UNIT- I INTRODUCTION TO COMPUTER GRAPHICS

UNIT – I Contents at a glance:

I. Application areas of computer graphics,
II. overview of graphics systems,
Video-display devices-raster-scan systems, random scan systems,
III. graphics monitors,
IV. work stations and input devices,
V. Graphics standards.

INTRODUCTION:

Graphics:

- Graphics (derived from Greek word "*graphikos"*) are visual presentations on some surface, such as a wall, canvas, screen, paper, or stone to brand, inform, illustrate, or entertain.
- Graphics word is derived from the word graph. A graph has x and y axis. Same way something which is created in digital word is seen on a digital screen, this screen also has x and y axis. So the output on any digital device is termed as graphics.

Computer Graphics:

- graphics created using computers with help from specialized graphics hardware and software
- Computer Graphics is concerned with all aspects of producing pictures or images in computer by using specialized graphics hardware and software.
- *computer graphics* refers to several different things:
  - the representation and manipulation of image data by a computer
  - the various technologies used to create and manipulate images
  - the sub-field of computer science which studies methods for digitally synthesizing and manipulating visual content

History of computer graphics development:

1 The word "computer graphics" first phrased by William fetter, a graphics designer in 1960
2 First graphical hardware devices are Sketch Pad(by IVAN Sutherland in 1963) and Light pen
3 IVAN SUTHERLAND considered as father of computer graphics.

Types of Computer Graphics:

Computer Graphics can be broadly divided into two

a) Non Interactive Computer Graphics
b) Interactive Computer Graphics

Non Interactive Computer Graphics: In non interactive computer graphics otherwise known as passive computer graphics, the observer has no control over the image. Familiar examples of this type of computer graphics include the titles shown on TV and other forms of computer art.

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

Working of an interactive graphics display

Interactive graphics display consists of three components

a) A display controller

b) A digital memory or frame buffer

c) A television monitor

d) A video controller

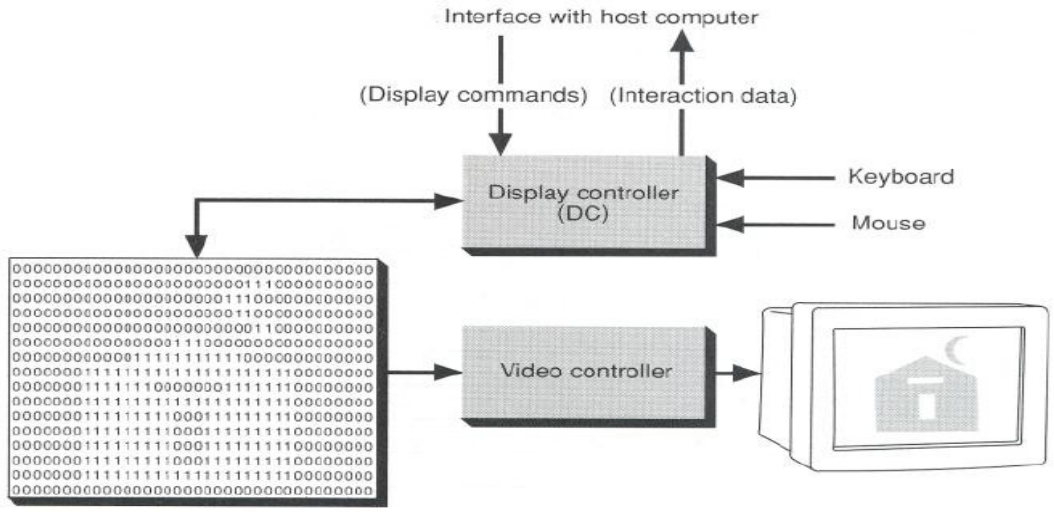
The display controller gets the inputs and commands from the user and determines the image to be displayed on the monitor. The display controller will divide the image into a number of pixels. This image which is to be displayed is stored in the frame buffer. The image will be stored as a matrix of intensity values. The image will be displayed onto the television monitor and the video controller will act as a simple interface that passes the contents of the frame buffer to the monitor. The image must be repeatedly passed to the monitor, 30 or more times a second. This helps you to maintain a steady picture on the screen.

In the frame buffer the image is stored as a pattern of binary digital numbers. These binary digital numbers represents a rectangular array of picture elements or pixels (a picture can be divided into a number of picture elements or pixels. You will learn more about pixels in the coming lectures.). So corresponding to each pixel you have a binary digital number in the frame buffer. If your image is a black and white image you can represent the black pixels by 0 s and white pixels by 1s. Therefore a 16 X 16 array of black and white pixels could be represented by the binary values stored in the 32 8-bit bytes. Now what happens to this data?

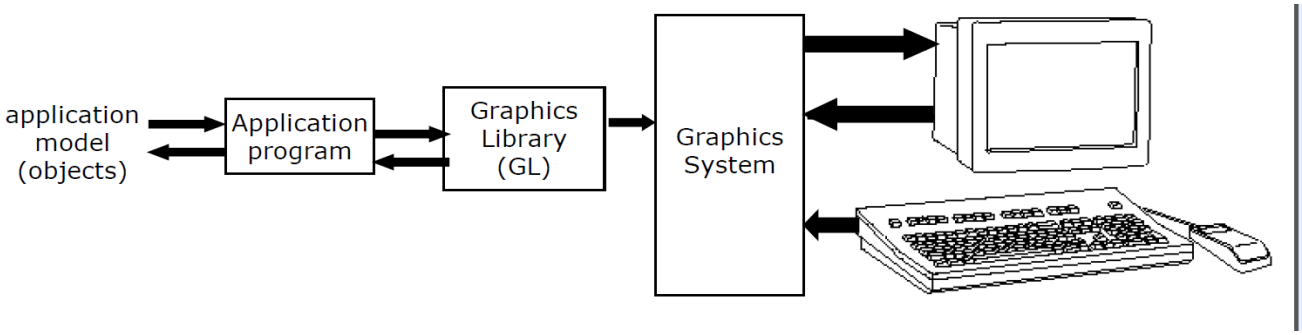
The video controller simply reads each successive byte of data from the frame buffer and converts its 0s and 1s into the corresponding video signal. This signal is then fed into the TV monitor, producing a black and white pattern on the screen. The video controller repeats this operation 30 times a second in order to maintain a steady picture on the TV screen.

All we need is to modify the frame buffer's contents. Set the frame buffer with a new set of values so that it represents the new image. In this way we can achieve effects like a rotating wheel and a wheel that grows and shrinks.

The figure given below gives an idea about the **interactive graphics display system**



Conceptual framework for interactive graphics:



I. Applications of Computer Graphics:

1. Education and Training
2. Entertainment
3. Computer Aided Design (CAD)
4. Graphs and charts or presentation graphics
5. Virtual Reality Environments
6. Data visualizations
7. Computer Art
8. Image Processing
9. Graphical User Interface (GUI)
10. Animation software

1. Education and Training:

- For some training applications, special systems are designed.
- Examples of such specialized systems are the simulators for practice sessions or training of
  - ship captains,
  - aircraft pilots,
  - heavy-equipment operators,
  - Air traffic control personnel.

The specialized pilot training systems consists of screens for visual display, key board which will be used by instructor to give input parameters which will affect airplane performance.

Education and Training

- For some training applications, special systems are designed.
- Examples of such specialized systems are the simulators for practice sessions or training of
  - ship captains,
  - aircraft pilots,
  - heavy-equipment operators,
  - Air traffic control personnel.

2.Entertainment:

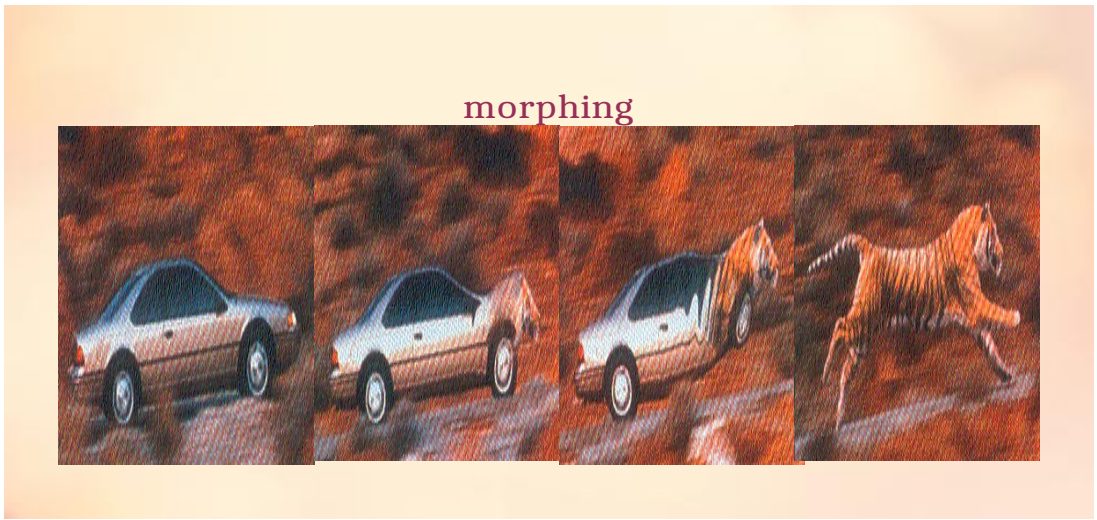
- Computer graphics methods are now commonly used in
  - making motion pictures,
  - music videos,
  - Television shows.
  - Animations

Morphing: changing object shape from one form to another.

Film making using computer rendering and animation technique

Computer graphics methods used to produce special effects to films or TV series

Figure shows example of animation characters that can be designed with help of graphics tools.



3.Computer Aided Design (CAD):

- A major use of computer graphics is in design processes, particularly for engineering and architectural systems, but almost all products are now computer designed.
- computer-aided design methods are now routinely used in the design of
  - buildings,
  - automobiles,
  - aircraft,
  - watercraft,
  - spacecraft,
  - computers,
  - textiles
  - Standard shapes of mechanical , electrical, electronic and logic circuits
- Animations are often used in CAD applications. Real-time animations using wireframe displays on a video monitor are useful for testing performance of a vehicle or system

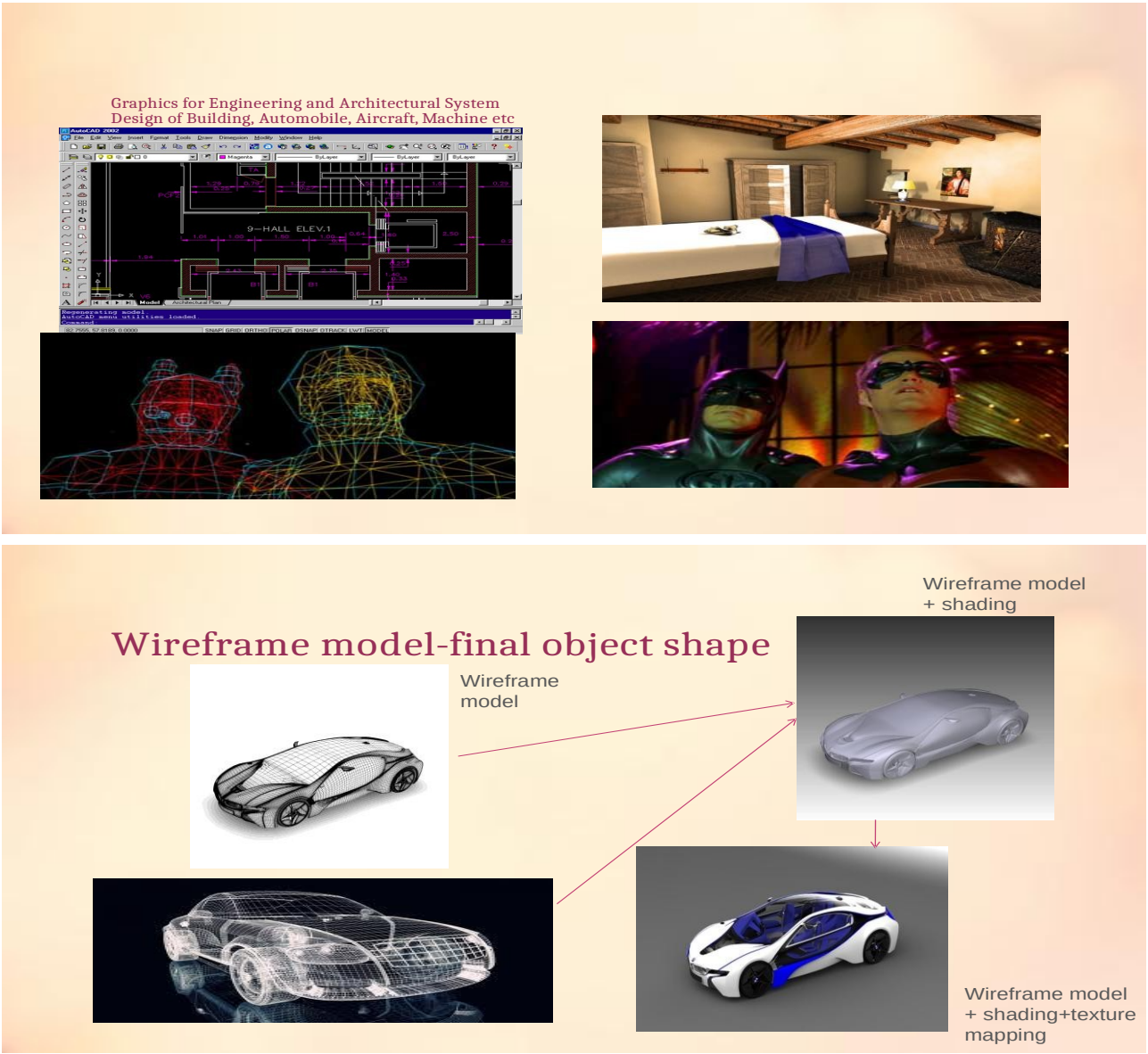


Figure: designs that can be created using CAD tool

4.Graphs and charts or presentation graphics:

- Presentation graphics is commonly used to summarize
  - financial,
  - statistical,
  - mathematical,
  - scientific, and
  - economic data for research reports,
  - Managerial reports, consumer information bulletins, and other types of reports.

