

**SRINIVAS INSTITUTE OF MANAGEMENT STUDIES**

**PANDESHWAR, MANGALORE**

**-**

**575 001**

**BACKGROUND STUDY MATERIAL**

Computer Graphics & Multimedia

B.C.A

-

IV SEMESTER

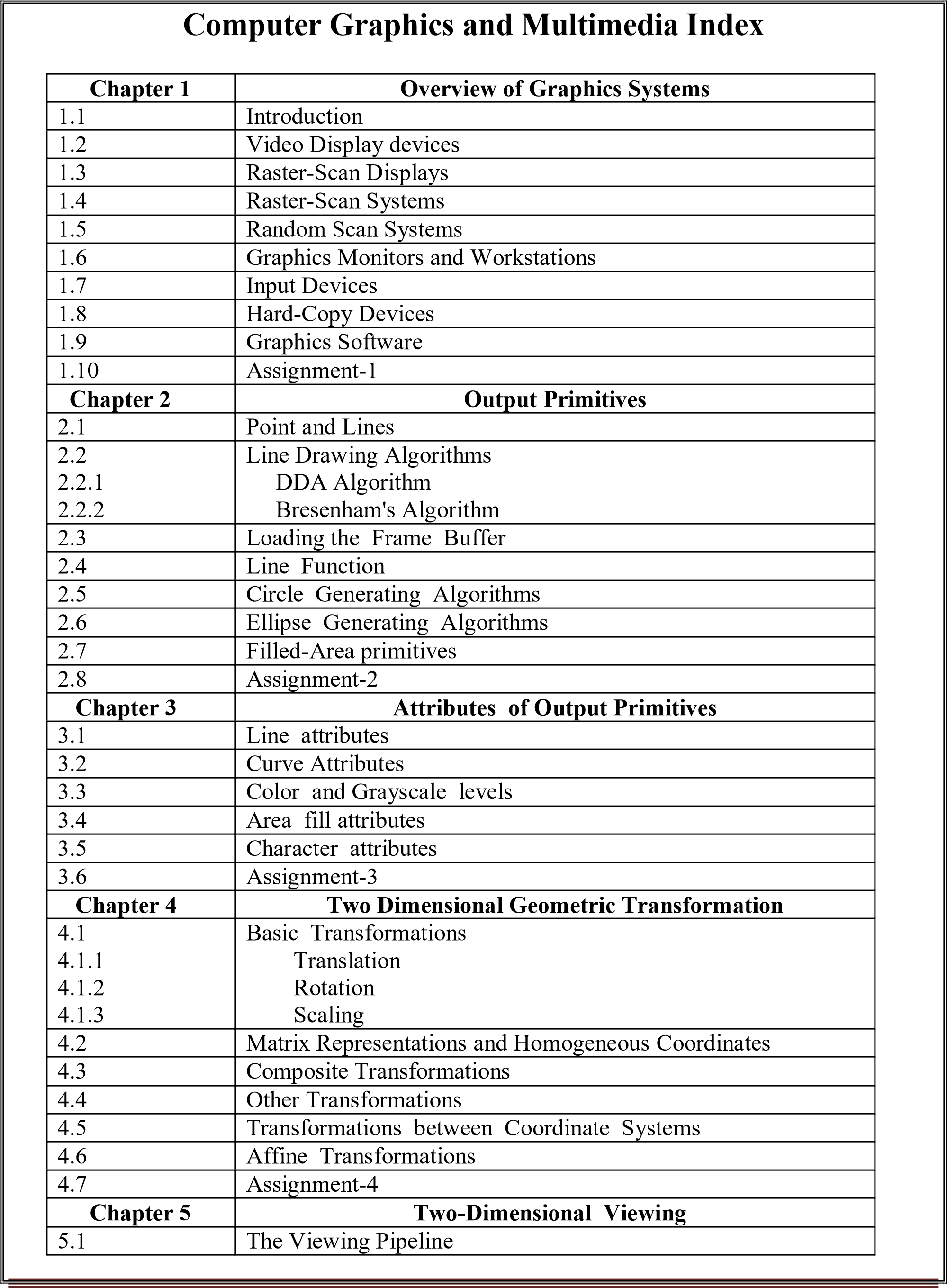
201

5

-

16





|  |  |  |  |
| --- | --- | --- | --- |
|  | 5.2 | Viewing Coordinate Reference Frame |  |
| 5.3 | Window-to-Viewport Coordinate Transformation |
| 5.4 | Two-Dimensional Viewing Functions |
| 5.5 | Clipping operations |
| 5.6 | Point clipping |
| 5.7  5.7.1 | Line clipping  Cohen- Sutherland Line clipping |
| 5.8  5.8.1 | Polygon clipping  Sutherland-Hodgeman Polygon clipping. |
| 5.9 | Assignment-5 |
| **Chapter 6** | **Introduction to Multimedia** |
| 6.1 | What is Multimedia |
| 6.2 | Definition |
| 6.3 | Use of Multimedia |
| 6.4 | Delivering Multimedia |
| 6.5 | Assignment-6 |
| **Chapter 7** | **Text** |
| 7.1 | The Power of meaning |
| 7.2 | About fonts and faces |
| 7.3 | Using fonts in Multimedia |
| 7.4 | Using text in Multimedia |
| 7.5 | Computers and text |
| 7.6 | Font editing and design tools |
| 7.7 | Hypermedia and hyper text |
| 7.8 | Assignment-7 |
| **Chapter 8** | **Images** |
| 8.1 | How to create images |
| 8.2 | Making still images |
| 8.3 | Color |
| 8.4 | Image file formats |
| 8.5 | Assignment-8 |
| **Chapter 9** | **Sound** |
| 9.1 | The Power of sound |
| 9.2 | Digital audio |
| 9.3 | MIDI audio |
| 9.4 | MIDI vs. Digital audio |
| 9.5 | Multimedia system Sounds |
| 9.6 | Audio File formats |
| 9.7 | Vaughan’s Law of Multimedia minimums |
| 9.8 | Adding sounds to multimedia Project |
| 9.9 | Assignment-9 |
| **Chapter 10** | **Animation** |
| 10.1 | The Power of motion |
| 10.2 | Principles of animation |
|  | | | |

|  |  |  |  |
| --- | --- | --- | --- |
|  | 10.3 | Animation by computer |  |
| 10.4 | Assignment-10 |
| **Chapter 11** | **Video** |
| 11.1 | Using video |
| 11.2 | How video works and is displayed |
| 11.3 | Digital Video container |
| 11.4 | Obtaining video clips |
| 11.5 | Shooting and editing videos |
| 11.6 | Assignment-11 |
| **Chapter 12** | **Making Multimedia** |
| 12.1 | The stages of Multimedia project |
| 12.2 | The needs for Multimedia project |
| 12.3 | Input and output devices needed |
| 12.4 | Software needed required authoring system |
| 12.5 | Assignment-12 |
| **Chapter 13** | **Value Added Sessions** |
| Session 1 | Computer-Animated language |
| Session 2 | Vector Drawing |
| Session 3 | Video Recording Formats |
| Session 4 | Broadcasting video Standards |
|  | Question Bank |
|  | Model Question Paper |
|  | |

**Teaching Plan**

**Subject: BCACAC401- Computer Graphics and Multimedia**

**Faculty: Mr. Krishna Prasad K.**

# Total Hours: 48 +4= 52 Hours UNIT- I 1. Introduction 6 Hrs

**Session 1:** Introduction to computer graphics, Video Display devices-Architecture, Discussion on Direct view storage tubes (DVST), Discussion on flat panel displays and Raster-Scan Displays-Architecture.

**Session 2:** Raster Scan Systems- Architecture, Explanations and Random Scan SystemsArchitecture and explanations, Difference between raster and random scan systems.

**Session 3:** Graphics Monitors and Workstations-Importance and Examples.

**Session 4:** Input Devices-Different input devices discussions like Keyboard, Mouse, Trackball and space ball, Joysticks, Data glove, Digitizer, Image scanner, Touch panel, Light pen and Voice system

**Session 5:** Hard-Copy Devices-Discussion on different types of printers like Dot matrix, Laser, Inkjet and Discussion on Plotters

**Session 6:** Graphics Software-Types of graphics software,Graphic Coordinate representation, Graphics Functions and Graphic Software Standards

# 2. Language Reference 6 Hrs

**Session 7:** Point and Lines-Definition, importance and line properties.

**Session 8:** Line Drawing Algorithms-Derivation of DDA algorithm, Steps involved in DDA line algorithm.

**Session 9:** Bresenham's Algorithm- Derivation of Bresenham's algorithm, Steps involved in Bresenham's line algorithm.

**Session 10:** Loading the Frame Buffer and Line Function-importance and discussion on its built in functions.

**Session 11:** Circle Generating Algorithms-Properties of Circle, Midpoint circle drawing algorithm derivation and Steps involved in midpoint circle drawing algorithm.

**Session 12:** Ellipse Generating Algorithms-Properties of Ellipse, Midpoint ellipse drawing algorithm and steps involved in midpoint ellipse drawing algorithm and Filled-Area primitives.

# UNIT- II 3. Attributes of Output Primitives 3 Hrs

**Session 13:** Line attributes and Curve Attributes-Discussion on type, width, color and pen and brush options of line and curve.

**Session 14:** Color and Grayscale levels-Its importance and discussion on color lookup table. **Session 15:** Area fill attribute-discussion on area fill and hatch fill patterns and Character attributes like color, width, height and discussion on many built in functions.

# 4. Two Dimensional Geometric Transformation 4 Hrs

**Session 16:** Basic Transformations- Discussion on Translation, Rotation and Scaling definition’s and equations.

**Session 17:**  Matrix Representations and Homogeneous Coordinates importance and definitions and homogenous coordinates for Translation, Rotation and Scaling

**Session 18:** Composite Transformations and Other Transformations like reflection and sheardefinition, importance and equations

**Session 19:** Transformations between Coordinate Systems and Affine Transformationsdefinition, importance and examples.

# 5. Two-Dimensional Viewing 5 Hrs

**Session 20:** Discussion on Viewing Pipeline and Viewing Coordinate Reference Frame and discussion on word coordinates and device coordinates.

**Session 21:** Window-to-Viewport Coordinate Transformation-Discussion on normalization transformation and workstation transformation and discussion on some built in functions of TwoDimensional Viewing.

**Session 22:** Clipping operations and Point clipping-Its properties, definitions of clipping and examples for clipping.

**Session 23:** Line Clipping- Explanations on Cohen- Sutherland Line clipping, examples and steps involved in Cohen- Sutherland Line clipping.

**Session 24:** Polygon clipping- Explanations on Sutherland-Hodgeman Polygon clipping, examples and steps involved in Sutherland-Hodgeman Polygon clipping.

# UNIT- III

**6.** **Introduction to Multimedia 3 Hrs Session 25:** Introduction to multimedia-Definition of Multimedia and examples for multimedia. **Session 26:** Applications of Multimedia in different areas like creative industries, commercial, entertainment and fine arts, Education, Industries and Medicine

**Session 27:** Delivering Multimedia-Discussion on different media involved in delivering multimedia like CD-ROM, DVD, Flash Drives, broadband and internet.

# 7. Text 3 Hrs

**Session 28:** The Power of meaning of text, Discussion about fonts and faces and naming conventions.

**Session 29:** Discussion on using fonts in Multimedia with its properties Discussion on using text in Multimedia with its properties

**Session 30:** Discussion on computers and text, font editing and design tools and discussion on Hypermedia and hyper-text

# 8. Images 3 Hrs

**Session 31:** Discussion on how to create images-Discussion on captured and bit map images and bitmap sources.

**Session 32:** Making still images-Discussion on bitmap softwares, capturing and editing Images and scanning images.

**Session 33:** Color-Discussion on understanding natural light and color, computerized color, Additive and subtractive color and color pallets. Discussion on image file formats like BMP, PICT, TIFF, JPEG, JIFF etc.

# 9. Sound 3 Hrs

**Session 34:** Discussion on Power of sound, Digital audio-Preparing digital audio files, and MIDI audio-Definition and importance.

**Session 35:** Discussion on MIDI vs. Digital audio and Discussion on Multimedia system Sounds and Audio file formats like WMA, MP3, RA, VOC etc.

**Session 36:** Discussion on Vaughan’s Law of Multimedia minimum and steps involved in adding sounds to multimedia-Project.

# UNIT-IV 10. Animation 3 Hrs

**Session 37:** Discussion on Power of motion, Definition of animation and its importance.

**Session 38:**  Discussion on Principles of animation-Difference between 2D and 3D animation. **Session 39:** Animation by computer-Discussion of Animation techniques like Cel animation, Kinametics and Morphing.

# 11. Video 5 Hrs

**Session 40:** Discussion on Video Definition and how to use video.

**Session 41:** Explanation about how video works and how video is displayed on the screen.

**Session 42:** Explanations on Digital Video container like MPEG, Codec etc.

**Session 43:** Explanation on obtaining video clips and tips for obtaining video clips.

**Session 44:** Shooting and editing videos-Recording Formats like S-VHS video, Component (YUV) and digital video.

# 12. Making Multimedia 4 Hrs

**Session 45:** Discussion on the stages of Multimedia project-planning and costing, designing and producing, testing and delivering.

**Session 46:** The needs for Multimedia project like creativity, Organization, Communication, Multimedia hardware and memory and storage devices etc.

**Session 47:** Input and output devices neededfor multimedia project likeOCR, Digital cameras, CRT and LCD monitors etc.

**Session 48:** Software needed for multimedia project like painting and drawing tools 3Dmodelling and animation tools and image editing tools etc. Discussion on multimedia authoring tools and its types.

# Value added Sessions 4Hrs

**Session 49:** Computer-Animated language-Discussion on Scene description and key- frame systems.

**Session 50:** Vector Drawing introduction and how vector drawing works.

**Session 51:** Discussion on different Video Recording Formats types.

**Session 52:** Broadcasting video Standards-importance, definition its types.

**Text Books:**

1. Donald Hearn, M. Pauline Baker, Computer Graphics - C version, 2nd

Edition, LPE Pearson. (Units: I and II)

1. Tay Vaughan, Multimedia: Making It Work, 8th Edition, Tata McGraw Hill, 2011. (Units: III and IV)

**Reference Books:**

1. Steven Harrington, Computer Graphics: A Programming Approach, McGraw Hill Education.
2. Ze-Nian Li and Mark S Drew, Fundamentals of Multimedia, PHI, 2009
3. Ralf Steinmetz and Klara Nahrstedt, Multimedia: Computing, Communication and Applications, LPE, Pearson Education

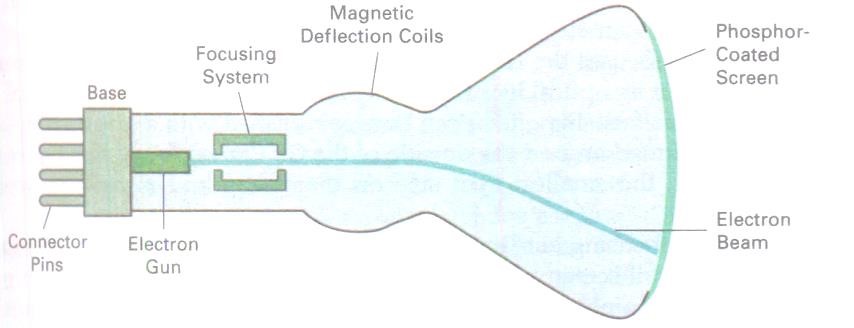
**Scheme of Examination:**

|  |  |  |
| --- | --- | --- |
| Internal marks | External Exam | Total |
| (Total Marks- 20)  3 Marks for first internal   1. Marks for second internal 2. Marks for preparatory 3. Marks for Attendance   5 Marks for Assignment | (Total Marks-100)  Part A-20 Marks  Part B-80 marks (From each unit 20 marks) | (20+80=100) |

# Unit-I Chapter 1 Overview of Graphics Systems

## 1.1 Introduction

The computer graphics is one of the most effective and commonly used ways to communicate the processed information to the user. It displays the information in the form of graphic objects such as pictures, charts, graphs and diagrams instead of simple text. In Graphics, pictures and objects are presented as a collection of picture elements called pixels. The pixel is the smallest addressable screen element. It is the smallest element of the display screen. The special procedures determine which pixel will provide the best approximation to the desired picture or graphics object.



## 1.2 Video display Devices

Typically, the primary output device in a graphics system is a video monitor. The operation of most video monitors is based on the standard cathode-ray tube (CRT) design.

# Figure 1.1: Basic design of a magnetic deflection CRT

Figure 1.1 illustrates the basic operation of a CRT. A beam of electrons (cathode rays), emitted by an electron gun, passes through focusing and deflection systems that direct the beam toward specified positions on the phosphor-coated screen. The phosphor then emits a small spot of light at each position contacted by the electron beam. Because the light emitted by the phosphor fades very rapidly, some method is needed for maintaining the screen picture. One way to keep the phosphor glowing is to redraw the picture repeatedly by quickly directing the electron beam back over the same points. This type of display is called a refresh CRT.

The primary components of an electron gun in a CRT are the heated metal cathode and a control grid (Figure 1.2). Heat is supplied to the cathode by directing a current through a coil of wire, called the filament, inside the cylindrical cathode structure. This causes electrons to be "boiled off" the hot cathode surface. In the vacuum inside the CRT envelope, the free, negatively charged electrons are then accelerated toward the phosphor coating by a high positive voltage. The accelerating voltage can be generated with a positively charged metal coating on the inside of the CRT envelope near the phosphor screen, or an accelerating anode can be used.

# Figure 1.2 Operation of an electron gun with an accelerating anode

Sometimes the electron gun is built to contain the accelerating anode and focusing system within the same unit. Intensity of the electron beam is controlled by setting voltage levels on the control grid, which is a metal cylinder that fits over the cathode. The amount of light emitted by the phosphor coating depends on the number of electrons striking the screen, which control the brightness of a display by varying the voltage on the control grid. The focusing system in a CRT is needed to force the electron beam to converge into a small spot as it strikes the phosphor. Otherwise, the electrons will repel each other, and the beam would spread out as it approaches the screen. Focusing is accomplished with either electric or magnetic fields.

Additional focusing hardware is used in high-precision systems to keep the beam in focus at all screen positions. The distance that the electron beam must travel to different points on the screen varies. Therefore, the electron beam will be focused properly only at the center of the screen. As the beam moves to the outer edges of the screen, displayed image become blurred. To compensate for this, the system can adjust the focusing according to the screen position of the beam.

Different kinds of phosphors are available for use in a CRT. Besides color, a major difference between phosphors is their persistence: how long they continue o emit light (that is, have excited electrons returning to the ground state) after the CRT beam is removed. Persistence is defined as the time it takes the emitted light from the screen to decay to one-tenth of its original intensity. Lower- persistence phosphors require higher refresh rates to maintain a picture on the screen without flicker.

The maximum number of points that can be displayed without overlap on a CRT is referred to as the resolution. A more precise definition of resolution is the number of points per centimeter that can be plotted horizontally and vertically, although it is often simply stated as the total number of points in each direction. The resolution of CRT is dependent on the type of phosphor, the intensity to be displayed, and the focusing and deflection systems. Typical resolution on highquality systems is 1280 by 1024, with higher resolutions available on many systems. High resolution systems are often referred to as high-definition systems. The physical size of a graphics monitor is given as the length of the screen diagonal, with size varying from about 12 inches to 27 inches or more.

Another property of video monitors is aspect ratio. This number gives the ratio of vertical points to horizontal points necessary to produce equal-length lines in both directions on the screen. An aspect ratio of 3/4 means that a vertical line plotted with three points has the same length as a horizontal line plotted with four points.

# Color CRT Monitors

A CRT monitor displays color pictures by using a combination of phosphors that emit differentcolored light. By combining the emitted light from the different phosphors, a range of colors can be generated. The two basic techniques for producing color displays with a CRT are the beampenetration method and the shadow-mask method.

The beam-penetration method for displaying color pictures has been used with random-scan monitors. Two layers of phosphor, usually red and green, are coated onto the inside of the CRT screen, and the displayed color depends on how far the electron beam penetrates into the phosphor layers. A beam of slow electrons excites only the outer red layer. Abeam of very fast electrons penetrates through the redlayer and excites the inner green layer. At intermediate beam speeds, combinations of red and green light are emitted to show two additional colors, orange and yellow. The speed of the electrons, and hence the screen color at any point, is controlled by the beam-acceleration voltage. Beam penetration has been an inexpensive way to produce color in random-scan monitors, but only four colors are possible, and the quality of pictures is not as good as with other methods.

Shadow-mask methods arecommonly usedin raster scan systems (including color TV**)** because they produce a much wider range of colors than the beam penetration method. A shadow-mask CRT has three phosphor color dots at each pixel position. One phosphor dot emits a redlight, another emits a green light, and the third emits a blue light. This typeof CRT has three electron guns, one for each color dot, and a shadow-mask grid just behind the phosphor-coated screen. The delta shadow-mask method commonly used in color CRTsystems. The three electron beams are deflected and focused as a group onto the shadow mask, which contains a series of holes aligned with the phosphor-dot patterns. When the three beams pass through a hole in the shadow mask, they activate a dot triangle, which appears as a small color spot on the screen. The phosphor dots in the triangles are arranged so that each electron beam can activate only its corresponding color dot when it passes through the shadow mask. Another configuration for the three electron guns is an in-line arrangement in which the three electron guns, and the corresponding red-green-blue color dots on the screen, are aligned along one scan line instead of in a triangular pattern. This in-line arrangement of electron guns is easier to keep in alignment and is commonly used in high-resolution color CRTs. We obtain color variations in a shadowmask CRT by varying the intensity levels of the three electron beams.

Color CRTs in graphics systems are designed as RGBmonitors. These monitors use shadowmask methods and take the intensity level for each electron gun (red, green, and blue) directly from the computer system without any intermediate processing. High-quality raster-graphics systems have 24 bits per pixel in the fame buffer, allowing 256voltage settings for each electron gun and nearly 17 million color choices for each pixel. An RGB color system with 24 bits of storage per pixel is generally referred to as a full-color system or a true-color system.

# Direct-View Storage Tubes

An alternative method for maintaining a screen image is to store the picture information inside the CRT instead of refreshing the screen. A direct-view storage tube (DVST) stores the picture information as a charge distribution just behind the phosphor-coated screen. Two electron guns are used in a DVST. One, the primary gun, is used to store the picture pattern; the second, the flood gun, maintains the picture display.

**Advantages:**

* No Refreshing is needed.
* Very complex pictures can be displayed at very high resolutions without flicker.

**Disadvantages:**

* Do not display color and the selected part of a picture cannot be erased.
* To eliminate a picture section, the entire screen must be erased and the modified picture redrawn.
* The erasing and redrawing process can take several seconds for a complex picture.

# Flat-Panel Displays

The term flat-panel display refers to a class of video devices that have reduced volume, weight, and power requirements compared to a CRT. A significant feature of flat-panel displays is that they are thinner than CRTs, and we can hang them on walls or wear them on our wrists. Since we can even write on some flat-panel displays, they will soon be available as pocket notepads. Current uses for flat-panel displays include small TVmonitors, calculators, pocket video games, laptop computers, armrest viewing of movies on airlines, as advertisement boards in elevators, and as graphics displays in applications requiring rugged, portable monitors.

We can separate flat-panel displays into two categories: emissive displays and non emissive displays. The emissive displays (or emitters) are devices that convert electrical energy into light. Plasma panels, thin-film electroluminescent displays, and Light-emitting diodes are examples of emissive displays. Nonemissive displays (or non emitters) use optical effects to convert sunlight or light from some other source into graphics patterns. The most important examples of nonemissive flat panel display is a liquid-crystal device.

Plasma panels, also called gas-discharge displays, are constructed by filling the region between two glass plates with a mixture of gases that usually includes neon. Thin-filmelectroluminescent displays are similar in construction to a plasma panel. The difference is that the region between the glass plates is filled with a phosphor, such as zinc sulfide doped with manganese, instead of a gas. Athird typeof emissive device is the light-emitting diode (LED).Amatrix of diodes is arranged to form the pixel positions in the display, and picture definition is stored in a refresh buffer. Asin scan-line refreshing of a CRT**,** information is read from the refresh buffer and converted to voltage levels that are applied to the diodes to produce the light patterns in the display.

The LCD monitor creates images with a special kind of liquid crystal that is normally transparent but which, when charged with electricity become opaque. One disadvantage of LCD monitors is that their images can be difficult to see in the brightest light. For this reasons, laptop computer users often look for shady places to sit when working outdoors or near windows. A bigger disadvantages of LCD monitors, however, is their limited viewing angle-that is, the angels from which the displays images clearly even when standing at the angel to the screen. In LCD monitors, however, the viewing shrinks, as you increase your angel to the screen. The images becomes fuzzy quickly .In many older flat-panel systems, the user must face the screen straight to see the image clearly.

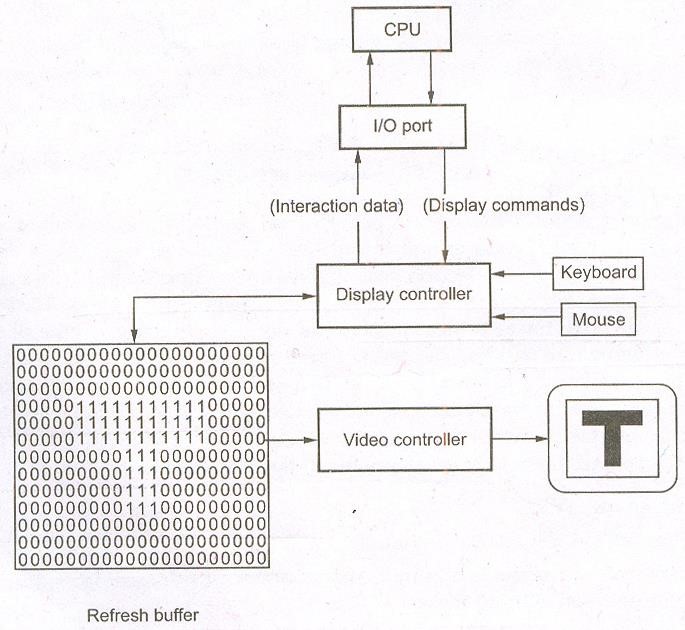
There are two main categories of liquid crystal displays;

Passive matrix LCD relies on transistors for each row and each column of pixels, thus creating a grid that defines the location of each pixels. The color displays by a pixels is determined by the electricity coming from the transistors at the end of the row and top of the column. Although the passive matrix monitor is inexpensive to manufacture, they narrow viewing angle. Another disadvantage is that they don’t ‘refresh’ the pixels very often. Most passive matrix screens now use dual –scan LCD technology, which scans the pixels twice as often.

Active matrix LCD technology assigns a transistor to each pixel and each pixel is turned on and off individually. This enhancement allows the pixels to be refreshed much more rapidly. So submarine is not the problems. Active matrix screens have a wider viewing angel than dual-scan screens. Many active matrix displays use thin-film (TFT) technology, which employs as many as four transistors per pixels.

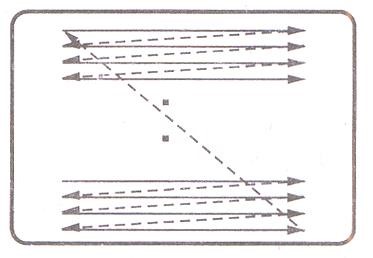
## 1.3 Raster Scan Display

The following figure shows the architecture of a raster display. It consists of display controller, central processing unit (CPU), video controller, refresh buffer, keyboard, mouse and the CRT.



# Figure 1.3 Raster scan display

**Architecture of a raster display:** As shown in the above figure the display images stored in the form of 1’s and 0’s in the **refresh buffer.** The video controller reads this refresh buffer and produces the actual image on the screen. It does this by scanning one scan line at a time, from top to bottom and then back to the top.



# Figure 1.4 Raster Scan CRT

In this method, the horizontal and vertical deflection signals are generated to move the beam all over the screen in a pattern shown in the above figure

Here, the beam is swept back forth from left to right across the screen. When the beam is moved from the left to right, it is ON. The beam OFF, when it is moved from right to left as shown by dotted line in the above figure

When the beam reaches the bottom of the screen, it is made OFF and rapidly retraced back to the top to start again. A display produced in this way is called **raster scan display.** Raster scanning process is similar to reading different lines on the page of a book. After completion of scanning of one line, the electron beam files back to the start of next line and process repeats. In the raster scan display, the screen images are maintained by repeating scanning the same image 60 times/sec. This process is known as **refreshing of screen.** At the end of the each scan line, the electron beam returns to the left side of the screen to begin displaying the next scan line. The return to the left of the screen, after refreshing each scan line, is called the **horizontal retrace** of the electron beam. And at the end of each frame (displayed in 1/80th to 1/60th of a second), the electron beam returns (**vertical retrace**) to the left corner of the screen to begin the next frame.

## 1.4 Raster scan systems

Interactive raster graphics systems typically employ several processing units. In addition to the central processing unit, or CPU, a special-purpose processor, called the video controller or display controller, is used to control the operation of the display device.

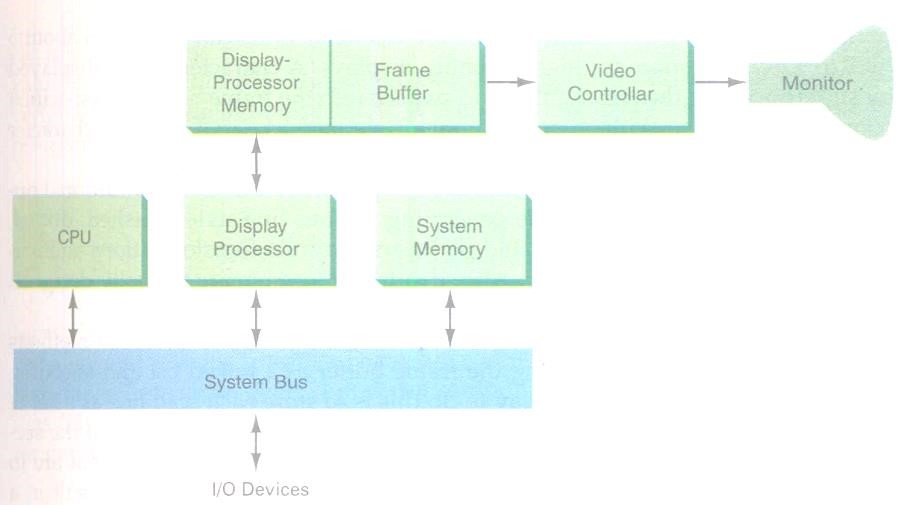
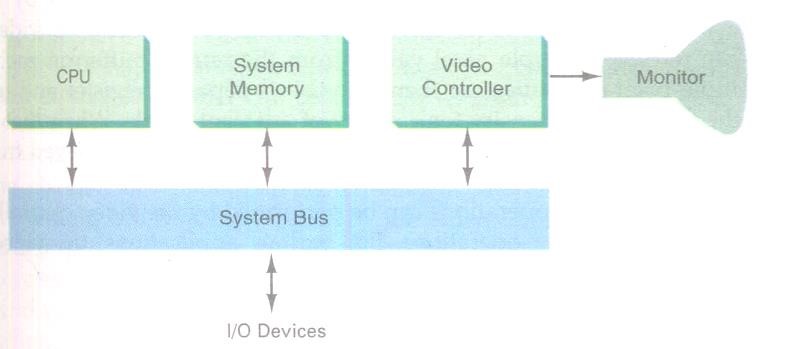
Frame-buffer locations, and the corresponding screen positions, are referenced in Cartesian coordinates. For many graphics monitors, the coordinate origin is defined at the lower left screen corner. The screen surface is represented as the first quadrant of a two-dimensional system, with positive values increasing to the right and positive y values increasing from bottom top. (On some personal computers, the coordinate origin is referenced at upper left corner of the screen, so the y values are inverted.) Scan lines are labeled from Ymax at the top of the screen to 0 at the bottom. Along each scan screen pixel positions are labeled from 0 to Xmax. The basic refresh operations of the video controller are programmed. Two registers are used to store the coordinates of the screen pixels. Initially, the x register is set to 0 and the Y register is set to Ymax.

# Figure 1.5 Architecture of a simple raster graphics system

The value stored the frame buffer for this pixel position is then retrieved and used to set the intensity of the CRT beam. Then the x register is incremented by 1, and the process repeated for the next pixel on the top scan line. This procedure is repeated for pixel along the scan line. After the last pixel on the top scan line has been processed, the x register is reset to 0 and the Y register is decremented by 1 pixel along this scan line are then processed in turn, and the procedure is repeated each successive scan line. After cycling through all pixels along the bottom scan line (y = 0), the video controller resets the registers to the first pixel position the top scan line and the refresh process starts over.

Since the screen must be refreshed at the rate of 60 frames per second, simple procedure cannot be accommodated by typing RAM chips. The cycle time is too slow. To speed up pixel processing, video controllers can retrieve multiple pixel values from the refresh buffer on each pass. The multiple pixel intensities are then stored in a separate register and used to control the CRT beam intensity for a group of adjacent pixels. When that group of pixels has been processed, the next block of pixel values is retrieved from frame buffer.

In high-quality systems, for example, two frame buffers are often provided so that one buffer can be used for refreshing while the other is being filled with intensity values. Then the two buffers can switch roles. This provides a fast mechanism for generating real-time animations, since different views of moving objects can be successively loaded into the refresh buffers. Also, some transformations can be accomplished by the video controller. Areas of the screen can be enlarged, reduced, or moved from one location to another during the refresh cycles. In addition, the video controller often contains a lookup table, so that pixel values in the frame buffer are used to access the lookup table instead of controlling the CRT beam intensity directly.



This provides a fast method for

changing

scree

n intensity values.

**Figure 1.6: Architecture of a raster**

**-**

**graphics**

**system with a display processor**

**1.5**

**Random Scan Systems**

When

operated as a

random

-

scan display unit,

a CRT has the electron beam di

rected

only to the

parts of the screen where a

-

picture is to be drawn. Random

scan

monitors draw a picture one line

at a time

and for this reason are also re

ferred to as vector displays (or stroke

-

wri

ting or

c

alligraphic displays). The

component

lines of a picture can be drawn and refreshed by a

random

-

scan system in any specified order.

**Figure 1.7 Random scan system**

As shown in figure 1.7

random or

vector scan CRT directly traces out onl

y the desir

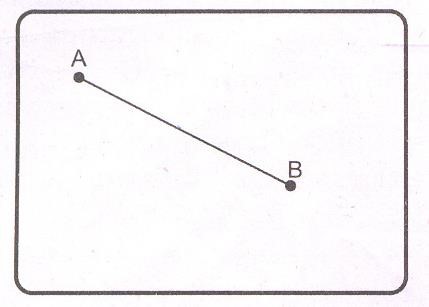
ed lines on

CRT i.e., i

f we want a line connecting point A with point B on the vector graphics display, we

simply drive the beam deflection circuitry which will cause the beam to go dir

ectly from point A



|  |  |
| --- | --- |
| **Vector Scan Display** | **Raster Scan Display** |
| In vector scan display the beam is moved between the end points of the graphics primitives | In raster scan display the beam is moved all over the screen one scan line at a time, from top to bottom and then back to top. |
| Vector display flickers when the number of primitives in the buffer becomes too large. | In raster display, the refresh process is independent of the complexity of the image. |
| Scan conversion is not required. | Graphics primitives are specified in terns of their endpoints and must be scan converted into their corresponding pixels in he frame buffer. |
| Scan conversion hardware is not required. | Because each primitive must be scan converted, real time dynamics is far more computational and requires separate scan conversion hardware. |
| Vector display draws a continuous and smooth lines | Raster display can display mathematically smooth line, polygons and boundaries of curved primitives only by approximating them with pixels on the raster grid. |
| Cost is more. | Cost is low. |
| Vector display only draws lines and characters. | Raster display has ability to display areas filled with solid colors or patterns. |

to point B. If we want to move the beam from point A to B without showing a line between points, we can blank the beam as we move it.

The vector scan system architecture consists of display controller, Central Processing Unit (CPU), display buffer memory and a CRT. A display controller is connected as an I/O peripheral to the central processing unit (CPU). The display buffer memory stores the computer produced display list or display program. The program contains point and line plotting commands with (x,y) or (x,y,z) end point coordinated, as well as character plotting commands. The display controller interprets commands for plotting points, lines and characters and sends digital and point coordinates to a vector generator. The vector generator then converts the digital coordinates values to analog voltages for beam-deflection circuits that displace an electron beam writing on the CRT’s phosphor coating.

In vector scan system beam is deflected from end point to end point, hence this technique is also called **random scan.** We know as beam, strikes phosphor it emits light. But phosphor light decays after few milliseconds and therefore it are necessary to repeat through the display list to refresh the phosphor at least 30 times per second to avoid flicker. As display buffer is used to store display list and it is used for refreshing, the display buffer memory is also called **refresh buffer.**

# Difference between Vector scan display and raster scan display

## 1.6 Graphics Monitor and Workstations

Most graphics monitors today operate as raster-scan displays. Graphics systems range from small general-purpose computer systems with graphics capabilities to sophisticated full-color systems that are designed specifically for graphics applications. A typical screen resolution for personal computer graphics workstations with keyboard and mouse input devices. Diagonal screen dimensions for general-purpose personal computer systems can range from 12 to 21 inches, and allowable color selections range from 16 to over 32,000. For workstations specifically designed graphics applications, typical screen resolution is 1280 by 1024, with a screen diagonal of 16 inches or more. Graphical workstations can be configured with from 8 to 24 bits per pixel (fullcolor systems), with higher screen resolutions, faster processors, and other options available in high-end systems.

In applications such as air traffic control, simulation, medical imaging, and CAD, a high definition graphics monitors are used. This system has a diagonal screen size of 27 inches, resolutions ranging from 2048 by 1536, 2560 by 2048, with refresh rates of 80 Hz or 60Hz non interlaced. A multi screen system called the MediaWall, provides large "wall-sized" display area. This system is designed for applications that quire large area displays in brightly lighted environments, such as at trial shows, conventions, retail stores, museums, or passenger terminals. Media operates by splitting images into a number of sections and distributing the selections over an array of monitors or projectors using a graphics adapter and satellite control units. An array of up to 5 by 5 monitors, each with a resolution of by 480, can be used in the MediaWall to provide an overall resolution of 3200 × 2400 for either static scenes or animations.

Many graphics workstations are configured with two monitors. Many graphics workstations are configured with two monitors, while the second monitor displays the detail in some part of the picture. Another use for dual-monitor systems is to view a picture on one monitor and display graphics options (menus) for manipulating the picture components on the other monitor.

Interactive graphics workstation contains multiple input and other devices. A typical setup for CAD applications consist of various keyboards, button boxes, tablets, and mice are attached to the video monitors for use in the design process.

## 1.7 Input Devices

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

**Keyboard**: An alphanumeric keyboard on a graphics system is used primarily as a device for entering text strings. The keyboard is an efficient device for inputting such non graphic data as picture labels associated with a graphics display. Keyboards can also be provided with features to facilitate entry of screen coordinates, menu selections, or graphics functions. Cursor-control keys and function keys are common features on general purpose keyboards. Function keys allow users to enter frequently used operations in a single keystroke, and cursor-control keys can be used to select displayed objects or coordinate positions by positioning the screen cursor. Other types of cursor-positioning devices, such as a trackball or joystick, are included on some keyboards.

