel to the microfilm storyboard.



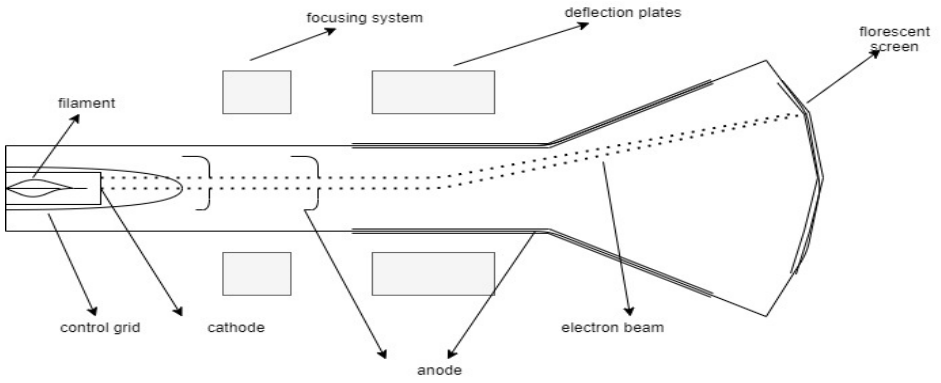
Film Projector[4]

7.8.2 Cathode Ray Tube

The primary output device in graphics system is video monitor and most of the video monitor works on Cathode ray tube. So whatever technology we are using it works on CRT design. CRT is a specialized vacuum tube in which images are produced when an electron beam strikes the phosphorous coated screen. Images produce when electron beam strikes the screen. CRT modulates, accelerates, deflects the electron beam on the screen to create an image. Let's see how CRT works.

It contains an electron gun. The electron gun contains heating filament and cathode. It generates negatively charge electrons and accelerates towards phosphorous coated screen.

Space for learners:



CATHODE RAY TUBE

The next component is control grid. Control grid controls the flow of electrons so that electron does not move freely inside the vacuum tube.

Focusing system is the next component. It is used to create clear picture by focusing the electron into a narrow beam. It is accomplished either magnetic or electric field. If a picture is not clear means there must be problem with focusing system.

The next component to control grid is deflection system. It controls the direction of electron beam by either using electric or magnetic fields. Two pairs of plates are used. One pair of parallel plates are use for horizontal and another for vertical deflection.

Final component is phosphorous coated screen. Inside of screen is coated with phosphorous. The phosphorous glows when highly energy electron beam hits screen and thus we see the picture on screen.

CHECK YOUR PROGRESS

4. What is the purpose of horizontal and vertical deflection plate in CRT?

5.What is the function of electron gun? Why it is required?

Space for learners:

7.9 SUMMING UP

In the context of computer graphics, thus we can say animation is produced by a series of geometric transformation that scaling, translation, rotation or any other mathematical technique, to produce sequence of scenes. It usually specifies the artificially drawn picture sequence. The computer is one of the latest tools in this area, its use powered by its graphics capabilities. Animation in scientific visualization enables an easy understanding of concepts evolved, even of processes that cannot be seen through videos. A micro film output with the projector generally results in more accurate translation from the color system displays than video tape. Film projection requires less resources of RAM, processor speed or graphics capabilities as compared to the graphics audio visual file displays in computer generated animation. Now a days, these devices also come with an interface to a computers standard output, AV or S-video connection. This enables the story board to be prepared and stored in computer audiovisual files.

Space for learners:

7.10 ANSWERS TO CHECK YOUR PROGRESS

1. There are four types of animation.
2. Animation shows accurate representation of an object. It allows us to create 3D representation which seems to be realistic in the display devices. It allows us to create model that are essential for research and study.
3. Key frame systems are specialized animation languages designed simply to generate the in-betweens from the user specified key frames. Parameterized systems describe characteristics of object motion as part of the object definitions.
4. First the electrons pass through the vertical deflection plates, yielding slightly higher sensitivity as distance of vertical deflection plates is more from phosphor coated screen as compared to horizontal deflection plates.

5. Electron gun is use to produce and accelerate the beam of an electron inside the vacuum tube of the CRT. In order to generate and accelerate the gun requires the heater, cathode electrodes, grid, and different types of anodes.

Space for learners:

7.11 MODEL QUESTIONS

1. Define the term Computer animation.
2. Compare various file format of animation.
3. Explain various animation techniques used in computer animation.
4. Discuss various devices for producing animation.
5. How key frame system different from parameterized system?
6. Which program is best for animation?
7. How film projector works? Explain.
8. Explain the working of CRT.
9. What are the functions of different components in a CRT?
10. Write application of animation.

7.12 REFERENCES AND SUGGESTED READINGS

- Computer Animation: Algorithms and Techniques-Book by Rick Parent
- Animation for Beginners: Book by Lisa Lee
- Computer Graphics –Zhigang Xiang and Roy Plastock.
- https://ssec.si.edu/stemvisions-blog/what-physics-behind-watching-movie