Data plotting is most common computer graphics application. Examples of data plots are:

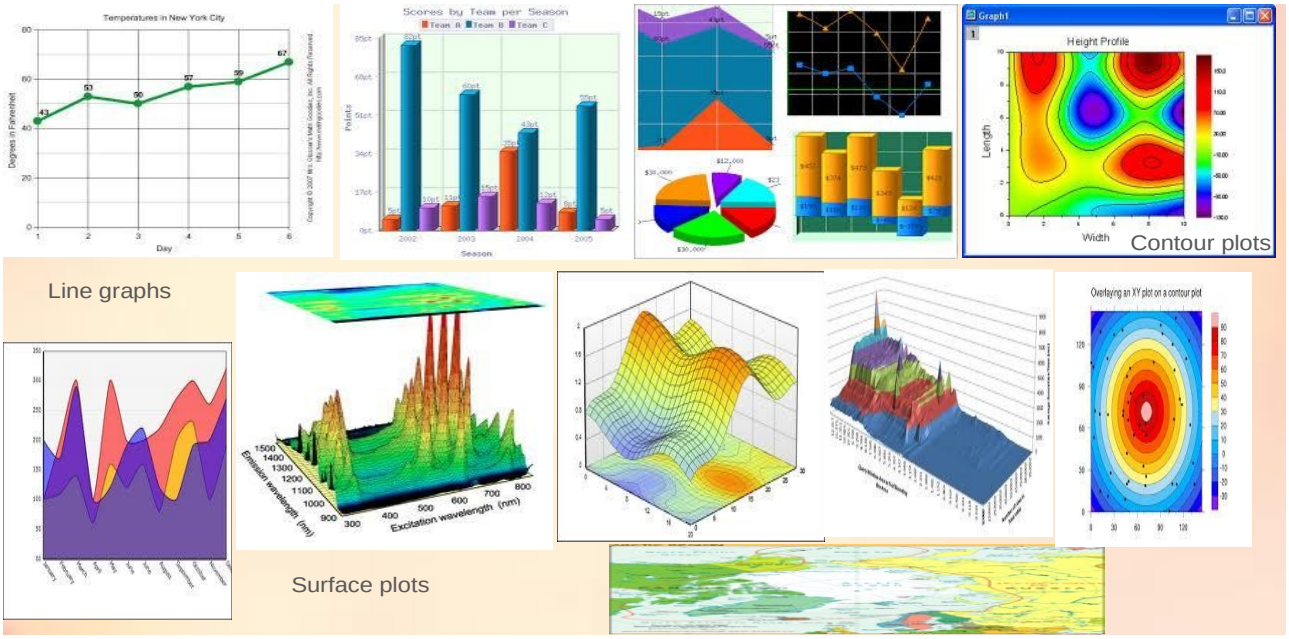
-line graphs

-bar charts

-pie charts

-surface graphs

-contour plots



5.Virtual Reality Environments:

- Virtual Reality is defined as:
  - Simulated environment
  - Interaction with human senses
  - The user will interact with objects in a three dimensional scene in virtual reality environments.
  - Specialized hardware devices provide 3D viewing effects and allow the user to pick-up objects in scene.
- Types of Virtual reality hardware devices
  - Glasses
  - Personal mobile suits
  - Head mounted visors
  - Ear phones
  - Tracking/Non-Tracking(position tracker)
  - Motion Capturing
  - Fully enclosed systems
- Virtual Reality is typically used in following applications
  - Military
  - Microsoft flight simulation
  - Medical
  - Construction

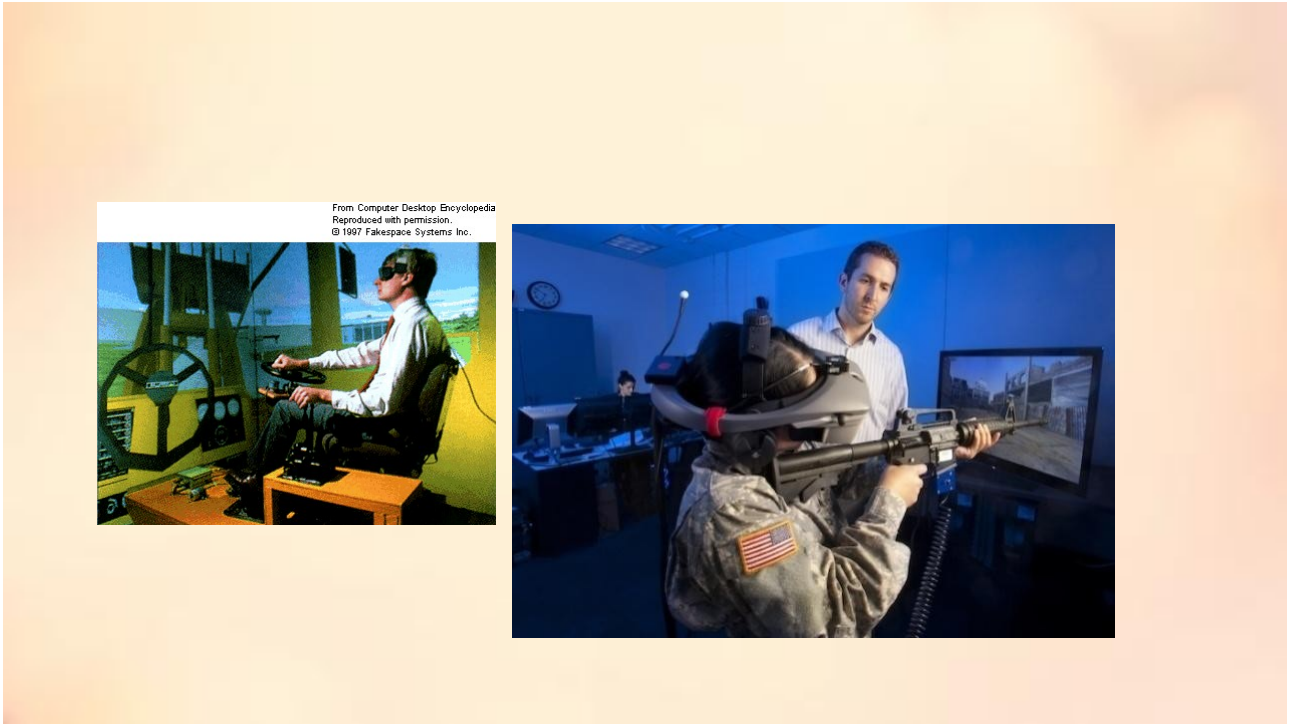


Figure: examples of virtual reality environments

6.Data Visualization:

- Scientists, engineers, medical personnel, business analysts, and others often need to analyze large amounts of information or to study the behavior of certain processes.
- Numerical simulations carried out on supercomputers frequently produce data files containing thousands and even millions of data values.

Two types of visualization:

1. Scientific Visualization
2. Business Visualization

Scientific Visualization:

Producing graphical representations for scientific, engineering and medical data sets is referred to as scientific visualization

Business Visualization:

Producing graphical representations with data sets related to commerce, industry and other non scientific areas is referred to as business visualization.

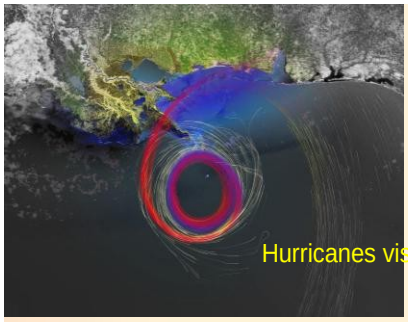
Large amount of data or information can be visualized by using visualizing techniques such as

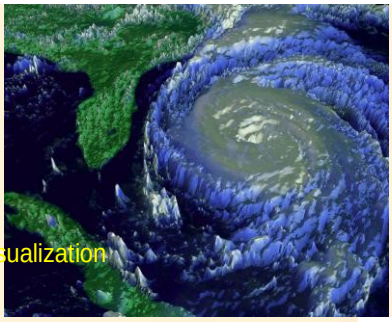
Color coding

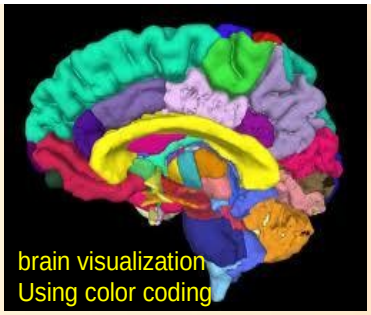
Contour plots

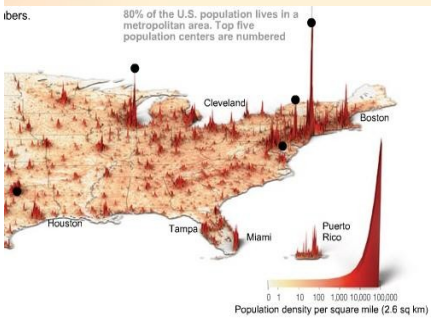
Rendering for constant-value surfaces or other spatial regions and specially designed shapes that are used to represent different data types.

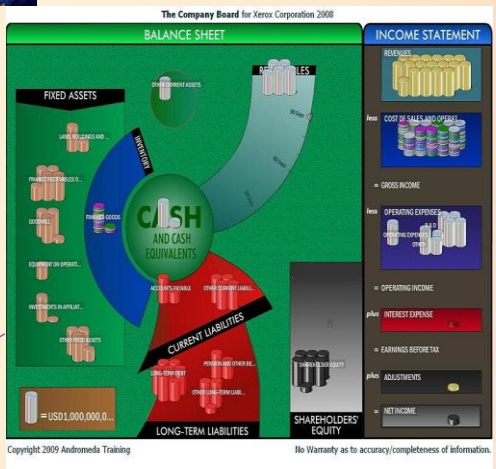
Visual techniques are also used to aid in understanding and analysis of complex processes and mathematical functions.











business visualization

7.Computer Art:

Fine art and commercial art make use of computer-graphics methods.

Artists now have available a variety of computer graphics methods and tools including specialized hardware, commercial software packages(such as Lumena), symbolic mathematics programs(mathematica), CAD packages, animation systems that provide facilities for designing object shapes and specifying object motions.

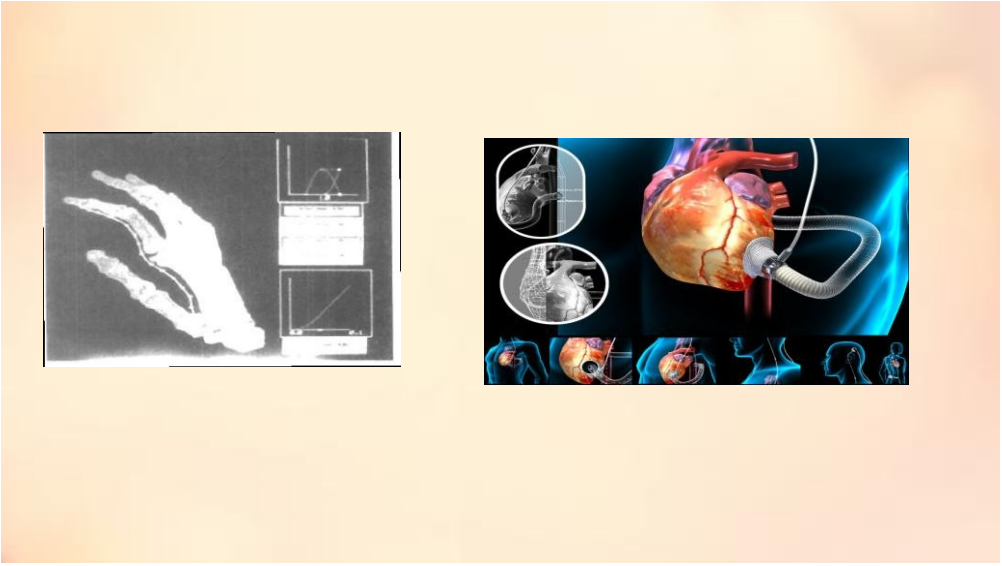
- Paintbrush
- Film animations,
- Advertisements design
- Morphing

8.Image Processing:

- Image processing, on the other hand applies techniques to modify or interpret existing pictures, such as photographs and TV scans.
- Two principal applications of image processing are
  - improving picture quality and
  - Machine perception of visual information, as used in robotics.
- Medical applications of image processing:
  - X ray
  - Tomography
  - Computed *X-ray tomography (CT) and position emission tomography (PET)* uses projection methods to reconstruct cross sections from digital data.

The improving picture quality of images is an application of image processing which is more used in commercial art applications that involve the retouching and rearranging of sections of photographs.

The image processing techniques are also used to analyze satellite photos of earth.



9.Graphical User Interface (GUI):

All the application software's provides a graphical user interface (GUI). Graphical interfaces have component windows manager.

Each application software consists of multiple windows called display windows which can be activated by a mouse click or an input from keyboard or a touch input.

GUI's generally consists of the following things:

- Interfaces
- Display menus
- icons

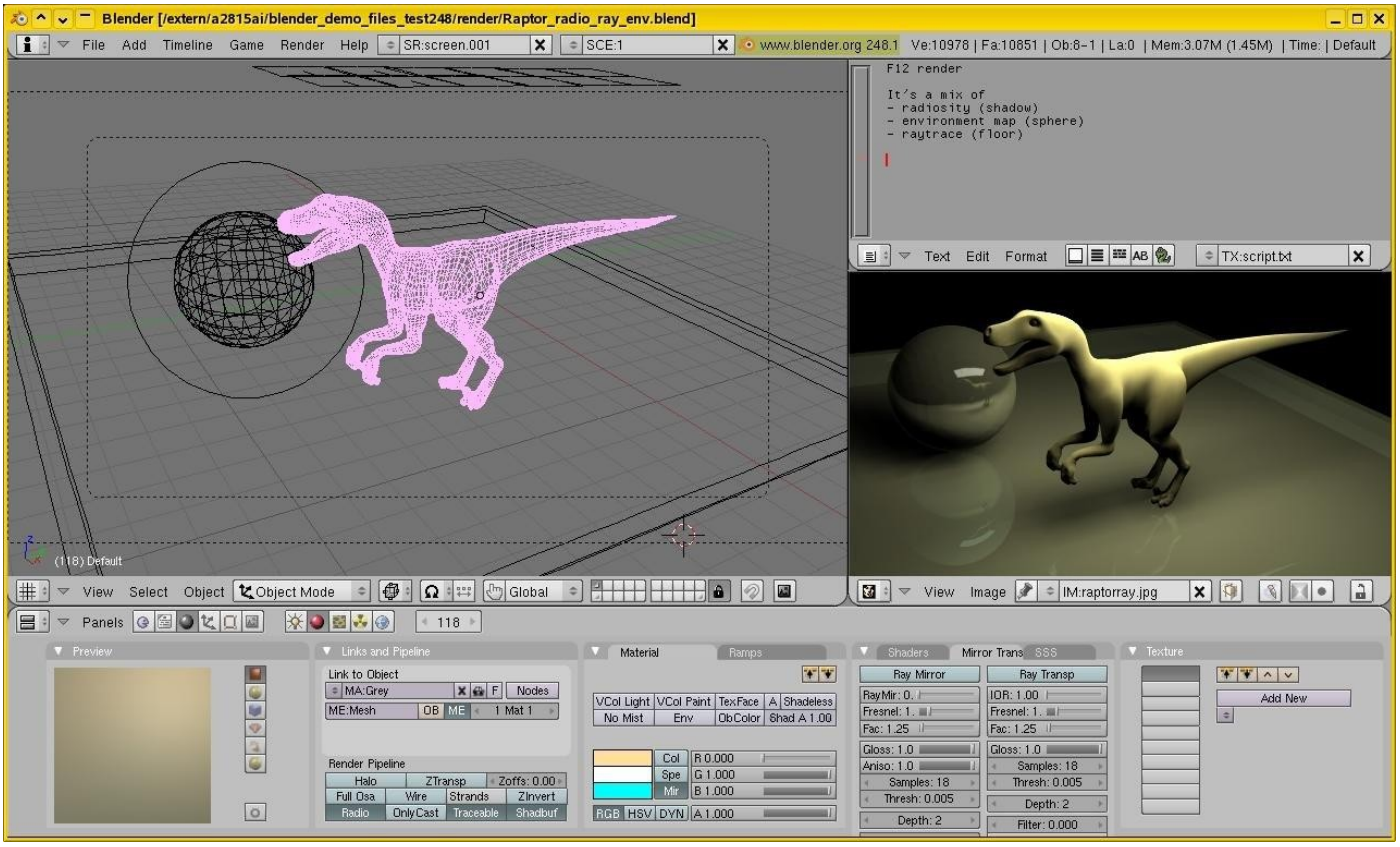


Figure: showing blender tool interface for designing a dynosarous character

II. OVERVIEW OF GRAPHICS SYSTEM:

The operation of most video monitors is based on the standard cathode-ray tube (CRT) design

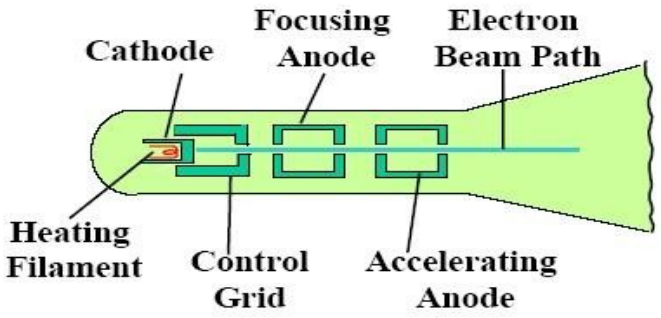
VIDEO DISPLAY DEVICES:

1 Refresh Cathode-Ray Tubes
2 Raster-Scan Displays
3 Random-Scan Displays
4 Color CRT Monitors
5 Direct-View Storage Tubes
6 Flat-Panel Displays
7 Three-Dimensional Viewing Devices
8 Stereoscopic and Virtual-Reality Systems

1. REFRESH CATHODE-RAY TUBE (CRT):

Basic operation of 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 phosphorous 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.