Additionally, a numeric keypad is often included on the keyboard for fast entry of numeric data. For specialized applications, input to a graphics application may come from a set of buttons, dials, or switches that select data values or customized graphics operations. Buttons and switches are often used to input predefined functions, and dials are common devices for entering scalar values. Real numbers within some defined range are selected for input with dial rotations. Potentiometers are used to measure dial rotations, which are then converted to deflection voltages for cursor movement.

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

Since a mouse can be picked up and put down at another position with change in cursor movement, it is used for making relative changes in the position of the screen cursor. One, two, or three buttons are usually included on the top the mouse for signaling the execution of some operation, such as recording cursor position or invoking a function. Most general-purpose graphics systems not include a mouse and a keyboard as the major input devices.

Additional devices can be included in the basic mouse design to increase the number of allowable input parameters. The Z mouse includes three buttons, a thumbwheel on the side, a trackball on the top, and a standard mouse ball underneath. This design provides six degrees of freedom to select spatial positions, rotations, and other parameters. With the Z mouse, we can pick an object, rotate it, and move it in any direction, or we can navigate our viewing position and orientation through a three-dimensional scene. Applications of the Z mouse include virtual reality, CAD, and animation.

**Trackball and Spaceball:** As the name implies, a trackball is a ball that can be rotated with the fingers or palm of the hand to produce screen-cursor movement. Potentiometers, attached to the ball, measure the amount and direction of rotation. Trackballs are often mounted on keyboards or other devices such as the Z mouse. While a trackball is a two-dimensional positioning device, a spaceball provides six degrees of freedom. Unlike the trackball, a spaceball does not actually move. Strain gauges measure the amount of pressure applied to the spaceball to provide input for spatial positioning and orientation as the ball is pushed or pulled in various directions. Spaceballs are used for three-dimensional positioning and selection operations in virtual-reality systems, modeling, animation, CAD, and other applications.

**Joysticks:** A joystick consists of a small, vertical lever (called the stick) mounted on a base it is used to steer the screen cursor around. Most joysticks select screen positions with actual stick movement; others respond to pressure on the stick. Some joysticks are mounted on a keyboard; others function as stand-alone units.

The distance that the stick is moved in any direction from its center position corresponds to screen-cursor movement in that direction. Potentiometers mounted at the base of the joystick measure the amount of movement, and springs return the stick to the center position when it is released. One or more buttons can be programmed to act as input switches to signal certain actions once a screen position has been selected. In another type of movable joystick, the stick is used to activate switches that cause the screen cursor to move at a constant rate in the direction selected. Eight switches, arranged in a circle, are sometimes provided, so that the stick select anyone of eight directions for cursor movement. Pressure-sensitive sticks, also called isometric joysticks, have a non movable stick.

**Data Glove:** Data glove can be used to grasp a virtual object. Data glove is constructed with a series of sensors that detect hand and finger motion. Electromagnetic coupling between transmitting antennas and receiving antennas is used to provide information about the position and orientation of the hand. The transmitting and receiving antennas can each be structured as a set of three mutually perpendicular coils, forming a three-dimensional Cartesian coordinate system. Input from the glove can be used to position or manipulate objects virtual scene.

**Digitizer:** A common device for drawing, painting, or interactively selecting coordinate positions on an object is a digitizer. These devices can be used to input coordinate values in either a two-dimensional or a three-dimensional space. Typically, a digitizer is used to scan over a drawing or object and to input a set of discrete coordinate positions, which can be joined with straight-line segments to approximate the curve or surface shapes.

One type of digitizer is the graphics tablet (also referred to as a data tablet), which is used to input two-dimensional coordinates by activating a hand or stylus at selected positions on a flat surface. Many graphics tablets are constructed with a rectangular grid of wires embedded in the tablet surface. Electromagnetic pulses are generated in sequence along the wires, and an electric signal is induced in a wire coil in an activated stylus or hand cursor to record a tablet position. Depending on the technology signal strength, coded pulses, or phase shifts can be used to determine position on the tablet.

**Image Scanner:** Drawings, graphs, color and black-and-white photos, or text can be stored for computer processing with an image scanner by passing an optical scanning mechanism over the information to be stored. The gradations of gray scale or color are then recorded and stored in an array. Once we have the internal representation of a picture, we can apply transformations to rotate, scale, or crop the picture to a particular screen area. We can also apply various imageprocessing methods to modify the array representation of the picture. For scanned text input, various editing operations can be performed on the stored documents.

**Touch Panels:** As the name implies, touch panels allow displayed objects or screen positions be selected with the touch of a finger. A typical application of touch panels is the selection of processing options that are represented with graphical icon. Touch input can be recorded using optical, electrical, or acoustical methods.

Optical touch panels employ a line of infrared light-emitting diodes (LED) along one vertical edge and along one horizontal edge of the frame. The opposite vertical and horizontal edges contain light detectors. These detectors are used to record which beams are interrupted when the panel is touched.

An electrical touch panel is constructed with two transparent plates separated by a small distance. One of the plates is coated with a conducting material, and the other plate is coated with a resistive material. When the outer plate is touched, it is forced into contact with the inner plate. This contact creates a voltage drop across the resistive plate that is converted to the coordinate values of the selected screen position.

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

**Light Pen:** Touch pencil-shaped devices are used to select screen positions by detecting the light coming from points on the CRT screen. They are sensitive to the short burst of light emitted from phosphor coating at the instant the electron beam strikes a particular point. An activated light pen, pointed at a spot on the screen as the electron beam lights up that spot, generates an electrical pulse that causes the coordinate position of the electron beam to be recorded. As with cursor-positioning devices, recorded light-pen coordinates can be used to position an object or to a processing option.

**Disadvantages:**

* When a light pen is pointed at the screen, part of the screen image is obscured by the hand and pen
* Prolonged use of the light pen can cause arm fatigue.
* Light pen requires special implementations for some applications because they cannot detect positions within black areas.
* Light pens sometimes give false reading due to background lighting in a room

**Voice System:** Speech recognizers are used in some graphics workstations as input devices accept voice commands. The voice-system input can be used to initiate graphics operations or to enter data. These systems operate by matching an input against a pre defined dictionary of words and phrases. A dictionary is set up for a particular operator by having the operator speak the command words to be used into the system. Each word is spoken several times, and the system analyzes the word and establishes a frequency pattern for that word in the dictionary along with the corresponding function to be performed. Later, when a voice command is given, the system searches the dictionary for a frequency-pattern match. Voice input is typically spoken into a microphone mounted on a headset.

## 1.8 Hard Copy Devices

The quality of the pictures obtained from a device depends on dot size, the number of dots per inch, or lines per inch, that can be displayed. Printers produce output by either impact or nonimpact methods. Impact printers press formed character faces against an inked ribbon onto the paper. Line printer is an example of an impact device. Nonimpact printers and plotters use laser techniques, ink-jet sprays, xerographic processes (as used in photocopying machines), electrostatic methods, and electro thermal methods to get images paper.

Impact printers press formed character faces against an inked ribbon onto the paper. A line printer is an example of an impact device, with the typefaces mounted on bands, chains, drums, or wheels. Nonimpact printers and plotters use laser techniques, ink-jet sprays, xerographic processes (as used in photocopying machines), electrostatic methods, and electro thermal methods to get images onto paper. Character impact printers often have a dot-matrix print head containing a rectangular array of protruding wire pins, with the number of pins depending on the quality of the printer. Individual characters or graphics patterns are obtained by retracting certain pins so that the remaining pins form the pattern to be printed.

In a laser device, a laser beam creates a charge distribution on a rotating drum coated with a photoelectric material, such as selenium. Toner is applied to the drum and then transferred to paper.

Ink jet printer creates an image directly on the paper by spraying ink through tiny nozzles. Compare to laser printers, the operating cost of an ink jet printer is relatively low. Expensive maintenance is rare, and the only part that needs routine replacement is the ink cartridge. Ink jet printers offer a cost-effective way to print in color. Color ink jet printers have four ink nozzles: Cyan (blue), magenta (red), yellow and black. These four colors are used in almost all color printing. Note that colors are different from primary additive colors used in monitors. Printed color is the result of light bouncing off the paper, not color transmitted directly from light source.

Laser printers are more expensive than ink jet printers, their print quality is higher, and most are faster. As their name implies, a laser is at the heart of these printers. A separate CPU and memory are built into the printer to interpret the data that it receives from the computer and to control the laser. The rotating mirror of laser printer reflects the laser and aim at any point on the drum, creating an electrical charge on the drum. Toner which is composed of tiny particles of oppositely charged ink, sticks to the drum in the places laser has charged. Then with pressure and heat, the toner is transferred off the drum onto the paper. The amount of special memory that the laser printer contains determines the speed at which documents are printed.

A plotter is a special kind of output device. Plotter is used to print large-format images such as construction or engineering drawings created in a CAD system. Early plotters were bulky, mechanical device that used robotic arms, which literally drew the image on a piece of paper.

## 1.9 Graphics Software

There are two general classifications for graphics software: general programming packages and special-purpose applications packages. A general graphics programming package provides an extensive set of graphics functions that can be used in a high-level programming language such as C or FORTRAN. An example of a general graphics programming package is the GL (Graphics Library) system on Silicon Graphics equipment. Basic functions in a general package include those for generating picture components (straight lines, polygons, circles, and other figures), setting color and intensity values, selecting views, and applying transformations. By contrast, application graphics packages are designed for nonprogrammers, so that users can generate displays without worrying about how graphics operations work. The interface to the graphics routines in such packages allows users to communicate with the programs in their own terms. Examples of such applications packages are the artist's painting programs and various business medical and CAD systems.

**Coordinate representation:** With few exceptions, general graphics packages are designed to be used with Cartesian coordinate specifications. If coordinate values for a picture are specified in some other reference frame (spherical, hyperbolic, etc.), they must be converted to Cartesian coordinates before they can be input to the graphics package. Special-purpose packages may allow use of other coordinate frames that are appropriate to the application. In general; several different Cartesian reference frames are usedto construct and display a scene. We can construct the shape of individual objects, such as trees or furniture**,** in a scene within separate coordinate reference frames called modeling coordinates, or sometimes local coordinates or master coordinates. Once individual object shapes have been specified, we can place the objects into appropriate positions within the scene using a reference frame called world coordinates. Finally, the world-coordinate description of the scene is transferred to one or more output-device reference frames for display. Thesedisplay coordinate systems are referred to as device coordinates or screen coordinates in the case of a video monitor. Modeling and world coordinate definitions allow us to set any convenient floating-point or integer dimensions without being hampered by the constraints of a particular output device. Generally, a graphics system first converts world-coordinate positions to normalized device coordinates, in the range from 0 to 1, before final conversion to specific device coordinates.

**Graphics Functions:** Ageneral-purpose graphics package provides users with a variety of functions for creating and manipulating pictures. These routines can be categorized according to whether they deal with output, input, attributes, transformations, viewing, or general control. The basic building blocks for pictures are referred to as output primitives. They include character strings and geometric entities, such as points, straight lines, curved **Lines,** filled areas etc. Attributes are the properties of the output primitives; that is, an attribute describes how a particular primitive is to be displayed. They include intensity and color specifications, line styles, text styles, and area-filling patterns.

We can change the size, position, or orientation of an object within a scene using geometric transformations. Similar modeling transformations are used to construct a scene using object descriptions given in modeling coordinates. Given the primitive and attribute definition of a picture in world coordinates, a graphics package projects a selected view of the picture on an output device. Viewingtransformations are used to specify the view that is to be presented and the portion of the output display area that is to be used**.** Interactive graphics applications use various kinds of input devices, such as a mouse, a tablet, or a joystick. Input functions are used to control and process the data flow from these interactive devices. Finally, a graphics package contains a number of housekeeping tasks, such as clearing a display screen and initializing parameter. We can lump the functions for carrying out these chores under the heading control operations.

**Software Standards:** The primary goal of standardized graphics software is portability. When packages are designed with standard graphics functions, software can he moved easily from one hardware system to another and used in different implementations and applications. International and national standards planning organizations in many countries have cooperated in an effort to develop a generally accepted standard for computer graphics. After considerable effort, this work on standards 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 the American National Standards Institute (ANSI). The second software standard to be developed and approved by the standards organizations was PHIGS (Programmer's Hierarchical Interactive Graphics standard), which is an extension of GKS. Standard graphics functions are defined as a set of specifications that is Independent of any programming language. Alanguage binding is then defined for a particular high-level programming language. This binding gives the syntax for accessing the various standard graphics functions from this language. Although PHIGS presents a specification for basic graphics functions, it does not provide a standard methodology for a graphics interface to output devices. Nor does it specify methods for storing and transmitting pictures. Separate standards have been developed for these areas. Standardization for device interface methods is given in the Computer Graphics Interface (CGI) system. And the Computer Graphics Metafile (CGM) system specifies standards for archiving and transporting pictures.

**1.10 Assignemt-1**

# Short Answer Questions (Each carries 2 marks)

1. Define refresh in case of CRT
2. What are horizontal and vertical retrace?
3. What are two categories of flat panel displays? Give example for each.
4. Write two drawbacks of DVST.
5. Define resolution.
6. Define persistence.
7. Define aspect ratio.
8. Write any two drawbacks of light pens.
9. What are impact and non-impact printers? Give example for each.
10. Mention the two advantages of raster graphic system over vector graphics systems (M. U. May/ June 2010, 2012)
11. Write a note on Data glove.
12. What are the advantages and disadvantages of DVST?
13. What are normalized device coordinates?
14. Expand GKS, DVST, PHIGS, CGM, DDA and CGI (1 mark each)
15. What are the different measures to find quality of picture obtained from a device?
16. List out any four hard copy devices.
17. Define world and screen coordinates.
18. Differentiate between active matrix LCD and Passive Matrix LCD.

# Long Answer Questions

1. Explain the architecture of Raster display system with neat diagram. (5 Marks)
2. Explain the architecture of Vector display system or random display system with neat diagram. (5 Marks)
3. Differentiate raster and vector display systems. (5 Marks)
4. List advantages and drawbacks of raster and vector display systems. (5 Marks)
5. List the drawbacks of DVST. (3 Marks)
6. Write a note on graphic monitors and workstations.
7. Explain Track ball, Joystick, Spaceball, Data Glove and digitizer. (2 marks each) 8. Write a note on image scanners. (3 Marks)
8. What are the drawbacks of light pens? (4 Marks)
9. Write a note on voice recognition or voice system. (3 Marks)
10. Explain different types of hard copy devices. (6 Marks)
11. Explain graphics functions. (4 Marks)
12. Write a note on graphics standards. (5 Marks)
13. Write a short note on key board and mouse (6 Marks).
14. Write a note on graphic softwares (4 Marks).
15. Write a short note on color CRT Monitor (5 Marks)

# UNIT-I Chapter 2 Output Primitives

## 2.1 Point and Lines

Point plotting is accomplished by converting a single coordinate position furnished by an application program into appropriate operations for the output device in use. With a CRT monitor, for example, the electron beam is turned on to illuminate the screen phosphor at the selected location. How the electron beam is positioned depends on the display technology. A random-scan (vector) system stores point-plotting instructions in the display list, and coordinate values in these instructions are converted to deflection voltages that position the electron beam at the screen locations to be plotted during each refresh cycle. For a black and-white raster system, on the other hand, a point is plotted by setting the bit value corresponding to a specified screen position within the frame buffer to 1. Then, as the electron beam sweeps across each horizontal scan line, it emits a burst of electrons (plots a point) whenever a value of 1 is encountered in the frame buffer. With an RGB system, the frame buffer is loaded with the color codes for the intensities that are to be displayed at the screen pixel positions.

Line drawing is accomplished by calculating intermediate positions along line path between two specified endpoint positions. An output device is then directed to fill in these positions between the endpoints. For analog devices, such a vector pen plotter or a random-scan display, a straight line can be drawn smoothly from one endpoint to the other. Linearly varying horizontal and vertical deflection voltages are generated that are proportional to the required changes in the x and y directions to produce the smooth line.

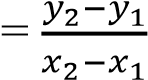
Digital devices display a straight line segment by plotting discrete points between the two endpoints. Discrete coordinate positions along the line path are ca1culated from the equation of the line. For a raster video display, the line color (intensity) is then loaded into the frame buffer at the corresponding pixel coordinates. Reading from the frame buffer, the video controller then "plots" the screen pixels. Screen locations are referenced with integer values, so plotted positions may only approximate actual line positions between two specified endpoints. A computed line position of (10.48, 20.51), for example, would be converted to pixel position (l1, 21).This rounding of coordinate values to integers causes lines to be displayed with a stairstep appearance. The characteristic stairstep shape of raster lines is particularly noticeable on systems with low resolution, and we can improve their appearance somewhat by displaying them on high-resolution systems.

For the raster-graphics, object positions are specified directly in integer device coordinates. For our convenience, we will assume that pixel positions are referenced according to scan-line number and column number (pixel position across a scan line). Scan lines are numbered consecutively from 0, left to right across each scan line.

To load a specified color into the frame buffer at a position corresponding to column x along scan line y, we can use a procedure, *setpixel (x, y).* We sometimes will also want to be able to retrieve the current frame-buffer intensity setting for a specified location by using method *getpixel (x, y).*

## 2.2 Line-drawing algorithms

The Cartesian slope-intercept equationfor a straight line is y=m.x + b …. (2.1)

with m representing the slope of the line and b as the y intercept. Given that the twoendpoints of a line segment are specified at positions *(*x1,y1)and *(*x2,y2), as shown in Figure 2.1**,** we can determine values for the slope m and y intercept b with the following calculations m  …… (2.2)

b=𝑦1- m. 𝑥1 …… (2.3)

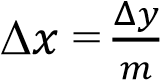


# Figure 2.1: Line path between endpoint positions *(*x1,y1) and *(*x2,y2)

Algorithms for displaying straight line are based on the line equation 2.1 and the calculations given in Equations 2.2 and 2.3. For any given *x*interval Δ𝑥 along a line, we can compute the corresponding y interval from 2.2 as

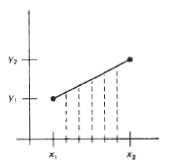
Δ𝑦 = m. Δ𝑥 ------- (2.4)

Similarly, we can obtain the *x*interval Δ𝑥 corresponding to a specified Δ𝑦 as

 …….. (2.5)

Theseequations form the basis for determining deflection voltages in analog devices. For lines with slope magnitudes | m |< 1**,** Δ𝑥can beset proportional to a small horizontal deflection voltage and the corresponding vertical deflection is set proportional to Δ𝑦 as calculated from Eq**.** 2.4. For lines whose slopes have magnitudes | m | > 1**,** Δ𝑦can beset proportional to a small vertical deflection voltage with the corresponding horizontal deflection voltage set proportional to Δ𝑥**,** calculated from Eq. 2.5. For lines with *m* = 1, Δ𝑥= Δ𝑦and the horizontal and vertical deflections voltages are equal. In each case, a smooth line with slope m is generated between the specified endpoints.

On Raster systems, lines are plotted with pixels and the step size horizontal and vertical directions are constrained by pixel separations. That is, we must "sample" a line at discrete positions and determine the nearest pixel to the line at each sampled position. This is called as scan conversion process, for a straight line is illustrated in Figure 2.2**,** for a near horizontal line with discrete sample positions along the xaxis.



# Figure 2.2: Straight line segment with five sampling positions along the x axis between

𝒙𝟏 **and** 𝒙𝟐***.***

## 2.2.1 DDA Algorithm Derivation

We know that the slope of a straight line is given as m = *y*  *y*2  *y*1 ... (1) *x x*2  *x*1

For any given x interval ∆x along, a line, we can compute the corresponding y interval ∆y from equation (1) as

*y*  *y*2 *y*1*x* …(2) *x*2 *x*1

Similarly, we can obtain the x interval ∆x corresponding to a specified ∆y as

*x*  *x*2  *x*1 *y* …(3) *y*2  *y*1

Once the intervals are known the values for next x and next y on the straight line can be obtained as follows xi+1= xi+∆x

*x*2  *x*1 *y* …( 4) and

= *xi* + 

*y*2  *y*1

yi+1=yi+∆y

= *yi* + *y*2  *y*1*x* … (5)

*x*2  *x*1

The equations 4 and 5 represent a recursion relation for successive values of x and y along the required line. Such a way of rasterizing a line is called a digital differential analyzer (DDA). For simple DDA either ∆x or ∆y, whichever is larger, is chosen as one raster unit, i.e. if *x* *y* then *x* 1 else *y* 1

With this simplification if *x* 1 then we have

*yi*1= *yi* + *y*2  *y*1*x* and *xi*1  *xi* 1 *x*2  *x*1

If *y* 1then we have *yi*1  *yi* 1 and *xi*1= *xi* + *x*2 *x*1 *x* *y*2 *y*1

# DDA line algorithm

1. Read the line end points (x1,y1 ) and (x2,y2) such that they are not equal.

[if equal then plot that point and exit]

1. If *x*2*x*1 >= *y*2*y*1 then length= *x*

else length= *y*

end if

1. *x* =(x2-x1)/length

*y* = (y2-y1)/length

This makes either *x* or *y* equal to 1 because the length is either | x2-x1| or |y2-y1|, the incremental value for either x or y is 1.

1. x=x1+0.5 \* sign( *x* ) y=y1+0.5\*sign( *y* )

[Here the sign function makes the algorithm work in all quadrants. It returns -1, 0,1 depending on whether its argument is <0, =0, >0 respectively. The factor 0.5 makes it possible to round the values in the integer function rather than truncating them]

1. i=1 [begins the loop, in this loop points are plotted]

1. while(ilength) begin

Plot (Integer(x), Integer(y)) x= x+∆x y= y+∆y i=i+1

end

1. stop

The DDA algorithm is a faster method for calculating pixel positions than the direct use of Equation. It eliminates the multiplication in Equation by making use of raster characteristics, so that appropriate increments are applied in the x or y direction to step to pixel positions along the line path. The accumulation of round off error in successive additions of the floating-point increment, however, can cause the calculated pixel positions to drift away from the true line path for long line segments. Furthermore, the rounding operations and floating-point arithmetic in procedure line DDA are still time-consuming. We can improve the performance of the DDA algorithm by separating the increments m and l/m into integer and fractional parts so that all calculations are reduced to integer operations.

## 2.2.2 Bresenham's Line Algorithm

Bresenham’s line algorithm uses only integer addition and subtraction and multiplication by 2, and we know that the computer can perform the operations of integer addition and subtraction very rapidly. The computer is also time-efficient when performing integer multiplication by powers of 2. Therefore, it is an efficient method for scan-converting straight lines.

The basic principle of Bresenham’s line algorithm is to select the optimum raster locations to represent a straight line.

Assume that we have just selected the pixel P(xi, yi) and now we have to select between the pixel one increment to the right or one increment to the right and one increment up. That is we have to choose between (xi+1, yi) or (xi+1, yi+1)

In Bresenham’s algorithm, the difference between pixel A to pixel C and pixel B to pixel C is calculated and sign of the difference is used to select the pixel whose distance from pixel C is smaller as the best approximation to the line.

The slope intercept form of the line is given by **y=mx + c.**

Distance between pixel A to pixel C given by **d1=y - yk**

Distance between pixel B to C is given by **d2= yk+1 – y**

# Therefore d1=m(xk+1)+c-yk --- 1)

d**2**= y**k**+1 –(m(x**k**+1)+c) **---2)**

d**1**-d**2**= m(x**k**+1)+c-y**k**-y**k**-1+m(x**k**+1)+c

But m=dy/dx

(d**1**-d**2**) dx =2dy(x**i**+1) -2dxy**i**+2cdx-dx

# di=2dy(xi+1) -2dxyi+2cdx-dx ---3)

To find the initial decision variable at (x**0**, y**0**),

di=2dy(xi+1) -2dxyi+2cdx-dx We know that **y=mx + c** and **c=y-mx** d0=2dy(x0+1)-2dxy0+2(y0-mx0) dx – dx

*dx*

We know that m=

*dy*

*dx*

d0=2dy(x0+1)-2dxy0+2(y0- x0) dx - dx *dy*

# d0=2dy-dx ---4)

**case 1:** if di < 0 means d1 < d2. We have chosen pixel A. and the value of di+1=d+2dy . Here we are increment x by one unit.

**Case 2:** if di > 0 means d1 > d2. In this case we choose pixel B. and the value of di+1=.di-2dx Here we increment x value by one unit and y value by one unit.

**Algorithm:**

1. Read the line end points (x1, y1) and (x2, y2) such that they are not equal.
2. dx=x2-x1, dy=y2-y1
3. initialize the starting point x=x1, y=y1
4. d=2dy-dx
5. while(x<x2) begin

if (d<0) begin x=x+1 d=d+2dy end else x=x+1 y=y+1 d=d-2dx

end

plot (x,y)

end

6. stop

## 2.3 Loading the frame Buffer

When straight line segments and other objects are scan converted for display with a raster system, frame-buffer positions must be calculated. We have assumed that this is accomplished with the setpixel procedure, which stores intensity values for the pixels at corresponding addresses within the frame-buffer array. Scan-conversion algorithms generate pixel positions at successive unit intervals. This allows us to use incremental methods to calculate frame-buffer address.

As a specific example, suppose the frame-buffer array is addressed in row major order and that pixel positions vary from (0, 0) at the lower left screen corner to (xmax, ymax) at the top right corner. For a bi level system (l bit per pixel), the frame-buffer bit address for pixel position (x, y) is calculated as addr(x, y) = addr (0, 0) + y (𝑥𝑚𝑎𝑥 + 1) + *x* (2.3.1)

Moving across a scan line, we can calculate the frame-buffer address for the pixel at (x+1, y) as the following offset from the address for position (x, y):

addr(x +1, y) = addr (x, y) + 1 (2.3.2)

Stepping diagonally up to the next scan line from (x, y), we get to the frame-buffer address of

(x + 1, y + 1) with the calculation

addr(x +1, y+1) = addr (x, y) + 𝑥𝑚𝑎𝑥 + 2 (2.3.3)

Where the constant 𝑎𝑥 + 2 is precomputed once for all line segments.

Similar incremental calculations can be obtained from Equation (2.1.1) for unit steps in the negative x and y screen directions. Each of these address calculations involves only a single integer addition.

Methods for implementing the setpixel procedure to store pixel intensity values depend on the capabilities of a particular system and the design requirements of the software package. With systems that can display a range of intensity for each pixel, frame-buffer address calculations would include pixel width (number of bits), as well as the pixel screen location.

## 2.4 Line Function

A procedure for specifying straight-line segments can be set up in a number of different forms.

The two-dimensional line function is defined as follows. polyline (n, wcPoints)

where parameter n is assigned an integer value equal to the number of coordinate positions to be input, and wcPoints is the array of input world-coordinate values for line segment endpoints. This function is used to define a set of n-1 connected straight line segments. Because series of connected line segments occur more often than isolated line segments in graphics applications, polyline provides a more general line function. 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.

Example: As an example of the use of polyline, the following statements generate two connected line segments, with endpoints at (50, 100), (150, 250), and (250, 100):

wcPoints[l].x = 50; wcPoints[l].y = 100; wcPoints[2].x = 150; wcPoints[2].y = 250; wcPoints[3].x = 250; wcPoints[3].y = 100; polyline (3, wcPoints);

Coordinate references in the polyline function are stated as absolute coordinate values. This means that the values specified are the actual point position in the coordinate system in use.

## 2.5 Circle Generating Algorithm

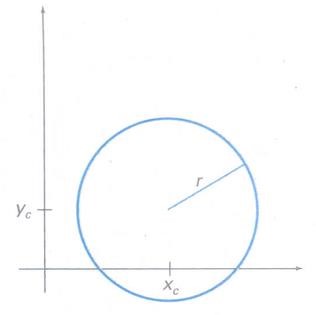
Since the circle is a frequently used component in pictures and graphs, a procedure for generating either full circles or circular arcs is included in most graphics packages. More generally, a single procedure can be provided to display either circular or elliptical curves.

# Properties of Circles

A circle is defined as the set of points that are all at a given distance r from a center position

(𝑥𝑐, 𝑦𝑐 ). This distance relationship is expressed by the Pythagorean theorem in Cartesian coordinates as

(x − 𝑥𝑐)2 + (y − 𝑦𝑐)2 = r2 (1)

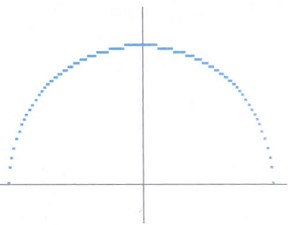


# Figure 2.3: Circle with center coordinates (𝒙𝒄, 𝒚𝒄) and radius r

We could use this equation to calculate the position of points on a circle circumference by stepping along the x axis in unit steps from 𝑥𝑐 - r to 𝑥𝑐 + r and calculating the corresponding y values at each position as

y = 𝑦𝑐 ± √𝑟2 − (𝑥𝑐 – x ) 2 (2)

But this is not the best method for generating a circle. One problem with this approach is that it involves considerable computation at each step. Moreover, the spacing between plotted pixel positions is not uniform. We could adjust the spacing by interchanging x and y (stepping through y values and calculating x values) whenever the absolute value of the slope of the circle is greater than 1. But this simply increases the computation and processing required by the algorithm.



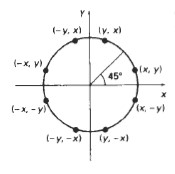
# Figure 2.4: Positive half of a circle plotted with Equation 2 and with (𝐱𝐜, 𝐲𝐜) = (0,0)

Another way to eliminate the unequal spacing shown in above figure is to calculate points along the circular boundary using polar coordinates r and e. Expressing the circle equation in parametric polar form yields the pair of equations

x = xc + r cos 𝜃 y = yc + r sin 𝜃

When a display is generated with these equations using a fixed angular step size, a circle is plotted with equally spaced points along the circumference. The step size chosen for 𝜃 depends on the application and the display device.

A circle is a symmetrical figure. It has eight-way symmetry as shown in the Figure. Thus, any circle generating algorithm can take advantage of the circle symmetry to plot eight points by calculating the coordinates of any one point. For example, if one point is calculated with a circle algorithm, seven more points could be found just by reflection. We can generate the circle section in the second quadrant of the xy plane by noting that the two circle sections are symmetric with respect to they axis. And circle sections in the third and fourth quadrants can be obtained from sections in the first and second quadrants by considering symmetry about the xaxis. We can take this one step further and note that there is also symmetry between octants. Circle sections in adjacent octants within one quadrant are symmetric with respect to the 45' line dividing the two octants. Taking advantage of the circle symme**t**ryin this way we can generate all pixel positions around a circle by calculating only the points within the sector from *x* = *0*to x = y.



# Figure 2.5: 8-way symmetry of a circle

More efficient circle algorithms are based on incremental calculation of decision parameters, as in the Bresenham’s line algorithm, which involves only simple integer operations. A method for direct distance comparison is to test the halfway position between two pixels to determine if this midpoint is inside or outside the circle boundary.

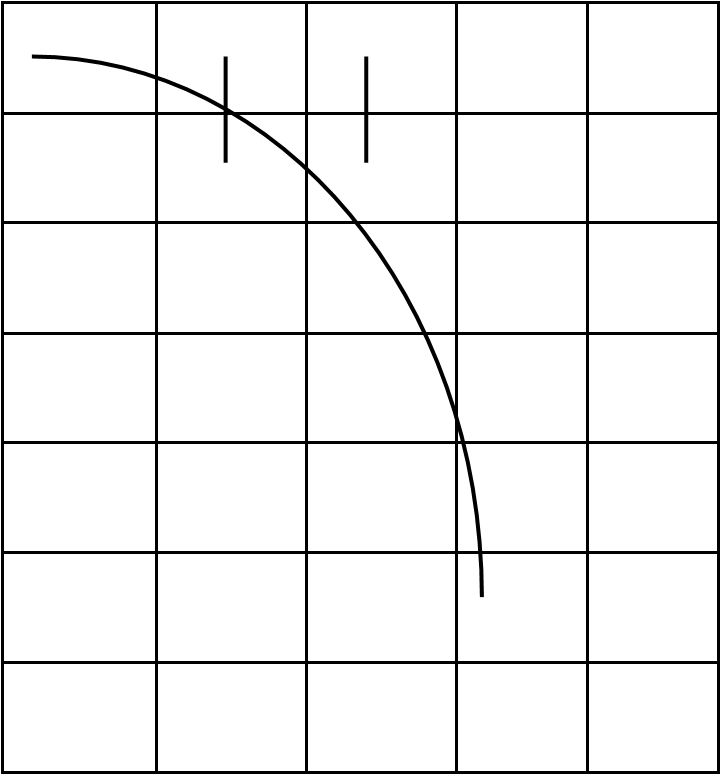
# Midpoint circle drawing Algorithm

The midpoint circle drawing algorithm uses the eight-way symmetry of the circle to generate it. It plots 1/8 part of the circle i.e from 900 to 450 as shown in the figure 3.18. as circle is drawn from 90 to 45 degree the x moves in the positive direction and y moves in the negative direction. To draw 1/8 part of the circle, we take unit steps in the positive x direction and make use of decision parameter to determine which of the 2 possible y position is closer to the circle path at each step.

In the above figure the 2 possible y positions are yi and yi-1 at xi+1. Therefore we have to determine whether the pixel at (xi+1, yi) or at position (xi+1, yi-1) is closer to the circle. The decision of choosing between these 2 pixels based on the value of a decision variable. For this we have to find the midpoint coordinate values and we have to substitute these values in the

circle function f(x,y)=x2+y2-r2 evaluated at the midpoint between these 2 pixels

|  |  |
| --- | --- |
| *di*1 *di*  3 2*xi* | …. (2) |
| We can write *d i*1 *di*  3 2(*xi*1 1) | *xi*1  *xi* 1 and xi=xi+1-1 |
| *di*1 *di* 1 2*xi*1 | **… (3)** |

*di*  *f* (*xi* 1, *yi*  1)  (*xi* 1)2  (*yi*  1)2 *r*2

2 2

*di*  *xi*2  *yi*2  2*xi*  *yi*  5 *r*2 -1)

4

if di < 0, we choose pixel A, di > 0, we choose B; and if di =0, we can choose either A or B

Now calculate di+1. There are 2 cases

# Case 1: if di < 0 ( if A is chosen)

The value of d for the next grid (di+1) depends on whether we chose A or B. To calculate di+1, again we have to find the midpoint coordinate values. If we have chosen pixel A, the midpoint coordinate values are (xi+2, yi-  ). Substituting these values in the circle function we get,

1.  2 (*yi*  1)2 *r*2 *f* (*xi*  2, *yi*  )=*d i* 1  (*xi*  2) 
2. 2

 *xi*2  4 4*xi*  *yi*2  1  *yi* *r*2 4

= *xi*2  *yi*2  4*xi*  *yi* *r*2

# Case 2: if di >0(if B is chosen)

If we have chosen pixel B, the midpoint coordinate values are (xi+2, yi-  ). Substituting these values in the circle function we get,

3  2 (*yi*  3)2 *r*2 *f* (*xi*  2, *yi*  )  *d i* 1  (*xi*  2) 

2 2

*d i*1  *xi*2  4  4*xi*  *yi*2  9 3*yi* *r*2 4

*di*1 *di* 5 2*xi*  2*yi* … (4)

We can write *di*1 *di*  5 2*xi*  2*yi* di+1 = di + 5 +2(xi+1-1)- 2(1+yi+1) *xi*1  *xi* 1 and yi+1=yi-1

*di*1 *di* 1 2*xi*1  2*yi*1 … (5)

The initial value of the decision parameter can be obtained by evaluating circle function at the start position (x0, y0) = (0,r)

*d*0  *f* ((01)2  (*r*  1)2 *r*2

2

1. 2 *r*2

*d*0 1 (*r*  ) 

1. *d*0 1.25 *r*

# Algorithm

1. read the radius r of the circle
2. initialize starting position as x=0, y=r
3. calculate the initial value of the decision parameter as d=1.25-r
4. do

{

Plot( x, y)

If(d<0) {

x=x+1 d=d+2x+1

}

Else

{

x=x+1 y=y-1 d=d+2x+1

} while(x < y)

1. determine symmetry points
2. stop

## 2.6 Ellipse Generating Algorithm

Loosely stated, an ellipse is an elongated circle. Therefore, elliptical curves can be generated by modifying circle-drawing procedures to take into account the different dimensions of an ellipse along the major and minor axes.

# Properties of Ellipse

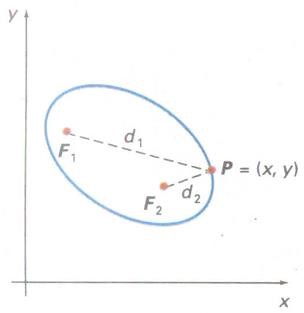
An ellipse is defined as the set of points such that the sum of the distances from two fixed positions (foci) is the same for all points. If the distance to the two foci from any point P = (x, y)

on the ellipse are labeled 𝑑1and 𝑑2, then the general equation of an ellipse can be stated as

𝑑1 + 2 = constant ( 1 )

Expressing distances d} and d2 in terms of the focal coordinates 𝐹1 = (𝑥1, 𝑦1) and 𝐹2= (𝑥1, 𝑦1) and 𝐹2= (𝑥2, 𝑦2), we have

√(x − x1) 2 + (y − y1) 2 + √(x − x2) 2 + (y − y2) 2 = constant (2)



# Figure 2.6: Ellipse generated about Foci 𝑭𝟏and 𝑭𝟐

By squaring this equation, isolating the remaining radical, and then squaring again, we can rewrite the general ellipse equation in the form

𝐴𝑥2 + 𝐵𝑦2 + Cxy + Dx + Ey + F =0 (3)

where the coefficients A, B, C, D, E, and F are evaluated in terms of the focal coordinates and the dimensions of the major and minor axes of the ellipse. The major axes are the straight line segment extending from one side of the ellipse to the other through the foci. The minor axis spans the shorter dimension of the ellipse, bisecting the major axis at the halfway position (ellipse center) between the two foci.

# Midpoint Ellipse Drawing Algorithm

Midpoint Ellipse Drawing Algorithm

We use the 4-way symmetry of the ellipse to draw only one fourth of it, i.e the ellipse is produced in the first quadrant. The other three parts can be drawn making use of the symmetry.

The one fourth of the ellipse, i.e the part of the ellipse in the first quadrant is split into two parts at the place where the slope of the curve is -1. (The slope of the curve at a point is defined as the slope of the tangent to the curve at the point.)

Drawing the first part is exactly similar to a circle, i.e we increment x every time but, we decrement y depending on the decision variable. We increment x here because, in the initial parts of the curve, we have more points in x direction, than in Y direction. The number of points in the X go on decreasing as the slope becomes more and more negative. When this slope reaches a value of -1, the number of points available in X and Y become the same.

Once the slope becomes less than -1, the number of points in Y is more than that in X. Hence, in the second part, we give unit steps to y (decrement y) every time, and increment X depending on the decision variable.