CRT design with electro static deflection

Components of CRT:

1) Connector pins
2) Base
3) Electron Gun
    1) Heated metal cathode
    2) Control grid
4) Focusing System
    1) Focusing anode
    2) Accelerating anode
5) Magnetic Deflection Coils
    1) Vertical Deflection plates
    2) Horizontal Deflection plates
6) Electron Beam & Phosphor Coated Screen

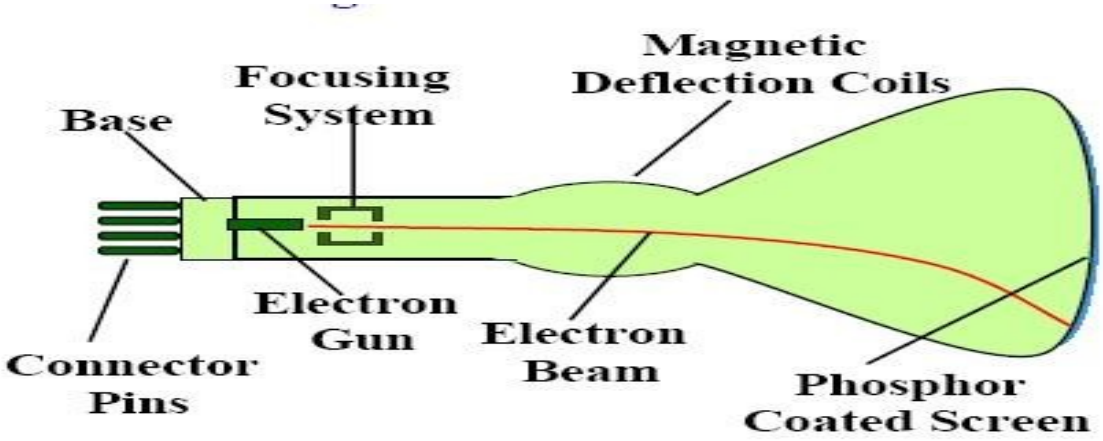


Figure: CRT design with magnetic deflection

Electron gun operation:

- The primary components of an electron gun in a CRT are the heated metal cathode and a control grid .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 towards the phosphor coating by a high positive voltage.
- The accelerating voltage can be generated with a positively charged metal coating on the in side of the CRT envelope near the phosphor screen, or an accelerating anode can be used
- Controlling of Intensity of electron beam
- Focusing System
- Controlling deflection of electron beam
  - Electric fields
  - Magnetic Fields

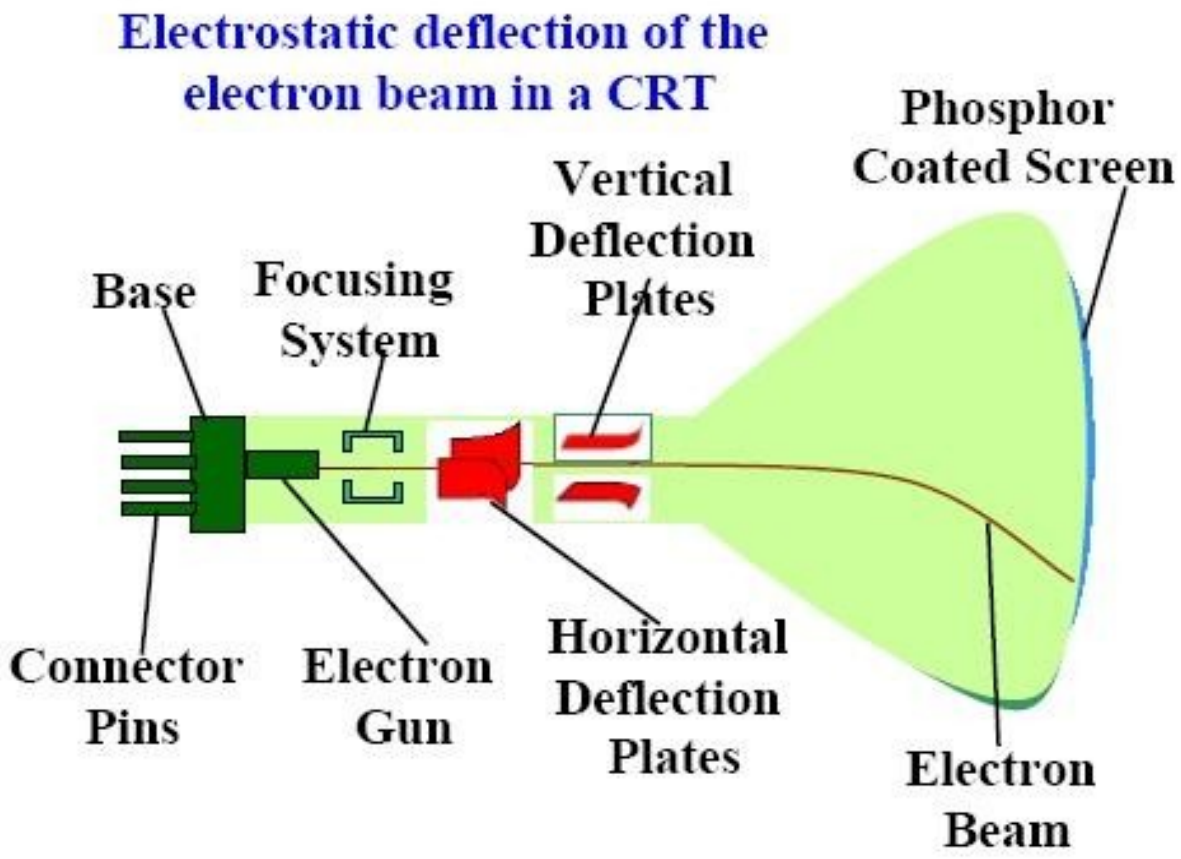
Excitation of phosphor electrons:

- Types of phosphorous screens:
  - High Persistence
  - Low persistence

- **Persistence:** Persistence is defined as the time it takes the emitted light from the screen to decay to one-tenth of its original intensity
- **Resolution:** 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.
- **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. (Sometimes aspect ratio is stated in terms of the ratio of horizontal to vertical points.) 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.

CRT design showing the operation of electron gun:



2. RASTER SCAN DISPLAYS:

Raster Scan methods have increasingly become the dominant technology since about 1975. These methods use the TV type raster scan. The growth in the use of such methods has been dependent on rapidly decreasing memory prices and on the availability of cheap scan generating hardware from the TV industry.

The screen is coated with discrete dots of phosphor, usually called pixels, laid out in a rectangular array. The image is then determined by how each pixel is intensified. The representation of the image used in servicing the refresh system is thus an area of memory holding a value for each pixel. This memory area holding the image representation is called the frame buffer.

The values in the frame buffer are held as a sequence of horizontal lines of pixel values from the top of the screen down. The scan generator then moves the beam in a series of horizontal lines with fly-back (non-intensified) between each line and between the end of the frame and the beginning of the next frame.

Features:

1. Adopts the Television Technology
2. Used by most common type of graphics monitor
3. Electron beam swept across each row from left to right ,pixel by pixel, from top to bottom & generates the series of on/off patterns
4. **Frame Buffer/ Refresh Buffer:**
   - Area holds the picture definition
   - Retrieved and used for approximation
   - Screen points can be referred as Pixel/Pel
   - Performance of the approximated picture (intensity range for pixels) gets decided by capability of raster scan system

5. Bitmap:

- Frame buffer which holds the intensity values for black and white system(1 bit per pixel for storing pixel intensity value)

6. Pixmap:

- Frame buffer which holds the intensity values for colour system(24 bits/pixel for storing pixel intensity value)

7. Refreshing rate:

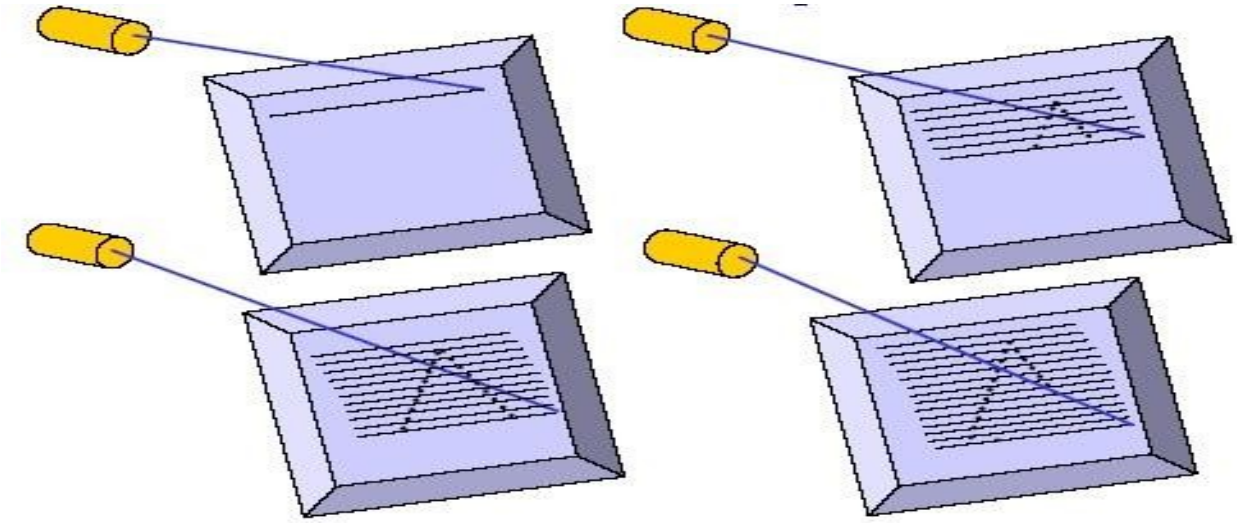
- 60 to 80 frames per second
- Represented as cycles per second/ Hertz(Hz)

8.Horizontal Retrace:

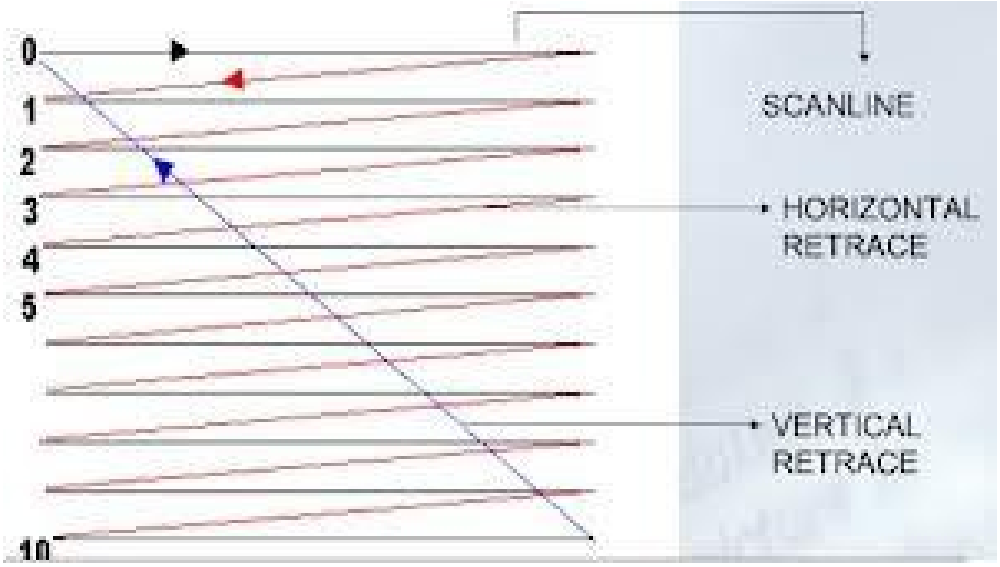
- The return of electron beam to the left of the screen after refreshing each scan line

9. Vertical Retrace:

- The return of electron beam to the left top corner of the screen after the completion of each frame

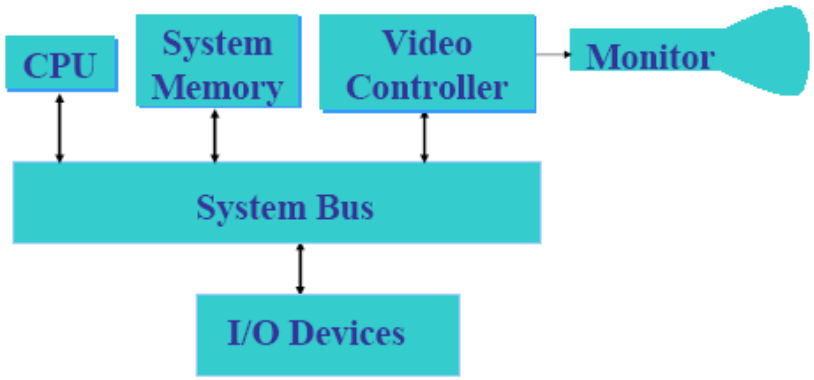


Raster Scan—Horizontal Retrace and Vertical Retrace:



RASTER SCAN SYSTEMS:

You know that interactive raster graphics system typically employs several processing units. In addition to central processing unit, or CPU, there is a special –purpose processor, called the video controller or display controller. It is used to control the operation of the display device.

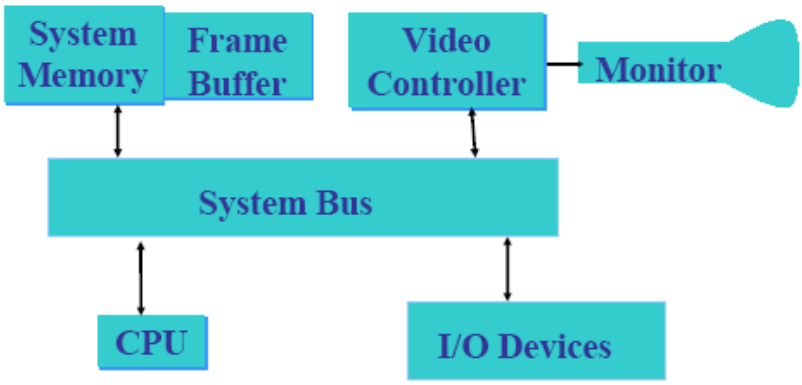


Architecture of a simple raster graphics system

Here the frame buffer can be anywhere in the system memory, and the video controller accesses the frame buffer to refresh the screen.

Video Controller:

In commonly used raster systems a fixed area is reserved for the frame buffer, and the video controller is given direct access to the frame buffer memory. Frame buffer locations and the corresponding screen positions are referenced in Cartesian coordinates.



Architecture of a raster system with a fixed portion of the system memory reserved for the frame buffer

For the graphics monitor, the origin I defined at the lower left screen corner. The screen is represented as a two dimensional system i.e. the x values increasing to the right and the y values increasing from bottom to top. Each scan line is labeled as y_{max} at the top of the screen to 0 at the bottom. Along each scan line, screen pixel positions are labeled from 0 to x_{max}.

Refresh operation of the video buffer

In the basic refresh operation, two registers are used. The purpose of this registers are to store the coordinates of the screen pixels. Initially the x register is set to 0 and the y register is set to y_{max}. The value stored in the frame buffer for this pixel position is 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 is repeated for each pixel along the scan line.

After the last pixel on the top of the scan line has been processed, the x register is reset to 0and the y register is decremented by 1. The pixels along this scan line are then processed and the procedure is repeated for each successive scan line.

After cycling through all the pixels along the bottom scan line(y=0), the video controller resets the registers to the first pixel position on the top scan line and refresh process starts over.

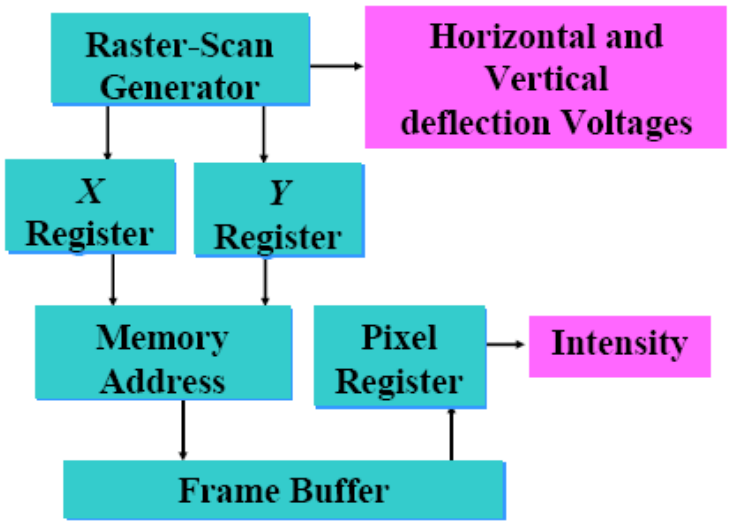
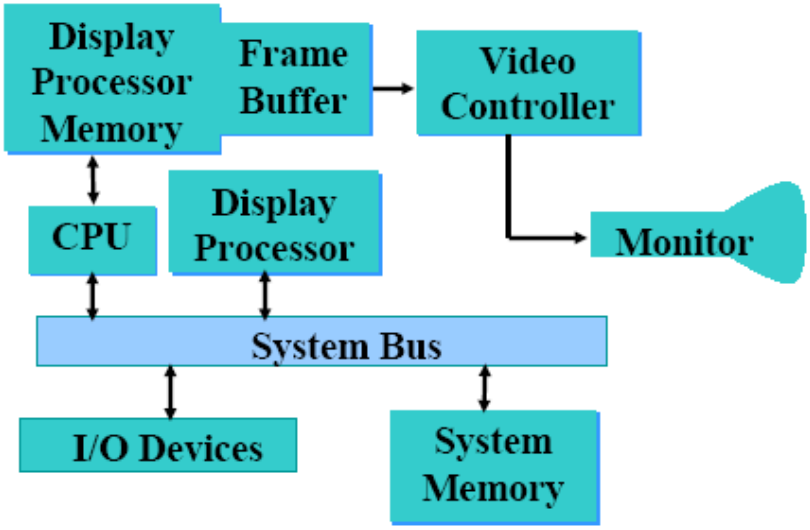


Figure: Basic Video Controller refresh operations

In high quality systems two buffers are used so that one is used for refreshing and other is being filled with intensity values. Then the two buffers can switch the roles. This provides a fast mechanism for generating real time animations, since different views of moving objects can be successively loaded into the refresh buffer.

Raster Scan Display processor

Raster Scan system may contain a separate display processor.



Raster Scan Display processor is sometimes referred to as graphics controller or a display coprocessor. The purpose of this processor is to free the CPU from the graphics tasks. But what is its major task? Its major task is to digitize a picture definition given in an application program into a set of pixel intensity values for storage in the frame buffer. This digitization process is called scan conversion. Graphics commands specifying straight lines and other geometric shapes can be can converted into a set of intensity points. When you scan convert a straight line segment we have to locate the pixel positions closest to the line path and store the intensity for each position in the frame buffer.

Display processor are deigned to perform different other functions. They are

- To generate various line styles (dashed , dotted, or solid) & To display color areas
- To perform certain manipulations and transformations on displayed objects
- To interface with interactive input devices, such as mouse

Organization of frame buffer in Raster Scan systems:

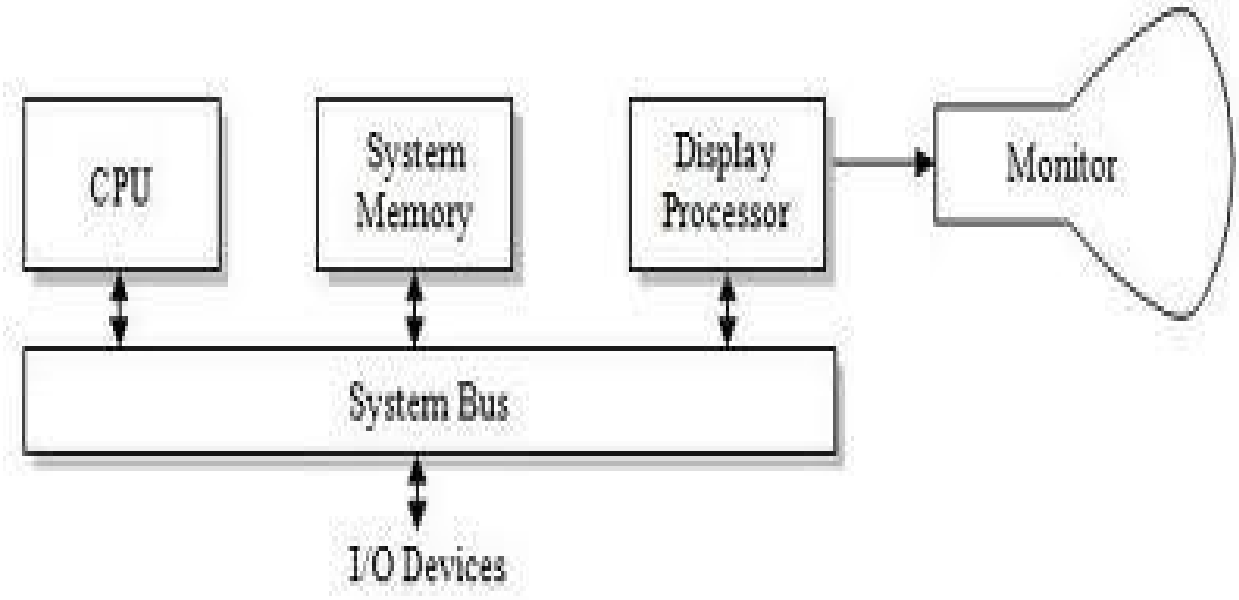
- Types of possibilities of organizing Frame buffer:
  - Frame buffer will be present somewhere inside of system memory
  - Some raster systems have a fixed portion of system memory reserved for frame buffer
  - Some raster systems have 2 frame buffers. one frame buffer will contains intensity values of present image. Second frame buffer contains picture pattern of next image which is to be displayed.
  - In some raster systems, frame buffer will be implemented using linked list representation
  - Run length encoding
  - Cell encoding
  - Some raster systems will have separate display processors. This display processor will contain separate display processor memory, frame buffer.

3. RANDOM SCAN DISPLAY:

Random scan displays, often termed vector Vector, Stroke, and Line drawing displays, came first and are still used in some applications. Here the characters are also made of sequences of strokes (or short lines). The electron gun of a CRT illuminates straight lines in any order. The display processor repeatedly reads a variable 'display file' defining a sequence of X,Y coordinate pairs and brightness or color values, and converts these to voltages controlling the electron gun

In Random Scan Systems an application program is input and stored in the system memory along with a graphics package. Graphics commands in the application program are translated by the graphics package into a display file stored in the system memory. This display file is then accessed by the display processor to refresh the screen. The display processor cycles through each command in the display file program once during every refresh cycle.

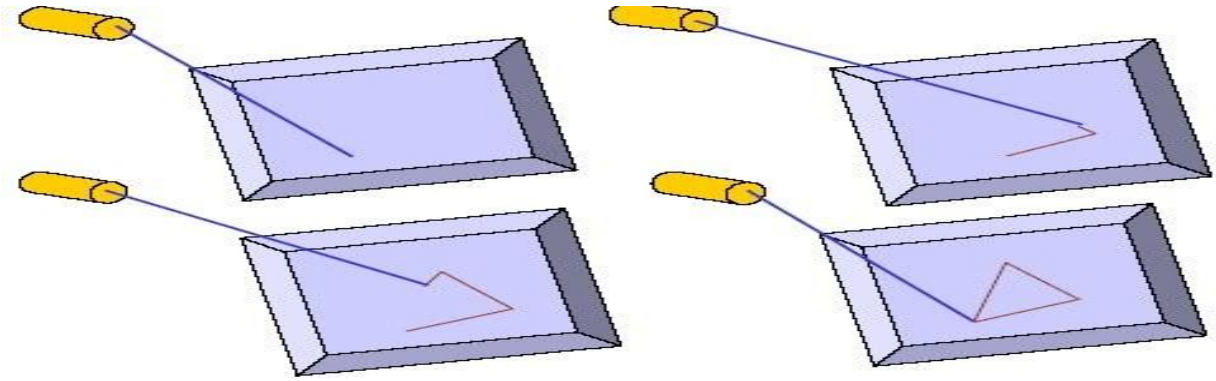
Sometimes the display processor in the random scan system is referred to as a display processing unit or a graphics controller. Graphics patterns are drawn on a random scan system by directing the electron beam along the component lines of picture. Lines are defined by the values for their coordinate endpoints, and these input coordinate values are converted to x and y deflection voltage. A scene is then drawn one line at a time by positioning the beam to fill in the line between specified endpoints.



Features of Random Scan display:

1 Electron beam directed only to the parts of the screen where, picture is to be drawn
2 Draws the picture one line at a time
3 Also be called as vector drawn displays/ stroke-writing/ calligraphic displays
4 The approximated elements of picture gets refreshed in any specific order.
5 Refresh rate depends upon the number of lines to be displayed
6 Picture definition will be the set of commands called as “display program” or display file.
7 Draws the lines 30 to 60 times per second
8 Mainly used in line drawing applications
9 Won’t be suitable for realistic displays

Random scan display



Advantages of Random scan display:

- Very high resolution
- Easy for animation
- Little memory requirements

Drawbacks of Random scan display:

- Very expensive
- Limited color capability
- Can’t draw the complex images

4. COLOR CRT MONITORS:

1 A CRT monitor displays color pictures by using a combination of phosphors that emit different-colored light.

2 The two basic techniques for producing color displays with a CRT are the

- beam-penetration method and
- the shadow-mask method.

Beam Penetration Method:

1 Used with random scan monitors

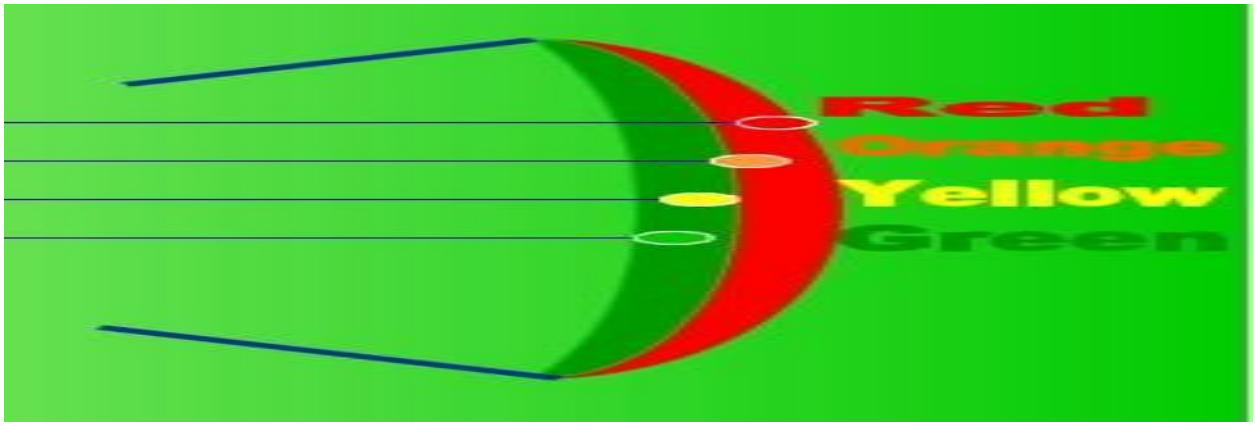
2 Two Layers of Phosphor is used Red and Green

3 The displayed color depends on how far the electron beam penetrates into the phosphoric layers

4 Only 4 colors are possible

Red, Green, Orange, Yellow

5 Intensity is based on the Speed of the beam



Shadow Masking Method

1 Used in Raster Scan Systems

2 Produces wider range of colors than beam penetration method

3 Each pixel position has three phosphor dots

4 This type of CRT possess 3 electron guns one for each dot

5 Shadow mask grid placed just behind the screen.

6 Color Variations

Varying the intensity levels of the three electron beams

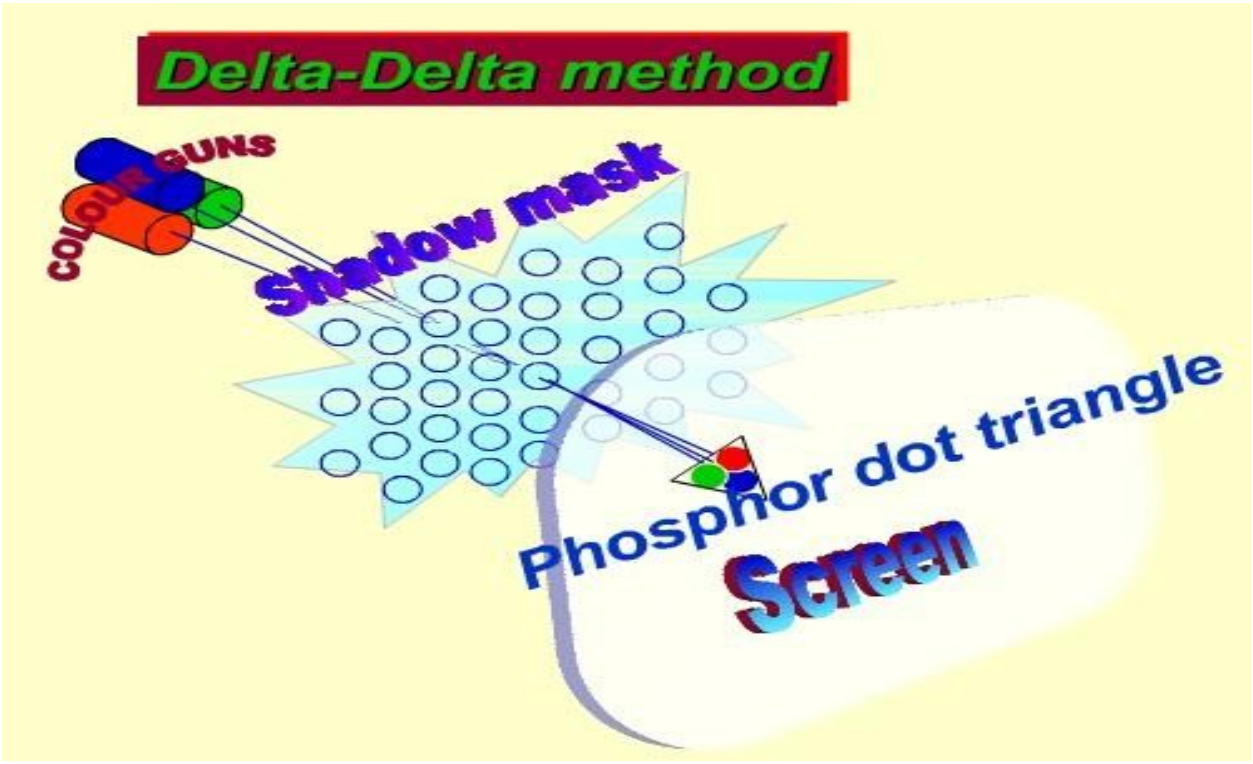
7 24 bits of storage per pixel

8 Arrangements

- Delta-Delta Method
- In-Line Method

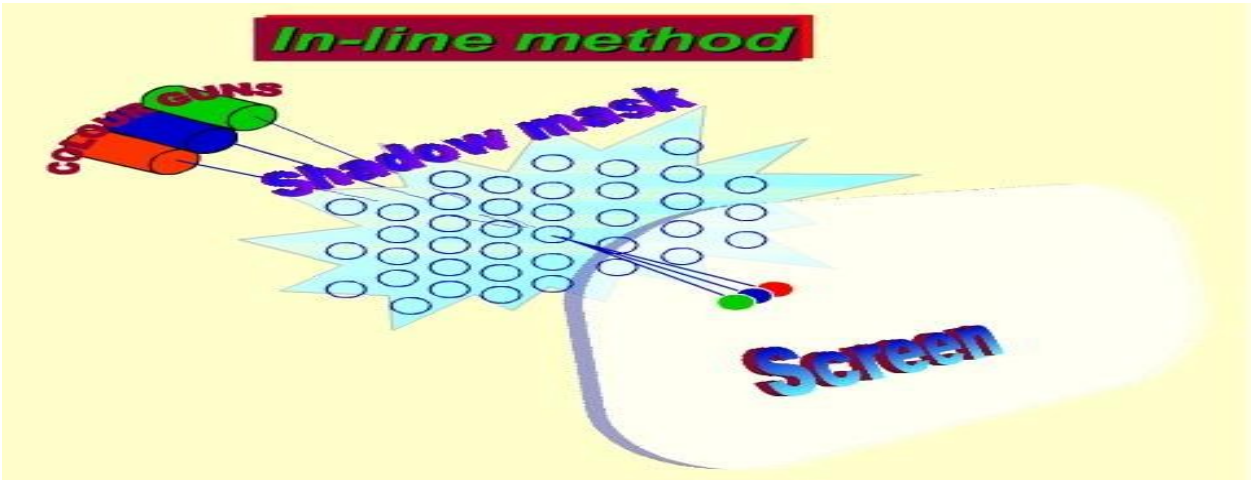
Delta-Delta Method

- Commonly used in color CRTs
- Mask holds the series of holes aligned with phosphor dot patterns (triangular pattern)
- Beam pass through the hole in the mask & activate the dot triangle which appears as a color spot