Derivation: The ellipse equation is given byb2a2+a2y2-a2b2

# Region 1( slope > -1)

In the above diagram, the 2 possible y values are yi and yi-1 at xi+1. We have to determine whether the pixel at (xi+1, yi) or (xi+1, yi-1) is closer to the ellipse boundary. The decision of choosing between these 2 points depends on the value of a decision variable. So it is necessary to find the midpoint coordinate values and substitute these values in the ellipse function f(x, y)= byb2a2+a2y2-a2b2.  The midpoint coordinate values are (*xi* 1, *yi*  1) . Substituting these values

2

in the function f(x, y)= byb2a2+a2y2-a2b2 we get,

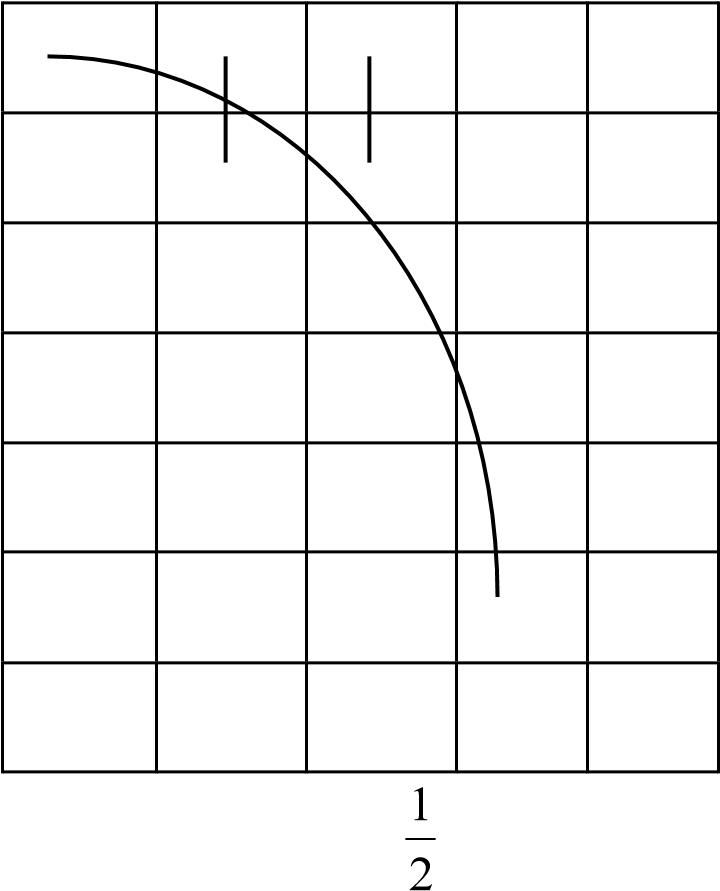
1. 2(*xi* 1)2  *a*2(*yi*  1)2 *a*2*b*2 *f* (*xi* 1, *yi*  )  *b*
2. 2

*di*  *b*2(*xi* 1)2  *a*2(*yi*  1)2 *a*2*b*2 **… (1)**

2

If the value of the decision variable di < 0, choose A, di >0 choose B, if di=0, choose either A or

# B. To calculate di+1: Case 1: if di < 0

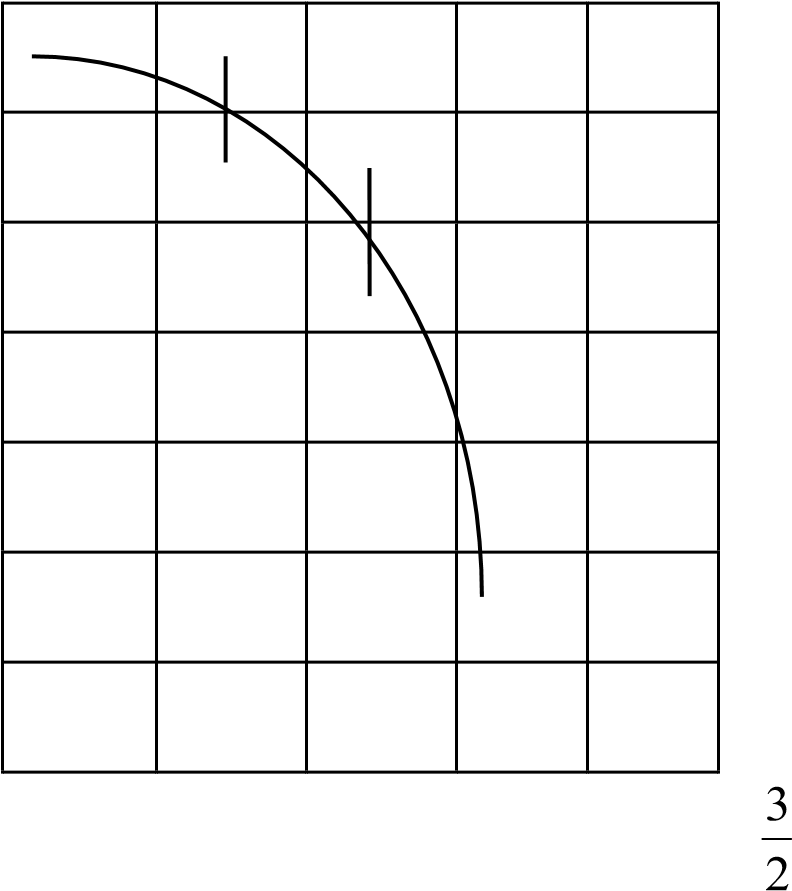
**The midpoint coordinate values are** (*xi*  2, *yi*  ) . Substituting these values in the ellipse function we get,

*di*1  *b*2(*xi*  2)2  *a*2(*yi*  1)2 *a*2*b*2

2 Subtract *di*1 *di* we get,

*di*1 *di* *b*2(3 2*xi* ) **…(2)**

**Case 2: if di > 0.**

**To calculate di+1,** the midpoint coordinate values are(*xi*  2, *yi*  ) . Substituting these value

in the ellipse function

*di*1  *b*2(*xi*  2)2 *a*2(*yi*  32)2 *a*2*b*2 Subtract *di*1 *di* we get, *di*1 *di* *b*2(3 2*xi* ) *a*2(2 *yi* ) **…(3)**

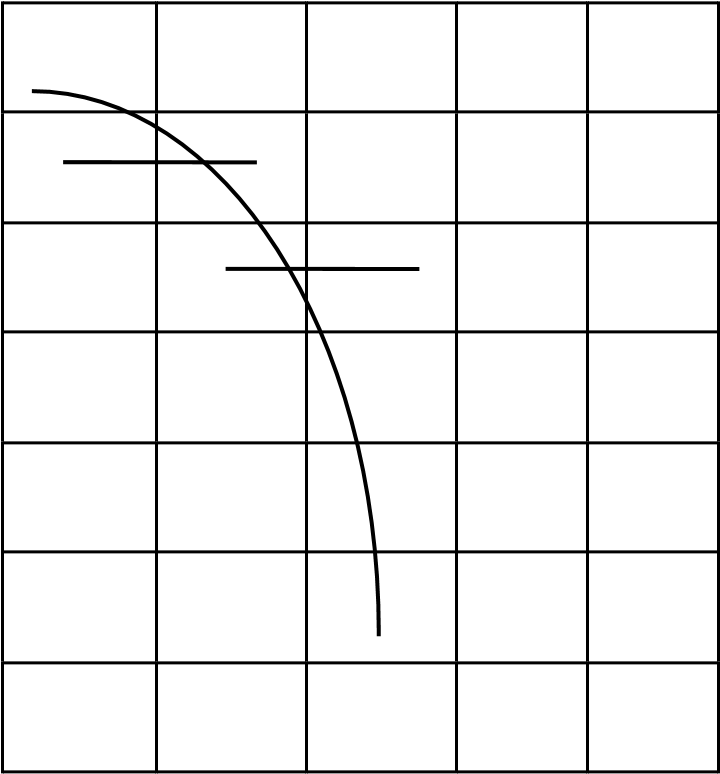
Now find the initial decision value for the region 1. The initial values are x=0, y=b. substituting these values in equation (1) we get**,**

*di*  *b*2(*xi* 1)2  *a*2(*yi*  1)2 *a*2*b*2  *b*2(01)2  *a*2(*b* 1)2 *a*2*b*2

2 2

*d*1 *b*2 *a*2*b* 0.25*a*2 …(4)

# Region 2(Slope < -1)

Once the slope becomes less than -1, the no. of points in the y axis is more than x. hence in the 2nd region we get unit steps in y( i.e decrement y) and increment x depending on the value of the decision variable.

1

# To calculate di the midpoint coordinate values are (*xi*  , *yi* 1) . So 2

*di*  *b*2(*xi*  1)  *a*2(*yi* 1)2 *a*2*b*2 **…(5)**

2

# Case 1: di < 0

we know that, if di < 0, midpoint is inside the ellipse boundary, so darken the right square. To

3 calculate di+1, we have to again find the midpoint coordinate values. They are(*xi* , *yi*  2) 2

*di*1  *b*2(*xi*  3)  *a*2(*yi*  2)2 *a*2*b*2 … (6)

2

Subtract *di*1 *di* we get,

*di*1 *di* *b*2(2*xi*  2) *a*2(3 2*yi* ) … (7)

# Case 2 if di > 0

Here midpoint is outside the ellipse boundary. So Left Square should be darkened. To calculate

1 di+1, find the midpoint coordinate values. They are(*xi*  , *yi*  2). 2

*di*1  *b*2(*xi*  1)  *a*2(*yi*  2)2 *a*2*b*2 …(8)

2

Subtract *di*1 *di* we get,

*di*1  *di*  *a*2(3 2*yi* ) …(9)

To find the initial value of the decision variable in the region 2, the values are at (x,y). substituting these values in di we get

*di*  *b*2(*xi*  1)2  *a*2(*yi* 1)2 *a*2*b*2 2

*d*2 *b*2(*xi*  0.5)2 *a*2(*yi* 1)2 *a*2*b*2

# Algorithm

1. Read the major axis diameter a and minor axis diameter b of the ellipse.
2. Initialize starting position as x=0 and y=b
3. Calculate the initial value of the decision variable for the 1st region as d1=b2-(a2b) + (0.25a2)
4. Plot(x , y)
5. While(a2(y-0.5)>b2(x+1))

Begin

If (d1<0)

Begin

d1=d1+b2 (2x+3)

X=x+1

End

Else

Begin

d1=d1+b2 (2x+3)+a2(2-2y)

y=y-1 x=x+1

End

Plot(x, y)

End

1. Initialize the decision variable for the 2nd region as d2=b2(x+0.5)2+a2(y-1)2-a2b2
2. While(y>0)

Begin

If ( d2 < 0)

Begin

d2=d2+b2(2x+2)+a2(3-2y)

x=x+1 y=y-1 End

Else

Begin

d2=d2+a2(3-2y)

y=y-1

End

Plot(x, y)

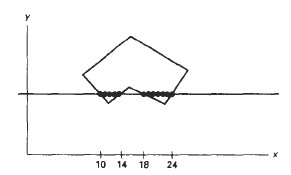
End

## 2.7 Filled-Area primitives

A standard output primitive in general graphics packages is a solid-color or patterned polygon area. Other kinds of area primitives are sometimes available, but polygons are easier to process since they have linear boundaries. There are two basic approaches to area filling on raster systems. One way to fill an area is to determine the overlap intervals for scan lines that cross the area. Another method for area filling is to start from a given interior position and paint outward from this point until we encounter the specified boundary conditions. This can-line approach is typically used in general graphics packages to fill polygons, circles, ellipses, and other simple curves. Fill methods starting from an interior point are useful with more complex boundaries and in interactive painting systems. Calculations performed in scan-conversion and other graphics algorithms typically take advantage of various coherence properties of a scene that is to be displayed. What we mean by coherence is simply that the properties of one part of a scene are related in some way to other parts of the scene so that the relationship can be used to reduce processing. Coherence methods often involve incremental calculations applied along a single scan line or between successive scan lines.

# Scan-Line Polygon F ill Algorithm

For each scan line crossing a polygon, the area-fill algorithm locates the intersection points of the scan line with the polygon edges. These intersection points are then sorted from left to right, and the corresponding frame-buffer positions between each intersection pair are set to the specified fill color.



# Figure 2.7: Interior pixels along a scan line passing through a polygon area

Calculations performed in scan-conversion and other graphics algorithms typically take advantage of various coherence properties of a scene that is to be displayed. Coherence is simply that the properties of one part of a scene are related in some way to other parts of the scene so that the relationship can be used to reduce processing. Coherence methods often involve incremental calculations applied along a single scan line or between successive scan lines.

# Inside-Outside Tests

Area-filling algorithms and other graphics processes often need to identify interior regions of objects. To identify interior regions of an object graphics packages normally use either:

1. Odd-Even rule
2. Nonzero winding number rule

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

1. Draw a line from any position P to a distant point outside the coordinate extents of the object and counting the number of edge crossings along the line.
2. If the number of polygon edges crossed by this line is **odd** then P is an **interior** point.

Else

P is an **exterior** point

# Nonzero Winding Number Rule

Counts the number of times the polygon edges wind around a particular point in the counterclockwise direction. This count is called the winding number, and the interior points of a two-dimensional object are defined to be those that have a nonzero value for the winding number.

1. Initializing the winding number to 0.
2. Imagine a line drawn from any position P to a distant point beyond the coordinate extents of the object.

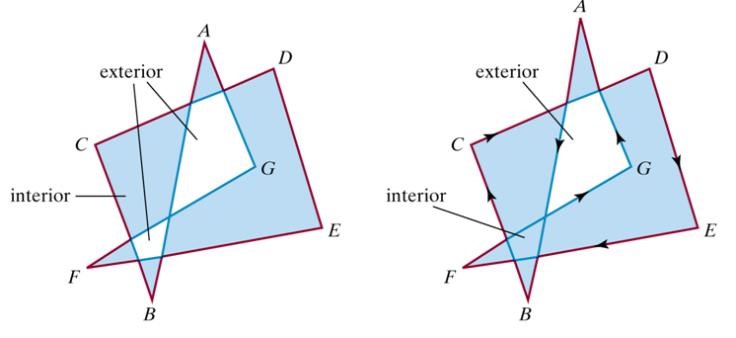
# Nonzero Winding Number Rule

3. Count the number of edges that cross the line in each direction. We add 1 to the winding number every time we intersect a polygon edge that crosses the line from right to left, and we subtract 1 every time we intersect an edge that crosses from left to right. 4. If the winding number is nonzero, then

*P* is defined to be an interior point

Else

*P* is taken to be an exterior point**.**



**(a) odd even rule (b) nonzero winding number rule**

# Figure 2.8: Identifying interior and exterior regions for a selfintersecting polygon

**Polygon Filling:** Filling the polygon means highlighting all the pixels which lie inside the polygon with any color other than background color. There are 2 basic approaches used to fill the polygon. One way to fill a polygon is to start from a given “seed point” known to be inside the polygon and highlight outward from this point i.e neighboring pixels until we encounter the boundary pixel. This approach is called **seed fill** because color flows from the seed pixel until reaching the polygon boundary. Another approach to fill the polygon is to apply the inside test i.e to check whether pixel is inside the polygon or outside the polygon and then highlight pixels which lie inside the polygon. This approach is known as scan line algorithm.

**Seed fill:** This seed fill algorithm is further classified as flood-fill algorithm and boundary fill algorithm. Algorithms that fill interior defines regions are called **flood-fill** algorithms, those that fill boundary-defined regions are called **boundary-fill** algorithms or edge-fill algorithms

**Boundary Fill algorithm:** In this method edges of a polygon are drawn. Then starting with some seed point, any point inside the polygon we examine the neighboring pixels to check whether the boundary pixel is reached. If boundary pixels are not reached, pixels are highlighted and the process is continued until boundary pixels are reached.

Boundary defined regions may be either 4-connected or 8-connected as shown in the fig if a region is 4-connected, then every pixel in the region may be reached by a combination of moves in only 4 directions: left, right, up and down. For an 8-connected region every pixel in the region may be reached by a combination of moves in the 2 horizontal, 2 vertical and 4 diagonal directions



Procedure: boundary\_fill(x,y, fillcolor, boundarycolor)

{

If(getpixel(x,y)!=boundarycolor && getpixel(x,y)!=fillcolor)

{

Putpixel(x,y,fillcolor)

Boundaryfill(x+1, y, fillcolor,boundarycolor)

Boundaryfill(x, y+1, fillcolor,boundarycolor)

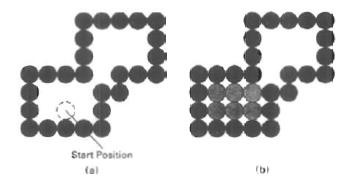
Boundaryfill(x-1, y, fillcolor,boundarycolor)

Boundaryfill(x, y-1, fillcolor,boundarycolor)

}

}

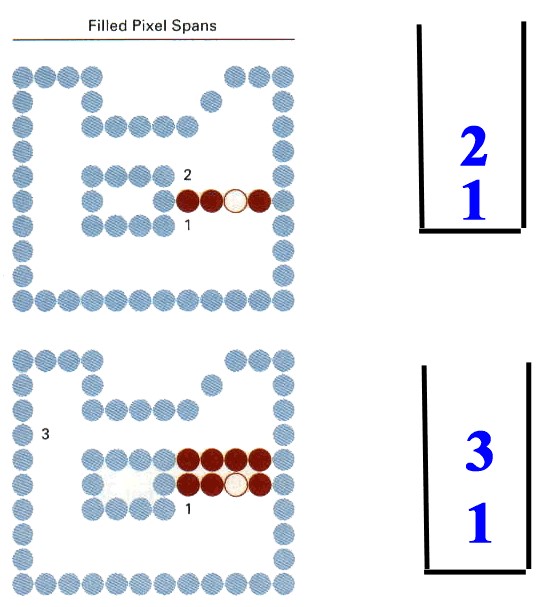
**getpixel()** gives the color of specified pixel and **putpixel()** draws the pixel with specified color.

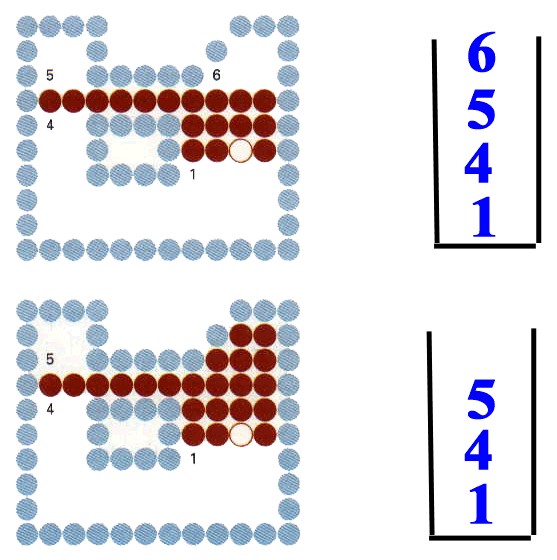


**Figure 2.9: The area defined within the color boundary (a) is only partially filled in (b) using a 4-connected boundary-fill algorithm.**

4-connected and 8-connected methods involve heavy recursion which may consume memory and time. More efficient methods are used. These methods fill horizontal pixel spans across scan line. This called a Pixel Span method. We need only stack a beginning position for each horizontal pixel span, instead of stacking all unprocessed neighboring positions around the current position, where spans are defined as the contiguous horizontal string of positions.

Boundary fill across pixel spans for a 4-connected area. First figure shows the filled initial pixel span, showing the position of the initial point (open circle) and the stacked positions for pixel spanson adjacent scan lines. Second figureshows Filled pixel span on the first scan line above the initial scan Line and the current contents of the stack. Third shows Filled pixel spans on the first two scanlines above the initial scanline and the current contents of the stack. Fourth figure shows completed pixel spans for the upper-right portion of the defined region and the remaining stacked positions to be processed.





**Flood Fill algorithm:** Sometimes it is required to fill an area that is not defined within a single color boundary. In such cases we can fill areas by replacing a specified interior color instead of searching for a boundary color. This approach is called flood fill algorithm. Here we start with some seed and examine the neighboring pixels. However, here pixels are checked for a specified interior color instead of a boundary color. And they are replaced by new color. Using either a 4connected or 8-connected approach we can step through pixel positions until all interior points have been filled. The following procedure illustrates the recursive method for filling 8-connected region using flood fill algorithm

Procedure: flood\_fill(x,y,old\_color, new\_color)

{

If (getpixel(x,y)=old\_color))

{

putpixel(x,y,new\_color) flood\_fill(x+1, y, old\_color, new\_color) flood\_fill(x-1, y, old\_color, new\_color) flood\_fill(x, y+1, old\_color, new\_color) flood\_fill(x, y-1, old\_color, new\_color) flood\_fill(x+1, y+1, old\_color, new\_color) flood\_fill(x-1, y-1, old\_color, new\_color) flood\_fill(x+1, y-1, old\_color, new\_color) flood\_fill(x-1, y+1, old\_color, new\_color)

}

}

**2.8 Assignment-2**

# Short Answer Questions (2 Marks Each)

1. Define scan conversion.
2. What is 8-way symmetry of a circle?
3. Write 8-way symmetry of a circle.
4. What is 4-way symmetry of an ellipse?
5. Write 4-way symmetry of an ellipse.
6. What are the different characteristics of a line?
7. Expand DDA. What is the equation for slope of the straight line?
8. Mention the drawback of DDA line algorithm (M. U. April/May 2011)
9. Write the two dimensional line functions with its syntax.
10. Explain eight way symmetry of a circle? (M. U. April/May 2012, 2011)
11. What is ellipse? Write the equation of ellipse.
12. Write boundary fill algorithm? (M. U. April/May 2011) 13. State the advantages of DDA. (M. U. April/May 2009)
13. Write flood fill algorithm.
14. List out two types line drawing algorithms.
15. What is coherence of a scene?
16. Write two drawbacks of DDA.
17. What are the two basic approaches to area filling on raster systems?
18. What are the two types of polygon filling?
19. What is the purpose of frame buffer?

# Long Answer Questions

1. Write Bresenham’s line drawing algorithm. (5 Marks)
2. Explain n different cases in scan-line polygon fill algorithm with suitable examples.
3. Explain odd-even rule and non-zero winding number rule.
4. Differentiate boundary fill and flood fill techniques.
5. What is a boundary fill algorithm? Explain (4 Marks-M. U. April/May 2013).
6. Derive midpoint circle algorithm (6 Marks-M. U. April/May 2013).
7. Explain the points and lines (5 Marks).
8. Write and explain the DDA algorithm (6 Marks-M. U. April/May 2012, 2011, 2010).
9. Explain the mid point algorithm for scan converting circles. (8 Marks-M. U. April/May 2008).
10. Explain about loading a frame buffer (4 Marks).
11. Write a note on filled area primitives. (4 Marks)
12. Explain flood fill algorithm (5 marks)
13. Write mid-point algorithm to draw an ellipse. (6 Marks) 14. Derive Mid-point algorithm to draw an ellipse.

15. Derive DDA algorithm (7 Marks)

# UNIT-II Chapter 3 Attributes of Output Primitives

## 3.1 Line Attribute

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

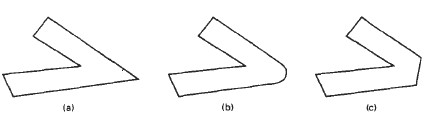
**Line type:** Possible selections for the line-type attribute include solid lines, dashed and dotted lines. We modify a line-drawing algorithm to generate such lines by setting the length and spacing of displayed solid sections along the path. To set line type attributes in a PHIGS application program, a user invokes function setLinetype (lt)

Where parameter lt is assigned a positive integer value of 1, 2, 3, or 4 to generate lines that are, respectively, solid, dashed, dotted, or dash-dotted. Other values for line-type parameter 1t could be used to display variations in the dot-dash patterns. .Once the line-type parameter has been set in a PHIGS application program all subsequent line-drawing commands produce lines with this line type. Raster line algorithms display line-type attributes by plotting pixel spans. For the various dashed, dotted, and dot-dashed patterns, the line-drawing procedure outputs sections of contiguous pixels along the line path, skipping number of intervening pixels between the solid spans. Pixel counts for length and inters pan spacing can be specified in a pixel mask, which is a string containing the digits 1 and 0 to indicate which positions to plot along path. The mask 1111000, for instance, could be used to display a dashed line with a dash length of four pixels and an interdash spacing of three pixels. On a bi-level system, the mask gives the bit values that should be loaded into the frame buffer along the line path to display the selected line type

**Line Width:** A line-width command is used to set the current line-width value in attribute list. This value is then used by line-drawing algorithms to control the thickness of lines generated with subsequent output primitive commands: setLinewidthScaleFactor (lw)

Line-width parameter 1w is assigned a positive number to indicate the width of the line to be displayed. A value of 1 specifies a standard-width line. On a pen plotter, for instance, a user could set lw to a value of 0.5 to plot a line whose width is half that of the standard line. Values greater than 1 produces lines thicker than the standard. While implementing width options using horizontal or vertical pixel spans is that the method produces lineswhose ends are horizontal or vertical regardless of the slope of the line. Thiseffect is more noticeable with very thick lines. We can adjust the shape of the lineends to give them a better appearance by adding line caps. One kind of line cap is the butt cap obtained by adjusting the end positions of the component parallel linesso that the thick line is displayed with square ends that are perpendicular to the line path. Another line cap is the round cap obtained by adding a filled semicircle to each butt cap. The circular arcs are centered on the line endpoints and have a diameter equal to the line thickness. Athird type of line cap is the projecting square cap. Here, we simply extend the line and add butt caps that are positioned one-half of the line width beyond the specified endpoints.

Displaying thick lines using horizontal and vertical pixel spans, for example, leaves pixel gaps at the boundaries between lines of different slopes where there is a shift from horizontal spans to vertical spans. We can generate thick polylines that are smoothly joined at the cost of additional processing at the segment endpoints. Amiter joinis accomplished by extending the outer boundaries of each of the two lines until they meet. Around join is produced by capping the connection between the two segments with a circular boundary whose diameter is equal to the line width. And a *bevel*join is generated by displaying the linesegments with buttcaps and filling in the triangular gap where the segments meet. If the angle between two connected line segments is very small, a miter join can generate a long spike that distorts the appearance of the polyline.



**Figure 3.1: Thick line segments connected with (a) miter join, (b) round join, and (c)bevel join.**

**Pen and Brush Option:** With some packages, lines can be displayed with pen or brush selections. Options in this category include shape, size, and pattern. The different shapes can be stored in a pixel mask that identifies the array of pixel positions that are to be set along the line path. To avoid setting pixels more than once in the frame buffer, we can simply accumulate the horizontal spans generated at each position of the mask and keep track of the beginning and ending x positions for the spans across each scan line. Lines generated with pen (or brush) shapes can be displayed in various widths by changing the size of the mask. For example, the rectangular pen line could be narrowed with a 2 × 2 rectangular mask or widened with a 4 × 4 mask.

**Line Color:** A polyline routine displays a line in the current color by setting this color value in the frame buffer at pixel locations along the line path using the setPixel procedure. The number of color choices depends on the number of bits available per pixel in the frame buffer. Weset the line color value in PHIGS with the function setPolylineColourIndex (lc)

Nonnegative integer values, corresponding to allowed color choices, are assigned to the line color parameter lc. A line drawn in the background color is invisible, and a user can erase a previously displayed line by re specifying it in the background color.

## 3.2 Curve Attributes

Parameters for curve attributes are the same as those for line segments. We display curves with varying colors, widths, dot-dash patterns, and available or brush options. Methods for adapting curve-drawing algorithms to accommodate attribute selections are similar to those for line drawing. The pixel masks discussed for implementing line-type options are also used in raster curve algorithms to generate dashed and dotted patterns. For example the mask 11100 produces the dashed circle. If we want to display constant-length dashes, we need to adjust the number of pixels plotted in each dash as we move around the circle circumference.

Raster curves of various widths can be displayed using the method of horizontal or vertical pixel spans.

Where the magnitude of the curve slope is less than 1, we plot vertical spans; where the slope magnitude is greater than 1, we plot horizontal spans.

Pen (or brush) displays of curves are generated using the same techniques discussed for straight line segments. We replicate a pen shape along the line path, for a circular arc in the first quadrant. Here, the center of the rectangular pen is moved to successive curve positions to produce the different shape. Curves displayed with a rectangular pen in this manner will thicker where the magnitude of the curve slope is 1.

## 3.3 Color and Grayscale levels

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

Ina color raster system, the number of color choices available depends on the amount of storage provided per pixel in the frame buffer. Also, color information can be stored in the frame buffer in two ways: We can store color codes directly in the frame buffer, or we can put the color codes in a separate table and use pixel values as an index into this table. With the direct storage scheme, when ever a particular color code is specified in an application program, the corresponding binary value is placed in the frame buffer for each component pixel in the output primitives to be displayed in that color. A minimum number of colors can be provided in this scheme with 3 bits of storage per pixel as shown in table 3.1.

Each of the three bit positions is used to control the intensity level (either on or off) of the corresponding electron gun in an RGB monitor. The leftmost bit controls the red gun, the middle bit controls the green gun, and the rightmost bit controls the blue gun. Adding more bits per pixel to the frame buffer increases the number of color choices.

Figure 3.1 illustrates a possible scheme for storing color values in a color lookup table (or video lookup table), where frame-buffer values are now used as indices into the color table. In this example, each pixel can reference anyone of the 256 table positions, and each entry in the table uses 24 bits to specify an RGB color.

For the color code 2081, a combination green-blue color is displayed for pixel location(x, y). Systems employing this particular lookup table would allow a user to select any 256 colors for simultaneous display from a palette 17 million colors. Compared to a full-color system, this scheme reduces number of simultaneous colors that can be displayed, but it also reduces the buffer storage requirements to 1 megabyte. Some graphics systems provide per pixel in the frame buffer, permitting a user to select 512 colors that used in each display.

Color tables are an alternate means for providing extended color capabilities to a user without requiring large frame buffers. Lower cost personal computer systems, in particular, often use color tables to reduce frame-buffer storage requirements.

**Table**

**3**

**.1:**

**The Eight color code for a three bit per pixel frame buffer**

**Figure**

**3**

**.**

**2**

**:**

**A Color Look table with 24 bit per entry accessed from a frame buffer with 8**

**bits per pixel.**

**Grayscale**

**:**

With monitors

that have no color capability, color

functions can be used in an

ap

plication program to set the shades of gray, or

grayscale, for displayed primi

tives. Numeric

values over the range from 0 to 1 can be used to specify grayscale

levels, which are then

converted to appropriate binary codes for

storage in the

raster. This

allows the intens

ity

settings

to be eas

ily adapted to systems with dif

fering grayscale

capabilities.

Table

3

.

lists the specifications for intens

2

ity codes for a four

-

level gray

scale system.

In this

example, any intensity input

value near 0.33 would be stored

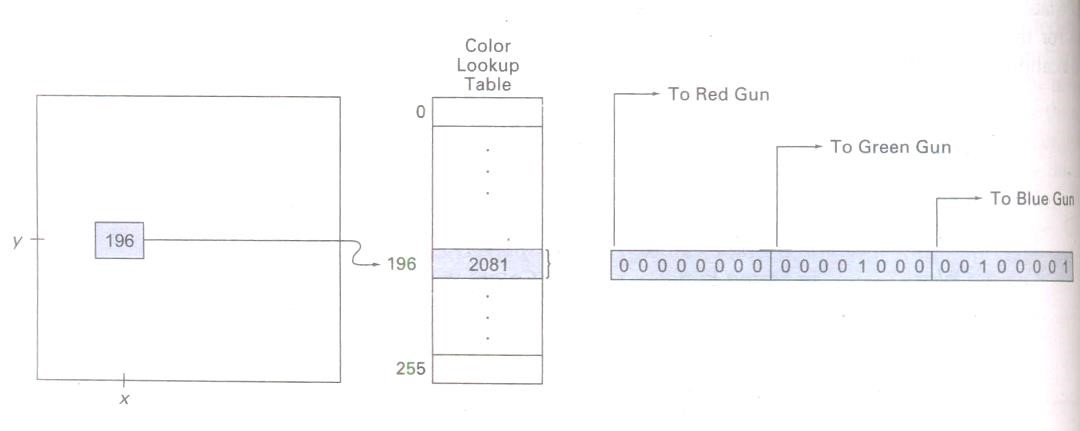
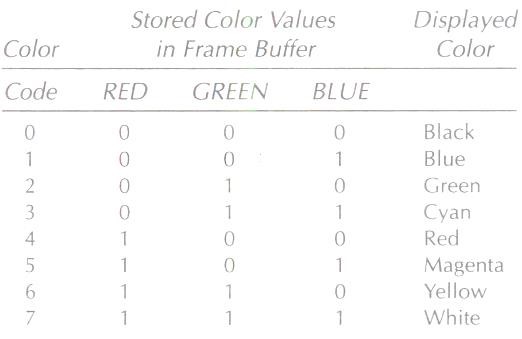
as

the

binary value 01 in the frame

buffer, and pixels with this value would be

displayed as dark gray. If additional bits per pixel are



available in the frame buffer, the value of 0.33 would be mapped to the nearest level. With 3 bits per pixel, we can accommodate 8 gray levels; while 8 bits per pixel would give us 256 shades of gray. An alternative scheme for storing the intensity information is to convert each intensity code directly to the voltage value that produces this grayscale level on the output device in use.

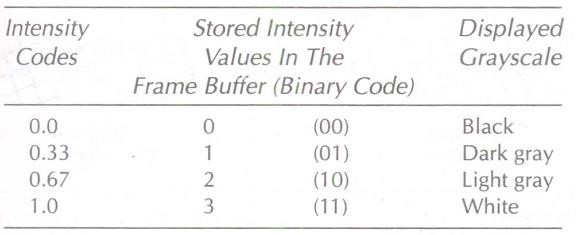
# Table 3.2: Intensity codes for a four-level Grayscale system

## 3.4 Area Fill Attribute

Options for filling a defined region include a choice between a solid color or a patterned fill and choices for the particular colors and patterns. These fill options can be applied to polygon regions or to areas defined with curved boundaries depending on the capabilities of the available package. In addition, areas painted using various brush styles, colors, and transparency parameters.

Fill styles: Areas are displayed with three basic fill styles: hollow with a color border, with a solid color, or filled with a specified pattern or design. A basic fill selected in a PHIGS program with the function setInteriorStyle (fs)

Values for the fill-style parameter fs include hollow, solid, and pattern. Another value for fill style is hatch, which is used to fill an area with selected hatching patterns- parallel lines or crossed lines. As with line attributes, a selected fill-style value is recorded in the list of system attributes and applied to fill the interiors of subsequently specified areas. Fill selections parrameter fs are normally applied to polygon areas, but they can also be mented to fill regions with curved boundaries.



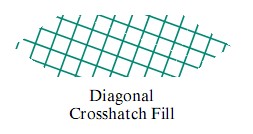
Hollow areas are displayed using only the boundary outline, with the interior color the same as the background color. A solid fill is displayed in a single color up to and including the borders of the region. The color for a solid interior or for a hollow area outline is chosen with setInteriorColourIndex (fc)

where fill-color parameter fc is set to the desired color code. A polygon hollow fill is generated with a line-drawing routine as a closed polyline. Solid fill of a region can be accomplished with the scan-line procedures.

Other fill options include specifications for the edge type, edge width, and edge color of a region. These attributes are set independently of the fills style or fill color, and they provide for the same options as the line-attribute parameters (line type, line width, and line color).

# Pattern fill

Fill patterns can be defined in rectangular color arrays that list different colors for different positions in the array. Or, a fill pattern could be specified as a bit array that indicates which relative positions are to be displayed in a single selected color. An array specifying a fill pattern is a mask that is to be applied to the display area. Some graphics systems provide an option for selecting an arbitrary initial position for overlaying the mask. From this starting position, the masks replicated in the horizontal and vertical directions until the display area are filled with no nonoverlapping copies of the pattern. Where the pattern overlaps specified fill areas, the array pattern indicates which pixels should be displayed in a particular color. This process of filling an area with a rectangular pattern is called tiling, and a rectangular fill pattern is sometimes referred to as a tiling pattern. Sometimes, pre defined fill patterns are available in a system, such as the hatch fill patterns shown in figure.



# Figure 3.3: Area filled with hatch patterns

We can implement a pattern fill by determining where the pattern overlaps those scan lines that cross a fill area. Beginning from a specified start position for a pattern fill, we map the rectangular patterns vertically across scanlines and horizontally across pixel positions on the scanlines. Each replication of the pattern array is performed at intervals determined by the width and height of the mask. Where the pattern overlaps the fill area, pixel colors are set according to the values stored in the mask.

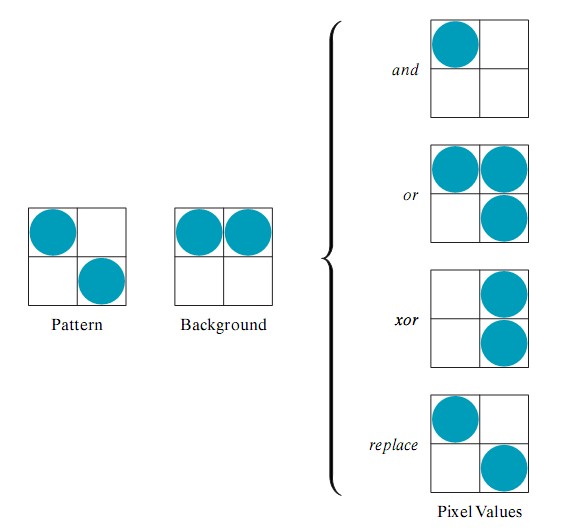
Hatch fill could be applied to regions by drawing sets of line segments to display either single hatching or crosshatching. Alternatively, hatch fill can be specified as a pattern array that produces sets of diagonal lines.

A reference point (𝑥𝑝, 𝑦𝑝) for the starting position of a fill pattern can be set at any convenient position, inside or outside the fill region. For instance, the reference point could be set at a polygon vertex. Or the reference point could be chosen as the lower left corner of the bounding rectangle (or bounding box) determined by the coordinate extents of the region. Always setting (𝑥𝑝, 𝑦𝑝) atthe coordinateorigin also simplifies the tiling operations when each element of a pattern is to be mapped to a single pixel.

**Color Blended Fill Regions:** It is also possible to combine a fill pattern with background colors in various ways. A pattern could be combined with background colors using a transparency factor that determines how much of the background should be mixed with the object color. Or we could use simple logical or replace operations. Figure 3.3 demonstrates how logical and replace operations would combine a 2 by 2 fill pattern with a background pattern for a binary (black-and-white) system

# Figure 3.4: Combining a fill pattern with a background pattern using logical operations and, or, and xor (exclusive or), and using simple replacement

Some fill methods using blended colors have been referred to as soft-fill or tint-fill algorithms. One use for these fill methods is to soften the fill colors at object borders that have been blurred to antialias the edges. Another application of a soft-fill algorithm is to allow repainting of a color area that was originally filled with a semitransparent brush, where the current color is then a mixture of the brush color and the background colors “behind” the area. In either case, we want the new fill color to have the same variations over the area as the current fill color.



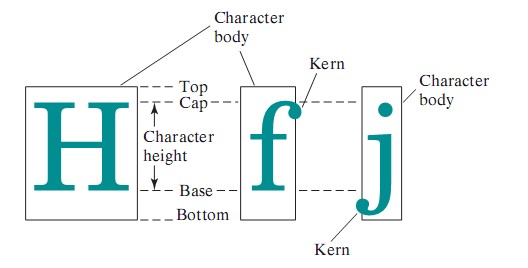
## 3.5 Character attributes

We control the appearance of displayed characters with attributes such as font, size, color, and orientation. In many packages, attributes can be set both for entire character strings (text) and for individual characters that can be used for special purposes such as plotting a data graph.

There are a great many possible text-display options. First of all, there is the choice of font (or type face), which is a set of characters with a particular design style such as New York, Courier, Helvetica, London, Times Roman, and various special symbol groups. The characters in a selected font can also be displayed with assorted underlining styles (solid, dotted, double), in boldface, in italics, and in outline or shadow styles.

Color settings for displayed text can be stored in the system attribute list and used by the procedures that generate character definitions in the frame buffer. When a character string is to be displayed, the current color is used to set pixel values in the frame buffer corresponding to the character shapes and positions.

We could adjust text size by scaling the overall dimensions (height and width) of characters or by scaling only the height or the width. Character size (height) is specified by printers and compositors in points, where one point is about 1/72 of a inch. The distance between the bottom line and the top line of the character body is the same for all characters in a particular size and typeface, but the body width may vary. Each character is positioned within the character body by a font designer in such a way that suitable spacing is attained along and between print lines when text is displayed with character bodies touching. Character height is defined as the distance between the baseline and the capline of characters. Kerned characters, such as f and j, typically extend beyond the character body limits, and letters with descenders (g, j, p, q, y) extend below the baseline.



# Figure 3.5: Examples of character bodies

Sometimes, text size is adjusted without changing the width-to-height ratio of characters with setCharacterHeight (ch)

Parameter ch is assigned a real value greater than 0 to set the coordinate height of capital letters. A character string can be displayed with three different character heights, while maintaining the ratio of width to height. The width only of text can be set with the function

setCharacterExpansionFactor (cw)

where character-width parameter cw is set to a positive real value that scales the body width of characters. Text height is unaffected by this attribute setting. Text can be displayed with a constant height and varying widths also.

Spacing between characters is another attribute that can often be assigned to a character string. Spacing between character is controlled separately with

setCharacterSpacing (cs)

where the character spacing parameter cs can be assigned any real value. The value assigned to cs determines the spacing between character bodies along print lines. A character string can be displayed with three different setting for the inter character spacing. The orientation for a character string can be set according to the direction of a character up vector.

setCharacterUpVector (upvact)

parameter upvect in this function is assigned two values that specify the x and y vector components. Text is then displayed so that the orientation of characters from baseline to capline is in the direction of the up vector. It is useful in many applications to be able to arrange character strings vertically or horizontally. An attribute parameter for this option is set with statement

setTextPath (tp)

where the text parameter tp can be assigned the value: right, left, up, or down. Character strings could also be oriented using a combination of up-vector and text-path specifications to produce slanted text.

Another possible attribute for character strings is alignment. This attributes specifies how text is to be displayed with respect to a reference position. Alignment attribute are set with

SetTextAlignment (h, v)

where parameter h and v control horizontal and vertical alignment, respectively. Horizontal alignment is set by assigning h a value of left, centre, or right. vertical alignment is set by assigning v a value of top, cap, half, base, or bottom.

In some graphics packages a text-precision attribute is also available. A precision specification for text display is given with

SetTextPrecision (tpr)

where text precision parameter tpr is assigned one of the values: string, char, or stroke. The highest-quality text is displayed when the precision parameter is the value stroke.

Finally, a library of text-processing routines often supplies a set of special characters, such as a small circle or cross, which are useful in various applications. Most often these characters are used as marker symbols in network layouts or in graphing data sets. The attributes for these marker symbols are typically color and size.

## 3.6 Assignment-3

**Short Answer Questions (Each carries 2 Marks)** 1. List the possible selections for Line Type attribute of a straight line

1. List any four character attributes.
2. What is the use of pixel mask while creating dashed lines? Give example
3. What is a line cap? List various types.
4. List the various joinsfor thick polyline and write the diagram to represent them
5. What is soft fill?
6. What is grayscale?
7. How the color for a solid interior or for a hollow area outline is chosen? 9. What are the two methods of pattern filling? (M. U. April/may 2011)
8. What is color blended fill region?
9. What are the two methods of fill style?
10. Write any two methods of character attributes?

# Long Answer Questions

1. List the different attributes of a straight line and explain any ONE in detail. (5 Marks)
2. Explain Line Type attribute of a straight line. (5 Marks)
3. Write a note on Line Width attribute of a straight line. (5 Marks)
4. Write a note on curve attributes. (5 Marks)
5. Explain how color values are stored in a color lookup table. (5 Marks)
6. What are the different basic fill styles for polygon areas? Explain. (5 Marks)
7. Explain color as an attribute of output primitive. (5 marks)
8. Explain grayscale as an attribute of output primitive (5 marks)
9. Explain area fill attribute (5 Marks)
10. Write a short note on pattern fill. (5 Marks)
11. Explain any five methods of text attribute (5 Marks).

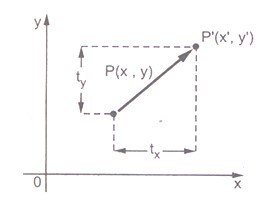
# UNIT-II Chapter 4 Two Dimensional Geometric Transformations

## 4.1 Basic Transformation

Almost all graphic systems allow the programmer to define picture that include a variety of transformations. In this chapter we discuss the 2D and 3D transformations primitives.

# 4.1.1Translation

A two dimensional translation is a process of changing the position of an object in a straight line path from one coordinate location to another.



# Figure 4.1: Translation

We can translate a two dimensional point by adding translation distances, tx and ty, to the original coordinate position (x,y) to move the point to a new position (*x*, *y*), as shown in figure. *x*  *x*  *tx* … (4.1.1) *y*  *y*  *t y* … (4.1.2)

The translation distance pair (tx, ty) is called a translation vector or shift vector.

It is possible to express the translation equations 4.1.1 and 4.1.2 as a single matrix equation by using column vectors to represent coordinate positions and the translation vector:

*x* *x* *tX* 

P=*y* *P*  *y* *T*  *ty* 



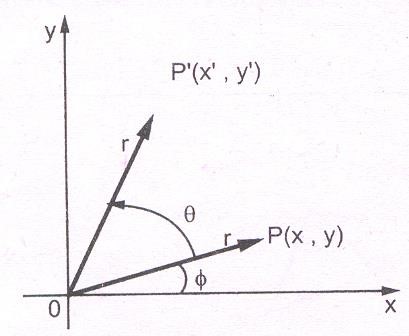
This allows us to write the two dimensional translation equations in the matrix form:

*P*  *P**T* … (4.1.3)

Translation is a rigid-body transformationthat moves objects without deformation. That is, every point on the object is translated by the same amount. A straight Line segment is translated by applying the transformation equation to each of the line endpoints and redrawing the line between the new endpoint positions will not change in size of the line. Polygons are translated by adding the translation vector to the coordinate position of each vertex and regenerating the polygon using the new set of vertex coordinates and the current attribute settings

## 4.1.2 Rotation

A two dimensional rotation is applied to an object by repositioning it along a circular path in the xy plane. To generate a rotation, we specify a rotation angle  and the position of the rotation point about which the object is to be rotated.



# Figure 4.2: Rotation of the object about the origin

Let us consider the rotation of the object about the origin. Here, r is the constant distance of the point from the origin, angle is the original angular position of the point from the horizontal, and is the rotation angle. Using standard trigonometric equations, we can express the transformed coordinates in terms of angle and  as

*x*  *r*cos()  *r*coscos*r*sinsin

… (4.1.4)

*y*  *r*sin()  *r*cossin *r*sincos

The original coordinates of the point in polar coordinates are given as x=r cos

y=r sin …(4.1.5)

Substituting equations 4.1.5 into 4.1.4 we get the transformation equations for rotating a point ( x,y) through an angle about the origin as:

*x*  *x*cos *y*sin

*y*  *x*sin *y*cos …(4.1.6)

The above equations can be represented in the matrix form as given below.

*x* *y* *x y* cossin cossin

*P*  *P*.*R* …(4.1.7)

Where R is the rotation matrix and it is given as

cos sin

R=sin cos ….( 4.1.8)



It is important to note that positive values for the rotation angle define counterclockwise rotations about the rotation point and negative values rotate objects in the clockwise sense. For negative values of  i.e for clockwise rotation, matrix becomes

 cos() sin()

R=sin() cos 



cos sin cos()  cos

=sin cos *and* sin() sin …(4.1.9)



Rotation of a point about an arbitrary pivot position or a point can be obtained using following formula

𝑥′ = 𝑥𝑟 + (x-𝑥𝑟) cos𝜃 - (y-𝑦𝑟) sin𝜃

𝑦′ = 𝑦𝑟 + (x-𝑥𝑟) sin𝜃 + (y-𝑦𝑟) cos𝜃

**Exercise:** Show that 90° rotation of unit vector along the x-axis produces a unit vector along yaxis.

The unit vector along x-axis is (1,0) and unit vector along y-axis is (0,1)

2D rotation formula is 𝑥′ = xcos𝜃 – ysin𝜃

𝑦′ = xsin𝜃 +ycos𝜃

𝑥′ = xcos90° – ysin90°

𝑦′ = xsin90° +ycos90°

𝑥′ = 1.0 – 0.1

𝑦′ = 1.1 +0.0

𝑥′ = 0

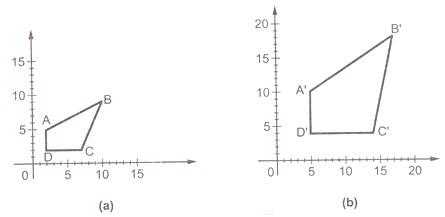
𝑦′ = 1 i.e. (1, 0) unit vector along y-axis

## 4.1.3 Scaling

A scaling transformation changes the size of an object. This operation can be carried out for polygons by multiplying the coordinate values ( x, y) of each vertex by scaling factors Sx and Sy to produce the transformed coordinates (*x*, *y*).

*x*  *x*.*Sx*

*and*  … (4.1.10) *y*  *y*.*Sy*



# Figure 4.3: Scaling

Scaling factor Sx scales object in the x direction and scaling factor Sy scales object in the y direction. The equations 4.1.10 can be written in the matrix form as given below:

*Sx* 0 

*x* *y* *x y*  0 *Sy* 

=*x*.*Sx y*.*S y*  … ( 4.1.11)

=P.S

Any positive numeric values are valid for scaling factors Sx and Sy. Values less than 1 reduce the size of the objects and values greater than 1 produce an enlarged object. For both Sx and Sy values equal to 1, the size of object does not change. To get uniform scaling it is necessary to assign same value for Sx and Sy. Unequal value for Sx and Sy result in a differential scaling. When Sx , and Sy, are assigned the same value, a uniformscaling is produced that maintains relative object proportions. Unequal values for Sx , and Sy, result in a differential scaling that is often used in design applications, whew pictures are constructed from a few basic shapes that can be adjusted by scaling and positioning transformations.

We can control the location of a scaled object by choosing a position, called the fixed point, that is to remain unchanged after the scaling transformation. Coordinates for the fixed point (𝑥𝑓, 𝑦𝑓,) can be chosen as one of the vertices, the object centroid, or any other position. Apolygon is then scaled relative to the fixed point by scaling the distance from each vertex to the fixed point. For a

vertex with coordinates (x, y) the scaled coordinates ( are calculated as

𝑥*′* = 𝑥𝑓 + (x-𝑥𝑓) , 𝑦*′* = 𝑦 + (y-𝑦𝑓)𝑠𝑦

We can rewrite these scaling transformations to separate the multiplicative and additive terms:

𝑥*′* = x. 𝑠𝑥 + 𝑥𝑓 (1-𝑠𝑥)

𝑦*′* = y. 𝑠𝑦 + 𝑦𝑓 (1-𝑠𝑦)

Where the additive terms 𝑥𝑓 (1-𝑠𝑥) and 𝑦𝑓 (1-𝑠𝑦) are constant for all points in the object.

## 4.2 Matrix Representations and Homogeneous Coordinates

|  |  |  |
| --- | --- | --- |
| For scaling: | Sx  P  P.   0   | 0  0       Sy  0 |

In design and picture formation process, many times we may require to perform translation, rotations, and scaling to fit the picture components into their proper positions. In the previous section we have seen that each of the basic transformations can be expressed in the general matrix form 𝑃′= P.M1 + M2 …… (4.2.1)

1 0 tx 

For translation: P' P.    

0 1 ty 

i.e. M1 = Identity matrix

M2 = Translation vector

 cos  sin 0

For rotation: P  P.    

sin cos  0

i.e. M1 = Rotational matrix

M2 = 0

i.e. M1 = Scaling matrix

M2 = 0

To produce a sequence of transformations with above equations, such as translation followed by rotation and then scaling, we must calculate the transformed co-ordinates one step at a time. First, coordinates are translated, then these translated coordinates are scaled, finally the scaled coordinates are rotated. But this sequential transformation process is not efficient. A more efficient approach is to combine sequence of transformations into one transformation so that the final coordinate positions are obtained directly from initial coordinates. This eliminates the calculation of intermediate coordinate values.

In order to combine sequence of transformations we have to eliminate the matrix addition associated with the translation term in M2 (Refer equation 4.2.1). To achieve this we have to represent matrix M1 as 3 × 3 matrix instead of 2 × 2 introducing an additional dummy coordinate W. Here, points are specified by three numbers instead of two. This coordinate system is called homogeneous coordinate system and it allows us to express all transformation equations as matrix multiplication. The homogeneous coordinate is represented by a triplet XW, YW, W, where

x  XW andy  YW

W W

For two dimensional transformations, we can have the homogeneous parameter W to be any non zero value. But it is convenient to have W = 1. Therefore, each two dimensional position can be represented with homogeneous coordinate as (x, y, 1).

|  |  |  |
| --- | --- | --- |
|    T   0 1 0     tx ty 1    Therefore, we have | …. (4.2.2) | |
|  1    x y1 x y1  0    tx | 0  1 ty | 0    0    1 |

Summering it all up, we can say that the homogeneous coordinates allow combined transformation, eliminating the calculation of intermediate coordinate values and thus save required time for transformation and memory required to store the intermediate coordinate values. Let us see the homogeneous coordinates for three basic transformations.

# Homogeneous Coordinates for Translation

The homogeneous coordinates for translation are given as

 1 0 0

= [x + tx y + ty 1 …. (4.2.3)

# Homogeneous Coordinates for Rotation

The homogeneous coordinates for rotation are given as

 cos sin 0

 

R  sin cos 0 …. (4.2.4)

 

 0 0 1

Therefore, we have

 cos  sin 0

 

x y 1  x y 1  sin cos  0

 

 0 0 1

= xcos  ysin xsin  ycos 1 …. (4.2.5)

# Homogeneous Coordinates for Scaling

The homogeneous coordinate for scaling are given as

Sx 0 0

 

S  0 Sy 0

 

 0 0 1

= x.Sx y.Sy 1 …. (4.2.6)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  1    T1.S.T   0     2 | 0 0 0.5 0 0 1 | | 1. 0      1. 0 2. 1 | |
|    1 0  0   2 1  0 |    0.5 0 0  0 1 2 |
|  | 0.5      0     1 | 1. 0 1      0.5 0 0  1 1 2 | 1. 0 0.5       1. 0   0 2. 1  1 | 0  0.5  1 | 0    0  1 |
|  | A 1 1      B  3 1  C 3 2  D 1 3   | 1   0.5 0  1     0 0.5  1  1 1  1  | 0    0  1 |  |  |
|  | 1.5  2.5     2.5    1.5 | 1.5 1  1.5 1     1. 1   2.5 1 |  |  |  |

Note: The object matrix is written first and it is then multiplied by the required transformation matrix. If we wish to write the transformation matrix first and then the object matrix we have to take the transpose of both the matrices and post-multiply the object matrix i.e.,

x 1 0 tx  x

     

y  0 1 ty  y

     

 1 0 0 1  1

Example: Find the transformation matrix that transforms the given square ABCD to half its size with centre still remaining at the same position. The coordinates of the square are: A(1, 1), B(3, 1), C(3, 2), D(1, 3) and centre at (2, 2). Also find the resultant coordinates of square.

Sol.: This transformation can be carried out in the following steps Translate the square so that its center coincides with the origin.

Scale the square with respect to the origin.

Translate the square back to the original position

Thus, the overall transformation matrix is formed by multiplication of three matrices.

## 4.3 Composite Transformations

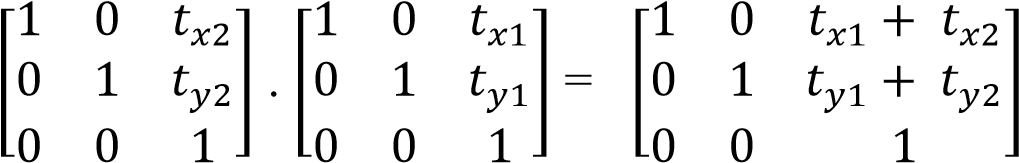
With the matrix representations of 2D transformations, we can set up a matrix or any sequence of transformations as a composite transformation matrix by calculating the matrix product of the individual transformations. Forming products of transformation matrices is often referred to as a concatenation, or composition of matrices. For column-matrix representation of coordinate positions, we form composite transformations by multiplying matrices in order from right to left. That is, each successive transformation matrix premultiplies the product of the preceding transformation matrices.

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

𝑃′ = T (𝑡𝑥2, 𝑡𝑦2) . { T (𝑡𝑥1, 𝑡𝑦1). P } …….. (4.3.1)

= { T (𝑡𝑥2, 𝑡𝑦2) . T (𝑡𝑥1, 𝑡𝑦1) } .P

Where P and 𝑃′ are represented as homogeneous-coordinate column vectors. We can verify this result by calculating the matrix product for the two associative groupings. Also, the composite transformation matrix for this sequence of translations is

 ………..(4.3.2)

or

T (𝑡𝑥2, 𝑡𝑦2) . T (𝑡𝑥1, 𝑡𝑦1) = T (𝑡𝑥1 + 𝑡𝑥2 , 𝑡𝑦1 + 𝑡𝑦2 ) ……….(4.3.3) which demonstrates that two successive translations are additive.

# Rotation

Two successive rotations applied to point p to produce the transformed position

𝑃′ = R(𝜃2) . { R(𝜃1) . P}

= { R(𝜃2) . R(𝜃1) } . P ……..(4. 3.4)

By multiplying the two rotation matrices, we can verify that two successive rotations are additive:

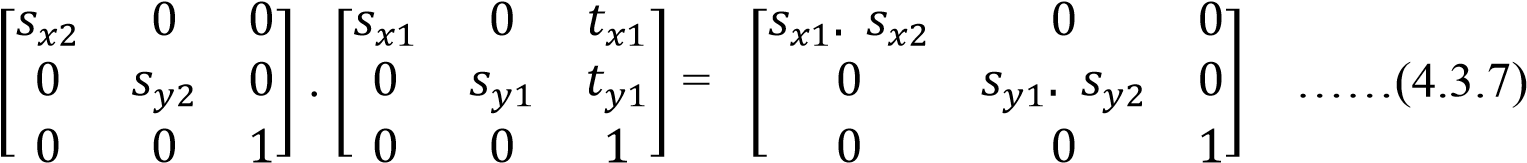
R(𝜃2) . R(𝜃1) = R(𝜃1 + 𝜃2 ) …………..(4.3.5)

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

𝑃′ = R(𝜃1 + 𝜃2 ) . P ……………..(4.3.6)

# Scaling

Concatenating transformation matrices for two successive scaling operation reduces the following composite scaling matrix:



or

S (𝑠𝑥2, 𝑠𝑦2) . S (𝑠𝑥1, 𝑡𝑦1) = S(𝑠𝑥1. 𝑠𝑥2 , 𝑠𝑦1. 𝑠𝑦2 ) …….(4.3.8)

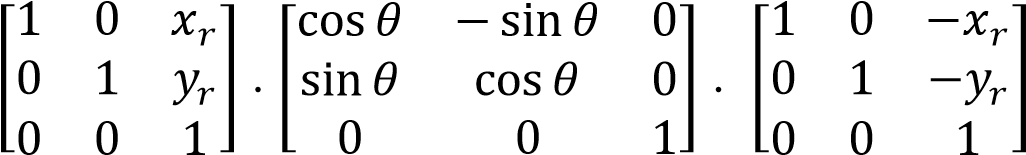
The resulting matrix in this case indicates that successive scaling operations are multiplicative. That is, if we were to triple the size of an object twice in succession, the final size would be nine times that of the original.

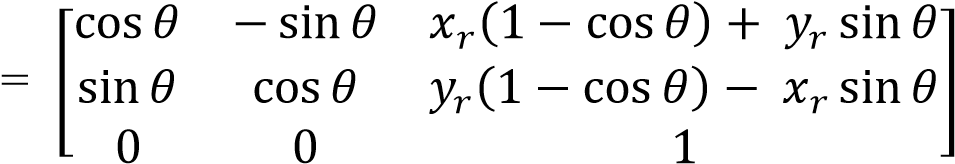
# General Pivot-Point Rotation

With a graphics package that only provides a rotate function for revolving objects about the coordinate origin, we can generate rotations about any selected pivot point (𝑥𝑟,) by performing the following sequence of translate-rotate-operations:

1. Translate the object so that the pivot-point position is moved to the coordinate origin.
2. Rotate the object about the coordinate origin.
3. Translate the object so that the pivot point is returned to its original position.

The composite transformation matrix for this sequence is obtained with concatenation.



 (4.3.9)

which can be expressed in the form

T ( , 𝑦𝑟) . R( 𝜃 ) . T(-𝑥𝑟 , -𝑦𝑟) = R(𝑥𝑟, 𝑦𝑟, 𝜃 ) (4.3.10)

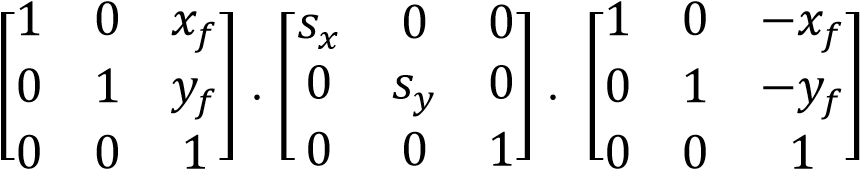
where T(-𝑥𝑟 , -𝑦𝑟 ) = 𝑇−1 (𝑥𝑟 , 𝑦𝑟 ). In general, a rotate function can be set up to accept parameters for pivot-point coordinates, as well as the rotation angle, and to generate automatically the rotation matrix of Eq. 4.3.9

# General Fixed Point Scaling

We can generate a transformation sequence to produce scaling with respect to a selected fixed position (𝑥𝑓,𝑦𝑓) using a scaling function that can only scale relative to the coordinate origin.

1. Translate object so that the fixed point coincides with the coordinate origin.
2. Scale the object with respect to the coordinate origin.
3. Use the inverse translation of step 1 to return the object to its original position.

Concatenating the matrices for these three operations produces the required scaling matrix

 (4.3.11)

or

T ( , 𝑦𝑓) . S (𝑠𝑥, 𝑠𝑦) . T((-𝑥𝑓 , -𝑦𝑓) = S(𝑥𝑓, 𝑦𝑓, 𝑠𝑥, 𝑠𝑦 ) (4.3.12)

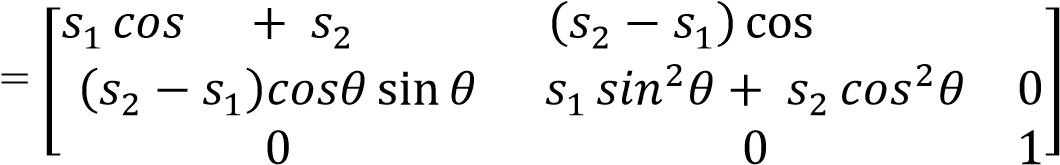
This transformation is automatically generated on systems that provide a scale function that accepts coordinates for the fixed point.

# General Scaling Directions

Parameters 𝑠𝑥 and 𝑠𝑦 scale objects along the x and y directions. We can scale an object in other directions by rotating the object to align the desired scaling directions with the coordinate axes before applying the scaling transformation.

Suppose we want to apply scaling factors with values specified by parameters 𝑠1 and 𝑠2 in the directions. To accomplish the scaling with out changing the orientation of the object, we first perform a rotation so that the directions for 𝑠1 and 𝑠2 coincide with the x and y axes, respectively. Then the scaling transformation is applied, followed by an opposite rotation to return points to their original orientations. The composite matrix resulting from the product of these three transformations is

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Reflection | Transfor matrix | mation | Original | Image | Reflected Image | |
| Reflection about y-axis | 1 0  0 1    0 0 | 0  0    1 |  |  |  |  |
|  |  |  |  |

R−1( 𝜃 ) . S(𝑠1, 𝑠2) . R(𝜃)

2𝜃 𝑠𝑖𝑛2𝜃 𝜃 sin 𝜃 0

(4.3.13)

# Concatenation Properties

Matrix multiplication is associative. For any three matrices, A, B, and C, the matrix product A. B. C can be performed by first multiplying A and B or by first multiplying B and C:

A . B. C = (A. B) . C = A. (B. C) (4.3.14)

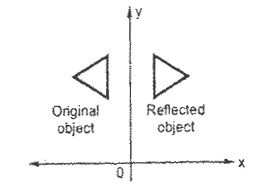
Therefore, we can evaluate matrix products using either a left-to-right or a right to-left associative grouping. On the other hand, transformation products may not be commutative: The matrix product A . B is not equal to B.A, in general. This means that if we want to translate and rotate an object, we must be careful about the order in which the composite matrix is evaluated. For some special cases, such as a sequence of transformations all of the same kind, the multiplication of transformation matrices is commutative.

## 4.4 Other Transformations

Basic transformations such as translation, rotation, and scaling are included in most graphics packages. Some packages provide a few additional transformations that are useful in certain applications. Two such transformations are reflection and shear.

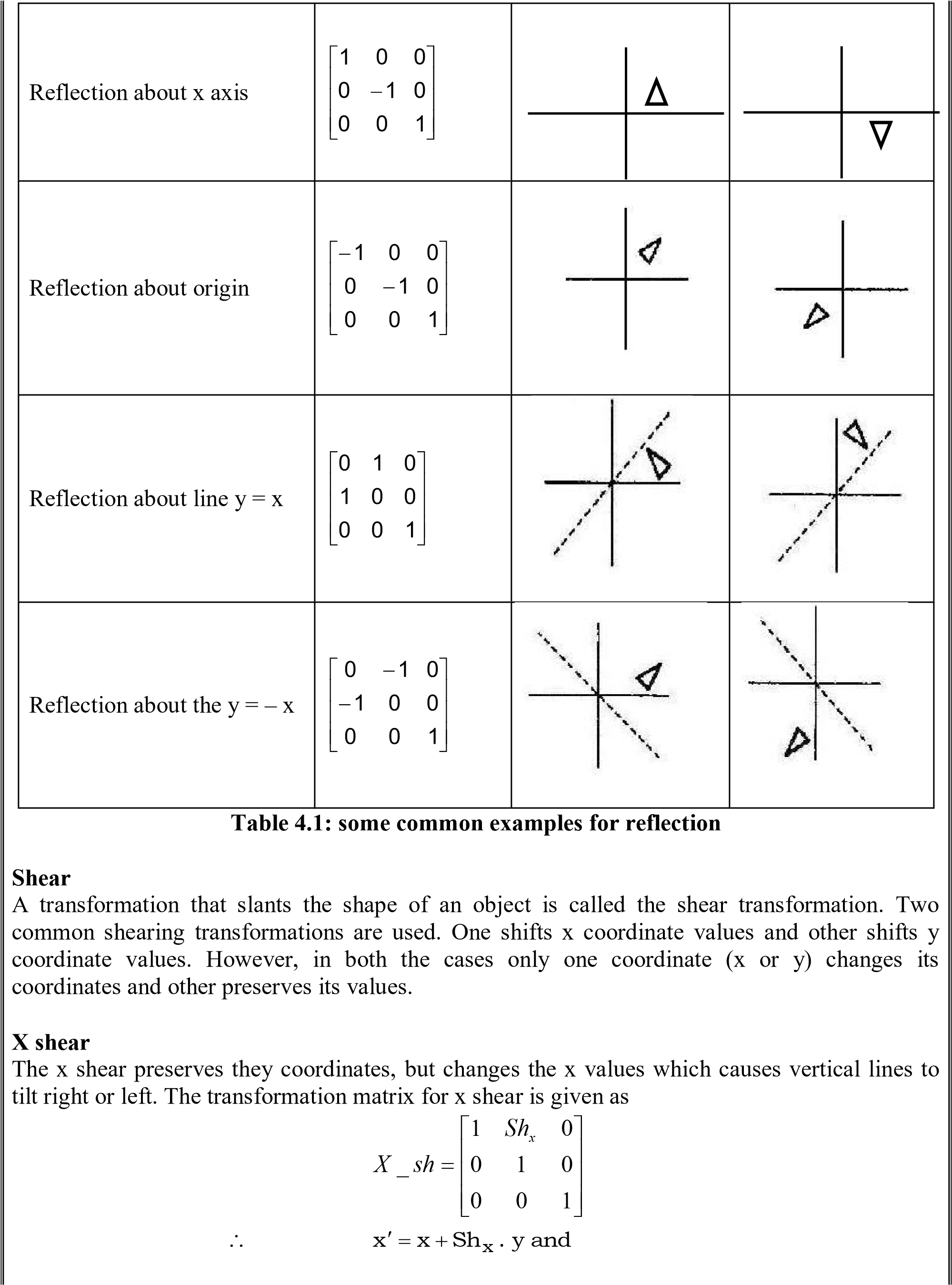
Reflection

A reflection is a transformation that produces a mirror image of an object relative to an axis of reflection. We can choose an axis of reflection in the xy plane or perpendicular to the xy plane.



# Figure 4.6: Reflection about y axis

Table 4.1 gives some examples of some common reflection



|  |  |  |
| --- | --- | --- |
|  1  X shear with y reference line :  Shx  Shx . yref | 0  1  0 | 0  0    1 |

y  y …. (4.4.1)

# Y Shear

The y shear preserves the x coordinates, but changes the y values which causes horizontal lines to transform into lines which slope up or down. The transformation matrix for y shear is given as

 1 0 0

*Y* \_*sh*  *Shy* 1 0

 0 0 1

 x  x and y  y  Sh. x

# Shearing Relative to Other Reference Line

We can apply x shear and y shear transformations relative to other reference lines. In x shear transformation we can use y reference line and in y shear we can use x reference line. The transformation matrices for both are given below:

1 Shy 0

 

Y shear with x reference line : 0 1 0

0  Shy . yref 0

Example: Show how shear transformation may be expressed in terms of rotation and scaling.

Solution:

The shear transformation matrix for x and y combinely can be given as

 1 Shy 0

Shx 1 0

 0 0 1

We have scaling matrix and rotation matrix as given below:

Sx 0 0  cos sin 0

S   0 Sy 0 R  sin cos 0

 0 0 1  0 0 1

If we combine scale matrix and rotation matrix we have,

 Sx cos Sx sin 0

S.R.  Sy sin Sy cos 0

 0 0 1

Comparing shear matrix and S.R. matrix we have

Shx Sy sin

Shy  Sx sin

Sx cos  1 and

Sy cos1

1

 Sx and

cos

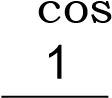
1

Sy 

cos

Substituting values of Sx and Sy we get,

1

Shx  .sintan 

Shy .sin tan

cos

Therefore, the shear transformation matrix expressed in terms of rotation and scales is

 1 tan 0

tan 1 0 Sx cos Sy cos1

 0 0 1

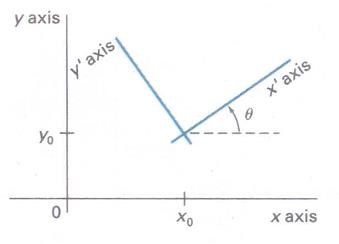
Where  : angle of rotation Sx : x scale and

Sy : y scale

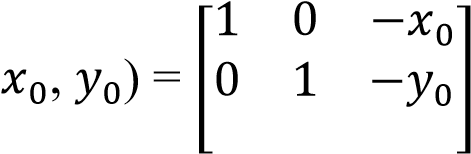
Graphics applications often require the transformation of object descriptions from one coordinate system to another.

## 4.5 Transformation between Co-ordinate Systems

Graphics applications often require the transformation of object descriptions from one coordinate system to another. We need to transform between two Cartesian systems. Figure 4.7 shows two Cartesian systems, with the coordinate origins at (0, 0) and (𝑥0, 𝑦0) and with an orientation angle 𝜃 between the x and 𝑥′ axes.



# Figure 4.7: A cartesian 𝒙′𝒚′ system positioned at (𝒙𝟎, 𝒚𝟎) with orientation 𝜽 in an xy cartesian system

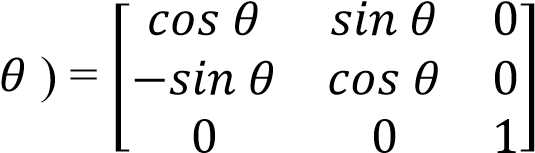
To transform object descriptions from xy coordinates to 𝑥′𝑦′coordinates, we need to set up a transformation that superimposes the 𝑥′𝑦′ axes onto the xy axes. This is done in two steps:

1. Translate so that the origin (𝑥0, 𝑦0) of the 𝑥′𝑦′ system is moved to the origin of the xy system.
2. Rotate the 𝑥′ axis onto the x axis.

T ( …………..(4.5.1)

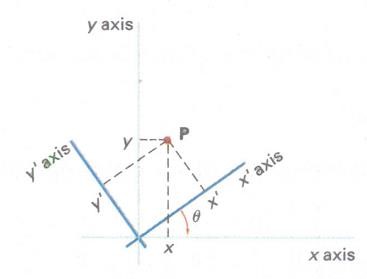
0 0 1

Translation of co-ordinate system expressed with matrix operation and the orientation of the two systems after the translation operation would appear as in Figure 4.8. To get the axes of the two systems into coincidence, we then perform the clockwise rotation

R (- …………..(4.5.2)

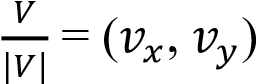
Concatenating these two transformations matrices gives us the complete composite matrix for transforming object descriptions from the xy system to the x'y' system:

𝑀𝑥𝑦, 𝑥′𝑦′ = R (-𝜃 ) . T(−𝑥0 , −𝑦0 ) …………..(4.5.3)



# Figure 4.8: Position of the reference frames shown in Fig.4.7 after translating the origin of the 𝑥′𝑦′ system to the coordinate origin of the xy system

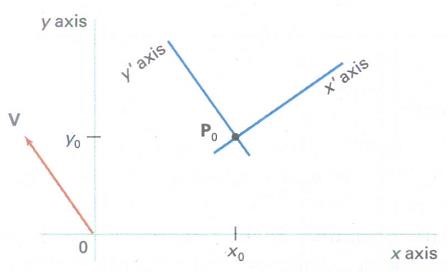
An alternate method for giving the orientation of the second coordinate system is to specify a vector V that indicates the direction for the positive y' axis, as shown in Figure. 4.9. Vector V is specified as a point in the xy reference frame relative to the origin of the xy system. A unit vector in the y' direction can then be obtained as

V =  (4.5.4)

And we obtain the unit vector u along 𝑥′ axes by rotating 90𝑜 clock wise

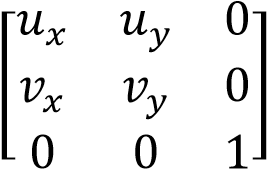
u = (𝑣𝑦, 𝑣𝑥)

(𝑢𝑥 , 𝑢𝑦 ) (4.5.5)



**Figure 4.9: Cartesian system x'y' with origin at Po = (**𝒙𝟎**,** 𝒚𝟎**) and y' axis parallel to x axis vector V.**

The elements of any rotation matrix could be expressed as elements of a set of orthogonal unit vectors. Therefore, the matrix to rotate the x'y' system into coincidence with the xy system can be written as

R= ………..(4.5.6)

## 4.6 Affine Transformation

A coordinate transformation of the form

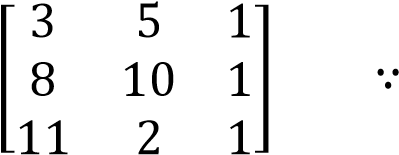
𝑥, **=** axx*x* **+** axy*y***+** bx **,** 𝑦, **=** ayx*x* **+** ayy*y***+** by………..(4.6.1)is called a two-dimensional affine transformation. Each of the transformed coordinates x' and y' is a linear function of the original coordinates x and y, and parameters 𝑎𝑖𝑗and 𝑏𝑘 are constants determined by the 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 examples of two-dimensional affine transformations. Any general two-dimensional affine transformation can always be expressed as a composition of these five transformations. Another affine transformation is the conversion of coordinate descriptions from one reference system to another, which can be described as a combination of translation and rotation. An affine transformation involving only rotation, translation, and reflection preserves angles and lengths, as well as parallel lines. For these three transformations, the lengths and angle between two lines remains the same after the transformation.