In-Line Method

- Three Electron guns, the tri-colour dots in the screen are aligned along one scan line instead of triangular shape
- Commonly used in high resolution color CRTs



5. Direct-View Storage Tubes:

Direct View Storage Tube (DVST)

1 Similar to random scan but persistent => no flicker

2 Can be incrementally updated but not selectively erased

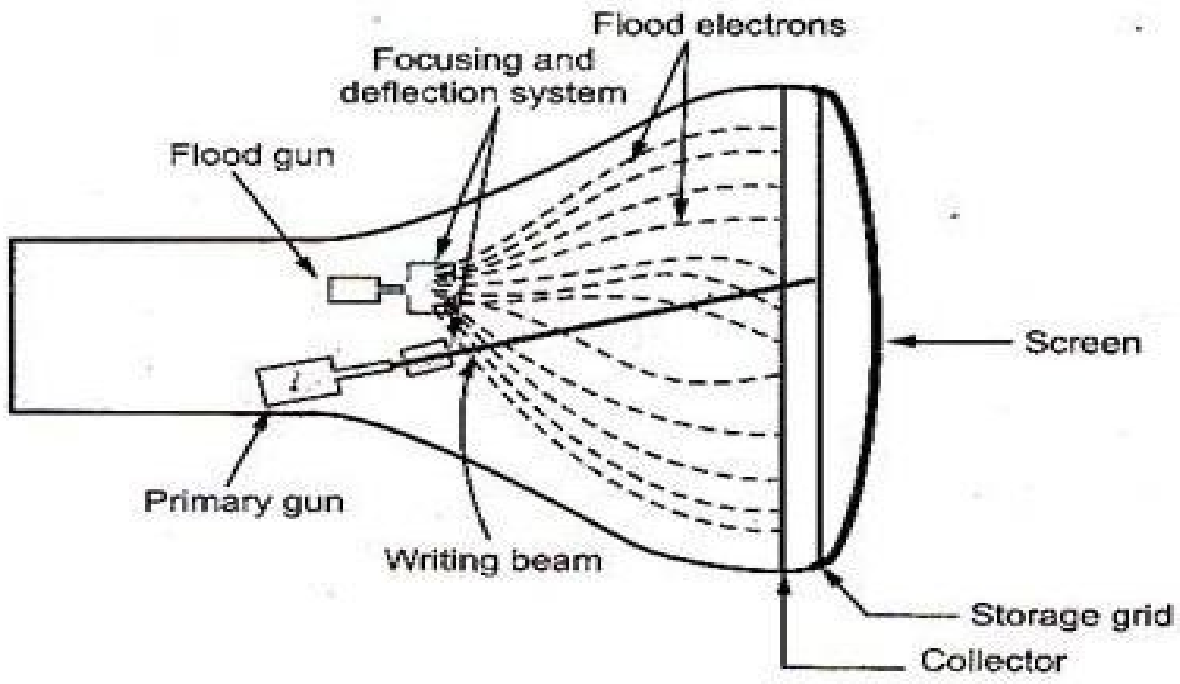
3 Used in analogue storage oscilloscopes

4 Achieve good resolution

5 Possess two electron guns

6 Primary gun -> to store the charge(intensity values of present image) in storage grid

7 Flood gun -> to retain the image in the screen. It contains the picture pattern which is to be displayed.



6. FLAT-PANEL DISPLAYS:

1 The term flat-panel display refers to a class of video devices that have reduced volume, weight and power requirements compared to a CRT.

2 2 types

a. Emissive Displays(plasma, LED's, Thin electroluminescent displays)

b. Non Emissive Displays(LCD)

3 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. Flat CRTs have also been devised, in which electron beams arts accelerated parallel to the screen, then deflected 90' to the screen. But flat CRTs have not proved to be as successful as other emissive devices.

4 Non emissive displays (or non emitters) use optical effects to convert sunlight or light from some other source into graphics patterns. The most important example of a non emissive flat-panel display is a liquid-crystal device.

LIQUID CRYSTAL DISPLAYS (LCD)

1 Smaller, lighter, with no radiation problems. Matrix addressable.

2 Found on portables and notebooks, and starting to appear more and more on desktops.

3 Similar in principle to that found in the digital watch

4 Thin layer of liquid crystal sandwiched between 2 glass plates.

5 Top plate transparent and polarized, bottom plate reflecting.

6 External light passes through top plate and crystal, and reflects back to eye. When voltage applied to crystal (via the conducting glass plates) it changes its polarisation, rotating the incoming light so that it cannot reflect back to the eye.

7 LCD requires refreshing at usual rates, but slow response of crystal means flicker not usually noticeable.

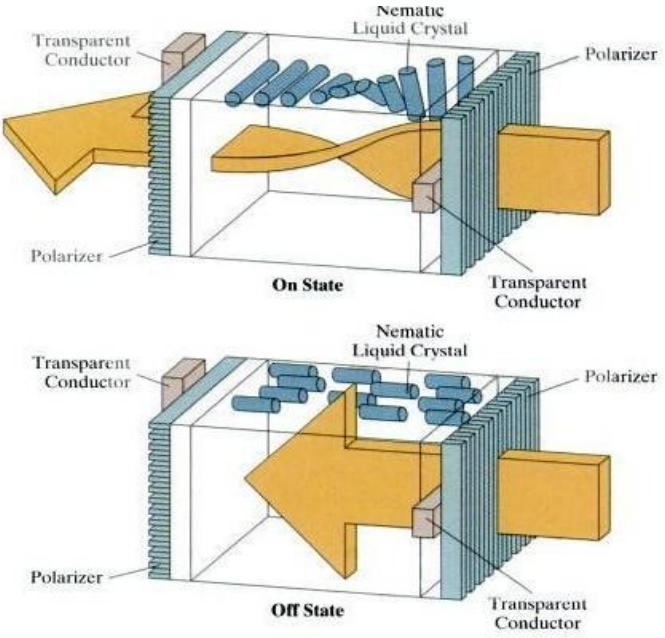


FIGURE 2-15 The light-twisting, shutter effect used in the design of most liquid-crystal display devices.

PLASMA display:

A plasma display panel (PDP) is a type of flat panel display common to large TV displays 30 inches (76 cm) or larger. They are called "plasma" displays because the technology utilizes small cells containing electrically charged ionized gases, or what are in essence chambers more commonly known as fluorescent lamps.

- Enhanced-definition plasma television
- High-definition plasma television
- HD Resolutions

What is Plasma?

- Plasma is referred to be the main element of a fluorescent light. It is actually a gas including ions and electrons. Under normal conditions, the gals has only uncharged particles. That is, the number of positive charged particles [protons] will be equal to the number of negative charged particles [electrons]. This gives the gas a balanced position.
- Suppose you apply a voltage onto the gas, the number of electrons increases and causes an unbalance. These free electrons hit the atoms, knocking loose other electrons. Thus, with the missing electron, the component gets a more positive charge and so becomes an ion.
- In plasma, photons of energy are released, if an electrical current is allowed to pass through it. Both the electrons and ions get attracted to each other causing inter collision. This collision causes the energy to be produced. Take a look at the figure illustrated below.

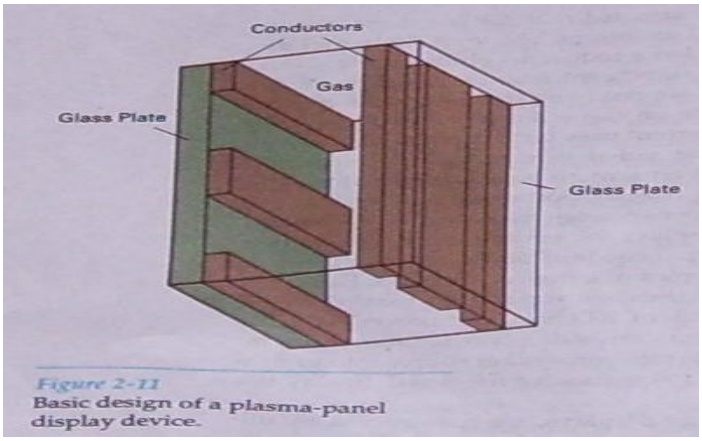
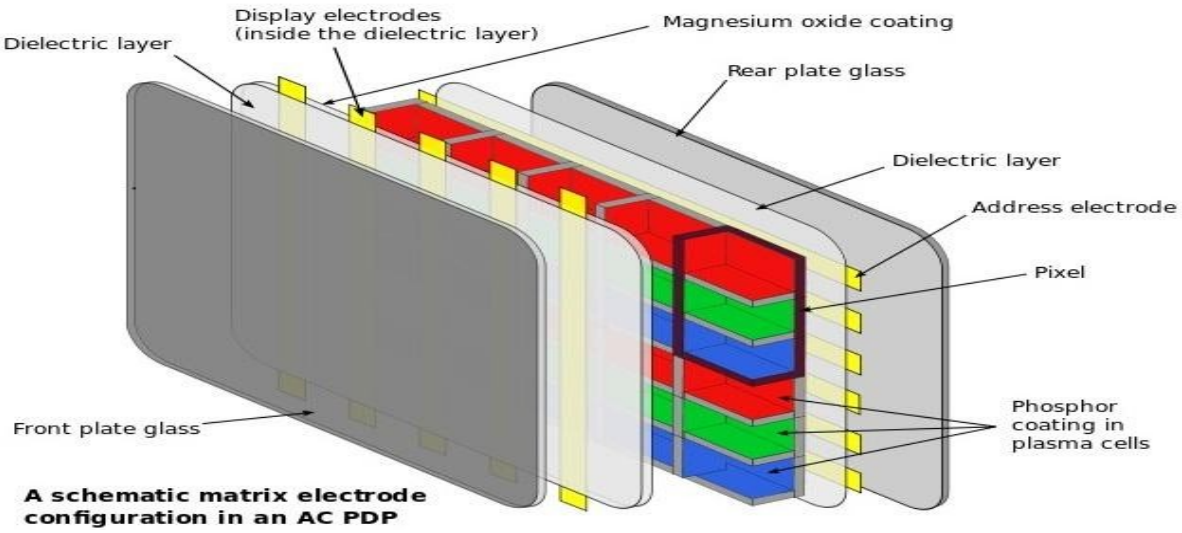


Figure 2-11
Basic design of a plasma-panel display device.

Working of Plasma Display



A schematic matrix electrode configuration in an AC PDP

1 The basic of any television is the Cathode Ray Tube [CRT]. We have been using this technology for the past 75 years. To know more about the working of CRT click the link given below.

2 But there are a lot of disadvantages for a CRT. The display lacks in clarity, and the size of the screen is huge. They become bulkier in size with the increase in screen width as the length f the tube has to be increased accordingly. This, inturn increases the weight.

3 Plasma display is the best remedy for this. Their wide screen display, small size and high definition clarity are their greatest advantages.

4 Plasma displays mostly make use of the Xenon and neon atoms. When the energy is liberated during collision, light is produced by them. These light photons are mostly ultraviolet in nature. Though they are not visible to us, they play a very important factor in exciting the photons that are visible to us.

5 In an ordinary TV, high beams of electrons are shot from an electron gun. These electrons hit the screen and cause the pixels to light up. The TV has three types of composite pixel colors which are distributed throughout the screen in the same manner. They are red, green and blue. These colors when mixed in different proportions can form the other colors. Thus the TV produces all the colors needed.

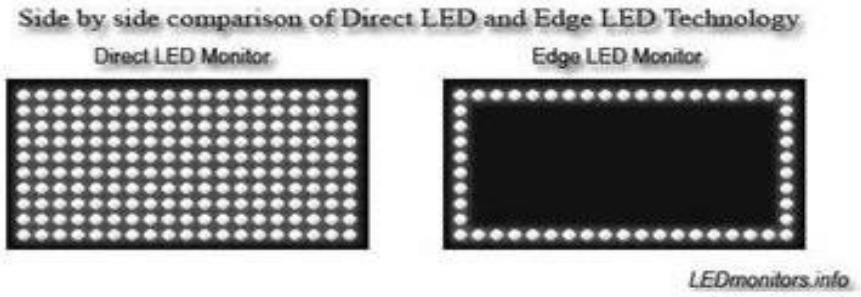
6 A plasma display consists of fluorescent lights which causes the formation of an image on screen. Like a CRT TV, each pixel has the three composite fluorescent color lights. These fluorescent lights are illuminated and the different colors are formed by combining the composite colors.

LED displays:

A **light-emitting diode** (**LED**) is a semiconductor diode that emits incoherent narrow-spectrum light when electrically biased in the forward direction of the p-n junction. This effect is a form of electroluminescence. An LED is usually a small area source, often with extra optics added to the chip that shapes its radiation pattern. The color of the emitted light depends on the composition and condition of the semi conducting material used, and can be infrared, visible, or near-ultraviolet.

An LED can be used as a regular household light source. Like a normal diode, an LED consists of a chip of semi conducting material impregnated, or *doped*, with impurities to create a *p-n junction*. As in other diodes, current flows easily from the p-side, or anode, to the n-side, or cathode, but not in the reverse direction. Charge-carriers—electrons and holes—flow into the junction from electrodes with different voltages. When an electron meets a hole, it falls into a lower energy level, and releases energy in the form of a photon. The wavelength of the light emitted, and therefore its color, depends on the band gap energy of the materials forming the *p-n junction*. In silicon or germanium diodes, the electrons and holes recombine by a *non-radiative transition* which produces no optical emission, because these are indirect band gap materials. The materials used for an LED have a direct band gap with energies corresponding to near-infrared, visible or near-ultraviolet light.

LEDs are usually built on an n-type substrate, with an electrode attached to the p-type layer deposited on its surface. P-type substrates, while less common, occur as well. Many commercial LEDs, especially GaN/InGaN, also use sapphire substrate. Substrates that are transparent to the emitted wavelength, and backed by a reflective layer, increase the LED efficiency. The refractive index of the package material should match the index of the semiconductor, otherwise the produced light gets partially reflected back into the semiconductor, where it may be absorbed and turned into additional heat, thus lowering the efficiency. An anti-reflection coating may be added as well.



7. Three Dimensional Viewing Devices:

Graphics monitors for the display of 3-dimensional scenes have been developed using a technique that reflects a CRT image from a vibrating , flexible mirror.

As the varifocal mirror vibrates, it changes the focal length.

These vibrations are synchronized with the display of an object on a CRT so that each point on the object is reflected from the mirror into a special position corresponding to the distance of that point from a specified viewing location.

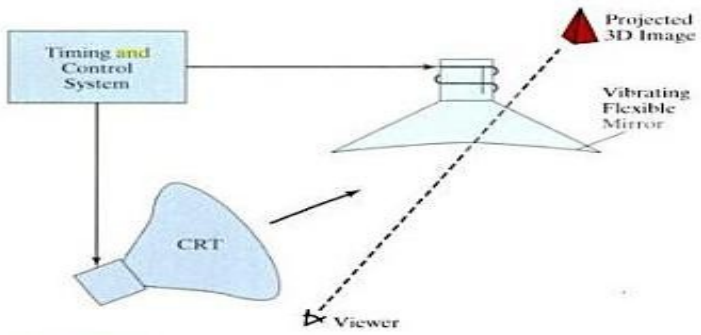


FIGURE 2-16 Operation of a three-dimensional display system using a vibrating mirror that changes focal length to match the depths of points in a scene.

8. Stereoscopic and Virtual Reality Systems:

We can represent 3-dimensional objects by displaying stereoscopic views of the object. This method gives a 3-dimensional effect by presenting a different view to each eye of an observer so that scene appears to have a depth.

We obtain stereoscopic projection by getting two views (left and right views) of a scene generated with viewing directions along the lines from the position of eyes.

When we look simultaneously at left view with left eye and right view with right eye, the two views merge into a single image and we perceive a depth in scene.

One way to produce a stereoscopic effect on a raster system is to display each of the two views on alternate refresh cycles. The screen is viewed through glasses.