# Problems related to transformation

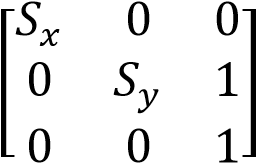
1. Scale the polygon with coordinates A(3,5), B (8,10), and C (11,2) by 3 units in X direction and 2 units in Y direction using matrix operation. (M.U. April/May-2008- 04 Marks)

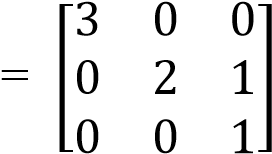
Polygon coordinates are A(3,5), B (8,10), and C (11,2) The Matrix form of given square is:

 A(3,5), B (8,10), C (11,2)

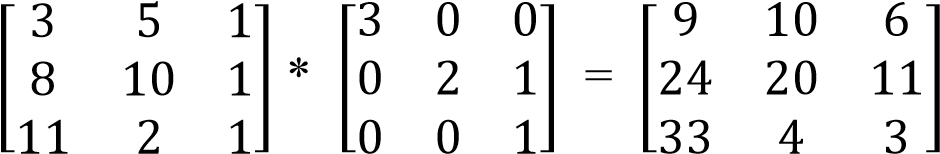
For increasing the size of polygon by 3 units in X direction and 2 units in Y direction, we have to write scaling factor as

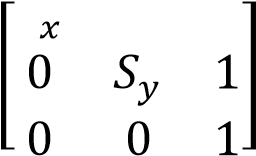
𝑆𝑥 = 3 𝑆𝑦 = 2

∴ Scaling matrix, S= 



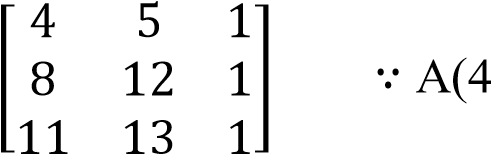
After scaling,



 ∴ New coordinate after scaling ABC by 3 units in X direction and 2 units in Y direction are- A(9,10), B(24,20) and C(33,4)

1. Scale a polygon with vertices (4,5), (8,12), (11,13) by 5 unit in x axis, 4 units in y axis.

Polygon coordinates are A(4,5), B (8,12), and C (11,13) (M.U. April/May-2009-04 Marks) The Matrix form of given square is:

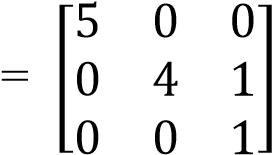
,5), B (8,12), C (11,13)

For increasing the size of polygon by 5 units in X direction and 4 units in Y direction, we have to write scaling factor as

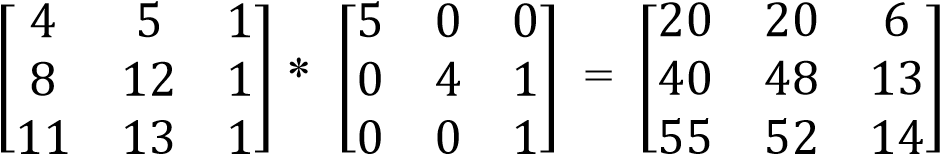
𝑆𝑥 = 5 𝑆𝑦 = 4

𝑆 0 0

∴ Scaling matrix, S=

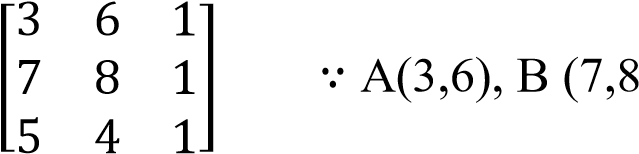


After scaling,



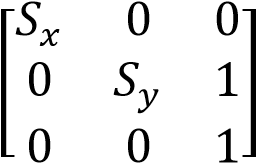
∴ New coordinate after scaling ABC by 3 units in X direction and 2 units in Y direction are- A(20,20), B(40,48) and C(55,52)

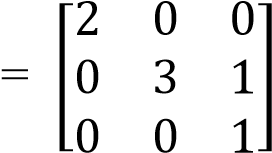
1. Scale a polygon with coordinates A(3,6), B(7, 8) and C(5, 4) by two units in X direction and three units in Y direction (M.U. April/May-2010- 06 Marks) Polygon coordinates are A(3,6), B (7,8), and C (5,4) The Matrix form of given square is:

), and C (5,4)

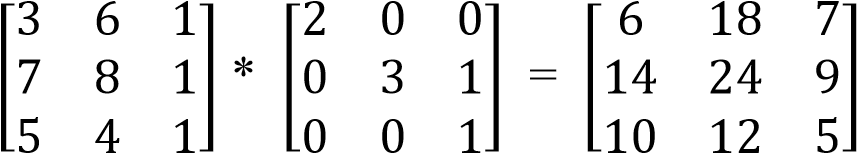
For increasing the size of polygon by 2 units in X direction and 3 units in Y direction, we have to write scaling factor as

𝑆𝑥 = 2 𝑆𝑦 = 3

∴ Scaling matrix, S= 



After scaling,



∴ New coordinate after scaling ABC by 2 units in X direction and 3 units in Y direction are- A(6,8), B(14,24) and C(10,12)

(

𝑥

′

)

=

100

(

+

150

-

100

)

√

2

2

(

𝑦

′

)

=

100

+

(

150

-

100

)

√

2

2

1. Consider the polygon with vertices A (l00 100), B(l50, 100), C(l50, 50), D(100, 50). Rotate this polygon by 45° about A(100, 100) and write the coordinates of new points. (M.U. April/May-2011- 05 Marks)

Rotation of a point about an arbitrary pivot position or a point can be obtained using following formula

𝑥′ = 𝑥𝑟 + (x-𝑥𝑟) cos𝜃 - (y-𝑦𝑟) sin𝜃

𝑦′ = 𝑦𝑟 + (x-𝑥𝑟) sin𝜃 + (y-𝑦𝑟) cos𝜃

Here polygon vertices are A (l00 100), B(l50, 100), C(l50, 50), D(100, 50) and its new coordinate are represented as A(𝑥′, 𝑦′ ), B(𝑥′, 𝑦′ ), C(𝑥′, 𝑦′ ), D(𝑥′, 𝑦′ ). Pivot point is (100,

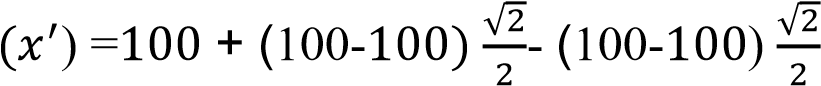
100)

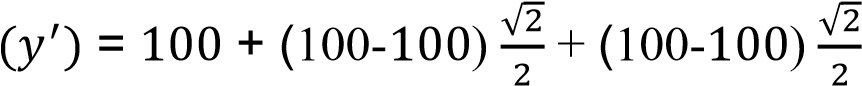
A(𝑥′) =𝑥𝑟 + (x-𝑥𝑟) cos𝜃 - (y-𝑦𝑟) sin𝜃

A(𝑦′) = 𝑦𝑟 + (x-𝑥𝑟) sin𝜃 + (y-𝑦𝑟) cos𝜃

A(𝑥′) =100 + (100-100) cos45° - (100-100) sin45°

A(𝑦′) = 100 + (100-100) sin45° + (100-100) cos45°

A

A

A(𝑥′) =100

A(𝑦′) = 100 ∴ A(𝑥′ , 𝑦′) = (100, 100)

B(𝑥′) =𝑥𝑟 + (x-𝑥𝑟) cos𝜃 - (y-𝑦𝑟) sin𝜃

B(𝑦′) = 𝑦𝑟 + (x-𝑥𝑟) sin𝜃 + (y-𝑦𝑟) cos𝜃

B(𝑥′) =100 + (150-100) cos45° - (100-100) sin45°

B(𝑦′) = 100 + (150-100) sin45° + (100-100) cos45°

B

B

B

B

B(𝑥′) =100 + 25√2 = 25(4+√2)

(

𝑥

′

)

=

100

+

50

√

2

2

(

𝑦

′

)

=

100

+

50

√

2

2

B(𝑦′) = 100 + 25√2 =25(4+√2) ∴ B(𝑥′ , 𝑦′) = (25(4+√2), 25(4+√2) )

C(𝑥′) =𝑥𝑟 + (x-𝑥𝑟) cos𝜃 - (y-𝑦𝑟) sin𝜃

(

𝑥

′

)

=

100

+

50

√

2

2

+50

√

2

2

(

𝑦

′

)

=

100

+

50

√

2

2

-

50

√

2

2

C(𝑦′) = 𝑦𝑟 + (x-𝑥𝑟) sin𝜃 + (y-𝑦𝑟) cos𝜃

C(𝑥′) =100 + (150-100) cos45° - (50-100) sin45°

C(𝑦′) = 100 + (150-100) sin45° + (50-100) cos45°

C

C

C(𝑥′) =100 + 25√2 +25√2

C(𝑦′) = 100 + 25√2 -25√2

C(𝑥′) =100 + 50√2 = 50(2+√2)

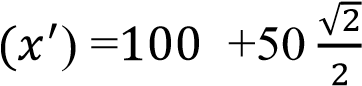
C(𝑦′) = 100 ∴ C(𝑥′ , 𝑦′) = (50(2+√2), 100)

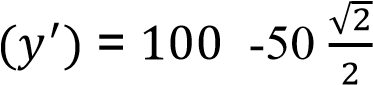
D(𝑥′) =𝑥𝑟 + (x-𝑥𝑟) cos𝜃 - (y-𝑦𝑟) sin𝜃

D(𝑦′) = 𝑦𝑟 + (x-𝑥𝑟) sin𝜃 + (y-𝑦𝑟) cos𝜃

D(𝑥′) =100 + (100-100) cos45° - (50-100) sin45°

D(𝑦′) = 100 + (100-100) sin45° + (50-100) cos45°

D

D

D(𝑥′) =100 + 25√2 =25(4+√2)

D(𝑦′) = 100 - 25√2 =25(4-√2 )

∴ D(𝑥′ , 𝑦′) = (25(4+√2), 25(4-√2 ))

Then new coordinates of polygon is A (100, 100), B (25(4+√2), 25(4+√2) ), C

(50(2+√2), 100), D (25(4+√2), 25(4-√2 ))

**4.7 Assignment-4**

# Short Answer Questions (Each carries 2 marks)

1. Write the equations for translation of a point at (x, y) to (x’, y’) position. Also write the column vectors.
2. Write the equations for scaling of a point at (x, y) to (x’, y’) position. Also write the column vectors
3. Write the equations for rotation of a point at (x, y) to (x’, y’) position about origin.
4. Define translation and scaling.
5. What is a rigid body transformation? Give example. (M. U. April/May-2012)
6. Write the matrices to represent translation and scaling in homogeneous coordinate system.
7. What is reflection?
8. Show that 900 rotation of unit vector along the x-axis produces a unit vector along y-axis(M. U. April/May-2013).
9. What is affine transformation? (M. U. April/May-2012, 2009) 10. Write the purpose of homogeneous coordinate system.
10. List the sequence of steps required to perform rotation of an object about a pivot point at

(xr, yr).

1. List the sequence of steps required to perform scaling of an object with respect to a fixed position at (xf, yf).
2. Discuss associative property with respect to transformation matrices.
3. Discuss commutative property with respect to transformation matrices.
4. Write the matrices for reflection about the x- axis and reflection about coordinate origin.
5. What is a shear? Write any one shear matrix.
6. Write the transformation matrix to rotate a point by an angle in clockwise direction (M.

U. April/May-2010).

1. Distinguish between affine and rigid body transformations (M. U. April/May-2010, 2008).
2. Show that 2 successive 2D scalings are multiplicative. (M. U. April/May-2009).
3. Write the steps used for rotation of an object about an arbitrary point. (M.U. April/may 2008)
4. What do you mean by composite transformation?

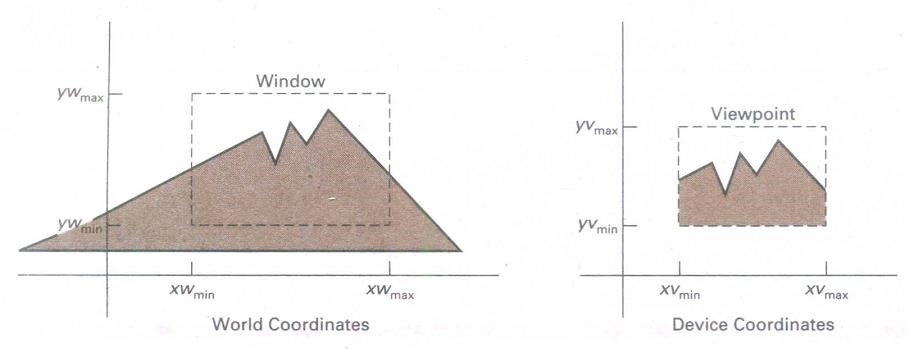
# Long Answer Questions

1. Prove that successive translations are additive and successive scaling are multiplicative (6 Marks-M. U. April/May 2013).
2. Consider the polygon with vertices A (100, 100), B (150,100) and C (150, 50). Rotate this by 450 degrees anticlockwise direction about A (100,100) and write co-ordinates of new point (3/5 Marks-M. U. April/May 2013, 2011).
3. Briefly explain homogeneous co-ordinates and matrix representation of 2D transformation (6 Marks-M. U. April/May 2013, 2012).
4. Derive a matrix for rotating an object about arbitrary point. (4 Marks-M. U. April/May 2012, 2009)
5. Show that successive 2D scaling is multiplicative (3 Marks-M. U. April/May 2012).
6. Scale the polygon with co-ordinates A(3, 7), B (7, 9) and C (6,5) by two units in X direction and two units in Y direction (4 Marks-M. U. April/May 2012).
7. Explain homogeneous co-ordinate system? Give 2D translation and scaling matrices using homogeneous co-ordinates (4 Marks-M. U. April/May 2011).
8. What are affine and rigid body transformations? Explain with example (3 Marks-M. U. April/May 2011).
9. Explain the sequence of transformations for rotating an object about arbitrary point in 2D transformation (8 Marks-M. U. April/May 2011).
10. Explain composite transformation. (8 Marks)
11. Explain different types of other transformation (7 Marks).
12. Prove that two successive 2D rotations are additive (4 Marks-M. U. April/May 2010, 2009).
13. Explain the homogenous co-ordinate system for 2D transformation (4 Marks-M. U. April/May 2010).
14. Scale a polygon with vertices (4,5), (8,12), (11,13) by 5 unit in x axis, 4 units in y axis (5 Marks-M. U. April/May 2009)
15. Scale the polygon with coordinates A(3,5), B (8,10), and C (11,2) by 3 units in X direction and 2 units in Y direction using matrix operation. (4 Marks -M. U. April/May 2008)
16. Prove that successive translations are additive and successive scaling are multiplicative (6 Marks -M. U. April/May 2008)
17. Discuss various Reflection transformations with suitable diagrams. (6 Marks)
18. Write a note on shear transformation. (4 Marks)

# UNIT-II Chapter 5 Two dimensional Viewing

## 5.1 The Viewing Pipeline

A world-coordinate area selected for display is called a window. An area on a display device to which a window is mapped is called a viewport. The window defines what is to be viewed; the viewport defines where it is to be displayed. Often, windows and viewports are rectangles in standard position, with the rectangle edges parallel to the coordinate axes. Other window or viewport geometries, such as general polygon shapes and circles, are used in some applications, but these shapes take longer to process. In general, the mapping of a part of a world-coordinate scene to device coordinates is referred to as a viewing transformation. Sometimes the twodimensional viewing transformation is simply referred to as the window-to-viewport transformation or the windowing transformation. But, in general, viewing involves more than just the transformation from the window to the viewport. Figure 5.1 illustrates the mapping of a picture section that falls within a rectangular window onto a designated rectangular viewport.



# Figure 5.1: A viewing transformation using standard rectangles for the window and viewport

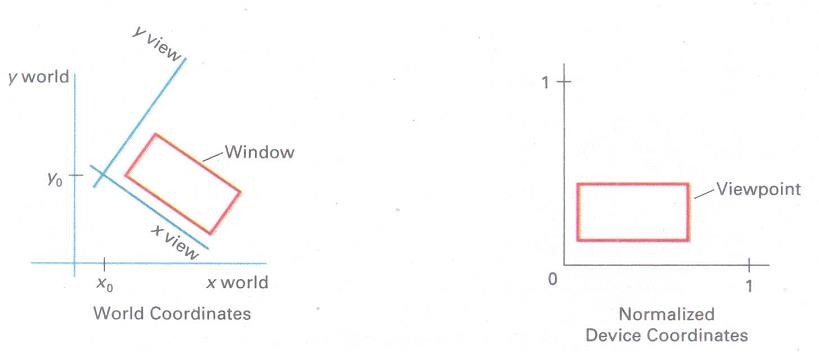
In computer graphics terminology, the term window originally referred to an area of a picture that is selected for viewing. Some graphics packages that provide window and viewport operations allow only standard rectangles, but a more general approach is to allow the rectangular window to have any orientation.

First, we construct the scene in world coordinates using the output primitives and attributes. Next, to obtain a particular orientation for the window, we can set up a two dimensional viewingcoordinate system in the world-coordinate plane, and define a window in the viewing-coordinate system. The viewing coordinate reference frame is used to provide a method for setting up arbitrary orientations for rectangular windows. Once the viewing reference frame is established, we can transform descriptions in world coordinates to viewing coordinates. We then define a viewport in normalized coordinates (in the range from 0 to 1) and map the viewing-coordinate description of the scene to normalized coordinates. At the final step, all parts of the picture that lie outside the viewport are clipped, and the contents of the viewport are transferred to device coordinates. Figure 5.2 illustrates a rotated viewing-coordinate reference frame and the mapping to normalized coordinates.

# Figure 5.2: Setting up a rotated world window in viewing coordinates and the corresponding normalized-coordinate viewport

By changing the position of the viewport, we can view objects at different positions on the display area of an output device. Also, by varying the size of viewports, we can change the size and proportions of displayed objects. We achieve zooming effects by successively mapping different-sized windows on a fixed-size viewport. As the windows are made smaller, we zoom in on some part of a scene to view details that are not shown with larger windows.

Viewports are typically defined within the unit square (normalized coordinates). This provides a means for separating the viewing and other transformations from specific output-device requirements, so that the graphics package is largely device-independent. When all coordinate transformations are completed, viewport clipping can be performed in normalized coordinates or in device coordinates. This allows us to reduce computations by concatenating the various transformation matrices.



## 5.2 Viewing Co-ordinate Reference Frame

This coordinate system provides the reference frame for specifying the world coordinate window. First, a viewing-coordinate origin is selected at some world position: Po = (𝑥0, 𝑦0). Then we need to establish the orientation, or rotation, of this reference frame. One way to do this is to specify a world vector V that defines the viewing 𝑦𝑣 direction. Vector V is called the view up vector.

Given V, we can calculate the components of unit vectors v = (𝑣𝑥, 𝑣𝑦) and u = (𝑢𝑥, 𝑢𝑦) for the viewing 𝑦𝑣 and axes, respectively. These unit vectors are used to form the first and second rows of the rotation matrix R that aligns the viewing 𝑥𝑣𝑦𝑣 axes with the world 𝑥𝑤𝑦𝑤 axes.

We obtain the matrix for converting world-coordinate positions to viewing coordinates as a twostep composite transformation: First, we translate the viewing origin to the world origin, then we rotate to align the two coordinate reference frames. The composite two-dimensional

transformation to convert world coordinates to viewing coordinates is

𝑀𝑊𝐶, 𝑉𝐶 = R. T (5.2.1)

# Figure 5.3: A viewing-coordinate frame is moved into coincidence with the world frame in two steps: (a) translate the viewing origin to the world origin, then (b) rotate to align the axes of the two system

Where T is the translation matrix that takes the viewing origin point 𝑃0 to the world origin, and R is the rotation matrix that aligns the axes of the two reference frames.

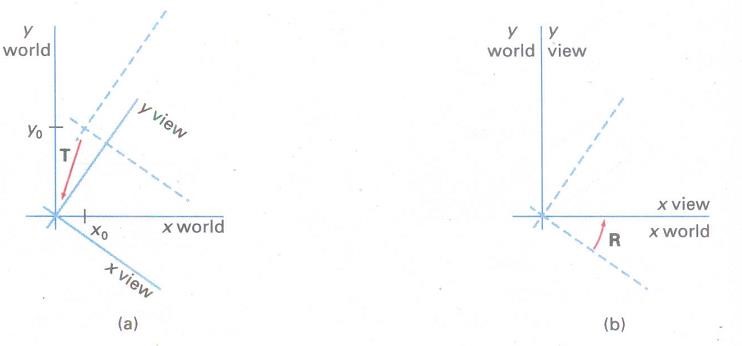
## 5.3 Window-to-Viewport Coordinate Transformation

We know that the picture is stored in the computer memory using any convenient Cartesian coordinate system, referred to as **world coordinate system** (WCS). However, when picture is displayed on the display device it is measured in **physical device coordinate system** (PDCS) corresponding to the display device: Therefore, displaying an image of a picture involves mapping the coordinates of the points and lines that form the picture into the appropriate physical device coordinate where the image is to be displayed. This mapping of coordinates is achieved with the use of coordinate transformation known as **viewing transformation.** The viewing transformation which maps picture coordinates in the WCS to display coordinates in PDCS is performed by the following transformations:

* Normalization transformation (N) and
* Workstation transformation (W)

# Normalization Transformation

We know that, different display devices may have different screen sizes as measured in pixels. Size of the screen in pixels increases as resolution of the screen increases. When picture is defined in the pixel values then it is displayed large in size on the low resolution screen while small in size on the high resolution screen. To avoid this and to make our programs to be device independent, we have to define the picture coordinates in some units other than pixels and use the interpreter to convert these coordinates to appropriate pixel values for the particular display device. The device independent units are called the **normalized device coordinates.** In these units, the screen measures 1 unit wide and 1 unit length. The lower left corner of the screen is the origin, and the upper-right is the point (1, 1). The point (0.5, 0.5) is the center of the screen no matter what the physical dimensions or resolution of the actual display device may be.



The interpreter uses a simple linear formula to convert the normalized device coordinates to the actual device coordinates.

1.  xn  XW
2.  yn  YH

where

x : Actual device x coordinate y : Actual device y coordinate xn : Normalized x coordinate yn : Normalized y coordinate

XW : Width of actual screen in pixels

YH : Height of actual screen in pixels

The transformation which maps the word coordinate to normalized device coordinate is called **normalization transformation.** It involves scaling of x and y, thus it is also referred to as scaling transformation.

# Workstation Transformation

The transformation which maps the normalized device coordinates to physical device coordinates is called workstation transformation.

The viewing transformation is the combination of normalization transformation and workstation transformations. It is given as V = N . W

We know that world coordinate system (WCS) is infinite in extent and the device display area in finite. Therefore, to perform a viewing transformation we select a finite world coordinate area for display called a **window.** An area on a device to which a window is mapped is called a **viewport.**

The window defined in world coordinates is first transformed into the normalized device coordinates. The normalized window is then transformed into the viewport coordinate. This window to viewport coordinate transformation is known as workstation transformation. It is achieved by performing following steps.

1. The object together with its window is translated until the lower left corner of the window is at the origin.
2. Object and window are scaled until the window has the dimensions of the viewport.
3. Translate the viewport to its correct position on the screen.

Therefore, the workstation transformation is given as

W = T.S.T–1

The transformation matrices for individual transformation are as given below:

 1 0 0

T   0 1 0

 Xwmin  Ywmin 1

Sx 0 0

S   0 Sy 0

 0 0 1

where Sx  xvmax  xvmin

xwmax  xwmin

Sy  yvmax  yvmin

ywmax  ywmin

 1 0 0

T1   0 1 0

xvmin yvmin 1

The overall transformation matrix for W is given as

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  | W = T.S.T–1 |  |  |  |
|     | 1  0 | 0  1 | 0 Sx 0 0   0  0 Sy 0  | 1  0 | 0  1 | 0  0  |

 Xwmin Yvmin 1  0 0 1 xvmin vmin 1

y

|  |  |  |  |
| --- | --- | --- | --- |
|  | Sx | 0 | 0 |

=  0 Sy 0

 1

xvmin  xw min.Sx yvmin  yw min.Sy 

## 5.4 Two Dimensional Viewing Functions

We define a viewing reference system in a PHIGS application program with the following function:

evaluateViewOrientationMatrix (𝑥0, 𝑦0, 𝑥𝑣, 𝑦𝑣, error, viewMatrix)

where parameters 𝑥0 and 𝑦0 are the coordinates of the viewing origin, and parameters 𝑥𝑣 and 𝑦𝑣 are the world-coordinate positions for the view up vector. An integer error code is generated if the input parameters are in error; otherwise, the view Matrix for the world-to-viewing transformation is calculated. Any number of viewing transformation matrices can be defined in an application.

To set up the elements of a window-to-viewport mapping matrix, we invoke the function

evaluateViewMappingMatrix ( 𝑥wmin, 𝑥wmax, 𝑦wmin, 𝑦wmax, 𝑥vmin, 𝑥vmax, 𝑦vmin, 𝑦vmax, error, viewMappingMatrix)

Here, the window limits in viewing coordinates are chosen with parameters 𝑥wmin , 𝑥wmax ,

𝑦wmin, 𝑦wmax and the viewport limits are set with the normalized coordinate positions 𝑥vmin,

𝑥vmax , 𝑦vmin , 𝑦vmax. As with the viewing-transformation matrix, we can construct several window-viewport pairs and use them for projecting various parts of the scene to different areas of the unit square.

we can store combinations of viewing and window-viewport mappings for various workstations in a viewing table with setViewRepresentation (ws, viewIndex, viewMatrix, viewMappingMatrix, xclipmin ,

xclipmax, ycIipmin, yclipmax, clipxy)

where parameter ws designates the output device (workstation), and parameter viewIndex sets an integer identifier for this particular window-viewport pair. The matrices viewMatrix and viewMappingMatrix can be concatenated and referenced by the viewIndex. Additional clipping limits can also be specified here, but they are usually set to coincide with the viewport boundaries. And parameter clipxy is assigned either the value noclip or the value clip.

The function SetViewIndex (ViewIndex), selects a particular set of options from the viewing table. At the final stage, we apply a workstation transformation by selecting a workstation window-viewport pair:

setWorkstationWindow (ws, xwsWindmin, xwsWindmax, ywswindmin, ywswindmax)

setWorkstationViewport (ws, xwsVPortmin, xwsVPortmax, ywsVPortmin, ywsVPortmax)

where parameter ws gives the workstation number. Window-coordinate extents are specified in the range from 0 to 1 (normalized space), and viewport limits are in integer device coordinates.

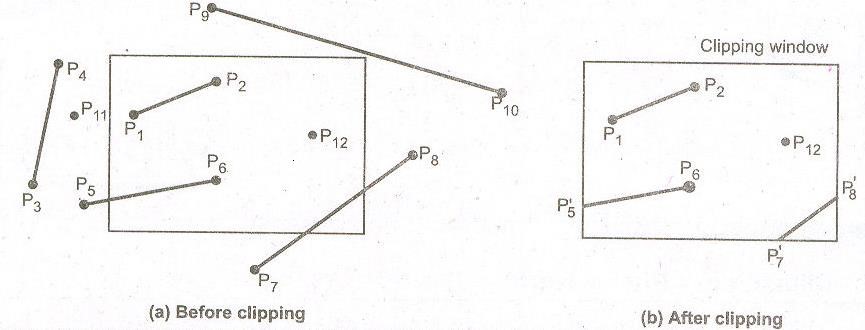
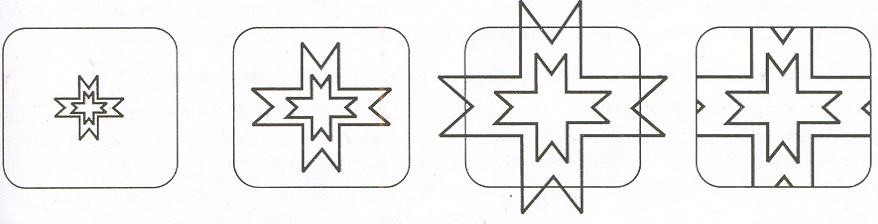
**5.5 Clipping Operations**

# Clipping in a Raster world

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

# Figure 5.4: Process of Clipping

The region against which an object is to be clipped is called **clip window or clipping window**. It usually is in a rectangular shape. The clipping algorithm determines which points, lines or portions of lines lie within the clipping window. These points, lines or portions of lines are retained for display. All others are discarded.



# Figure 5.5: Clipping a line using clip window

## 5.6 Point clipping

Assuming that the clip window is a rectangle in standard position, we save a point P = (x, y) for display if the following inequalities are satisfied:

𝑥𝑤𝑚𝑖𝑛 ≤ x ≤ 𝑥𝑤𝑚𝑎𝑥

𝑦𝑤𝑚𝑖𝑛 ≤ y ≤ 𝑦𝑤𝑚𝑎𝑥

where the edges of the clip window (𝑥𝑤𝑚𝑖𝑛, 𝑥𝑤𝑚𝑎𝑥, 𝑦𝑤𝑚𝑖𝑛 , 𝑦𝑤𝑚𝑎𝑥) can be either the worldcoordinate window boundaries or viewport boundaries. If anyone of these four inequalities is not satisfied, the point is clipped (not saved for display).

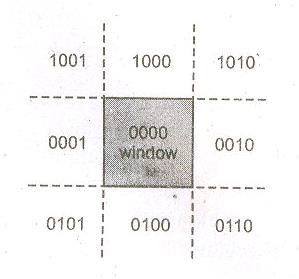
Although point clipping is applied less often than line or polygon clipping, some applications may require a point-clipping procedure. For example, point clipping can be applied to scenes involving explosions or sea foam that are modeled with particles (points) distributed in some region of the scene.

## 5.7 Clipping Lines

The lines are said to be interior to the clipping window and hence visible if both end points are interior to the window, e.g., line P1 P2 in Fig. However, if both end points of a line are exterior to the window, the line is not necessarily completely exterior to the window, e.g. line P7 P8 in Figure 5.5. If both end points of a line are completely exterior to the right of completely to the left of, completely above, or completely below the window, then the line is completely exterior to the window and hence invisible. For example, line P3 P4 in fig.

### 5.7.1 Sutherland and Cohen Subdivision Line Clipping Algorithm

This is one of the oldest and most popular line clipping algorithm developed by Dan Cohen and Ivan Sutherland. This algorithm uses a four digit (bit) code to indicate which of nine regions contain the end point of line. The four bit codes are called **region codes** or **outcodes**. These codes identify the location of the point relative to the boundaries of the clipping rectangle as shown in the figure 5.6.



# Figure 5.6: Four-bit codes for nine regions

Each bit position in the region code is used to indicate one of the four relative coordinate positions of the point with respect to the clipping window: to the left, right, top or bottom. The rightmost bit is the first bit and the bits are set to 1 based on the following scheme:

Set Bit 1 – if the end point is to the left of the window

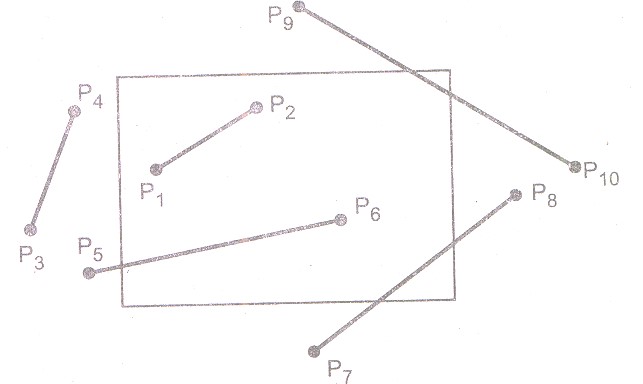
Set Bit 2 – if the end point is to the right of the window

Set Bit 3 – if the end point is below the window

Set Bit 4 – if the end point is above the window Otherwise, the bit is set to zero.

Once we have established region codes for all the line endpoints, we can determine which lines are completely inside the clipping window and which are clearly outside. Any lines that are completely inside the window boundaries have a region code of 0000 for both endpoints and we trivially accept these lines. Any lines that have a 1 in the same bit position in the region codes for each endpoint are completely outside the clipping rectangle, and we trivially reject these lines. A method used to test lines for total clipping is equivalent to the logical AND operator. If the result of the logical AND operation with two end point codes is not 0000, the line is completely outside the clipping region. The lines that cannot be identified as completely inside or completely inside or completely outside a clipping window by these tests are checked for intersection with the window boundaries.

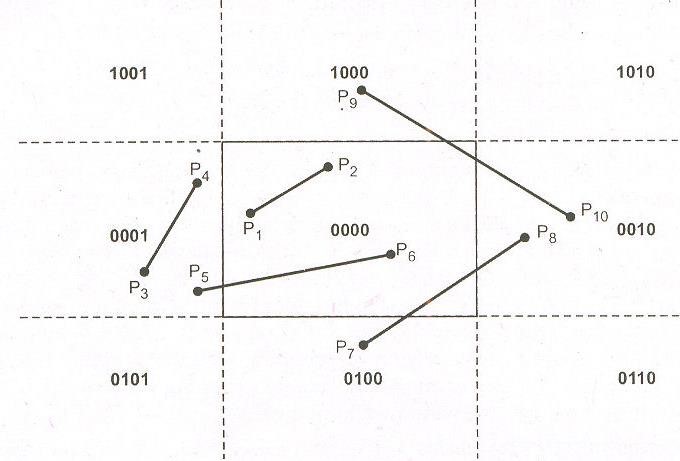
Example: Consider the clipping window and the lines shown in Fig. Find the region codes for each end point and identify whether the line is completely visible, partially visible or completely invisible.



# Figure 5.7: Line clipping example using Cohen-Sutherland algorithm

**Sol**. : The Fig. shows the clipping window and lines with region codes. These codes are tabulated and end point codes are logically AND’ed to identify the visibility of the line in the table below.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Line | End Point Code |  | Logical ANDing | Result |
| P1 p2 | 0000 | 0000 | 0000 | Completely  visible |
| P3 P4 | 0001 | 0001 | 0001 | Completely invisible |
| P5 P6 | 0001 | 0000 | 0000 | Partially visible |
| P7 P8 | 0100 | 0010 | 0000 | Partially visible |
| P9 P10 | 1000 | 0010 | 0000 | Partially visible |



# Figure 5.8: coded window regions

The **Sutherland – Cohen algorithm** begins the clipping process for a partially visible line by comparing an outside endpoint to a clipping boundary to determine how much of the line can be discarded. Then the remaining part of the line is checked against the other boundaries, and the process is continued until either the line is totally discarded or a section is found inside the window.

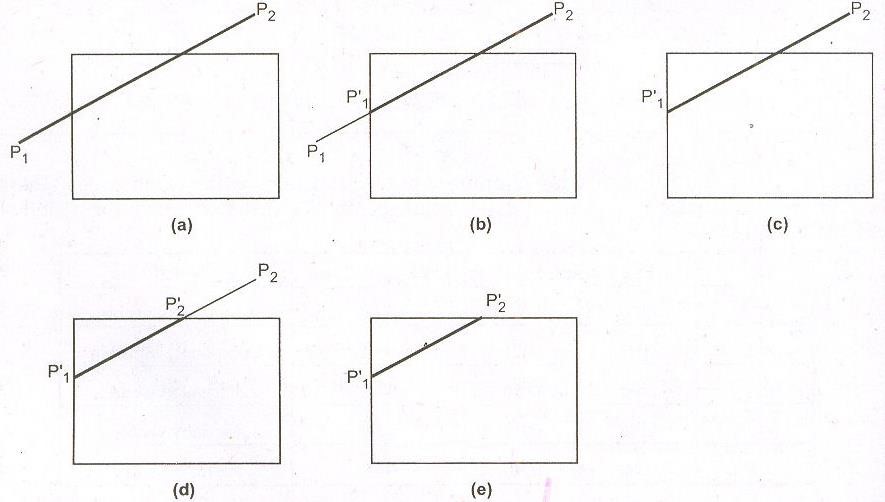
As shown in the above figure 5.9. line P1 P2 is a partially visible and point P1 is outside the window. Starting with point P1 the intersection point *P*1 is found and we get two line segments

*P*1  *P*1 and *P*1  *P*2 We know that, for *P*1  *P*1one end point i.e. P1 is outside the window and thus the line segment *P*1  *P*1is discarded. The line is now reduced to the section from *P*1 to P2. Since P2 is outside the clip window, it is checked against the boundaries and intersection point

*P*2 is found. Again the line segment is divided into two segments giving *P*1  *P*2and *P*2  *P*2 .

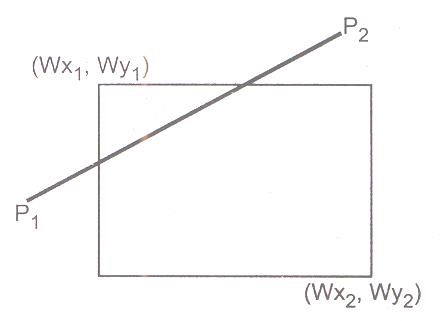
We know that, for *P*2  *P*2 one end point i.e. P2 is outside the window and thus the line segment

*P*2  *P*2 .is discarded. The remaining line segment *P*1  *P*2 is completely inside the clipping window and hence made visible.



# Figure 5.9: Line clipping Process

**Sutherland and Cohen subdivision line clipping algorithm:**



# Figure 5.10: Sutherland and Cohen subdivision line clipping algorithm

1. Read two end points of the line say P1 (x1, y1) and P2 (x2, y2).
2. Read two corners (left-top and right-bottom) of the window, say (Wx1, Wy1 and Wx2, Wy2).
3. Assign the region codes for two endpoints P1 and P2 using following steps :

Initialize code with bits 0000

Set Bit 1 - if (x < Wx1)

Set Bit 2 - if (x > Wx2)

Set Bit 3 – if(y< Wy1)

1. Check for visibility of line P1 P2
   1. If region codes for both endpoints P1 P2 are zero then the line is completely visible. Hence draw the line and go to step 9.
   2. If region codes for endpoints are not zero and the logical ANDing of them is also nonzero then the line is completely invisible, so reject the line and go to step 9.
   3. If region codes for two endpoints do not satisfy the conditions in 4a) and 4b) the line is partially visible
2. Determine the intersecting edge of the clipping window by inspecting the region codes of two endpoints.
   * 1. If region codes for both the end points are non-zero, find intersection points

*P*1 and *P*2 with boundary edges of clipping window with respect to point P1 and point P2 respectively

* + 1. If region code for any one end point is non zero then find intersection point

*P*1 or *P*2 with the boundary edge of the clipping window with respect to it.

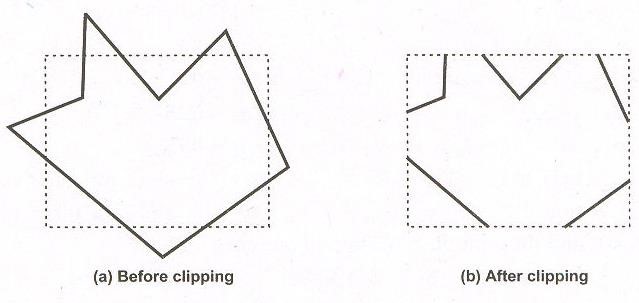
1. Divide the line segments considering intersection points.
2. Reject the line segment if any one end point of it appears outsides the clipping window.
3. Draw the remaining line segments.
4. Stop

## 5.8 Polygon Clipping

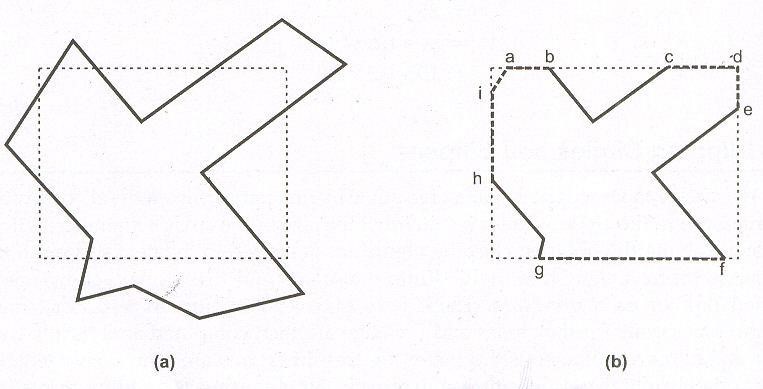
A polygon is nothing but the collection of lines. However, when a closed polygon is clipped as a collection of lines with line clipping algorithm, the original closed polygon becomes one or more open polygon or discrete lines as shown in the Figure.5.11. Thus, we need to modify the line clipping algorithm to clip polygons.

We consider a polygon as a closed solid area. Hence after clipping it should remain closed. To achieve this we require an algorithm that will generate additional line segment which make the polygon as a closed area. For example, in the Figure 5.12, the lines a-b, c-d, d-e, f-g and h-i are added to polygon to make it closed.