For creating a **virtual reality environment**, we can do the following things:

1) we can use stereoscopic viewing as component in virtual reality systems, where user can step into scene and interact with the environment. A headset containing optical system is used to generate stereoscopic views. A sensing system in the headset keeps track of viewer's position.

2) We can use projectors for creating a virtual reality environment.

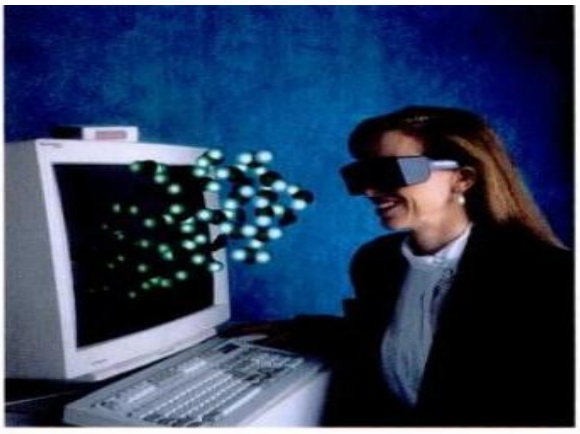


FIGURE 2–18 Simulated viewing of a stereoscopic projection. (*Courtesy of StereoGraphics Corporation.*)

FIGURE 2–20 Glasses for viewing a stereoscopic scene and an infrared synchronizing emitter. (*Courtesy of StereoGraphics Corporation.*)

FIGURE 2–21 A headset used in virtual-reality systems. (*Courtesy of Virtual Research.*)

III. GRAPHICS MONITORS:

1 Most graphics monitors today operate as raster scan displays

2 High-definition graphics monitor used in applications such as air traffic control, simulation, medical imaging, and CAD.

3 A multi screen system called the Media Wall, provides a large "wall-sized display area. This system is designed for applications that require large area displays in brightly lighted environments, such as at trade shows, conventions, retail stores, museums, or passenger terminals.

4 Following are the requirements for graphics monitors:

a. High Resolution

b. High interacting capabilities like monitors with touch sensing system

c. High speed processors

5 Some examples of graphics monitors are:

a. Dual display monitors

b. Large screen monitors(combination of small screen monitors)

FIGURE 2-33 The SGI Reality Center 2000D, featuring an ImmersaDesk R2 and displaying a large-screen stereoscopic view of pressure contours in a vascular blood-flow simulation superimposed on a volume-rendered anatomic data set. (*Courtesy of Silicon Graphics, Inc. and Professor Charles Taylor, Stanford University. © 2003 SGI. All rights reserved.*)

FIGURE 2-34 A wide-screen view of a molecular system displayed on the three-channel SGI Reality Center 3300W. (*Courtesy of Silicon Graphics, Inc. and Molecular Simulations. © 2003 SGI. All rights reserved.*)

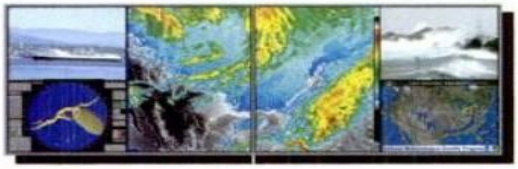


FIGURE 2-35 A multi-panel display system called the "Super Wall". (*Courtesy of RGB Spectrum.*)

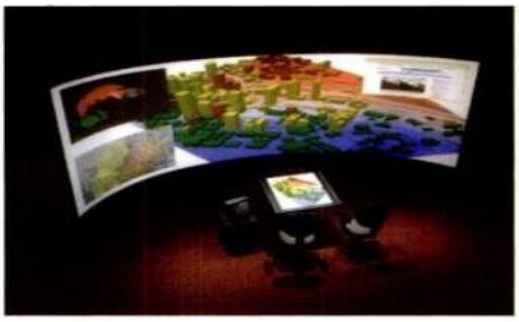


FIGURE 2-36 A homeland security study displayed using a system with a large curved viewing screen. (*Courtesy of Silicon Graphics, Inc. © 2003. All rights reserved.*)

IV. WORKSTATIONS:

- Workstation refers to a group of input and output devices connected to a single computer or multiple computers for a particular task.
- Graphics workstation may be group of only interactive graphical input devices or only output devices or a combination of both connected to computer.

FIGURE 2-38 A geophysical visualization presented on a 25-foot semicircular screen, which provides a 160° horizontal and 40° vertical field of view. (*Courtesy of Silicon Graphics, Inc., the Landmark Graphics Corporation, and Trimension Systems. © 2003 SGI. All rights reserved.*)

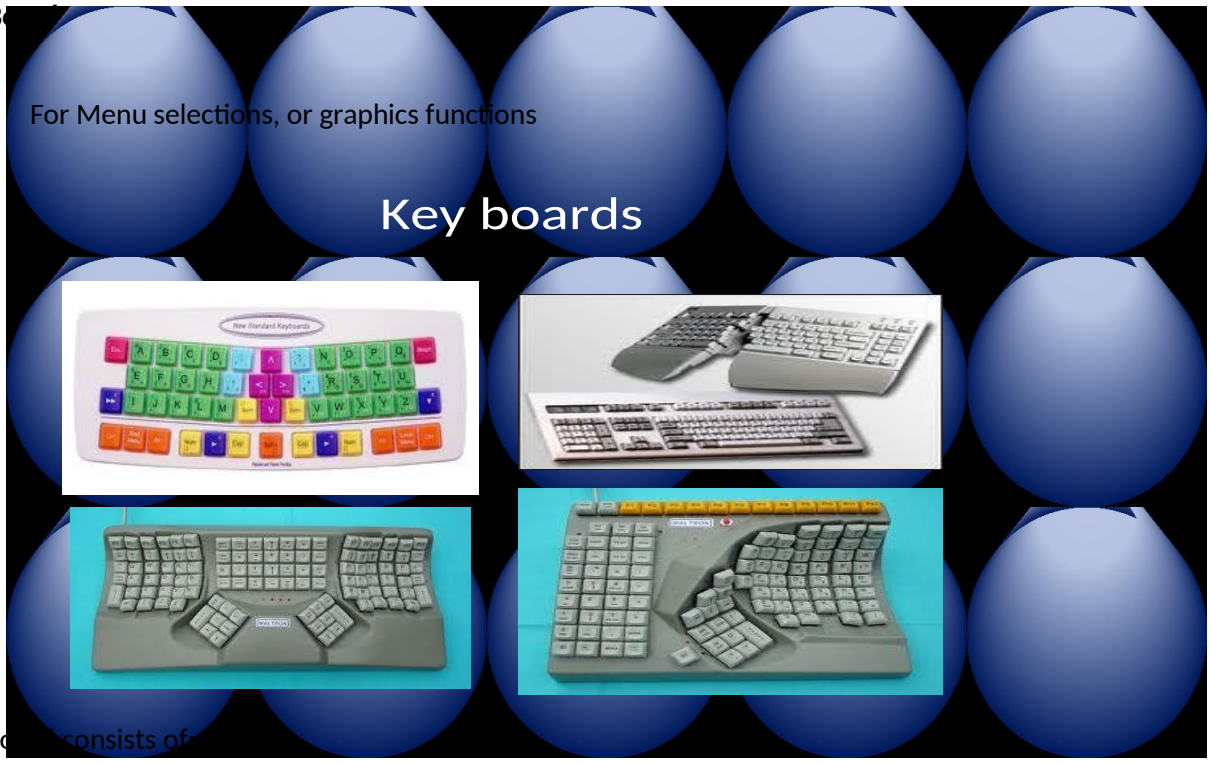
FIGURE 2-39 The 360° viewing screen in the NASA airport control-tower simulator, called the FutureFlight Central Facility. (*Courtesy of Silicon Graphics, Inc. and NASA. © 2003 SGI. All rights reserved.*)

INPUT DEVICES:

- Keyboards
- Mouse
- Trackball and Space ball
- Joysticks
- Data Glove
- Digitizers
- Image Scanners
- Touch Panels
- Light Pens
- Voice Systems

Key B

-
- For Menu selections, or graphics functions



Key b consists of:

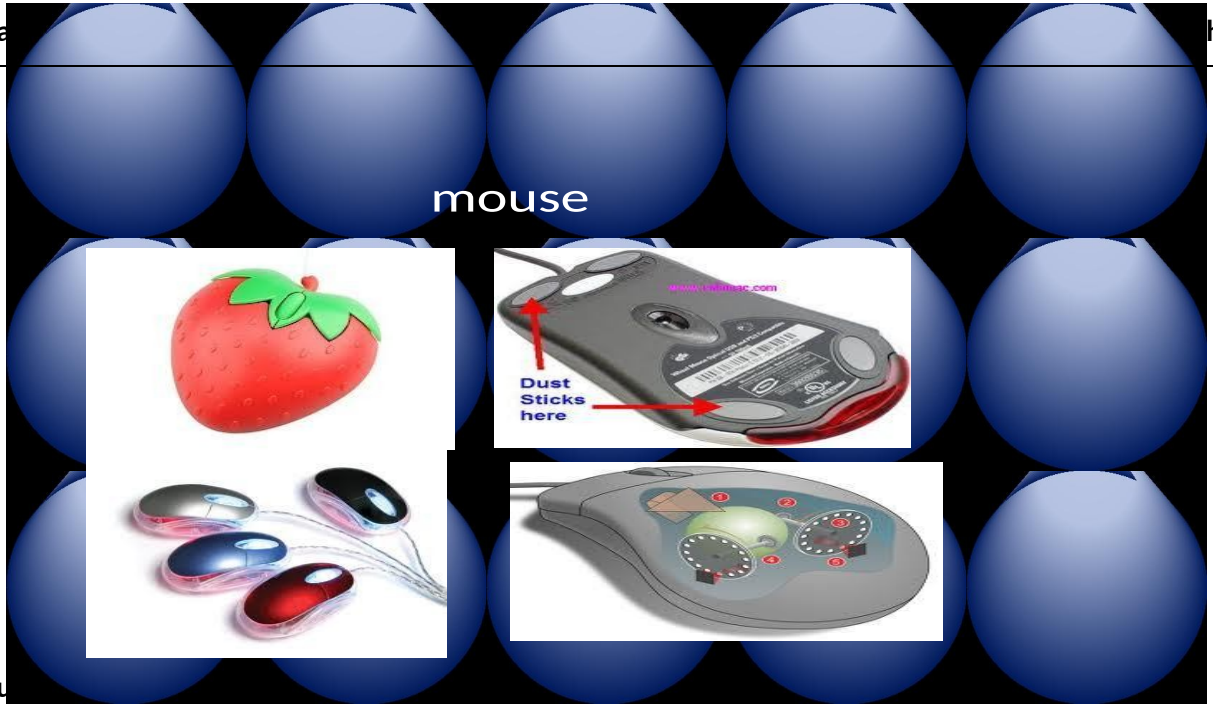
- Cursor-control keys
- function keys
- numeric keypad

Buttons and switches are often used to input predefined functions, and dials are common devices for entering scalar values.

- Potentiometers are used to measure dial rotations, which are then converted to deflection voltages for cursor movement.

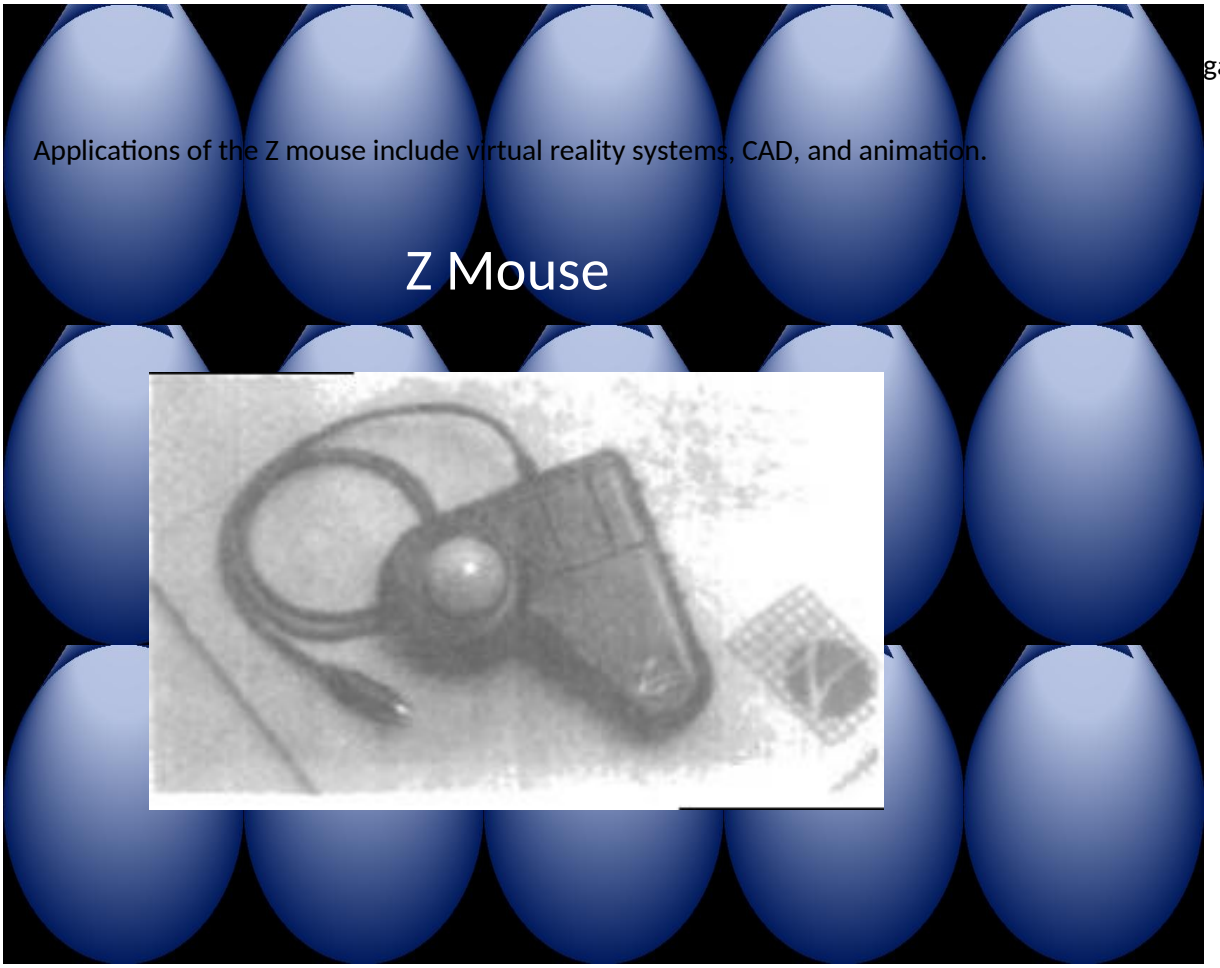
Mouse:

- 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.
- OPTICAL SENSOR is another method for detecting mouse motion.
- used for making relative changes in the position of the screen cursor.



Z Mou

- Z mouse consists of 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 Input Devices spatial positions, rotations, and other
- gate our
- Applications of the Z mouse include virtual reality systems, CAD, and animation.



Track ball and Space ball:

Track ball:

1 Trackball is a ball that can be rotated with the fingers or palm of the hand

2 Potentiometers attached to the ball, measure the amount and direction of rotation.

3 Trackballs are o mounted on key rds

Space ball:

1 Space ball provides six degrees of freedom.

2 Unlike the trackball, a space ball does not actually move.

3 al

4 ystems, modeling, animation, CAD, and other applications.

Joystick:

1 A joystick consists of a small, vertical lever (called the stick) mounted on a base that is used to steer the screen cursor around.

2 Most joysticks select screen positions with actual stick movement;

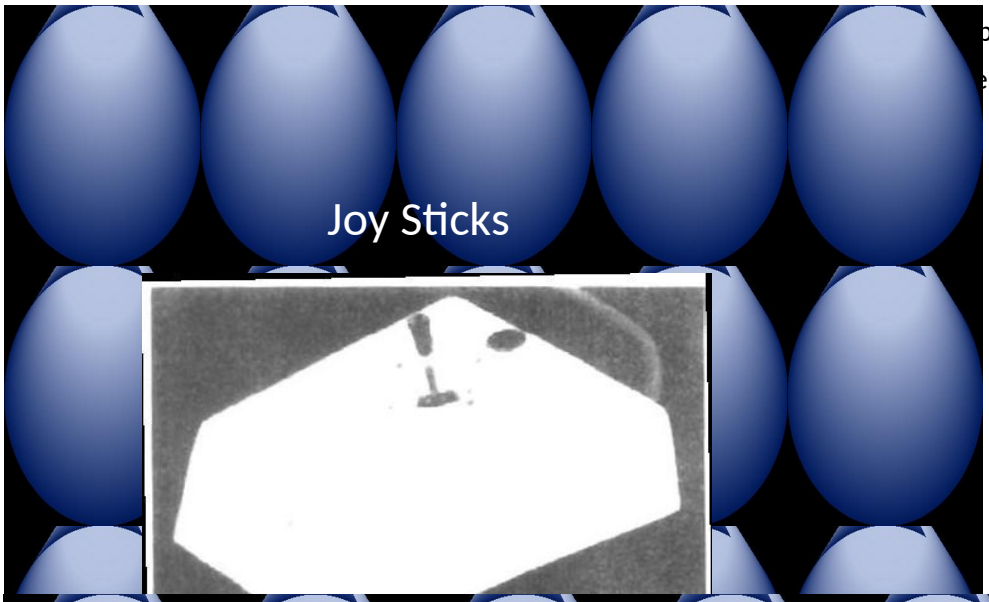
3 others respond to pressure on the stick

4 The distance that the stick is moved in any direction from its center position corresponds to screen-cursor movement in that direction

5 Potentiometers mounted at the base of the joystick measure the amount of movement

6 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.

7 Eight switches, arranged in a circle, are sometimes provided, so that the stick can select any one of eight directions for cursor movement.



ble stick.

ement of the cursor in the

- 

Data Gloves:

- Used to grasp a virtual object
- The glove is constructed with a series of sensors that detect hand and finger motions.
- Electromagnetic coupling between transmitting antennas and receiving antennas is used to provide
- Input from the glove can be used to position or manipulate objects in a virtual scene.

Digitizers

1 A common device for drawing, painting, or interactively selecting coordinate positions on an object is a digitizer.

2 These devices can be used to input coordinate values in either a two-dimensional or a three-dimensional space

3 Types:

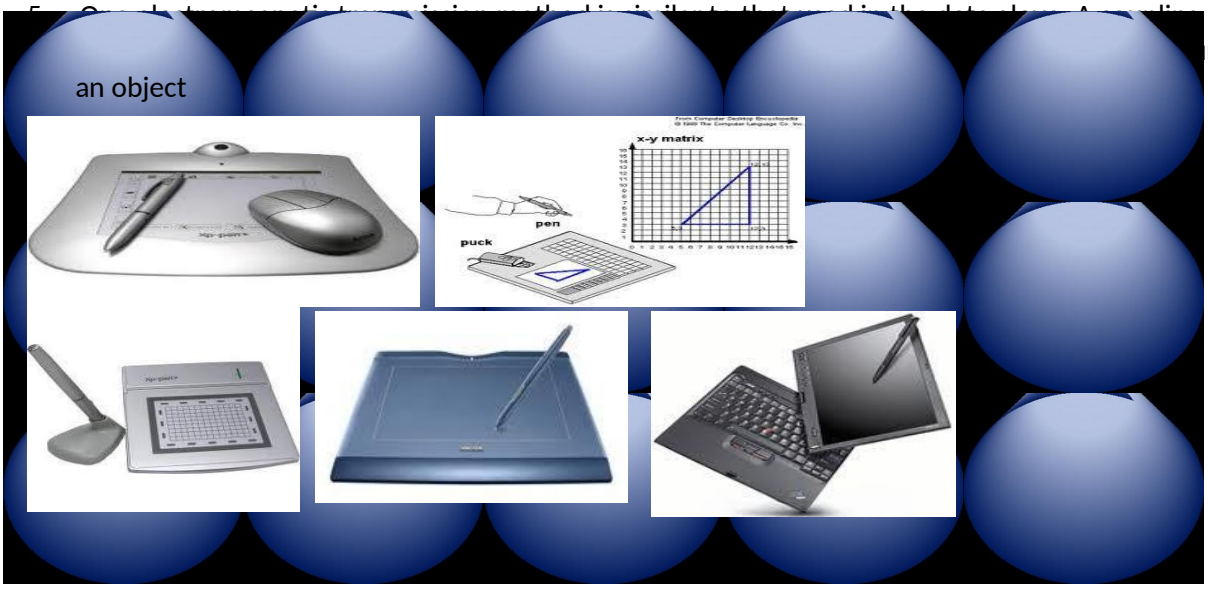
   a. Graphics tablet (based on hand cursor)

   b. Graphics tablet (based on stylus)

   c. Acoustic (or sonic) tablets use sound waves to detect a stylus position. Either strip microphones or point microphones can be used to detect the sound emitted by an electrical spark from a stylus tip.

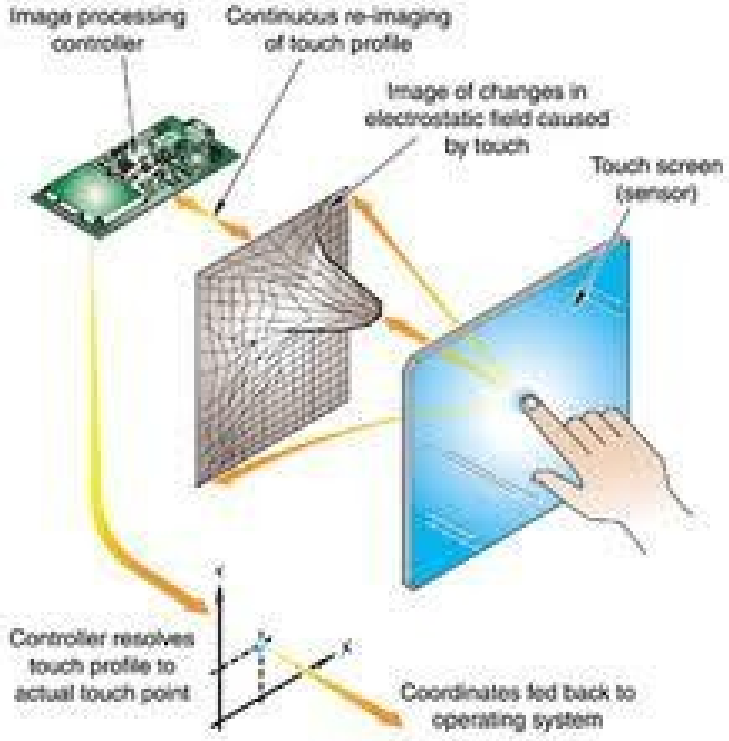
4 Three-dimensional digitizers use sonic or electromagnetic transmissions to word positions.

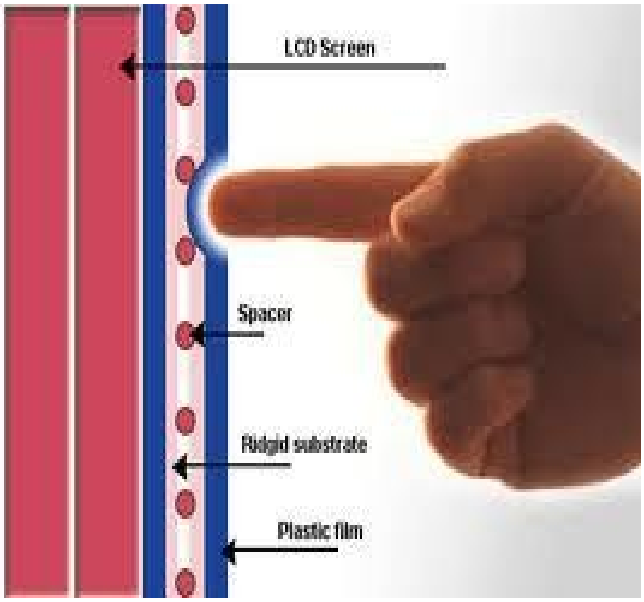
5 One electromagnetic transmission method is similar to that used in the data glove. A coupling between transmitter and receiver is used to report the position and orientation of an object



Touch Panels:

- Touch Panels are used for accessing applications in Computer or mobiles with help of finger touch.
- Touch systems are used for interacting graphically with objects or folders or text on screen.
- Touch System can be maintained in 3 different ways.
  - Optical touch panels(by using infra red LED's on edges of screen)
  - An electrical touch panel(by using conductive and resistive materials)
  - In acoustical touch panels(with the help of sound waves)



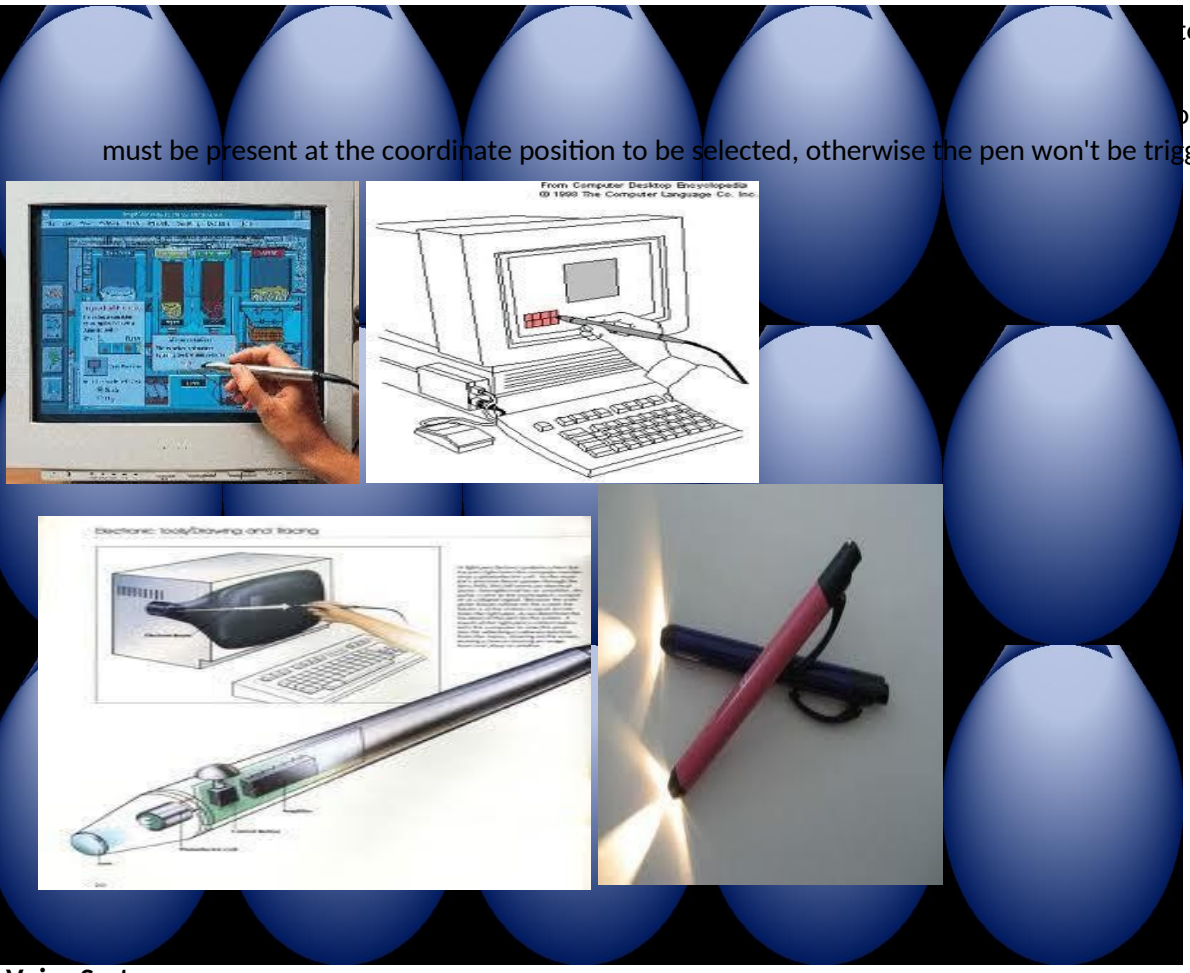


Light Pens:

- A light pen is a computer input device in the form of a light-sensitive and used in conjunction with a computer's CRT display.

touchscreen but

intensity level must be present at the coordinate position to be selected, otherwise the pen won't be triggered.



Voice Systems:

1 These are used for interacting with any system based on the audio input given by the user.

2 Voice systems have a dictionary of audio words.

3 The dictionary contains audio sounds and list of actions to be done for a particular audio sound.

4 Voice systems identify the sound based on frequency pattern of audio sound.

GRAPHICS SOFTWARE:

- Graphics software are of two types:
  - General Programming Languages(High level-Ex: C, FORTRAN etc)
  - Special purpose application Packages(Ex: CAD, Paint application, Maya, Flash, pixer's Render man)

COORDINATE SYSTEM:

- We use Cartesian coordinate system in computer graphics.
- 3 types of coordinates:
  - Modeling coordinates or local coordinates
  - World coordinates
  - Device coordinates.

GRAPHICS FUNCTIONS:

- Graphics Functions include functions for drawing simple shapes like line, circle, rectangle, ellipse etc
- Graphics Functions are also available for giving attributes to simple shapes. Example are giving line color or line style or font color or font style or filling styles etc

V. GRAPHICS STANDARDS:

1 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. Without standards, programs design for one hardware system often cannot be transferred to another system without extensive rewriting of the programs.

2 Portability is an important feature for any programming language.

3 It makes the programs written in PC to work in another PC.

4 For achieving portability, there should be common standards for developing programs.

5 The ISO first took initiation and developed first standard for developing graphics applications.

6 Two standards:

a. **1)GKS(Graphics Kernel System**—for drawing simple 2D shapes)

- GKS second Version(it extends from 2D drawing to 3D drawing)

b. **2) PHIGS (Programmers Hierarchical Interactive Graphics Standard)—**Allows us to draw not only 2D, 3D shapes, but also allows to do some manipulations.

- -PHIGS+(Extended version of PHIGS for doing complete manipulations on 2D,3D objects. surface rendering, polygons filling, changing colors etc are possible).

7 Graphical Kernel System (GKS)-first graphics software standard by ISO

8 PHIGS (Programmer's Hierarchical Interactive Graphics standard)-extension of GKS

PROBLEMS

1) **Assuming Z buffer algorithm allows 128 depth value levels to be used, approximately how much memory would a 512x512 pixel display require to store the Z-buffer?If the scene consists of 14 objects, what is the frame buffer memory requirements?**

Ans: depth values=128=2^7

No of bits per pixel =7

Resolution= 512 x 512

Memory required= 7x 512 x 512 bits

Memory for 14 objects= 14x(7x 512 x512) bits

2) **Consider three different raster systems with resolutions of 640 by 400, 1280 by 1024, and 2560 by 2048. What size frame buffer (in byte is needed for each of these systems to store 12 bits per pixel? How, much storage is required for each system if 24 bits per pixel are to be stored?**

Ans: the frame buffer sizes are : (640 x 400 x 12)/8 bits

(1280x 1024x12)/8 bits

(2560 x 2048 x 12)/8 bits

For 24 bits per pixel, the storage required is (640 x 400 x 24)/8 bits

(1280x 1024x24)/8 bits

(2560 x 2048 x 24)/8 bits

3) **Consider a raster display system with resolution of 800 by 400. How many pixels could be accessed per second by a display controller that refreshes the screen at the rate of 60 frames per second?**

Ans: access rate is (800 x 400)/60 pixels per second

4) **Assuming that a certain full-color (24 bit per pixel) RGB raster system has a 512 by 512 frame buffer, how many distinct color choices (intensity levels) would be available?**

Ans: for n bit planes, 2^n intensity levels are possible

Primary colors are 3..red, blue, green.