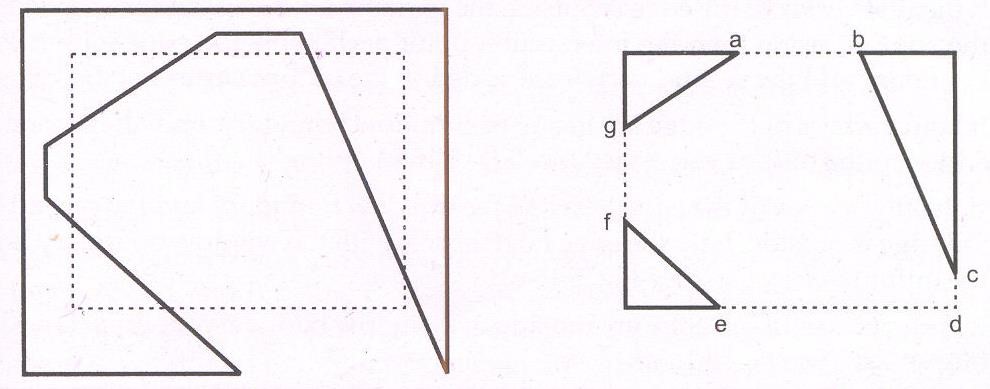
Adding lines c-d and d-e is particularly difficult. Considerable difficulty also occurs when clipping a polygon results in several disjoint smaller polygons as shown in the Figure 5.13. For example, the lines a-b, c-d, d-e and g-f are frequently included in the clipped polygon description which is not desired.



# Figure 5.11: Polygon clipping done by line clipping algorithm



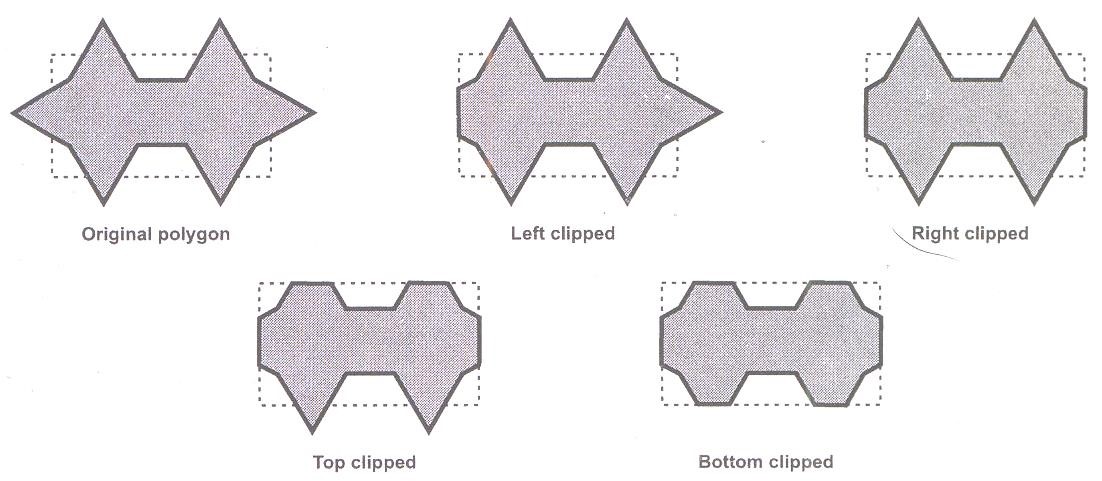
# Figure 5.12: Modifying the line clipping algorithm for polygon



# Figure 5.13: Disjoint polygons in polygon clipping

## 5.8.1 Sutherland-Hodgeman Polygon Clipping

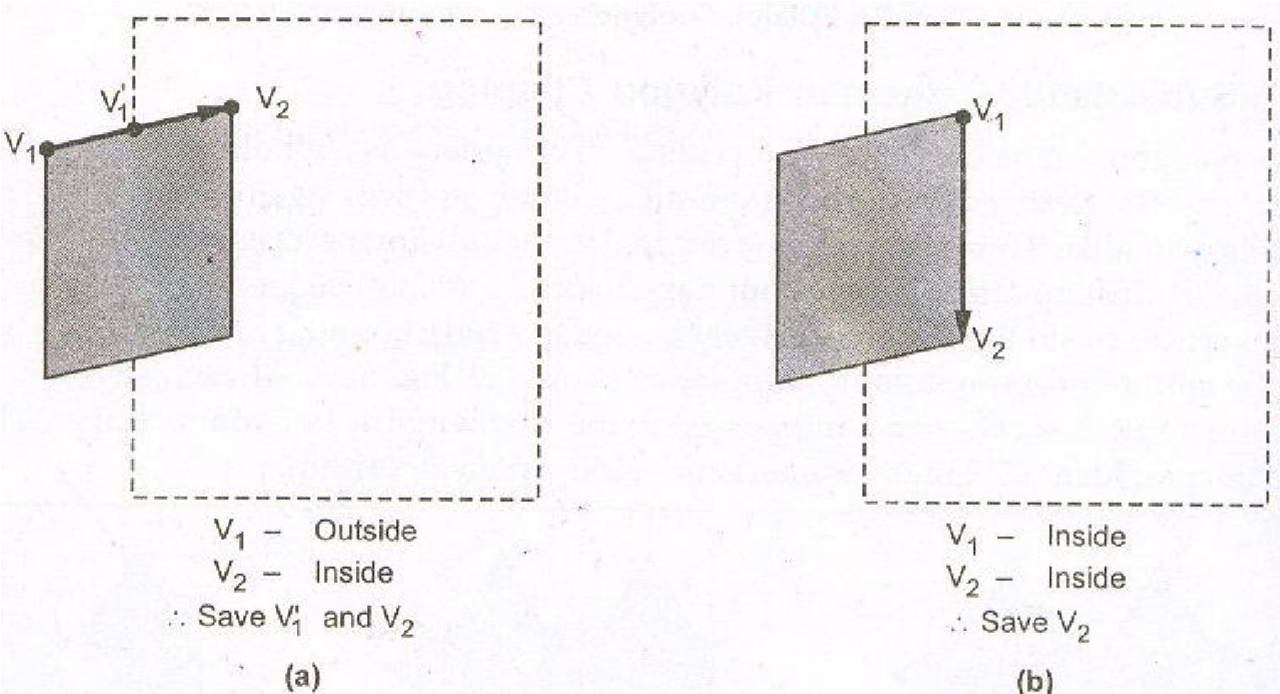
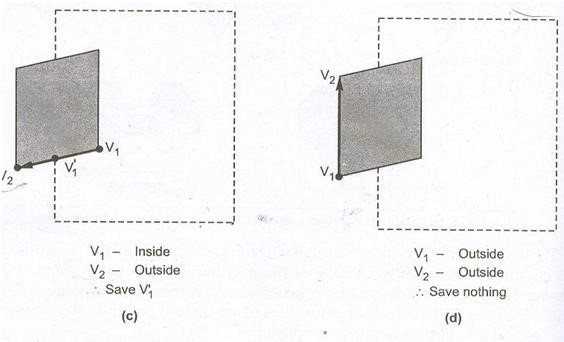
A polygon can be clipped by processing its boundary as a whole against each window edge. Beginning with the original set of polygon vertices, we could first clip the polygon against the left rectangle boundary to produce a new sequence of vertices. The new set of vertices could then be successively passed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



# Figure 5.14: Clipping a polygon against successive window boundaries

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

1. If the first vertex of the edge is outside the window boundary and the second vertex of the edge is inside then the intersection point of the polygon edge with the window boundary and the second vertex are added to the output vertex list (see fig a)
2. If both vertices of the edge are inside the window boundary, only the second vertex is added to the output vertex list ( see fig b)
3. If the first vertex of the edge is inside the window boundary and the second vertex of the edge is outside, only the edge intersection with the window boundary is added to the output vertex list (see fig c)
4. If both vertices of the edge are outside the window boundary, nothing is added to the output list (see fig d)



# Figure 5.15: Processing of Edges of polygon against left window boundary

Once all vertices are processed for one clip window boundary, the output list of vertices is clipped against the next window boundary.

# Sutherland-Hodgeman polygon clipping algorithm

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

**5.9 Assignment-5**

# Short Answer Questions (Each carries 2 Marks) 1. Define clipping. List any two types

1. Define window and viewport.
2. What is point clipping? Write the inequalities to be satisfied for saving a point for the purpose of display.
3. Write a diagram to show region codes which may be assigned to the end points of a straight line.
4. Define word coordinate system.
5. What is viewing transformation?
6. What is normalized device coordinates?
7. Differentiate between normalized transformation and workstation transformation.

# Long Answer Questions

1. Explain the technique of Sutherland Hodgman polygon clipping (4 Marks- M. U. April/May 2013)
2. Explain window to viewport transformation (5 Marks-M. U. April/May 2013, 2012)
3. Explain trivial accept, trivial reject and partial accept cases in Cohen-Sutherland method for line clipping with suitable examples. (6 Marks-M. U. April/May 2013)
4. Explain Cohen-Sutherland polygon clipping technique. (8 Marks- M. U. April/May 2012, 2010, 2009, 2008)
5. Explain viewing pipeline? (6 Marks)
6. What is clipping? Explain trivial accept, trivial reject and partial accept cases with suitable example (6 Marks-M. U. April/May 2011).
7. Explain Cohen Sutherland line clipping algorithm (8 Marks- M. U. April/May 2009, 2008)
8. Describe window to view port transformation. Derive the net matrix which represents this transformation (8 Marks- M. U. April/May 2008).
9. Write a short note on clipping circle and Ellipse (4 Marks).
10. Write a short note on point clipping (4 Marks)
11. Explain two dimensional viewing functions (6 Marks)
12. Explain about viewing coordinate reference frame (5 Marks)

# UNIT-III Chapter 6 Introduction to Multimedia

**6.1 What is multimedia?**

The simplest explanation of multimedia is “the combination of two or more media”. However multimedia is vast complex from than the term implies. It interlinks multiple media elements and, as a result makes for a more comprehensive end product than media components are experienced independently.

Media, by definition, is the plural of medium. It has facilitated to mean “facilitating or linking communication”-be it via phone, the web, TV, or some other instrument. The purpose of medium is to assist in the conveying of a message. At one time media is applied to newspaper as a way to broadcast news and information to the masses. Now media encompasses many forms of communication. Interactive multimedia comprises of Graphics, Animation, Photos, Sound, Video, and Text.

Multimedia is any combination of text, graphic art, sound, animation, and video delivered to you by computer or other electronic means. It is richly presented sensation. When you weave together the sensual elements of multimedia like dazzling pictures and animations, engaging sounds, compelling video clips, and raw textual information, you can electrify the thought and action centers of people’s minds. When you give them interactive control of the process, they can be enchanted. Multimedia stimulates eyes, ears, fingertips, and, most importantly, the head.

## 6.2 Definition

Multimedia is, a woven combinations of text, graphic art, sound, animation, and video elements. When you allow an end user (the viewer of a multimedia project) to control what and when the elements are delivered, it is interactive multimedia. When you provide a structure of linked elements through which the user can navigate, interactive multimedia becomes hypermedia.

Although the definition of multimedia is a simple one, making it work can be complicated. Not only do you need to understand how to make each multimedia element stand up and dance, but you also need to know how to use multimedia computer tools and technologies to weave them together. The people who weave multimedia into meaningful tapestries are multimedia developers.

The software vehicle, the messages, and the content presented on a computer or television screen, together constitute a multimedia project. If the project will be shipped or sold to consumers or end users, typically in a box or sleeve or on the Internet, with or without instructions, it is a multimedia title.

A multimedia project need not be interactive to be called multimedia: users can sit back and watch it just as they do a movie or the television. In such cases a project is linear, starting at a beginning and running through to an end. When users are given navigational control and can wander through the content at will, multimedia becomes nonlinear and interactive, and is a powerful personal gateway to information. When you allow an end user—also known as the viewer of a multimedia project—to control what and when the elements are delivered, it is called **interactive multimedia**. When you provide a structure of linked elements through which the user can navigate, interactive multimedia becomes **hypermedia**.

Determining how a user will interact with and navigate through the content of a project requires great attention to the message, the scripting or **storyboarding**, the artwork, and the programming. When the content is ready, it must be assembled in a process called **multimedia authoring.** These software tools are designed to manage individual multimedia elements and provide user interaction. **Integrated multimedia** is the “weaving” part of the multimedia definition, where source documents such as montages, graphics, video cuts, and sounds merge into a final presentation. In addition to providing a method for users to interact with the project, most authoring tools also offer facilities for creating and editing text and images, and they have extensions to drive videodisc players, videotape players, and other relevant hardware peripherals.

## 6.3 Use of Multimedia

Multimedia finds its application in various areas including, but not limited to, advertisements, art, education, entertainment, engineering, medicine, mathematics, business, scientific research and spatial, temporal applications. A few application areas of multimedia are listed below:

**Creative industries:** Creative industries use multimedia for a variety of purposes ranging from fine arts, to entertainment, to commercial art, to journalism, to media and software services provided for any of the industries listed below. An individual multimedia designer may cover the spectrum throughout their career. Request for their skills range from technical, to analytical and to creative.

**Commercial:** Much of the electronic old and new media utilized by commercial artists is multimedia. Exciting presentations are used to grab and keep attention in advertising. Industrial, business to business, and interoffice communications are often developed by creative services firms for advanced multimedia presentations beyond simple slide shows to sell ideas or liven-up training. Commercial multimedia developers may be hired to design for governmental services and nonprofit services applications as well.

**Entertainment and Fine Arts:** In addition, multimedia is heavily used in the entertainment industry, especially to develop special effects in movies and animations. Multimedia games are a popular pastime and are software programs available either as CD-ROMs or online. Some video games also use multimedia features. Multimedia applications that allow users to actively participate instead of just sitting by as passive recipients of information are called Interactive Multimedia.

**Education:** In Education, multimedia is used to produce computer-based training courses (popularly called CBTs) and reference books like encyclopedia and almanacs. A CBT lets the user go through a series of presentations, text about a particular topic, and associated illustrations in various information formats. Edutainment is an informal term used to describe combining education with entertainment, especially multimedia entertainment.

**Engineering:** Software engineers may use multimedia in Computer Simulations for anything from entertainment to training such as military or industrial training. Multimedia for software interfaces are often done as collaboration between creative professionals and software engineers.

**Industry:** In the Industrial sector, multimedia is used as a way to help present information to shareholders, superiors and coworkers. Multimedia is also helpful for providing employee training, advertising and selling products all over the world via virtually unlimited web-based technologies.

**Mathematical and Scientific Research**: In Mathematical and Scientific Research, multimedia is mainly used for modeling and simulation. For example, a scientist can look at a molecular model of a particular substance and manipulate it to arrive at a new substance. Representative research can be found in journals such as the Journal of Multimedia.

**Medicine**: In Medicine, doctors can get trained by looking at a virtual surgery or they can simulate how the human body is affected by diseases spread by viruses and bacteria and then develop techniques to prevent it.

**Multimedia in Public Places:** In hotels, railway stations, shopping malls, museums, and grocery stores, multimedia will become available at stand-alone terminals or kiosks to provide information and help. Such installation reduce demand on traditional information booths and personnel, add value, and they can work around the clock, even in the middle of the night, when live help is off duty.

A menu screen from a supermarket kiosk that provide services ranging from meal planning to coupons. Hotel kiosk list nearby restaurant, maps of the city, airline schedules, and provide guest services such as automated checkout. Printers are often attached so users can walk away with a printed copy of the information. Museum kiosk are not only used to guide patrons through the exhibits, but when installed at each exhibit, provide great added depth, allowing visitors to browser though richly detailed information specific to that display.

**Virtual reality:** At the convergence of technology and creative invention in multimedia is virtual reality, or VR. VR requires terrific computing horsepower to be realistic. In VR, your cyberspace is made up of many thousands of geometric objects plotted in three-dimensional space: the more objects and the more points that describe the objects, the higher resolution and the more realistic your view. As the user moves about, each motion or action requires the computer to recalculate the position, angle size, and shape of all the objects that make up your view, and many thousands of computations must occur as fast as 30 times per second to seem smooth.

Virtual reality (VR) is an extension of multimedia and it uses the basic multimedia elements of imagery, sound, and animation. Because it requires instrumented feedback from a wired-up person, VR is perhaps interactive multimedia at its fullest extension.

## 6.4 Delivering Multimedia

Multimedia requires large amounts of digital memory when stored in an end user’s library, or large amounts of bandwidth when distributed over wires, glass fiber, or airwaves on a network. The greater the bandwidth, the bigger the pipeline, so more content can be delivered to end users quickly.

**CD-ROM, DVD, Flash Drives:** CD-ROM (compact disc read-only memory) discs can be massproduced for pennies and can contain up to 80 minutes of full-screen video, images, or sound. The disc can also contain unique mixes of images, sounds, text, video, and animations controlled by an authoring system to provide unlimited user interaction.

Multilayered Digital Versatile Disc (DVD) technology increases the capacity and multimedia capability of CDs to 4.7GB on a single-sided, single-layered disc to as much as 17.08GB of storage on a double-sided, double-layered disc. CD and DVD burners are used for reading discs and for making them, too, in audio, video, and data formats.

Flash drives and thumb drives that do not require moving parts. As high speed connections become more and more pervasive and users become better connected, copper wire, glass fiber, and radio/cellular technologies may prevail as the most common delivery means for interactive multimedia files. These are served across the broadband Internet or from dedicated computer farms and storage facilities

**The Broadband Internet:** Today telecommunications networks are global, so when information providers and content owners determine the worth of their products and how to charge money for them, information elements will ultimately link up online as distributed resources on a data highway, where you will pay to acquire and use multimedia-based information.

Interactive multimedia is delivered to many homes throughout the world. Interest is a convergence of entertainment mega-corps, information publishers and providers, cable and telephone companies, and hardware and software manufacturers is driving this inevitable evolution, and profound changes in global communications strategy are on the drawing boards. What will be piped through this new system for entertainment, reference, and lifelong learning experiences are the very multimedia elements, including text, graphics, animation, sound, and video.

## 6.5 Assignment-6

**Short Answer Questions (Each carries 2 Marks)** 1. What is multimedia? (M. U. April/May-2013, 2012, 2011, 2009, 2008)

1. What is interactive media?
2. Define hypermedia.
3. What is virtual reality? Give examples.
4. Define multimedia authoring?
5. List out any four application areas of multimedia.
6. What are linear and nonlinear multimedia projects?
7. Define storyboarding.
8. Define integrated multimedia.
9. What is the purpose of multimedia authoring tools?

# Long Answer Questions

1. Write a short note on Multimedia. (4 Marks)
2. Briefly explain the multimedia uses (10 Marks)
3. Write a short note on virtual reality. (4 Marks)
4. Write a short note on CD-ROM, DVD and Flash Drives**.** (4 Marks)
5. Write a short note on broadband and internet. (4 Marks)
6. Briefly explain delivering multimedia. (6 Marks)
7. Write a short note on multimedia in Education and in public places (6 Marks)
8. Write a short note on multimedia in Industry, Entertainment and Fine arts. (4 Marks)
9. Write a short note on following (5 Marks)
   1. Multimedia in creative industry (ii) Multimedia in commercial
10. Write a short note on following (5Marks)
    1. Multimedia in engineering (ii) Multimedia in Mathematical and Scientific **UNIT-III**

# Chapter 7 Text

## 7.1 The Power of Meaning

The text in the multimedia is used to communicate information to the user. Proper use of text and words in multimedia presentation will help the content developer to communicate the idea and message to the user.

In multimedia, text is the words that will appear in your titles, menus, and navigation aids as well as in your narrative or content. It's important to design labels for title screens, menus, and buttons or tabs using words that have the most precise and powerful meanings to express what you need to say. Words and symbols in any form, spoken or written, are the most common system of communication. They deliver the most widely understood meaning to the greatest number of people-accurately and in detail. Because of this, they are vital elements of multimedia menus, navigation systems, keyword lists, and content.

## 7.2 About Fonts and Faces

A typeface is family of graphic characters that usually includes many type sizes and styles. A font is a collection of characters of a single size and style belonging to a particular typeface family. Typical font styles are bold face and italic. Other style attributes such as underlining and outlining of characters, may be added at the users choice.

The size of a text is usually measured in points. One point is approximately 1/72 of an inch i.e. 0.0138. The size of a font does not exactly describe the height or width of its characters. This is because the x-height (the height of lower case character x) of two fonts may differ.

Typefaces of fonts can be described in many ways, but the most common characterization of a typeface is serif and sans serif. The serif is the little decoration at the end of a letter stroke.

Times, Times New Roman, Bookman are some fonts which come under serif category. Arial, Optima, Verdana are some examples of sans serif font. Serif fonts are generally used for body of the text for better readability and sans serif fonts are generally used for headings. The following font shows a few categories of serif and sans serif fonts.

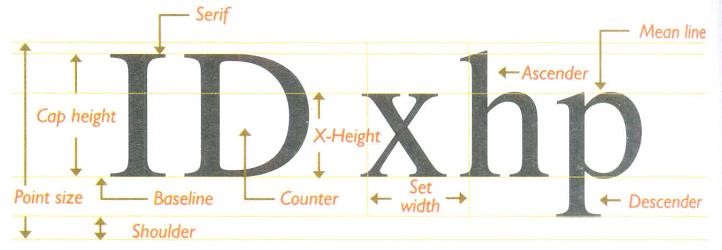


A font's size does not exactly describe the height or width of its characters. This is because the xheight (the height of the lowercase letter x) of two fonts may vary, while the height of the capital letters of those fonts may be the same (see Figure 7.1). Computer fonts automatically add space below the descender (and sometimes above) to provide appropriate line spacing, or leading (pronounced "ledding," named for the thin strips of lead inserted between the lines by traditional typesetters).

# Figure 7.1: The measurement of type

Leading can be adjusted in most programs on both the Macintosh and the PC. Character metrics are the general measurements applied to individual characters; kerning is the spacing between character pair. The metrics of a font can be altered to create interesting effects. For example, you can adjust the body width of each character from regular to condensed to expanded. We can adjust the spacing between characters (tracking) and the kerning between pairs of characters:

When a computer converts the letter A from a mathematical representation to a recognizable symbol displayed on the screen or in printed output (a process called rasterizing), the computer must know how to represent the letter using tiny square pixels (picture elements), or dots. It does this according to the hardware available and your specification, from a choice of available typefaces and fonts. Search for "free fonts." High-resolution monitors and printers can make more attractive-looking and varied characters because there are more fine little squares or dots per inch (dpi).



Placing an uppercase letter in the middle of a word, called an intercap, is a trend that emerged from the computer programming community, where coders discovered they could better recognize the words they used for variables and commands when the words were lowercase but intercapped.

## 7.3 Using fonts in Multimedia

It is a very difficult process to choose the fonts to be used in a multimedia presentation. Following are a few guidelines which help to choose a font in a multimedia presentation.

As many number of type faces can be used in a single presentation, this concept of using many fonts in a single page is called ransom-note topography.

* For small type, it is advisable to use the most legible font.
* In large size headlines, the kerning (spacing between the letters) can be adjusted
* In text blocks, the leading for the most pleasing line can be adjusted.
* Drop caps and initial caps can be used to accent the words.
* The different effects and colors of a font can be chosen in order to make the text look in a distinct manner.
* Anti aliased can be used to make a text look gentle and blended. This can give a more professional appearance. Anti-aliasing blends the colors along the edges of the letters (called dithering) to create a soft transition between the letter and its background.
* For special attention to the text the words can be wrapped onto a sphere or bent like a wave. Most word processors and text editors will let you create drop caps and SMALL CAPS in your text.
* Meaningful words and phrases can be used for links and menu items.
* In case of text links (anchors) on web pages the messages can be accented.
* If you are using centered type in a text block, keep the number of lines and their width to a minimum.
* For attention-grabbing results with single words or short phrases, try graphically altering and distorting your text and delivering the result as an image.
* Pick the fonts that seem right to you for getting your message across, and then doublecheck your choice against other opinions. Learn to accept criticism.
* Bold or emphasize text to highlight ideas or concepts, but do not make text look like a link or a button when it is not.
* On a web page, put vital text elements and menus in the top 320 pixels.

## 7.4 Using Text in Multimedia

A single item of menu text accompanied by a single action (a mouse click, keystroke, or finger pressed to the monitor) requires little training and is clean and immediate. Use text for titles and headlines (what it's all about), for menus (where to go), for navigation (how to get there), and for content (what you see when you get there).

**Designing with text:** A Computer screen provides a very small workspace for developing complex ideas. At some time or another, we will need to deliver high-impact or concise text messages on the computer screen in as condensed a form as possible. From a design perspective, our choice of font size and the number of headlines placed on a particular screen must be related both to the complexity of message and to its venue.

If messages are part of an interactive project or web site where we know the user is seeking information, we can pack a great deal of text information onto the screen. Seekers want dense material, and while they travel along the navigational pathways, they will scroll through relevant text and study the details. On the other hand, if used for creating presentation slides for public speaking support, the text will be keyed to a live presentation where the text accents the main message. In this case, use bulleted points in large fonts and few words with lots of white space. Coding an initial cap for a web page is simple. Use CSS (Cascading Style Sheets) attributes: CSS is used in web pages to create dynamic web pages.

**Animating Text:** There are plenty of ways to retain a viewer's attention when displaying text. For example, we can animate bulleted text and have it "fly" onto the screen. We can "grow" a headline a character at a time. For public speakers, simply highlighting the important text works well as a pointing device. When there are several points to be made, we can stack keywords and flash them past the viewer in a timed automated sequence.

**Symbols and Icons:** Symbols are concentrated text in the form of stand-alone graphic constructs. Symbols convey meaningful messages. Though we may think of symbols as belonging strictly to the realm of graphic art, in multimedia we should treat them as text-or visual words-because they carry meaning. Symbols such as the familiar trash are more properly called icons. When early computers began to display bitmaps as well as lines of text, there was a flurry of creative attempts by graphic artists to create interesting navigational symbols to alleviate the need for text.

**Menus for Navigation:** An interactive multimedia project or web site typically consists of a body of information, or content, through which a user navigates by pressing a key, clicking a mouse, or pressing a touch screen. The simplest menus consist of text lists of topics. Users choose a topic, click it, and go there. As multimedia and graphical user interfaces become persistent in the computer community, certain intuitive actions are being widely learned. Text is helpful to users to provide persistent cues about their location within the body of content.

**Buttons for Interaction:** In multimedia, buttons are the objects, such as blocks of text, a pretty blue triangle, or a photograph, that make things happen when they are clicked. They were invented for the sole purpose of being pushed or prodded with cursor, mouse, key, or finger. These buttons are used to manifest properties such as highlighting or other visual or sound effects to indicate that we can hit the target. The automatic button-making tools supplied with multimedia and HTML page authoring systems are useful, in creating the text for user. They offer little opportunity to fine-tune the look of the text.

**Fields for reading:** Unless the very purpose of our multimedia project or web site is to display large blocks of text, try to present to the user only a few paragraphs of text per page. Use a font that is easy to read rather than a prettier font that is illegible. Try to display whole paragraphs on the screen, and avoid breaks where users must go back and forth between pages to read an entire paragraph.

**Portrait vs. Landscape:** Traditional hard-copy and printed documents in the taller-than-wide orientation are simply not readable on a typical monitor with a wider than tall aspect ratio. The taller-than-wide orientation used for printed documents is called **portrait**; this is the 8.5-by-11inch size unique to the United States or the internationally designated standard A4 size, 8.27 by 11.69 inches. The wider-than-tall orientation normal to monitors is called **landscape**. Shrinking an 11-inch-tall portrait page of text into your available monitor height usually yields illegible chicken tracks.

## 7.5 Computers and Text

Postscript fonts are a method of describing an image in terms of mathematical constructs (Bezier curves), so it is used not only to describe the individual characters of a font but also to describe illustrations and whole pages of text. Since postscript makes use of mathematical formula, it can be easily scaled bigger or smaller. Apple and Microsoft announced a joint effort to develop a better and faster quadratic curves outline font methodology, called true type. In addition to printing smooth characters on printers, TrueType would draw characters to a low resolution (72 dpi or 96 dpi) monitor. Adobe and Microsoft then developed a new and improved font management system incorporating the best features of both PostScript and TrueType, and by 2007, OpenType became a free, publicly available international standard. If a specified font doesn’t exist on the target machine, a substitute must be provided that does exist on the target. This is **font substitution**. Translating or designing multimedia (or any computer-based material) into a language other than the one in which it was originally written is called **localization**.

## 7.6 Font editing and design tools

There is several software that can be used to create customized font. These tools help an multimedia developer to communicate his idea or the graphic feeling.

Using these software different typefaces can be created. In some multimedia projects it may be required to create special characters. Using the font editing tools it is possible to create a special symbols and use it in the entire text. Following is the list of software that can be used for editing and creating fonts:

* Fontographer
* Fontmonger
* Cool 3D text

Special font editing tools can be used to make your own type so you can communicate an idea or graphic feeling exactly. With these tools professional typographers create distinct text and display faces.

**Fontographer:** It is macromedia product; it is a specialized graphics editor for both Macintosh and Windows platforms. You can use it to create postscript, truetype and bitmapped fonts for Macintosh and Windows. Fontographer's features include a freehand drawing tool to create professional and precise inline and outline drawings of calligraphic and script characters, using either the mouse or alternative input method

**Making Pretty Text:** To make your text look pretty you need a toolbox full of fonts and special graphics applications that can stretch, shade, color and anti-alias your words into real artwork. Pretty text can be found in bitmapped drawings where characters have been tweaked, manipulated and blended into a graphic image. We can colorize the text, stretch, squeeze, and rotate it, and you can filter it through various plug-ins to generate wild graphic results.

## 7.7 Hypermedia and Hypertext

Multimedia is the combination of text, graphic, and audio elements into a single collection or presentation, becomes interactive multimedia when you give the user some control over what information is viewed and when it is viewed.

When text is stored in a computer instead of on printed pages, the computer's powerful processing capabilities can be applied to make the text more accessible and meaningful. The text can then be called hypertext; because the words, sections, and thoughts are linked, the user can navigate through text in a nonlinear way, quickly and intuitively. Because hypertext is the organized cross-linking of words not only to other words but also to associated images, video clips, sounds, and other exhibits, hypertext often becomes simply an additional feature within an over all multimedia design.

**Hypermedia Structures:** Two Buzzwords used often in hypertext are link and node. Links are connections between the conceptual elements, that is, the nodes that may consists of text, graphics, sounds or related information in the knowledge base. A link anchor is where you come from; a link end is the destination node linked to the anchor. Some hypertext systems provide unidirectional navigation and offer no return pathway; others are bidirectional. The simplest way to navigate hypermedia structures is via buttons that let you access linked information (text, graphics, and sounds) that is contained at the nodes.

**Searching for words:** There are typical methods for a word searching in hypermedia systems: Categories: Selecting or limiting the documents, pages, or fields of text within which to search for a word or words.

* Word relationship: Searching for words according to their general proximity and order.
* Adjacency: Searching for words occurring next to one another, usually in phrases and proper names.
* Alternates: Applying an OR criterion to search for two or more words.
* Association: Applying an AND criterion to search for two or more words.
* Negation: Applying a NOT criterion to search exclusively for references to a word that are not associated with the word.
* Truncation: Searching for a word with any of its possible suffixes.
* Intermediate words: Searching for words that occur between what might normally be adjacent words.
* Frequency: Searching for words based on how often they appear.

**7.8 Assignment-7**

# Short Answer Questions (Each carries 2 Marks)

1. Differentiate between typefaces and fonts.
2. Differentiate between leading, kerning and tracking.
3. Two types of typeface widely used and give example.
4. What are CSS?
5. Differentiate symbols and icons.
6. What are buttons? Where and all we can apply the actions.
7. Define font.
8. List any four font styles.
9. Define leading and kerning.
10. Differentiate serif and sans serif fronts.
11. Differentiate portrait and landscape documents.
12. What is font substitution?
13. What is localization with respect to multimedia?
14. List different values for alignment.
15. What is typeface?
16. List four attributes of a font.

**Long Answer Questions**

1. Write a note on power of meaning of text. (4 Marks)
2. List and explain different attributes of font. (5 Marks) 3. Explain different ways of presenting text in multimedia.
3. Explain about fonts and faces (7 Marks).
4. Explain about using fonts in multimedia (7 Marks).
5. Briefly explain about using text in multimedia. (8 Marks)
6. Write a short note on computers and text (4 Marks)
7. Explain about font editing and designing tools (6 Marks)
8. Write a short note on hypermedia and hypertext (5 Marks)
9. How to search for words? Explain (5 Marks)

# UNIT-III Chapter 8 Images

## 8.1 How to create images

Still images are the important element of a multimedia project or a web site. In order to make a multimedia presentation look elegant and complete, it is necessary to spend ample amount of time to design the graphics and the layouts. Competent, computer literate skills in graphic art and design are vital to the success of a multimedia project.

A digital image is represented by a matrix of numeric values each representing a quantized intensity value. When I is a two-dimensional matrix, then I (r, c) is the intensity value at the position corresponding to row r and column c of the matrix.

The points at which an image is sampled are known as picture elements, commonly abbreviated as pixels. The pixel values of intensity images are called gray scale levels (we encode here the “color” of the image). The intensity at each pixel is represented by an integer and is determined from the continuous image by averaging over a small neighborhood around the pixel location. If there are just two intensity values, for example, black, and white, they are represented by the numbers 0 and 1; such images are called binary-valued images. If 8-bit integers are used to store each pixel value, the gray levels range from 0 (black) to 255 (white).

There are different kinds of image formats in the literature. We shall consider the image format that comes out of an image frame grabber, i.e., the captured image format, and the format when images are stored, i.e., the stored image format.

Captured Image Format: Captured Image Format is specified by two main parameters: spatial resolution, which is specified as pixels (eg. 640x480) and color encoding, which is specified by bits per pixel. Both parameter values depend on hardware and software for input/output of images.

Stored Image Format: When we store an image, we are storing a two-dimensional array of values, in which each value represents the data associated with a pixel in the image. For a bitmap, this value is a binary digit.

A bitmap is a simple information matrix describing the individual dots that are the smallest elements of resolution on a computer screen or other display or printing device. A onedimensional matrix is required for monochrome (black and white); greater depth (more bits of information) is required to describe more than 16 million colors the picture elements may have, as illustrated in following figure. The state of all the pixels on a computer screen make up the image seen by the viewer, whether in combinations of black and white or colored pixels in a line of text, a photograph-like picture, or a simple background pattern.

**Bitmap Sources:** Make a bitmap from scratch with paint or drawing program. Grab a bitmap from an active computer screen with a screen capture program, and then paste into a paint program or your application. Capture a bitmap from a photo, artwork, or a television image using a scanner or video capture device that digitizes the image.

# 8.2: Making still images

Still images may be small or large, or even full screen. Whatever their form, still mages are generated by the computer in two ways: as bitmap (or paint graphics) and as vector-drawn (or just plain drawn) graphics.

Bitmaps are used for photo-realistic images and for complex drawing requiring fine detail. Vector-drawn objects are used for lines, boxes, circles, polygons, and other graphic shapes that can be mathematically expressed in angles, coordinates, and distances. A drawn object can be filled with color and patterns, and you can select it as a single object. Typically, image files are compressed to save memory and disk space; many image formats already use compression within the file itself – for example, GIF, PEG, and PNG. Still images may be the most important element of multimedia project. If you are designing multimedia by yourself, put yourself in the role of graphic artist and layout designer.

**Bitmap Software:** The abilities and feature of image-editing programs for both the Macintosh and Windows range from simple to complex. The Macintosh does not ship with a painting tool, and Windows provides only the rudimentary Paint (see following figure), so you will need to acquire this very important software separately, often bitmap editing or painting programs come as part of a bundle when you purchase your computer, monitor, or scanner.

**Capturing and Editing Images:** The image that is seen on a computer monitor is digital bitmap stored in video memory, updated about every 1/60 second or faster, depending upon monitor’s scan rate. When the images are assembled for multimedia project, it may often be needed to capture and store an image directly from screen. It is possible to use the PrtScr key available in the keyboard to capture a image.

A clip art collection may contain a random assortment of images, or it may contain a series of graphics, photographs, sound, and video related to a single topic. For example, Corel, Micrografx, and Fractal Design bundle extensive clip art collection with their image-editing software.

Morphing is another effect that can be used to manipulate still images or to create interesting and often bizarre animated transformations. Morphing allows you to smoothly blend two images so that one image seems to melt into the next, often producing some amusing results.

**Scanning Images:** After scanning through countless clip art collections, if it is not possible to find the unusual background you want for a screen about gardening. Sometimes when you search for something too hard, you don’t realize that it’s right in front of your face. Open the scan in an image-editing program and experiment with different filters, the contrast, and various special effects. Be creative, and don’t be afraid to try strange combinations, sometimes mistakes yield the most intriguing results.

## 8.3 Color

Color is a vital component of multimedia. Management of color is both a subjective and a technical exercise. Picking the right colors and combinations of colors for your project can involve many tries until you feel the result is right.

**Understanding Natural Light and Color:** The letters of the mnemonic ROY G. BIV, learned by many of us to remember the colors of the rainbow, are the ascending frequencies of the visible light spectrum: red, orange, yellow, green, blue, indigo, and violet. Ultraviolet light, on the other hand, is beyond the higher end of the visible spectrum and can be damaging to humans. The color white is a noisy mixture of all the color frequencies in the visible spectrum. The cornea of the eye acts as a lens to focus light rays onto the retina. The light rays stimulate many thousands of specialized nerves called rods and cones that cover the surface of the retina. The eye can differentiate among millions of colors, or hues, consisting of combination of red, green, and blue.

**Computerized Color:** Because the eye's receptors are sensitive to red, green, and blue light, by adjusting combinations of these three colors, the eye and brain will interpolate the combinations of colors in between. Although the eye perceives colors based upon red, green, and blue, there are actually two basic methods of making color: additive and subtractive.

* **Additive color:** In additive color model, a color is created by combining colored light sources in three primary colors: red, green and blue (RGB). This is the process used for a TV or computer monitor. On the back of the glass face of a CRT are thousands of phosphorescing chemical dots. These dots are each about 0.30mm or less in diameter (the dot pitch), and are positioned very carefully and very close together, arranged in triads of red, green, and blue. These dots are bombarded by electrons that "paint" the screen at high speeds (about 60 times a second). The red, green, and blue dots light up when hit by the electron beam. Your eye sees the combination of red, green, and blue light and interpolates it to create all other colors.

* **Subtractive Color:** In subtractive color method, a new color is created by combining colored media such as paints or ink that absorb (or subtract) some parts of the color spectrum of light and reflect the others back to the eye. Subtractive color is the process used to create color in printing. The printed page is made up of tiny halftone dots of three primary colors, cyan, magenta and yellow (CMY).

**Computer color Models:** Models or methodologies are used to specify colors in computer terms are RGB, HSB (hue, saturation, and brightness), HSL (hue, saturation, and lightness), CMYK, CIE, and others. Using the 24-bit RGB (red, green, blue) model, you specify a color by setting each amount of red, green, and blue to a value in a range of 256 choices, from 0 to 255. Eight bits of memory are required to define those 256 possible choices, and that has to be done for each of the three primary colors; a total of 24 bits of memory (8 + 8 + 8 = 24) are therefore needed to describe the exact color, which is one of "millions" (256 x 256 x 256 = 16,777,216).

**Color Palettes:** Palettes are mathematical tables that define the color of a pixel displayed on the screen. The most common palettes are 1, 4, 8, 16, and 24 bits deep.

**Dithering:** InDithering process, the color value of each pixel is changed to the closest matching color value in the target palette, using a mathematical algorithm. Often the adjacent pixels are also examined, and patterns of different colors are created in the more limited palette to best represent the original colors.

|  |  |
| --- | --- |
| Format | Extension |
| Microsoft Windows DIB | .bmp .dib .rle |
| Microsoft Palette | .pal |
| AutoCAD format 2D | .dxf |
| JPEG | .jpg |
| Windows Meta file | .wmf |
| Portable network graphic | .png |
| CompuServe gif | .gif |
| Apple Macintosh | .pict .pic .pct |

## 8.4 Image File Formats

There are many file formats used to store bitmaps and vectored drawing.

**BMP:** A graphics format native to Windows and OS/2. BMP is widely used on the PCs for icons and wallpaper. The BMP file format is not recommended for use with high-quality graphics images or photographs because it uses only 256 unique colors.

**PICT:** The native format defined by Apple for the Mac. It is widely used on Macs but less so on PCs.

**TIFF:** (Tagged Image File Format) Bitmap format defined in 1986 by Microsoft and Aldus and widely used on Macs and PCs. This format is usually the best to use when exchanging bitmap files that will be printed or edited further.

**JPEG:** (Joint Photographic Experts Group) JPEG is often abbreviated as JPG (pronounced JAYpeg). This bitmap format is common on the World Wide Web and is often used for photos and other high resolution (24 bit or millions of colors) images that will be viewed on the screen.

**GIF:** (Graphic Interchange Format) Like JPEG images, Gif images are often found on WWW pages. Unlike JPEG images, GIF images are reduced to 256 or fewer unique colors.

**PNG:** (Portable Network Graphics) This format was developed as an alternative to GIF and JPEG, PNG, like JPG, can store color images in a small amount of space, but PNG files can also store transparency information the way GIF files do. The PNG format designed mainly for use in Web pages.

Following are a list of few other image file formats.

# Table 8.1: Image file Formats

**8.5 Assignment-8**

# Short answer questions (Each carries 2 Marks)

1. What is an image? (M. U. April/May 2013)
2. Differentiate between stored image format and captured image format.
3. What is Morphing?
4. Differentiate between additive color and subtractive color.
5. What is meant by dithering? (M. U. April/May 2013) 6. Write a note on sampling. (M. U. April/May 2008)
6. Expand JPEG and PNG.
7. List out any four image file formats.
8. Expand CYMK and HSB with respect to computer color models.
9. What are Color Palettes?

# Long Answer Questions

1. How to create images explain? (6 Marks)
2. Briefly explain about making still images (7 Marks)
3. Write a short note on capturing and editing images and scanning images (4 marks)
4. Write a short note on computerized color (5 Marks)
5. Explain different image file formats (5 Marks)
6. Explain about color as a vital component in multimedia. (4 Marks)
7. Write a short note on Captured image format (4 marks)
8. Write a short note on bitmap images
9. Write a short note on bitmap softwares (3 Marks)
10. Write a short note on capturing and editing images (5 Marks)

# UNIT-III Chapter 9 Sound

## 9.1 The power of Sound

Sound is perhaps the most important element of multimedia. It is meaningful “speech” in any language, from a whisper to a scream. It can provide the listening pleasure of music, the startling accent of special effects or the ambience of a mood setting background. Sound is the terminology used in the analog form, and the digitized form of sound is called as audio.

When something vibrates in the air is moving back and forth it creates wave of pressure. These waves spread like ripples from pebble tossed into a still pool and when it reaches the eardrums, the change of pressure or vibration is experienced as sound. Acoustics is the branch of physics that studies sound. Sound pressure levels are measured in decibels (db); a decibel measurement is actually the ratio between a chosen reference point on a logarithmic scale and the level that is actually experienced.

The waveform repeats the same shape at regular intervals and this portion is called a period. Sounds that display a recognizable periodicity tend to be more musical than those that are non periodic. Examples of periodic sounds sources are musical instruments, vowel sounds, the whistling wind and bird songs. Non periodic sound sources include unpitched percussion instruments, coughs and sneezes and rushing water.

The frequency of a sound is the reciprocal value of the period; it represents the number of period in a second and is measured in hertz (Hz) or cycles per second (cps). A convenient abbreviation, KHz (kilohertz), is used to indicate thousands of oscillation per second: 1 kHz equals 1000Hz The frequency range is divided into:

Infra Sound: from 0 t 20Hz.

Human hearing frequency range: from 20Hz to 20 kHz. Ultrasound: from 20 kHz to 1GHZ

Hypersound: from 1GHZto 10THz.

## 9.2 Digital Audio

Digital audio is created when a sound wave is converted into numbers, a process referred to as digitizing. It is possible to digitize sound from a microphone, a synthesizer, existing tape recordings, live radio and television broadcasts, and popular CDs. You can digitize sounds from a natural source or prerecorded.

Digitized sound is sampled sound. Ever nth fraction of a second, a sample of sound is taken and stored as digital information in bits and bytes. The quality of this digital recording depends upon how often the samples are taken. The rate at which a continuous waveform is sampled is called the sampling rate. Sampling rates are measured in Hz.

# Preparing digital Audio Files

Preparing digital audio files is fairly straight forward method, if you have analog source materials, music or sound effects that you have recorded on analog media such as cassette tapes.

* The first step is to digitize the analog material and recording it onto a computer readable digital media.
* It is necessary to focus on two crucial aspects of preparing digital audio files:
  + Balancing the need for sound quality against your available RAM and Hard disk resources.
  + Setting proper recording levels to get a good, clean recording.

Remember that the sampling rate determines the frequency at which samples will be drawn for the recording. Sampling at higher rates more accurately captures the high frequency content of your sound. Audio resolution determines the accuracy with which a sound can be digitized.

Formula for determining the size of the digital audio

Monophonic = Sampling rate × duration of recording in seconds × (bit resolution / 8) × 1 Stereo = Sampling rate × duration of recording in seconds ×(bit resolution / 8) × 2

The sampling rate is how often the samples are taken.

* The sample size is the amount of information stored. This is called as bit resolution.
* The number of channels is 2 for stereo and 1 for monophonic.  The time span of the recording is measured in seconds.

Editing Digital Recording: Once a recording has been made, it will almost certainly need to be edited. The basic sound editing operations that most multimedia procedures needed are described in the paragraphs that follow

1. Multiple Tasks: Able to edit and combine multiple tracks and then merge the tracks and export them in a final mix to a single audio file.
2. Trimming: Removing dead air or blank space from the front of a recording and an unnecessary extra time off the end is your first sound editing task.
3. Splicing and Assembly: Using the same tools mentioned for trimming, you will probably want to remove the extraneous noises that inevitably creep into recording.
4. Volume Adjustments: If you are trying to assemble ten different recordings into a single track there is a little chance that all the segments have the same volume.
5. Format Conversion: In some cases your digital audio editing software might read a format different from that read by your presentation or authoring program.
6. Resampling or downsampling: If you have recorded and edited your sounds at 16 bit sampling rates but are using lower rates you must resample or downsample the file.
7. Equalization: Some programs offer digital equalization capabilities that allow you to modify recording frequency content so that it sounds brighter or darker.
8. Digital Signal Processing: Some programs allow you to process the signal with reverberation, multitap delay, and other special effects using DSP routines.
9. Reversing Sounds: Another simple manipulation is to reverse all or a portion of a digital audio recording. Sounds can produce a surreal, other wordly effect when played backward.
10. Time Stretching: Advanced programs let you alter the length of a sound file without changing its pitch. This feature can be very useful but watch out: most time stretching algorithms will severely degrade the audio quality.

## 9.3 MIDI audio

MIDI (Musical Instrument Digital Interface) is a communication standard developed for electronic musical instruments and computers. MIDI files allow music and sound synthesizers from different manufacturers to communicate with each other by sending messages along cables connected to the devices.

MIDI provides a protocol for passing detailed descriptions of a musical score, such as the notes, the sequences of notes, and the instrument that will play these notes. But MIDI data is not digitized sound; it is a shorthand representation of music stored in numeric form. A MID file is a list of time-stamped commands that are recordings of musical actions (the pressing down of a piano key or a sustain pedal, for example, or the movement of a control wheel or slider).

Creating your own original score can be one of the most creative and rewarding aspects of building a multimedia project, and MIDI (Musical Instrument Digital Interface) is the quickest, easiest and most flexible tool for this task.

The process of creating MIDI music is quite different from digitizing existing audio. To make MIDI scores, however you will need sequencer software and a sound synthesizer. The MIDI keyboard is also useful to simply the creation of musical scores.

MIDI software creates data about each note as it is played on a MIDI keyboard (or another MIDI device)—which note it is, how much pressure was used on the keyboard to play the note, how long it was sustained, and how long it takes for the note to decay or fade away, for example. This information, when played back through a MIDI device, allows the note to be reproduced exactly. Because the quality of the playback depends upon the end user’s MIDI device rather than the recording, MIDI is device dependent.

An advantage of structured data such as MIDI is the ease with which the music director can edit the data. To change instruments, you just change that value. Instruments that you can synthesize are identified by a General MIDI numbering system that ranges from 0 to 127.

Since MIDI is device dependent and the quality of consumer MIDI playback hardware varies greatly, MIDI’s true place in multimedia work may be as a production tool rather than a delivery medium. MIDI is by far the best way to create original music, so use MIDI to get the flexibility and creative control you want.

## 9.4 MIDI v/s Digital Audio

In contrast to MIDI data, digital audio data is the actual representation of a sound, stored in the form of thousands of individual numbers (samples). The digital data represents the instantaneous amplitude (or loudness) of a sound at discrete slices of time. MIDI data is to digital audio data what vector or drawn graphics are to bitmapped graphics. That is, MIDI data is device dependent; digital data is not.

MIDI has several advantages over digital audio and two huge disadvantages. First, the advantages:

* MIDI files are much more compact than digital audio files, and the size of a MIDI File is completely independent of playback quality. Because MIDI files are small, they don’t take up as much memory, disk space, or bandwidth.
* Because they are small, MIDI files embedded in web pages load and play more quickly than their digital equivalents.
* In some cases, if the MIDI sound source you are using is of high quality, MIDI files may sound better than digital audio files.
* You can change the length of a MIDI file (by varying its tempo) with out changing the pitch of the music or degrading the audio quality. MIDI data is completely editable
* Because they represent the pitch and length of notes, MIDI files can generally be converted to musical notation, and vice versa.

Following are the MIDI’s disadvantages:

* Because MIDI data does not represent sound but musical instruments you can be certain that playback will be accurate only if the MIDI play back device is identical to the device used for production.
* Also, MIDI cannot easily be used to play back spoken dialog, although expensive and technically tricky digital samplers are available.

In general, use MIDI in the following circumstances:

* Digital audio won’t work because you don’t have enough memory or bandwidth.  You have a high-quality MIDI sound source.
* You have complete control over the machines on which your program will be delivered, so you know that your users will have high-quality MIDI playback hardware.
* You don’t need spoken dialog.

The most important advantage of digital audio is its consistent playback quality, but this is where MIDI is the least reliable. With digital audio you can be more confident that the audio track for your multimedia project will sound as good in the end as it did in the beginning when you created it.

There are two additional and often more compelling reasons to work with digital audio:

* A wider selection of application software and system support for digital audio.
* The preparation and programming required for creating digital audio do not demand knowledge of music theory, while working with MIDI data usually does require a modicum of familiarity with musical scores, keyboards, and notation, as well as audio production.

In general, use digital audio in the following circumstances:

* You don’t have control over the playback hardware.
* You have the computing resources and bandwidth to handle digital files.
* You need spoken dialog.

## 9.5 Multimedia System Sounds

In Windows, system sounds are WAV files, and they reside in the Windows Media subdirectory. System event sounds include start.wav, chimes.wav, chord.wav, ding.wav, logof.wav, notify.wav, recycle.wav, tada.wav, and the Microsoft sound.wav that typically plays when Window starts up.We can assign these sounds to system events such as Windows startup, warnings from other applications. In OS X on a Macintosh, you can only change your system alert sound. We can Put custom sound file (in AIF format) into ~/System/Library/Sounds, then select it in the Sound preference pane.

## 9.6 Audio File Formats

A file format determines the application that is to be used for opening a file. A sound file’s format is simply a recognized methodology for organizing and (usually) compressing the digitized sound’s data bits and bytes into a data file. The file name extension identifies which method of storage is used. There are many ways to store the bits and bytes that describe a sampled waveform sound. The method used for consumer-grade music CD is Linear Pulse Code Modulation (LPCM), often shortened to PCM. LPCM tracks from an audio CD are usually converted and stored on a computer in uncompressed AIFF (Audio Interchange File Format) or wave format (WAV) files when copied from the CD.

The MP3 format was developed by the Moving Picture Experts Group (MPEG) the most common method for storing consumer audio. It incorporates a “lossy” compression algorithm to save space. WMA (Windows Media Audio) is a proprietary Microsoft format developed to improve MP3. OGG was developed as an open-source. MP4 is a format based on Apple’s QuickTime movie (.mov) “container” model and is similar to the MOV format, which stores various types of media, particularly time-based streams such as audio and video.

A codec (compressor-decompressor) is software that compresses a stream of audio or video data for storage or transmission, then decompresses it for playback. Following is the list of different file formats and the software that can be used for opening a specific file.

1. \*.AIF, \*.SDII in Macintosh Systems
2. \*.SND for Macintosh Systems
3. \*.WAV for Windows Systems
4. MIDI files – used by north Macintosh and Windows
5. \*.WMA –windows media player
6. \*.MP3 – MP3 audio
7. \*.RA – Real Player
8. \*.VOC – VOC Sound
9. AIFF sound format for Macintosh sound files
10. \*.OGG – Ogg Vorbis

## 9.7 Vaughan’s Law of Multimedia minimums

A classic physical anthropology law (Liebig’s Law of the Minimum) proposes that the evolution of eyesight, locomotor speed, sense of smell, or any other species trait will cease when that trait becomes sufficiently adequate to meet the survival adequate requirements of the competitive environment. If the trait is good enough, the organism expends no more effort improving it. Thus, if consumer-grade electronics and a handheld microphone are good enough for making your sound, and if you, your client, and your audience are all satisfied with the results, conserve your energy and money and avoid any more expenditure. And keep this Law of Minimums in mind when all trade-of decisions involving other areas of high technology and multimedia, too.

*Vaughan’s Law of Multimedia Minimums*: There is an acceptable minimum level of adequacy that will satisfy the audience, even when that level may not be the best that technology, money, or time and effort can buy.

## 9.8 Adding sounds to multimedia Project

Whether working on a Macintosh or in Windows, we need to follow certain steps to bring an audio recording into multimedia project. Here is a brief overview of the process:

1. Determine the File formats that are compatible with your multimedia authoring software and the delivery medium(s) you will be using (for File storage and bandwidth capacity).
2. Determine the sound playback capabilities (codecs and plug-ins) that the end user’s system offers.
3. Decide what kind of sound is needed (such as background music, special sound effects, and spoken dialog). Decide where these audio events will occur in the low of your project. Fit the sound cues into your storyboard or make up a cue sheet.
4. Decide where and when you want to use either digital audio or MIDI data.
5. Acquire source material by creating it from scratch or purchasing it.
6. Edit the sounds to it your project.
7. Test the sounds to be sure they are timed properly with the project’s images. This may involve repeating steps 1 through 4 until everything is in sync

When it’s time to import your compiled and edited sounds into your project, you’ll need to know how your particular multimedia software environment handles sound data. Each multimedia authoring program or web browser handles sound a bit differently.

In multimedia authoring environments, it is usually a simple matter to play a sound when the user clicks a button, but this may not be enough. If the user changes screens while a long file are playing, for example, you may need to program the sound to stop before leaving the current screen. If the file to be played cannot be found, you may need to code an entire section for error handling and file location. For web pages, you will need to embed a player and point to your sound file using HTML code.

**9.9 Assignment-9**

# Short Answer Questions (Each carries 2 marks)

1. Define the power of sound?
2. Define frequency and period?
3. Define digitizing of sound.
4. Write the formula for determining the size of the digital audio.
5. Define sampling rate.
6. Expand MIDI.
7. Mention the disadvantages of MIDI?
8. Mention any two advantages of MIDI over digital audio?
9. Mention any four multimedia system sounds.
10. Mention any four audio file formats.
11. What is Vaughan’s Law of Multimedia Minimums?

# Long Answer Questions

1. Write a short note on the power of sound. (5 Marks)
2. How to prepare digital audio files? Explain. (5 Marks) 3. Write a short note on editing digital recording (5 Marks).
3. Explain about MIDI audio (6 Marks).
4. Compare MIDI and Digital audio. (6 Marks)
5. Write a short note on multimedia system sounds. (3 Marks) 7. Explain about multimedia Audio file formats (5 Marks).
6. Write a short note on Vaughan’s Law of Multimedia Minimums. (4 Marks)
7. How to add sounds to multimedia project? Explain (5 Marks)
8. Explain in which circumstances digital audio and MIDI are preferred. (5 Marks) **UNIT-IV**

# Chapter 10 Animation

## 10.1 The power of motion

Animation makes static presentations come alive. It is visual change over time and can add great power to our multimedia projects. Carefully planned, well-executed video clips can make a dramatic difference in a multimedia project. Animation is created from drawn pictures and video is created using real time visuals. Visual effects such as wipes, fades, zooms, and dissolves are available in most multimedia authoring packages, and some of these can be used for primitive animation. For example, you can slide images onto the screen with a wipe, or you can make an object implode with an iris/close effect.

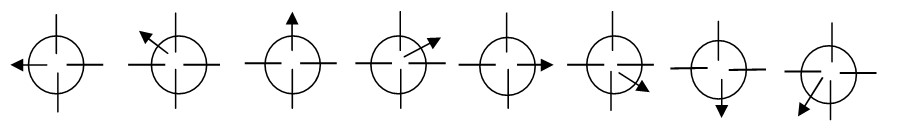
But animation is more than wipes, fades, and zooms. Animation is an object actually moving across or into or out of the screen; a spinning globe of our earth; a car driving along a line-art highway.

## 10.2 Principle of Animation

Animation is the rapid display of a sequence of images of 2-D artwork or model positions in order to create an illusion of movement. It is an optical illusion of motion due to the phenomenon of persistence of vision, and can be created and demonstrated in a number of ways. The most common method of presenting animation is as a motion picture or video program, although several other forms of presenting animation also exist.

Animation is possible because of a biological phenomenon known as persistence of vision and a psychological phenomenon called phi. An object seen by the human eye remains chemically mapped on the eye’s retina for a brief time after viewing. Combined with the human mind’s need to conceptually complete a perceived action, this makes it possible for a series of images that are changed very slightly and very rapidly, one after the other, to seemingly blend together into a visual illusion of movement. The following shows a few cells or frames of a rotating logo. When the images are progressively and rapidly changed, the arrow of the compass is perceived to be spinning.

Digital television video builds 24, 30, or 60 entire frames or pictures every second, depending upon settings; the speed with which each frame replaced by the next one makes the images appear to blend smoothly movement. Movies on film are typically shot at a shutter rate of 24 frames per second, but using projection tricks (the projector's shutter flashes light through each image twice), the flicker rate is increased to 48 times per second, and the human eye thus sees a motion picture. Television video builds entire frames or pictures every second; the speed with which each frame is replaced by the next one makes the images appear to blend smoothly into movement. To make an object travel across the screen while it changes its shape, just change the shape and also move or translate it a few pixels for each frame.



## 10.3 Animation by Computer

Computer animation programs typically employ the same logic and procedural concepts as cell animation, using layer, keyframe, and tweening techniques, and even borrowing from the vocabulary of classic animators. On the computer, paint is most often filled or drawn with tools using features such as gradients and anti aliasing. The word links, in computer animation terminology, usually means special methods for computing RGB pixel values, providing edge detection, and layering so that images can blend or otherwise mix their colors to produce special transparencies, inversions, and effects.

Computer Animation is same as that of the logic and procedural concepts as cel animation and use the vocabulary of classic cel animation – terms such as layer, Keyframe, and tweening. The primary difference between the animation software program is in how much must be drawn by the animator and how much is automatically generated by the software

In 2D animation the animator creates an object and describes a path for the object to follow. The software takes over, actually creating the animation on the fly as the program is being viewed by your user. A blinking word, a color-cycling logo (where the colors of an image are rapidly altered according to a formula), a cel animation (described more fully later on in this chapter),or a button or tab that changes state on mouse rollover to let a user know it is active are all examples of 2-D animations.

Path animation in 2-D space increases the complexity of an animation and provides motion, changing the location of an image along a predetermined path (position) during a specified amount of time (speed).

In 2 1⁄2-D animation, an illusion of depth (the z axis) is added to an image through shadowing and highlighting, but the image itself still rests on the flat x and y axes in two dimensions. Embossing, shadowing, beveling, and highlighting provide a sense of depth by raising an image or cutting it into a background.

In 3-D animation, software creates a virtual realm in three dimensions, and changes (motion) are calculated along all three axes (x, y, and z), allowing an image or object that itself is created with a front, back, sides, top, and bottom to move toward or away from the viewer, or, in this virtual space of light sources and points of view, allowing the viewer to wander around and get a look at all the object's parts from all angles.

On the computer, paint is most often filled or drawn with tools using features such as gradients and anti-aliasing. The word inks, in computer animation terminology, usually means special methods for computing color values, providing edge detection, and layering so that images can blend otherwise mix their colors to produce special transparencies, inversions and effects.

# Animation techniques

When you create an animation, organize its execution into a series of logical steps. First, gather up in your mind all the activities you wish to provide in the animation; if it is complicated, you may wish to create a written script with a list of activities and required objects. Choose the animation tool best suited for the job. Then build and tweak your sequences; experiment with lighting effects. Allow plenty of time for this phase when you are experimenting and testing. Finally, post-process your animation, doing any special rendering and adding sound effects.

# Cel Animation

The term cel derives from the clear celluloid sheets that were used for drawing each frame, which have been replaced today by acetate or plastic. Cels of famous animated cartoons have become sought-after, suitable-for-framing collector’s items.

Cel animation artwork begins with keyframes (the first and last frame of an action). For example, when an animated figure of a man walks across the screen, he balances the weight of his entire body on one foot and then the other in a series of falls and recoveries, with the opposite foot and leg catching up to support the body. The animation techniques made famous by Disney use a series of progressively different on each frame of movie film which plays at 24 frames per second. A minute of animation may thus require as many as 1,440 separate frames. The term cel derives from the clear celluloid sheets that were used for drawing each frame, which is been replaced today by acetate or plastic. Cel animation artwork begins with keyframes.

# Kinametics

It is the study of the movement and motion of structures that have joints, such as a walking man. Inverse Kinematics is in high-end 3D programs, it is the process by which you link objects such as hands to arms and define their relationships and limits. Once those relationships are set you can drag these parts around and let the computer calculate the result.

# Morphing

Morphing is popular effect in which one image transforms into another. Morphing application and other modeling tools that offer this effect can perform transition not only between still images but often between moving images as well.

The morphed images were built at a rate of 8 frames per second, with each transition taking a total of 4 seconds. Some product that uses the morphing features are as follows

* Black Belt’s Easy Morph and WinImages,
* Human Software’s Squizz
* Valis Group’s Flo , MetaFlo, and MovieFlo.

**10.4 Assignment-10**

# Short Answer Questions (Each carries 2 Marks)

1. What is power of motion?
2. Define Animation.
3. Differentiate between persistence of vision and phi?
4. What are the different types of animation by computer?
5. Mention different animation techniques.
6. What is Kinametics?
7. What is Morphing?
8. Define cel animation.
9. What is path animation?
10. Differentiate between 2-D and 3-D animation.

**Long Answer Questions**

1. Write a short note on power of motion (3 Marks).
2. Explain principle of animation (6 Marks).
3. Explain Animation by computer (6 Marks).
4. Explain about animation techniques (7 Marks).
5. Write a short note on cel animation (4 Marks).
6. Write a short note on Kinametics and morphing (5 Marks).

# UNIT-IV Chapter 11 Video

## 11.1 Using Video

Video standards and formats are still being refined as transport, storage, compression, and display technologies take shape in laboratories and in the marketplace and while equipment and post-processing evolves from its analog beginnings to become fully digital, from capture to display.

Compression (and decompression), using special software called a codec, allows a massive amount of imagery to be squeezed into a comparatively small data file, which can still deliver a good viewing experience on the intended viewing platform during playback.

If you control the delivery platform for your multimedia project, you can specify special hardware and software enhancements that will allow you to work with high-definition, fullmotion video, and sophisticated audio for high-quality surround sound. Or you can design a project to meet a specific compression standard, such as MPEG2 for Digital Versatile Disc (DVD) playback or MPEG4 for home video. You can install a superfast RAID (Redundant Array of Independent Disks) system that will support high-speed data transfer rates.

## 11.2 How Video works and is displayed

When light reflected from an object passes through a video camera lens, that light is converted into an electronic signal by a special sensor called a charge-coupled device (CCD). Top-quality broadcast cameras and even camcorders may have as many as three CCDs (one for each color of red, green, and blue) to enhance the resolution of the camera and the quality of the image.

It's important to understand the difference between analog and digital video. Analog video has a resolution measured in the number of horizontal scan lines (due to the nature of early cathodetube cameras), but each of those lines represents continuous measurements of the color and brightness along the horizontal axis, in a linear signal that is analogous to an audio signal. Digital video signals consist of discrete color brightness (RGB) value for each pixel. Digitizing analog video involves reading the analog signal and breaking it into separate data packets. This process is similar to digitizing audio, except that with video the vertical resolution is limited to the number of horizontal scan lines.

# Analog Video

In an analog system, the output of the CCD is processed by the camera into three channels of color information and synchronization pulses (sync) and the signals are recorded onto magnetic tape. There are several video standards for managing analog CCD output, each dealing with the amount of separation between the components-the more separation of the color information, the higher the quality of the image (and the more expensive the equipment). If each channel of color information is transmitted as a separate signal on its own conductor, the signal output is called component. Lower in quality is the signal that makes up Separate Video (S-Video), using two channels that carry luminance and chrominance information. The least separation (and thus the lowest quality for a video signal) is composite, when all the signals are mixed together and carried on a single cable as a composite of the three color channels and the sync signal.

# Digital Video

In digital systems, the output of the CCD is digitized by the camera into a sequence of single frames, and the video and audio data are compress before being written to a tape or digitally stored to disc or flash memory in one of several proprietary and competing formats. Digital video data formats, especially the codec used for compressing and decompressing video (and audio) data, are important.

# Display

Colored phosphors on a cathode ray tube (CRT) screen glow red, green, or blue when they are energized by an electron beam. Because the intensity of the beam varies as it moves across the screen, some colors glow brighter than others. Finely tuned magnets around the picture tube aim the electrons precisely onto the phosphor screen, while the intensity of the beam is varied according to the video signal.

If a computer displays a still image or words onto a CRT for a long time without changing, the phosphors will permanently change, and the image or words can become visible, even when the CRT is powered down. Screen savers were invented to prevent this from happening. Flat screen displays are all-digital, using either liquid crystal display (LCD) or plasma technologies, and have supplanted CRTs for computer use. Full integration of digital video in cameras and on computers eliminates the analog television form of video, from both the multimedia production and the delivery platform. Computer monitors use a different progressive-scan technology, and draw the lines of an entire frame in a single pass, without interlacing them and without flicker.

## 11.3 Digital Video Container

A digital video architecture is made up of an algorithm for compressing and encoding video and audio, a container in which to put the compressed data, and a player that can recognize and play back those files. Common containers for video are Flash Video (.flv), MPEG (.mp4), QuickTime (.mov), Windows Media Format (.wmv), WebM (.webm), and RealMedia (.rm). Containers may include data compressed by a choice of codecs, and media players may recognize and play back more than one video file container format.

Container formats may also include metadata-important information about the tracks contained in them-and even additional media besides audio and video. The QuickTime container, for example, allows inclusion of text tracks, chapter markers, transitions, and even interactive sprites.

**Codecs:** To digitize and store a 10-second clip of full-motion video in your computer requires the transfer of an enormous amount of data in a very short amount of time. Reproducing just one frame of digital video component video at 24 bits requires almost 1MB of computer data. Fullsize, full-motion uncompressed video requires that the computer deliver data at about 30MB per second.

**MPEG:** The MPEG standard has been developed by the Moving Picture Experts Group, a working group convened by the International Standards Organization (ISO) and the International Electro-technical Commission (IEC) to create standards for digital representation of moving pictures and associated audio and other data. MPEG1 and MPEG2 are the current standards. Using MPEG1, you can deliver 1.2 Mbps of video and 250 Kbps of two-channel stereo audio using CD-ROM technology. MPEG2, a completely different system from MPEG1, requires higher data rates (3 to 15 Mbps) but delivers higher image resolution, picture quality, interlaced video formats, multi resolution scalability, and multichannel audio features.

**Video Format Converters:** Be prepared to produce more than one version of your video (codecs in a container) to ensure that the video will play on all the devices and in all the browsers necessary for your project's distribution. We should able to convert our material into multiple formats. There are many free, shareware, and inexpensive file format converters available for multiple platforms.

## 11.4 Obtaining video clips

For projects that are focused on training, particularly training people to use software applications, video screen capture of mouse and key activity is widely used along with a voiceover sound track. Video screen capture tools for both PC and Macintosh systems will generate video that can then be edited and integrated with audio. One eLearning and courseware authoring program, Adobe's Captivate will not only capture your own screen activity but allow you to import video in a wide variety of formats (AVI, MOV, FLV, MPEG) and edit it into your final project.

## 11.5 Shooting and Editing Video

To add full-screen, full-motion video to your multimedia project, you will need to invest in specialized hardware and software or purchase the services of a professional video production studio. In many cases, a professional studio will also provide editing tools and post-production capabilities that you cannot duplicate with your Macintosh or PC.

**Video Tips:** A useful tool easily implemented in most digital video editing applications is “blue screen,” “Ultimate,” or “chromo key” editing. Blue screen is a popular technique for making multimedia titles because expensive sets are not required. Incredible backgrounds can be generated using 3-D modeling and graphic software, and one or more actors, vehicles, or other objects can be neatly layered onto that background. Applications such as VideoShop, Premiere, Final Cut Pro, and iMovie provide this capability.

# Recording Formats S-VHS video

In S-VHS video, color and luminance information are kept on two separate tracks. The result is a definite improvement in picture quality. This standard is also used in Hi-8. still, if your ultimate goal is to have your project accepted by broadcast stations, this would not be the best choice.

# Component (YUV)

In the early 1980s, Sony began to experiment with a new portable professional video format based on Betamax. Panasonic has developed their own standard based on a similar technology, called “MII,” Betacam SP has become the industry standard for professional video field recording. This format may soon be eclipsed by a new digital version called “Digital Betacam.” **Digital Video:** Full integration of motion video on computers eliminates the analog television form of video from the multimedia delivery platform. If a video clip is stored as data on a hard disk, CD-ROM, or other mass-storage device, that clip can be played back on the computer’s monitor without overlay boards, videodisk players, or second monitors. This playback of digital video is accomplished using software architecture such as QuickTime or AVI, a multimedia producer or developer; you may need to convert video source material from its still common analog form (videotape) to a digital form manageable by the end user’s computer system. So an understanding of analog video and some special hardware must remain in your multimedia toolbox.

Analog to digital conversion of video can be accomplished using the video overlay hardware described above, or it can be delivered direct to disk using FireWire cables. To repetitively digitize a full-screen color video image every 1/30 second and store it to disk or RAM severely taxes both Macintosh and PC processing capabilities–special hardware, compression firmware, and massive amounts of digital storage space are required.

**11.6 Assignment-11**

# Short Answer Questions (Each carries 2 Marks)

1. Expand DVD and RAID.
2. What is CCD?
3. Expand CRT and LCD.
4. What is Codecs?
5. List out any four video formats.
6. Expand MPEG.
7. Expand ISO and IEC.
8. Mention few digital video editing applications.
9. Expand ADC and DAC.
10. Differentiate between analog and digital video.

# Long Answer Questions

1. Explain about using video in Multimedia (4 Marks)
2. Briefly explain how video works and displayed (6 Marks) 3. Compare analog video and digital video (5 Marks).
3. How video is displayed? Explain (4 Marks)
4. Explain about digital video container (5 Marks)
5. Write a short note on video format converter and MPEG (5 marks)
6. How to obtain video clips? Explain. (4 Marks)
7. Explain about shooting and editing video. (7 Marks)
8. Write a short note on recording formats. (5 Marks)

# UNIT-IV Chapter 12 Making Multimedia

## 12.1 The stages of Multimedia Project

A Multimedia application is developed in stages as all other softwares are being developed. In multimedia application development a few stages have to complete before other stages being, and some stages may be skipped or combined with other stages. Following are the four basic stages of multimedia project development:

**Planning and Costing:** This stage of multimedia application is the first stage which begins with an idea or need. This idea can be further refined by outlining its messages and objectives. Before starting to develop the multimedia project, it is necessary to plan what writing skills, graphic art, music, video and other multimedia expertise will be required. It is also necessary to estimate the time needed to prepare all elements of multimedia and prepare a budget accordingly. After preparing a budget, a prototype or proof of concept can be developed.

**Designing and Producing:** The next stage is to execute each of the planned tasks and create a finished product. Perform each of the planned tasks to create a finished product. During this stage, there may be many feed back cycles with a client until the client is happy.

**Testing:** Test the programs to make sure that they meet the objectives of your project, work properly on the intended delivery platforms, and meet the needs of client or end user.

**Delivering:** The final stage of the multimedia application development is to pack the project and deliver the completed project to the end user. This stage has several steps such as implementation, maintenance, shipping and marketing the product.

## 12.2 The needs for Multimedia project

We need hardware, software, and good ideas to make multimedia. To make good multimedia, we need talent and skill. We also need the help of other people. Multimedia development of any scale greater than the most basic level is inherently a team effort: artwork is performed by graphic artists, video shoots by video producers, sound editing by audio producers, and programming by programmers. Some of the needs for multimedia projects are listed below.

**Creativity:** Before beginning a multimedia project, we must first develop a sense of its scope and content. The most precious asset that can bring to the multimedia workshop is creativity. We have to reverse-engineer the product, and then produce knockoffs using similar approaches and techniques. Taking inspiration from earlier experiments, developers modify and add their own creative touches for designing their own unique multimedia projects.

**Organization:** It's essential to develop an organized outline and a plan that rationally details the skills, time, budget, tools, and resources needed for a project. These should be in place before start to render graphics, sounds, and other components, and a protocol should be established for naming the files so that organizes them for quick retrieval when you need them.

**Communication:** Many multimedia applications are developed in workgroups comprising instructional designers, writers, graphic artists, programmers, and musicians located in the same office space or building. The workgroup members computers are typically connected on a local area network (LAN).The client's computers, however, may be thousands of miles distant, requiring other methods for good communication. Communication among workgroup members and with the client is essential to the efficient and accurate completion of your project. In the workplace, use quality equipment and software for your communications setup.

# Multimedia Hardware

Apple Macintosh operating system (OS) and the Microsoft Windows OS are the most commonly used platforms for the development and delivery of today's multimedia. The basic principles for creating and editing multimedia elements are the same for all platforms. From Macintosh to Windows format, and vice versa, using known file formats or even binary compatible files that require no conversion at all. Selection of the proper platform for developing your multimedia project may be based on personal preference of computer, budget constraints, project delivery requirements, and the type of material and content in the project.

**Windows v/s Macintosh:** Microsoft, primarily a software company, Apple is a hardware manufacturing company that developed its own proprietary software to run the hardware. Apple adopted Intel's processor architecture, an engineering decision that allows Macintoshes to run natively with any x86 operating systems, same as Windows. Macs can also run the Windows operating system.

**Networking windows and Macintosh Computers:** While working in a multimedia development environment consisting of a mixture of Macintosh and Windows computers, we want them to communicate with each other. LANs allow direct communication and sharing of peripheral resources such as file servers, printers, scanners, and network routers. Ethernet is only a method for wiring up computers, so you still will need client/server software to enable the computers to speak with each other and pass files back and forth.

**Connecting Devices:** Among the many hardwares there are enough wires which connect these devices. The data transfer speed the connecting devices provide will determine the faster delivery of the multimedia content. The most popularly used connecting devices are SCSI, USB, MCI and IDE. SCSI (Small Computer System Interface) is a set of standards for physically connecting and transferring data between computers and peripheral devices. Universal Serial Bus (USB) is a serial bus standard to interface devices. USB can connect all types of computer. The Media Control Interface, MCI in short, is an aging API for controlling multimedia peripherals connected to a Microsoft Windows or OS/2 computer. Usually storage devices connect to the computer through an Integrated Drive Electronics (IDE) interface. Essentially, an IDE interface is a standard way for a storage device to connect to a computer.

**Memory and Storage Devices:** Toestimate the memory requirements of a multimedia project- the space required on a floppy disk, hard disk, or CD-ROM, not the random access sense of the project’s content and scope. Color images, Sound bites, video clips, and the programming code that glues it all together require memory. For making multimedia, you will also need to allocate memory for storing and archiving working files used during production, original audio and video clips, edited pieces, and final mixed pieces, production paperwork and correspondence, and at least one backup of your project files, with a second backup stored at another location.

**Random Access Memory (RAM):** In spite of all the marketing hype about processor speed, this speed is ineffective if not accompanied by sufficient RAM. A fast processor without enough RAM may waste processor cycles while it swaps needed portions of program code into and out of memory. In some cases, increasing available RAM may show more performance improvement on your system than upgrading the processor chip.

**Read-Only Memory (ROM):** read-only memory (ROM) is not volatile. ROM is typically used in computers to hold the small BIOS program that initially boots up the computer, and it is used in printers to hold built-in font.

**Hard Disks:** Adequate storage space production environment can be provided by large-capacity hard disks, server-mounted on a network. As multimedia has reached consumer desktops, makers of hard disks have built smaller-profile, larger-capacity, faster, and less-expensive hard disks.

**Flash Memory or Thumb Drives:** The flash memory data storage devices can be integrated with USB or FireWire interfaces to store from eight megabytes to several GB of data. These storage devices are used in digital cameras, cell phones, and audio recording devices.

**CD-ROM Discs:** Compact disc read-only memory (CD-ROM) players have become an integral part of the multimedia development workstation and are an important delivery vehicle for massproduced projects. CD- ROM players have typically been very slow to access and transmit data. But developments have led to double-, triple-, quadruple-speed, 24x, 48x, and 56x drives designed specifically for computer.

**Digital Versatile Discs (DVD):** Digital Versatile Disc (DVD) capable of not only of gigabyte storage capacity but also full-motion video (MPEG2) and high-quality audio in surround sound, this is an excellent medium for delivery of multimedia projects.

**Blu-ray Discs:** Blu-ray is promoted not only for high definition television recording and high definition video distribution, but also for high definition camcorder archiving, mass data storage, and digital asset management and professional storage when used as a recording medium in BD- R format.

## 12.3 Input and output Devices needed

**Input Devices:** A great variety of input devices-from the familiar keyboard and handy mouse to touch screens and voice recognition setups-can be used for the development and delivery of a multimedia project. Scanners enable you to use optical character recognition (OCR) software, such as Omni Page from ScanSoft, a division of Nuance or Recore from Maxsoft-Ocron. With OCR software and a scanner, you can convert paper documents into a word processing document on your computer without retyping or rekeying. Barcode readers are probably the most familiar optical character recognition devices in use today. Using photo cells and laser beams, barcode reader recognize the numeric characters of the Universal Product Code (UPC) that are printed in a pattern of parallel black bars on merchandise label. With OCR, or barcoding, retailers can efficiently process goods in and out of their stores and maintain better inventory control.

An OCR terminal can be of use to a multimedia developer because it recognizes not only printed characters but also handwriting. For hands-free interaction with your project, try voice recognition systems. These behavioral biometric systems usually provide a unidirectional cardioid, noise-canceling microphone that automatically filters out background noise and learns to recognize voiceprints.

Dragon's Naturally Speaking takes dictation, translates text to speech, and does command-to- dick, a serious aid for people unable to use their hands. The quality of your audio recordings is greatly affected by the caliber of your microphone and cables. A unidirectional microphone helps filter external noise, and good cables help reduce noise emitted from surrounding electronic equipment.

Digital cameras use the same CCD technology as video cameras. They capture still images of a given number of pixels (resolution), and the images are stored in the camera's memory to be uploaded later to a computer. The resolution of a digital camera is determined by the number of pixels on the CCD chip, and the higher the megapixel rating, the higher the resolution of the camera. Digital cameras are small enough to fit in a cell phone, and in a more complicated manner they can be used in a television studio or spy camera on an orbiting spacecraft.

**Output Devices:** Presentation of the audio and visual components of your multimedia project requires hardware that may or may not be included with the computer itself, such as speakers, amplifiers, projectors, and motion video devices. Often the speakers you use during a project's development will not be adequate for its presentation.

The monitor you need for development of multimedia projects depends on the type of multimedia application you are creating, as well as what computer you're using. A wide variety of monitors is available for both Macintoshes and PCs. Serious multimedia developers will often attach more than one monitor to their computers because they can work with several open windows at a time.

When you need to show your material to more viewers than can huddle around a computer monitor, you will need to project it onto a large screen or even a white-painted wall. Cathode-ray tube (CRT) projectors, liquid crystal display (LCD) panels, Digital Light Processing (DLP) projectors, and liquid crystal on silicon (LCOS) projectors, Grating-Light-Valve (GLV) technologies, are available.

CRT projectors have been around for quite a while, they are the original “big-screen” televisions and use three separate projection tubes and lenses (red, green, and blue). The three color channels of light must “converge” accurately on the screen.

Hard-copy printed output has also entered the multimedia scene. From storyboards to presentations to production of collateral marketing material, printouts are an important part of the multimedia development environment.

## 12.4 Software need

The basic tool set for building multimedia projects contains one or more authoring systems and various editing applications for text, images, sounds, and motion video. A few additional applications are also useful for capturing images from the screen, translating file formats, and moving files among computers when it is required for a team. The software in multimedia toolkit and skill at using it determines what kind of multimedia work user can do and how fancy can render it. Making good multimedia means picking a successful route through the software swamp. The tools used for creating and editing multimedia elements on both Windows and Macintosh platforms do image processing and editing, drawing and illustration, 3-D and CAD, OCR and text editing, sound recording and editing, video and moviemaking, and various utilitarian housekeeping tasks.

**Text Editing and Word Processing Tools:** A word processor is usually the first software tool computer users rely upon for creating text. The word processor is often bundled with an office suite. Word processors such as Microsoft Word and WordPerfect are powerful applications that include spellcheckers, table formatters, thesauruses and prebuilt templates for letters, resumes, purchase orders and other common documents.

**OCR Software:** Often there will be multimedia content and other text to incorporate into a multimedia project, but no electronic text file. With optical character recognition (OCR) software, a flat-bed scanner, and a computer, it is possible to save many hours of rekeying printed words, and get the job done faster and more accurately than a roomful of typists.

OCR software turns bitmapped characters into electronically recognizable ASCII text. A scanner is typically used to create the bitmap. Then the software breaks the bitmap into chunks according to whether it contains text or graphics, by examining the texture and density of areas of the bitmap and by detecting edges. The text areas of the image are then converted to ASCII character using probability and expert system algorithms.

**Painting and Drawing Tools:** Painting and drawing tools, as well as 3-D modelers, are perhaps the most important items in the toolkit because, of all the multimedia elements, the graphical impact of the project will likely have the greatest influence on the end user. If the artwork is amateurish, or flat and uninteresting, both the creator and the users will be disappointed.

Painting software, such as Photoshop, Fireworks, and Painter, is dedicated to producing crafted bitmap images. Drawing software, such as CorelDraw, FreeHand, Illustrator, Designer, and Canvas, is dedicated to producing vector-based line art easily printed to paper at high resolution.

Some software applications combine drawing and painting capabilities, but many authoring systems can import only bitmapped images. Typically, bitmapped images provide the greatest choice and power to the artist for rendering fine detail and effects, and today bitmaps are used in multimedia more often than drawn objects. Some vector based packages such as Macromedia’s Flash are aimed at reducing file download times on the Web, and may contain both bitmaps and drawn art. The anti-aliased character shown in the bitmap of Color Plate 5 is an example of the fine touches that improve the look of an image.

Look for these features in a drawing or painting packages:

* An intuitive graphical user interface with pull-down menus, status bars, palette control, and dialog boxes for quick, logical selection
* Scalable dimensions, so you can resize, stretch, and distort both large and small bitmaps
* Paint tools to create geometric shapes, from squares to circles and from curves to complex polygons
* Ability to pour a color, pattern, or gradient into any area
* Ability to paint with patterns and clip art
* Customizable pen and brush shapes and sizes
* Eyedropper tool that samples colors
* Auto trace tool that turns bitmap shapes into vector-based outlines
* Support for scalable text fonts and drop shadows
* Multiple undo capabilities, to let you try again
* Painting features such as smoothing coarse-edged objects into the background with anti-aliasing, airbrushing in variable sizes, shapes, densities, and patterns; washing colors in gradients; blending; and masking
* Support for third-party special effect plug-ins
* Object and layering capabilities that allow you to treat separate elements independently Zooming, for magnified pixel editing
* All common color depths: 1-, 4-, 8-, and 16-, 134-, or 313- bit color, and grayscale
* Good color management and dithering capability among color depths using various color models such as RGB, HSB, and CMYK
* Good palette management when in 8-bit mode
* Good file importing and exporting capability for image formats such as PIC, GIF, TGA, TIF, WMF, JPG, PCX, EPS, PTN, and BMP

**3-D Modeling and Animation Tool:** With 3-D modeling software, objects rendered in perspective appear more realistic; you can create stunning scenes and wander through them, choosing just the right lighting and perspective for your final rendered image. Powerful modeling packages such as VectorWorks, AutoDesk’s Maya, Strata 3D, and Avid’s SoftImage are also bundled with assortments of prerendered 3-D clip art objects such as people, furniture, buildings, cars, airplanes, trees, and plants. Blender is a powerful (and free) cross-platform 3-D modeling program offering an extensive feature set.

A good 3-D modeling tool should include the following features:

* Multiple windows that allow you to view your model in each dimension, from the camera’s perspective, and in a rendered preview
* The ability to drag and drop primitive shapes into a scene
* The ability to create and sculpt organic objects from scratch
* Lathe and extrude features
* Color and texture mapping
* The ability to add realistic effects such as transparency, shadowing, and fog
* The ability to add spot, local, and global lights, to place them anywhere, and manipulate them for special lighting effects
* Unlimited cameras with focal length control
* The ability to draw spline-based paths for animation

**Image Editing Tools:** Image-editing application is specialized and powerful tools for enhancing and retouching existing bitmapped images. These applications also provide many of the feature and tools of painting and drawing programs and can be used to create images from scratch as well as images digitized from scanners, video frame-grabbers, digital cameras, clip art files, or original artwork files created with a painting or drawing package.

There are some features typical of image-editing applications and of interest to multimedia developers:

* Multiple windows that provide views of more than one image at a time
* Conversion of major image-data types and industry-standard file formats
* Direct inputs of images from scanner and video sources
* Employment of a virtual memory scheme that uses hard disk space as RAM for images that require large amounts of memory
* Capable selection tools, such as rectangles, lassos, and magic wands, to select portions of a bitmap
* Image and balance controls for brightness, contrast, and color balance
* Good masking features
* Multiple undo and restore features
* Anti-aliasing capability, and sharpening and smoothing controls
* Color-mapping controls for precise adjustment of color balance
* Tools for retouching, blurring, sharpening, lightening, darkening, smudging, and tinting
* Geometric transformation such as flip, skew, rotate, and distort and perspective changes
* Ability to resample and resize an image

**Sound Editing Tools:** Sound editing tools for both digitized and MIDI sound lets hear music as well as create it. By drawing a representation of a sound in fine increments, whether a score or a waveform, it is possible to cut, copy, paste and otherwise edit segments of it with great precision. System sounds are shipped both Macintosh and Windows systems and they are available as soon the Operating system is installed. For MIDI sound, a MIDI synthesizer is required to play and record sounds from musical instruments. For ordinary sound there are varieties of software such as Soundedit, MP3cutter, Wavestudio.

**Animation, Video and Digital Movie Tools:** Animation and digital movies are sequences of bitmapped graphic scenes (frames, rapidly played back). Most authoring tools adapt either a frame or object oriented approach to animation. Moviemaking tools typically take advantage of Quicktime for Macintosh and Microsoft Video for Windows and lets the content developer to create, edit and present digitized motion video segments.

# What you need: Authoring System

Multimedia authoring tools provide the important framework you need for organizing and editing the elements of multimedia project, including graphics, sounds, animations, and video clips. Authoring tools are used for designing interactivity and the user interface, for presenting the project on screen, and for assembling diverse multimedia elements into a single, cohesive product.

Authoring software provides an integrated environment for binding together the content and functions of your project, and typically includes everything you need to create, edit, and import specific types of data; assemble raw data into a playback sequence or cue sheet; and provide a structured method or language for responding to user input. With multimedia authoring software, we can make Video productions, Animations, Games, Interactive web sites, Demo disks and guided tours, Presentations, Kiosk applications, Interactive training, simulations, prototypes, and technical visualization.

# Types of Authoring Tools

The various multimedia authoring tools can be categorized into three groups, based on the method used for sequencing or organizing multimedia elements and events:

* Card- or page-based tools
* Icon-based, event-driven multimedia and game-authoring tools
* Time-based tool

# Card- and Page-Based Authoring Tools

Card-based or page-based tools are authoring systems, wherein the elements are organized as pages of a book or a stack of cards. Thousands of pages or cards may be available in the book or stack. These tools are best used when the bulk of your content consists of elements that can be viewed individually, letting the authoring system link these pages or cards into organized sequences. Page-based authoring systems such as LiveCode from Runtime Revolution (www.runrev.com) and ToolBook (www.toolbook.org) contain media objects: buttons, text fields, graphic objects, backgrounds, pages or cards, and even the project itself. Each object may contain a programming script; usually a property of that object, activated when an event (such as a mouse click) related to that object occurs. Card- and page-based systems typically provide two separate layers on each card: a background layerthat can be shared among many cards, and a foreground layer that is specific to a single card.

# Icon- and Object-Based Authoring Tools

Icon- and Object-BasedAuthoringtools are authoring systems, where in multimedia elements and interaction cues (events) are organized as objects in a structural framework or process. Icon- or object-based, event-driven tools simplify the organization of your project and typically display flow diagrams of activities along branching paths. In complicated navigational structures, this charting is particularly useful during development. Icon-based, event-driven tools provide a visual programming approach to organizing and presenting multimedia. First you build a structure or flowchart of events, tasks, and decisions, by dragging appropriate icons from a library. These icons can include menu choices, graphic images, sounds, and computations. The flowchart graphically depicts the project’s logic. When the structure is built, you can add your content: text, graphics, animation, sounds, and video movies. With icon-based authoring tools, non-technical multimedia authors can build sophisticated applications without scripting.

# Time-Based Authoring Tools

Time-based toolsare authoring systems, wherein elements and events are organized along a timeline, with resolutions as high as or higher than 1/30 second. Time-based tools are best to use when you have a message with a beginning and an end. Sequentially organized graphic frames are played back at a speed that you can set. Other elements (such as audio events) are triggered at a given time or location in the sequence of events. Each tool uses its own distinctive approach and user interface for managing events over time.

**Flash:** Flash is a time-based development environment. Flash, however, is also particularly focused on delivery of rich multimedia content to the Web.

**Director:** Adobe’s Director is a powerful and complex multimedia authoring tool with a broad set of features to create multimedia presentations, animations, and interactive multimedia applications.

# Object

In multimedia authoring systems, multimedia elements and events are often treated as objects that live in a hierarchical order of parent and child relationships. Messages passed among these objects order them to dothings according to the properties or modifiers assigned to them. Objects typically take care of themselves. Send them a message and they do their thing without external procedures and programming. Objects are particularly useful for games, which contain many components with many “personalities,” all for simulating real-life situations, events, and their constituent properties. Object-based authoring programs typically provide objects pre programmed with sensible properties, messages, and functions. A video object, for example, will likely have a duration property and a source property.

# Choosing an Authoring Tool

Authoring tools are constantly being improved by their makers, who add new features and increase performance with upgrade development cycles of six months to a year.

# Editing Features

The elements of multimedia-images, animations, text, digital audio and MIDI music, and video clips need to be created, edited, and converted to standard file formats, using the specialized applications. In many cases, however, the editors that may come with an authoring system will offer only a subset of the substantial features found in dedicated tools.

# Organizing Features

The organization, design, and production process for multimedia involves storyboarding and flowcharting. Some authoring tools provide a visual flowcharting system or overview facility for illustrating the project’s. Storyboards or navigation diagrams can also help organize a project and can help focus the overall project scope for all involved. Many web-authoring programs such as Dreamweaver include tools that create helpful diagrams and links among the pages of a web site

# Programming Features

Multimedia authoring systems offer one or more approaches, as explained in the following paragraphs:

* Visual programming with cues, icons, and objects
* Programming with a scripting language
* Programming with traditional languages, such as Basic or C
* Document development tools

Visual programmingwith icons or objects is perhaps the simplest and easiest authoring process. If you want to play a sound or put a picture into your project, just drag the element’s icon into the playlist—or drag it away to delete it.

Authoring tools, which offer a very high level language (VHLL) or interpreted scripting environment for navigation control and for enabling user inputs or goal-oriented programming languages. The more commands and functions provided in the scripting language, the more powerful the authoring system. As with traditional programming tools, look for an authoring package with good debugging facilities, robust text editing, and online syntax reference.

# Interactivity Features

Interactivity empowers the end users of your project by letting them control the content and flow of information. Authoring tools should provide one or more levels of interactivity:

* Simple branching, which offers the ability to go to another section of the multimedia production.
* Conditional branching, which supports a go-to based on the results of IF-THEN decisions or events.
* A structured language that supports complex programming logic, such as nested IF-

THENs, subroutines, event tracking, and message passing among objects and elements

# Performance Tuning Features

Complex multimedia projects require exact synchronization of events for example, the animation of an exploding balloon with its accompanying sound effect. Accomplishing synchronization is difficult because performance varies widely among the different computers used for multimedia development and delivery. Some authoring tools allow you to lock a production’s playback speed to a specified computer platform, but others provide no ability whatsoever to control performance on various systems.

# Playback Features

Authoring system should let you build a segment or part of your project and then quickly test it as if the user were actually using it. You should spend a great deal of time going back and forth between building and testing as you refine and smooth the content and timing of the project.

# Delivery Features

Delivering your project may require building a run-time version of the project using the multimedia authoring software. A run-time version or standalone allows your project to play back without requiring the full authoring software and all its tools and editors. Often, the runtime version does not allow users to access or change the content, structure, and programming of the project.

# Cross-Platform Features

It is also increasingly important to use tools that make transfer across platforms easy. If you develop on a Macintosh, look for tools that provide a compatible authoring system for Windows or offer a run-time player for the other platform**.**

# Internet Playability

Because the Web has become a significant delivery medium for multimedia, authoring systems typically provide a means to convert their output. So that it can be delivered within the context of HTML or DHTML. Test your authoring software for Internet delivery before you build your project.

**12.5 Assignment-12**

# Short Answer Questions (each carries 2 Marks)

1. Mention the different stages of multimedia project.
2. What are the different non tangible needs for multimedia project?
3. Expand SCSI and USB.
4. Expand LCOS and GLV.
5. Mention two text and word editing tools.
6. Mention four drawing and painting tools.
7. Mention four 3-D modeling and animation tools.
8. List any two features of image editing tools.
9. List out any two features of 3-D modeling and animation tools.
10. List out any two features of drawing and painting tools.

# Long Answer Questions

1. Explain the stages of multimedia project. (6 Marks)
2. Explain the non tangible need for multimedia project. (5 Marks)
3. Briefly explain the needs for multimedia project. (10 Marks)
4. Explain multimedia hardware as the needs for multimedia project. (6 Marks)
5. Explain memory and storage devices as the needs for multimedia project. (6 Marks)
6. Explain about input devices needed for multimedia project (6 Marks)
7. Explain about output devices needed for multimedia project (6 Marks)
8. Briefly explain software needed authoring system (10 Marks)
9. Write a short note on following (5Marks)
   1. Text and Word editing tools (ii) OCR software
10. Explain painting and drawing tools with its features. (6 Marks)
11. Explain 3-D modeling and animation tools with its features. (5Marks)
12. Explain image editing tools with its features. (6 Marks)
13. Write a short note on following (5Marks)
    1. Sound editing tools (ii) Animation, Video and Digital movie tools

**Chapter 13 Value Added Session**

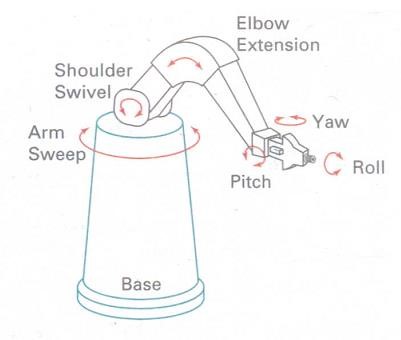
# Session 1: Computer Animated Languages

Design and control of animation sequences are handled with a set of animation Languages. A general-purpose language, such as C, Lisp, Pascal, or FORTRAN, is then used to program the animation functions, but several specialized animation languages have been developed. Animation functions include a graphics editor, a buffer frame generator, an in-between generator, and standard graphics routines. Graphics editor allows us to design and modify object shapes, using spline surfaces, constructive solid-geometry methods, or other representation schemes.

A typical task in an animation specification is scene description. This includes the positioning of objects and light sources, defining the photometric parameters light source intensities and surface-illumination properties), and setting the different parameters like position, orientation, and lens characteristics. Another standard function is action specification. This involves the layout of motion paths for the objects and camera. And we need the usual graphics routines: viewing and respective transformations, geometric transformations to generate object movements as a function of accelerations or kinematic path specifications, visible-surface identification, and the surface-rendering operations.

Key-frame systems are specialized animation languages designed simply generate the inbetweens from the user-specified key frames. Usually, each object in the scene is defined as a set of rigid bodies connected at the joints and with limited number of degrees of freedom. As an example, the single-arm robot has six degrees of freedom, which are called arm sweep, shoulder elbow extension, pitch, yaw, and roll. We can extend the number of degree of freedom for this robot arm to nine by allowing three-dimensional transformation for the base (Fig.1).

Parameterized systems allow object-motion characteristics to be specified as a part of the object definitions. The adjustable parameters control such object characteristics as degrees of freedom, motion limitations, and allowable shape languages.



# Figure 1: Degrees of freedom for a stationary, single-arm robot

Scripting systems allow object specifications and animation sequences to be defined with a userinput script. From the script, a library of various objects and motions can be constructed.

# Session 2: Vector Drawing

Most multimedia authoring systems provide for use of vector-drawn objects such as lines, rectangles, ovals, polygons, and text. Computer-aided design (CAD) programs have traditionally used vector-drawn object systems for creating the highly complex and geometric rendering needed by architects and engineers.

Graphic artists designing for print media use vector-drawn objects because the same mathematics that put a rectangle on your screen can also place that rectangle on paper without jaggies. This requires the higher resolution of the printer, using a page description language such as PostScript.

Programs for 3-D animation also use vector-drawn graphics. For example, the various changes of position, rotation, and shading of light required to spin the extruded.

**How Vector Drawing Works:** Vector-drawn objects are described and drawn to the computer screen using a fraction of the memory space required to describe and store the same object in bitmap form. A vector is a line that is described by the location of its two endpoints. A simple rectangle, for example, might be defined as follows:

RECT 0, 0, 200, 200

# Session 3: Video Recording Formats

**S-VHS Video:** In S-VHS video color and luminance information are kept on two separate tracks. The result is a definite improvement in picture quality. This standard is also used in Hi-8. Although basically oriented toward consumers, this format is gaining rapid acceptance among lower-end broadcasters. Still, if your ultimate goal is to have your project accepted by broadcast stations, this would not be the best choice.

**Component (YUV):** In the early 1980s, Sony began to experiment with a new portable professional video format based on Betamax. Called Betacam, it required speeding the tape up considerably (a 2-hour tape was used up in 20 minutes) and laying the signal on the tape in three channels, one for red, one for blue, and one for luminance information. The resulting format (called "component") produced images that had none of the problems of traditional composite video such as color shift and bleed and crawling edges on graphics. Betacam SP has become the industry standard for professional video field recording. This format may soon be eclipsed by a new digital version called "Digital Betacam." Betacam SP is without a doubt the choice for the broadcast industry.

**Component Digital:** Around the same time Sony was developing Betacam, they also began research into digital video, where the signal was converted to digital information before it was recorded as bits and bytes. The advantages were many, including better color and image resolution and an ability to make almost unlimited copies without loss of quality. The result of this research was D-l, or "Component Digital." Using a 19 mm (3/4-inch) tape, and recording the signal in a digital version of the component technology developed for Betacam, it quickly proved itself as the format of choice for graphics. D-l is at the pinnacle of NTSC video and is the mastering standard of choice among high-end editing facilities. However, this quality comes with an extremely high price tag. The result is that this format really only fits super-high-end broadcast projects and not your standard multimedia title broadcast-quality equipment but is not built to suffer wear and tear.

# Session 4: Broadcasting video Standards

Four broadcast and video standards and recording formats are commonly in use around the world: NTSC, PAL, SECAM, and HDTV. Because these standards and formats are not easily interchangeable, it is important to know where your multimedia project will be used.

# NTSC

The United States, Japan, and many other countries use a system for broadcasting and displaying video that is based upon the specifications set forth by the 1952 National Television Standards Committee. These standards define a method for encoding information into the electronic signal that ultimately creates a television picture. As specified by the NTSC standard, a single frame of video is made up of 525 horizontal scan lines drawn onto the inside face of a phosphorcoated picture tube every 1/30th of a second by a fast-moving electron beam.

# PAL

The Phase Alternate Line (PAL) system is used in the United Kingdom, Europe, Australia, and South Africa. PAL is an integrated method of adding color to a black-and-white television signal that paints 625 lines at a frame rate 25 frames per second.

# SECAM

The Sequential Color and Memory (SECAM) system is used in France, Russia, and few other countries. Although SECAM is a 625-line, 50 Hz system, it differs greatly from both the NTSC and the PAL color systems in its basic technology and broadcast method.

# HDTV

High Definition Television (HDTV) provides high resolution in a 16:9 aspect ratio (see following Figure). This aspect ratio allows the viewing of Cinemascope and Panavision movies. There is contention between the broadcast and computer industries about whether to use interlacing or progressive-scan technologies.

# Computer Graphics and Multimedia Question Bank Unit I Questions carrying TWO marks

1. Define refresh in case of CRT
2. What are horizontal and vertical retrace?
3. What are two categories of flat panel displays? Give example for each.
4. Write two drawbacks of DVST.
5. Define resolution.
6. Define persistence.
7. Define aspect ratio.
8. Write any two drawbacks of light pens.
9. What are impact and non-impact printers? Give example for each.
10. What do you mean by modeling coordinates?
11. Define world and screen coordinates.
12. Define scan conversion.
13. What are normalized device coordinates?
14. Expand GKS, DVST, PHIGS, CGM, DDA and CGI (1 mark each) 15. What is 8-way symmetry of a circle?
15. Write 8-way symmetry of a circle.
16. What is 4-way symmetry of an ellipse?
17. Write 4-way symmetry of an ellipse.
18. Write boundary fill algorithm.
19. Write flood fill algorithm.
20. What is coherence of a scene?
21. Write two drawbacks of DDA.
22. What is the purpose of frame buffer?

**Long answer questions:**

1. Explain the architecture of Raster display system with neat diagram.
2. Explain the architecture of Vector display system with neat diagram.
3. Differentiate raster and vector display systems.
4. List advantages and drawbacks of raster and vector display systems.
5. List the drawbacks of DVST.
6. Write a note on graphic monitors and workstations.
7. Explain Track ball, Joystick, Spaceball, Data Glove and digitizer. (2 marks each)
8. Write a note on image scanners.
9. What are the drawbacks of light pens?
10. Write a note on voice recognition.
11. Explain different types of hard copy devices.
12. Explain graphics functions.
13. Write a note on graphics standards. 14. Write DDA line drawing algorithm
14. Write Bresenham’s line drawing algorithm.
15. Write mid-point algorithm to draw a circle.
16. Explain n different cases in scan-line polygon fill algorithm with suitable examples.
17. Explain odd-even rule and non-zero winding number rule with suitable examples.
18. Differentiate boundary fill and flood fill techniques.
19. Write a note on boundary fill algorithm
20. Write a note on flood fill algorithm

# 7 or more than 7 marks

1. Derive Bresenham’s line drawing algorithm. 2. Derive Mid-point algorithm to draw a circle.

1. Derive Mid-point algorithm to draw an ellipse.
2. Write mid-point algorithm to draw an ellipse.

**Unit II TWO marks questions:**

1. List the possible selections for Line Type attribute of a straight line
2. List any four character attributes.
3. What is the use of **pixel mask** while creating dashed lines? Give example
4. What is a line cap? List various types.
5. List the various **joins** for thick polyline and write the diagram to represent them
6. What is soft fill?
7. Write the equations for translation of a point at (x, y) to (x’, y’) position. Also write the column vectors
8. Write the equations for scaling of a point at (x, y) to (x’, y’) position. Also write the column vectors
9. Write the equations for rotation of a point at (x, y) to (x’, y’) position about origin.
10. Define translation and scaling.
11. What is a rigid body transformation? Give example.
12. Differentiate **uniform** scaling and **differential** scaling
13. Write the matrices to represent **translation** and **scaling** in homogeneous coordinate system.
14. What is reflection?
15. Write matrices to represent rotation and scaling in homogeneous coordinate system.
16. Write the purpose of homogeneous coordinate system.
17. List the sequence of steps required to perform rotation of an object about a pivot point at

**(xr, yr).**

1. List the sequence of steps required to perform scaling of an object with respect to a fixed position at **(xf, yf).**
2. Discuss associative property with respect to transformation matrices.
3. Discuss commutative property with respect to transformation matrices.
4. Write the matrices for reflection about the x- axis and reflection about coordinate origin.
5. What is affine transformation?
6. What is a shear? Write any one shear matrix.
7. Define window and viewport.
8. Define clipping. List any two types
9. What is point clipping? Write the inequalities to be satisfied for saving a point for the purpose of display.
10. Write a diagram to show **region codes** which may be assigned to the end points of a straight line.

**Long answer questions:**

1. List the different attributes of a straight line and explain any ONE in detail. (5)
2. Explain Line Type attribute of a straight line. (5)
3. Write a note on Line Width attribute of a straight line. (5)
4. Explain curve attributes in detail. (5)
5. Explain how color values are stored in a color lookup table. (5)
6. What are the different basic fill styles for polygon areas? Explain. (5)
7. Write a note on pattern fill.(5)
8. Explain the different **text** attributes**.** (5)
9. Explain the **translation** transformation applied to a 2- dimensional object. (4)
10. Explain the **rotation** transformation applied to a 2- dimensional object with a suitable diagram. (5)
11. List the various basic 2- dimensional transformations and explain any one in detail.

(5)

1. Write an explanatory note on 2- dimensional scaling. (5)
2. What is the need for homogeneous coordinate system? Write the matrices to represent the various basic transformations in homogeneous coordinate system.(4)
3. Write an explanatory note on homogeneous coordinate system. (4)
4. Prove that successive translations and successive rotations are additive. (6)
5. Prove that successive scalings are multiplicative. Also explain general fixed point scaling with a suitable diagram. (6)
6. Explain general pivot-point rotation with a suitable diagram. (4)
7. Discuss various Reflection transformations with suitable diagrams. (6)
8. Write a note on shear transformation. (4)
9. Derive a composite matrix to transform object descriptions from **xy**coordinate system to **x’y’** coordinate system while the **x’y’** coordinate system has origin at (x0, y0) with an orientation angle Ө between x and x’ axes. (4)
10. Explain window to viewport transformation with a suitable diagram. (5)
11. Write and explain Cohen- Sutherland line clipping algorithm. (10)
12. Explain the procedure of Cohen- Sutherland line clipping. (5)
13. Explain the procedure of Sutherland- Hodgeman polygon clipping along with suitable diagrams. (5)

# Unit III Questions carrying TWO marks

1. Define multimedia.
2. What is interactive media?
3. Define hypermedia.
4. What is virtual reality? Give examples.
5. Differentiate between typefaces and fonts
6. Differentiate between leading, kerning and tracking.
7. Two types of typeface widely used and give example.
8. What are CSS?
9. Differentiate symbols and icons.
10. What are buttons? Where and all we can apply the actions.
11. What are linear and nonlinear multimedia projects?
12. Define storyboarding.
13. Define integrated multimedia.
14. What is the purpose of multimedia authoring tools?
15. Define font.
16. List any four font styles.
17. Define leading and kerning.
18. Differentiate serif and sans serif fronts.
19. Differentiate portrait and landscape documents.
20. What is font substitution?
21. What is localization with respect to multimedia?
22. List different values for alignment.
23. What is typeface?
24. List four attributes of a font.
25. What is the purpose of free and shareware fonts?
26. List two ways of generating images using computers.
27. What is E-book? Give some formats and extension.
28. What is HTML? How we can embed styles in it by using CSS?
29. What are tags? Give some examples.
30. What do you mean by anti-aliasing?
31. What is hypertext?
32. What are bitmaps?
33. Define autotracing.
34. What is extruding?
35. Define lathing.
36. Define dithering.
37. What are palettes? List most common color palettes.
38. Write and define 2 computerized colors. 39. Expand DIP, IGS, PDF, TIFF
39. What is morphing?
40. Define audio resolution.
41. Define sampling and quantization
42. Why digital audio is called device independent. 44. Expand MIDI and MPEG 45. What is Codec?
43. List the components of MIDI envelop.
44. Define attack sustain and decay with respect to sound.

# Long answer questions

1. Write application of multimedia in various fields
2. Write a note on virtual reality.
3. Explain different methods of delivering multimedia contents to the users.
4. List and explain different attributes of font.
5. Explain different ways of presenting text in multimedia.
6. What are different methods for searching in hypermedia?
7. Write a note on dithering.
8. Write a note on 3D modeling and rendering.
9. List the capabilities and limitations of bitmap
10. Differentiate bitmap, vector and 3D image formats
11. Write a note on various image types used in multimedia.
12. List and explain two methods of making colors in computers.
13. List and explain different sound editing operations.
14. Write a note on MIDI audio.
15. List the advantages of MIDI over digital audio.
16. List the drawbacks of MIDI
17. Write a note on
    1. Space Considerations (3)
    2. Audio Recording (3)
    3. Testing and Copyright Issues (3)
18. Give the advantages of Digital audio and in what circumstances we can use it? (5)
19. Write steps involved to bring an audio recording into multimedia project. (6)
20. How wave is converted into digital audio? And also explain quantizing and clipping of it with a neat diagrams (5)
21. Write any 3 or 4 or 5 editing digital recordings. (3 or 4 or 5)
22. Write applications of MIDI. (4)
23. Write any 3 or 4 types of searches in Web. (3 or 4)
24. Write a note on bitmaps. (4)
25. Write a note on vector drawn images. (4)
26. Write a note on additive and subtractive colors.(4)
27. Give most widely used typefaces with examples to it (in terms of fonts). (3)
28. Give any 6 design suggestions for text fonts. (3)
29. Write a note on Menus for Navigation. (3)
30. Write a note on Buttons for Interaction. (4)
31. Write a note on HTML Documents. (4)
32. Write a note on ASCII and ANSI(or Extended Character Set) character sets. (4)
33. Write a note on Unicode. (3)
34. Write a note on Unicode and Extended Character Set. (4)
35. How can you include special characters in Web pages give some examples. (3)
36. How pretty text can be created? (3)
37. Write any 4 types of searches in Web. ( 4)
38. Write any 5 editing digital recordings. (5)

# Unit IV Questions carrying 2 marks

1. What is animation?
2. What are principles of animation?
3. What is 2-D animation?
4. What is 2½-D animation?
5. What is 3-D animation?
6. What is cel animation?
7. What is tweening?
8. What is computer animation?
9. What is kinematics?
10. What is inverse kinematics?
11. What is Morphing in animation?
12. What is CCD? Where it is used?
13. What is helical scan recording?
14. What is NTSC?
15. What is PAL?
16. What is SECAM?
17. What is HDTV?
18. Expand MPEG, RAID
19. What is overscan and underscan?
20. What is MPEG? Write down their releases.
21. What are Chroma key technology in digital video editing.
22. What is pagemaking?
23. What do you mean by text formatting?
24. What is cropping?
25. What is generation loss?
26. Define prototype.
27. Expand SCSI, USB, UPC ATA (1 mark each)

Long answer questions

1. What are the logical steps of animation techniques? (5)
2. What is principles of animation?(3)
3. Write a note on cel animation. (5)
4. Write a note on computer animation. (5)
5. What are analog video? How it is recorded? (6)
6. Write a note on Analog Video. (5)
7. Which are the three analog broadcast video standards available? Explain in brief. (7)
8. How Digital Video is recorded? (5)
9. With diagram differentiate between video broadcast standards. (7)
10. Write a note on MPEG. (4)
11. Write a note on Shooting Platforms. (5)
12. Give some suggestions for creating good titles for video. (6)
13. List the essential capabilities of camcorder while setting up the production environment
14. List and explain different stages of a multimedia project.
15. Discuss intangible elements needed to make good multimedia.
16. List and explain different features to be considered while choosing multimedia authoring tools.
17. List and explain different types of authoring tools for multimedia.

**BCACAC 260** **Reg. NO.**

**Credit Based Fourth Semester B.C.A. Degree Examination**

**April/May 2015**

**(New Syllabus-2013-14 Batch)**

**COMPUTER GRAPHICS AND MULTIMEDIA**

**Time: 3 Hours**  **Max. Marks: 80**

# Note: Answer any ten questions from Part - A and anyone full question from each Unit of Part-B PART – A (10 x 2=20) Note: Answer any 10 questions each carries 2 marks

1. a) What are horizontal and vertical retrace?

1. What are impact and non-impact printers? Give example for each.
2. What is the purpose of frame buffer?
3. Define translation and scaling.
4. Write the purpose of homogeneous coordinate system.
5. List any two character attributes.
6. Define storyboarding.
7. Why digital audio is called device independent?
8. What are palettes? List anyone common color palettes.
9. What is 3-D animation?
10. What is morphing in animation?
11. What are chroma key technologies in digital video editing?

# PART-B UNIT-I

1. a) Explainthe architectureof raster display system with neat diagram. (4+4+7)
   1. Write a note on Graphics Software.
   2. Derive Mid-point algorithm to draw an ellipse.

1. a) Explain Track ball, Joystick, Space ball. (6+9)
   1. Derive mid-point algorithmto draw an ellipse.

# UNIT-II

1. a) Prove that successive rotations is additive. (5+3+7)
   1. Explain the different Text attributes.
   2. Write and explain Cohen-Sutherland line clipping algorithm.

1. a) Explain window to viewport transformation with a suitable diagram. (5+4+6)
2. Explain the Pivot Point Rotation with an example.
3. Discuss various Reflection transformations with suitable diagrams.

# UNIT-III

1. a) List and explain the various application of multimedia in various fields. (5+5+5) b) Write a note on bitmaps.
   1. Write a note on MIDI audio.

1. a) Write a note on additive and subtractive colors. (5+5+5)

b) What are different methods for searching in hypermedia?

* 1. List and explain the various sound editing operations.

# UNIT-IV

1. a) List and explain different stages of a multimedia project. (5+5+5)
   1. Write a note on Shooting Platforms.
   2. Write a note on MPEG.

1. a) List and explain different types of authoring tools for multimedia. (5+5+5)
   1. Write a note on video compression (CODEC).
   2. Discuss intangible elements needed to make good multimedia.

|  |  |
| --- | --- |
| 2. a) Explain Refresh CRT monitor.  b) Write a note on Graphics Software. | (4+4+7) |
| c) Derive Mid-point algorithm to draw an ellipse. |  |
| 3. a) List the differences between Raster and Random scan system. | (4+4+7) |

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* **BCACAC 260** **Reg. NO.**

**Credit Based Fourth Semester B.C.A. Degree Examination**

**April/May 2014**

**(New Syllabus-2013-14 Batch)**

**COMPUTER GRAPHICS AND MULTIMEDIA**

**Time: 3 Hours**  **Max. Marks: 80**

# Note: Answer any ten questions from Part - A and anyone full question from each Unit of Part-B PART – A (10 x 2=20) Note: Answer any 10 questions each carries 2 marks

1. a) What are the advantages and disadvantages of DVST ?

1. What are the two types of printers? Give example for each.
2. Write the nested loop C statements for filling a Rectangle.
3. Explain Region in ellipse.
4. What is Shear transformation? Write the matrix for x-direction Shear.
5. Write Rotation and Translation matrix for homogenous coordinates.
6. What is a Line Cap? List various types.
7. Define the following:
8. Window ii) View port

i) Define the following:

1. Multimedia ii) Interactive Multimedia
2. What are the text elements used in Multimedia?
3. What is Sound? Write the characteristics of Sound.
4. List four video broadcast standards.

# PART-B UNIT-I

1. Explain Flood fill algorithm.
2. Write and explain DDA line drawing algorithm.

# UNIT-II

1. a) Explain the different Text attributes. (5+6+4)
   1. What is transformation? Discuss three basic 20 transformations with suitable example.
   2. Explain general Pivot-Point Rotation with example.
2. a) Explain window to view port coordinate transformation with a suitable

diagram. (4+4+7)

1. Prove that two successive scaling are multiplicative.
2. Write and explain Cohen-Southerland line clipping algorithm

# UNIT-III

1. a) List and explain the applications of multimedia in various fields. (4+4+5+2)
   1. What are the points to be considered while choosing text font in multimedia?
   2. List and explain various Sound editing operations.
   3. Write a note on Image File formats used in Multimedia

1. a) List and explain different attributes of font. (5+4+3+3)
   1. Differentiate MIDI and Digital Audio.
   2. Explain Bitmap and Vector image.
   3. Write a note on Additive and Subtractive colors.

# UNIT-IV

1. a) What is Animation? Explain basic principles of animation. (5+4+3+3)
   1. List and explain the stages of multimedia project.
   2. List the features of 3D modeling tools?
   3. List and explain various types of Multimedia Authoring Tools.

(4+4+4+3)

1. a) List the features of Image editing tools
   1. List the various suggestions for creating good title in video.
   2. Explain capabilities of multimedia Authoring tools.
   3. Write a note on video compression (CODEC).

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*