For 24 bit planes, 8 bits for red, 8 bits for blue, 8 bits for green

2^8 intensity levels for red, 2^8 intensity levels for blue, 2^8 intensity levels for green

Total color choices are= $2^8 x 2^8 x 2^8$ = 16, 777, 216 color choices

5) **Suppose an RGB raster system is to be designed using an 8-inch by 10-inch screen with a resolution of 100 pixels per inch in each direction. If we want to store h bit5 per pixel in the frame buffer, how much storage (in bytes) do we need for the frame buffer?**

Ans : storage for frame buffer is:

(((8 x 100) × (10 x 100) × 5) /8) bit

6) **How long would it take to load a 640 by 480 frame buffer with 12 bits per pixel, if 10^5 bits can be transferred per second!**

Ans: time taken to load is X ;

(640 x 480 x 12)=100000 x X.

X=(640 x 480 x 12)/100000 seconds

7) **How long would it take to load a 24-bit per pixel frame buffer with a resolution of 1280 by 1024 using this frame transfer rate?**

Ans: time taken to load is X ;

(1280 x 1024 x 24)=100000 x X.

X=(1280 x 1024 x 24)/100000 seconds

Filled Area Primitives:

Note: For filling a given picture or object with color's, we can do it in two ways in C programming. The two ways are given below:

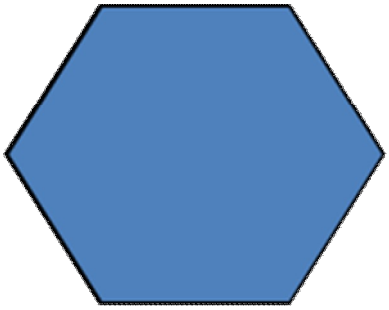
i. Using filling algorithms such as Floodfill algorithm, Boundary fill algorithm and scanline polygon fill algorithm, we can color the objects.
ii. Using inbuilt graphics functions such as floodfill(),setfillstyle() we can fill the object with color's directly without using any filling algorithm.

Here we will see the **filling algorithms**

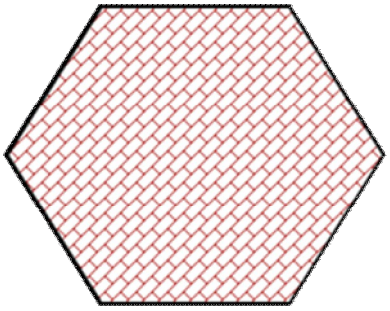
Polygon Filling

Types of filling

- Solid-fill
  All the pixels inside the polygon's boundary are illuminated.



- Pattern-fill
  the polygon is filled with an arbitrary predefined pattern.



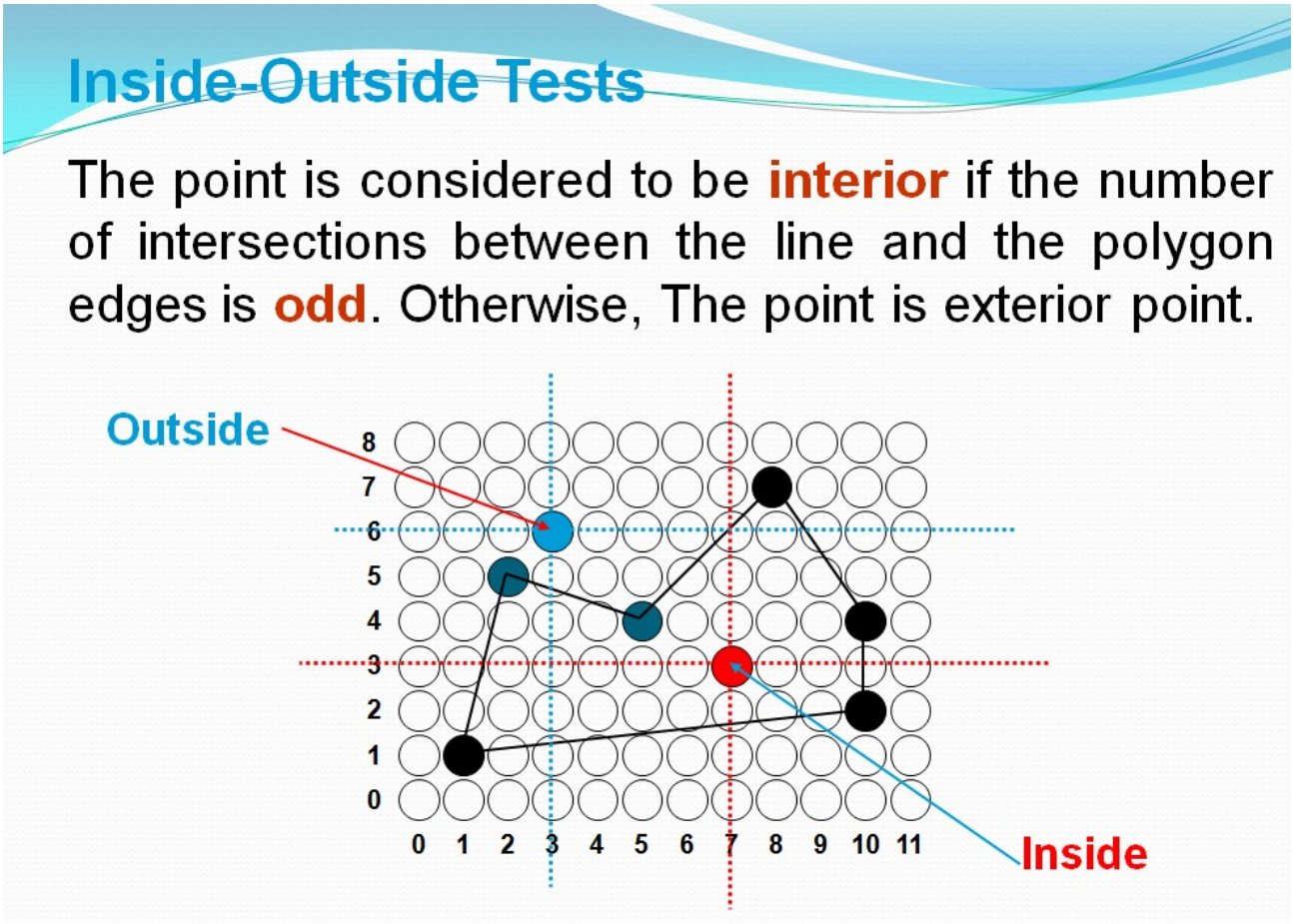
Polygon Representation

- The polygon can be represented by listing its n vertices in an ordered list.
  $P = \{(x_1, y_1), (x_2, y_2), \ldots\ldots, (x_n, y_n)\}$.
- The polygon can be displayed by drawing a line between (x_1, y_1), and (x_2, y_2), then a line between (x_2, y_2), and (x_3, y_3), and so on until the end vertex. In order to close up the polygon, a line between (x_n, y_n), and (x_1, y_1) must be drawn.
- One problem with this representation is that if we wish to translate the polygon, it is necessary to apply the translation transformation to each vertex in order to obtain the translated polygon.

Inside-Outside Tests

- When filling polygons we should decide whether a particular point is interior or exterior to a polygon.
- A rule called the odd-parity (or the odd-even rule) is applied to test whether a point is interior or not.
- To apply this rule, we conceptually draw a line starting from the particular point and extending to a distance point outside the coordinate extends of the object in any direction such that no polygon vertex intersects with the line.

Inside-Outside Tests



The Filling Algorithms are 3:

They are:

Three Algorithms for filling areas:

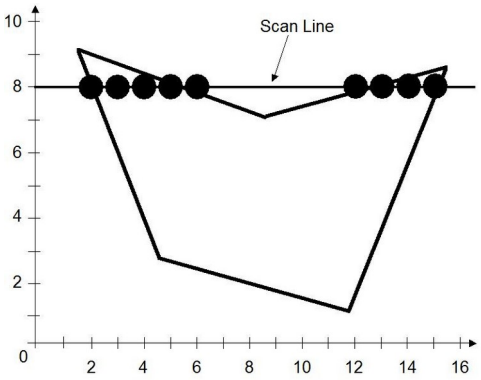
- 1) Scan Line Polygon Fill Algorithm
- 2) Boundary Fill Algorithm
- 3)Flood fill Algorithm

Scan Line Polygon Fill Algorithm:

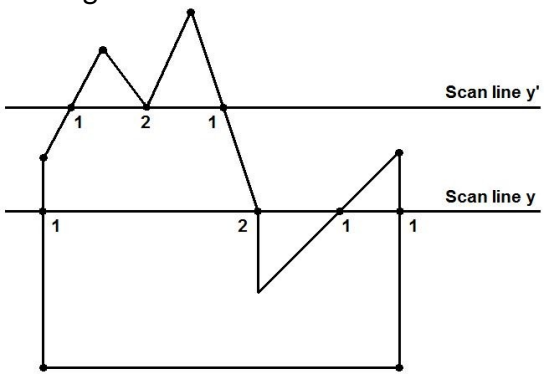
- The basic scan-line algorithm is as follows:
  - Find the intersections of the scan line with all edges of the polygon
  - Sort the intersections by increasing x coordinate
  - Fill in all pixels between pairs of intersections that lie interior to the polygon

The scan-line polygon-filling algorithm involves

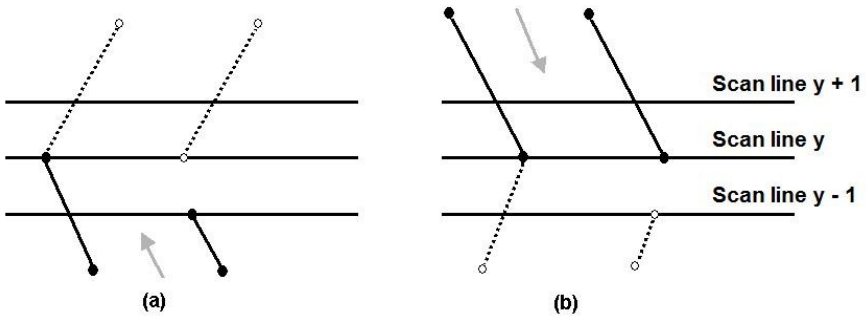
- the **horizontal scanning** of the polygon from its **lowermost** to its **topmost** vertex,
- identifying which edges intersect the scan-line,
- and finally drawing the interior horizontal lines with the specified fill color. process.



Dealing with vertices:



- When the endpoint **y** coordinates of the two edges are **increasing**, the **y** value of the upper endpoint for the **current edge** is decreased by one (a)
- When the endpoint **y** values are **decreasing**, the **y** value of the **next edge** is decreased by one (b)

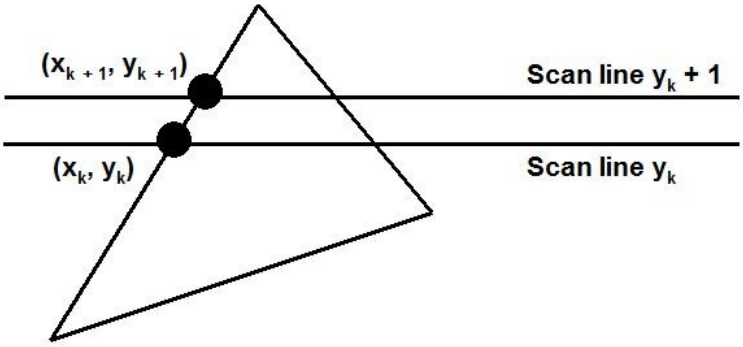


- **Determining Edge Intersections**

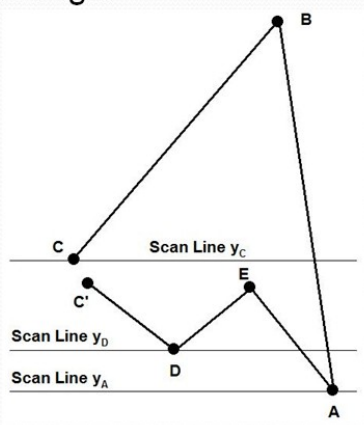
  $m = (y_{k+1} - y_k) / (x_{k+1} - x_k)$

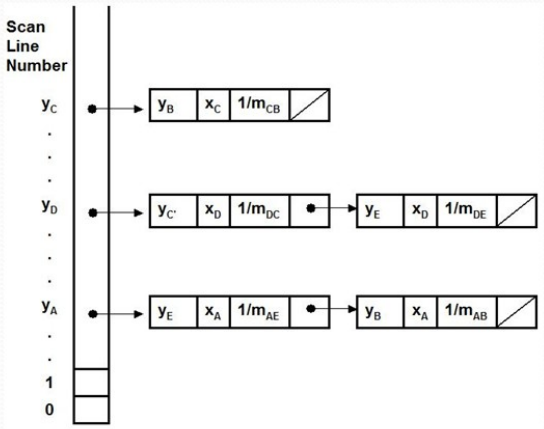
  $y_{k+1} - y_k = 1$

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



• Each **entry** in the table for a particular scan line contains the **maximum y** value for that edge, the **x-intercept** value (**at the lower vertex**) for the edge, and the **inverse slope** of the edge.





Steps in algorithm:

1. Find the minimum enclosed rectangle
2. Here the number of scanlines are equal to (ymax-ymin+1)
3. For each scanline, do

   a. Obtain the intersection points of scanline with polygon edges
   b. Sort the intersection edges from left to right
   c. Form the pairs of intersection points from the obtained list
   d. Fill with colors with in each pair if intersection points
   e. Repeat above procedure until ymax

Algorithm Steps:

1. the horizontal scanning of the polygon from its lowermost to its topmost vertex
2. identify the edge intersections of scan line with polygon
3. Build the edge table
    a. Each entry in the table for a particular scan line contains the maximum y value for that edge, the x-intercept value (at the lower vertex) for the edge, and the inverse slope of the edge.
4. Determine whether any edges need to be splitted or not. If there is need to split, split the edges.
5. Add new edges and build modified edge table.
6. Build Active edge table for each scan line and fill the polygon based on intersection of scanline with polygon edges.

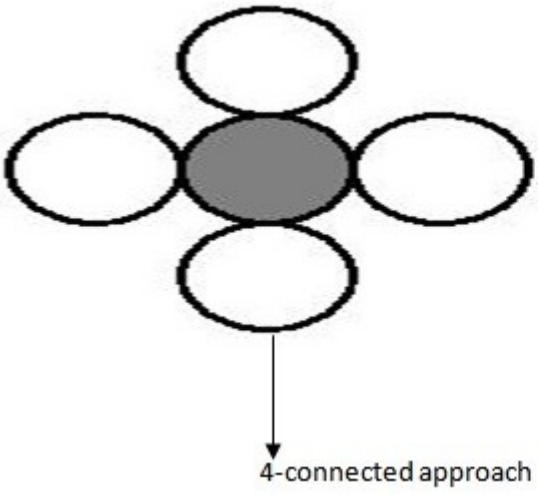
Boundary Fill Algorithm:

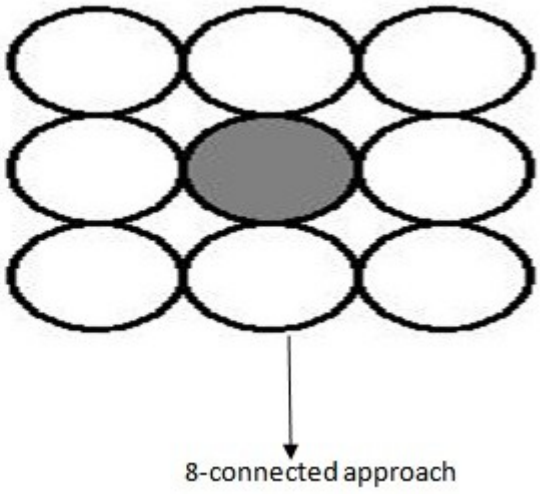
- Start at a point inside a region and paint the interior outward toward the boundary.
- If the boundary is specified in a single color, the fill algorithm processed outward pixel by pixel until the boundary color is encountered.
- A boundary-fill procedure accepts as input the coordinate of the interior point (x, y), a fill color, and a boundary color.

Algorithm:

The following steps illustrate the idea of the recursive boundary-fill algorithm:

1. Start from an interior point.
2. If the current pixel is not already filled and if it is not an edge point, then set the pixel with the fill color, and store its neighboring pixels (**4 or 8-connected**). Store only neighboring pixel that is not already filled and is not an edge point.
3. Select the next pixel from the stack, and continue with step 2.





In 4 connected approach, we can fill an object in only 4 directions. We have 4 possibilities for proceeding to next pixel from current pixel.

In 8 connected approach, we can fill an object in 8 directions. We have 8 possibilities for proceeding to next pixel from current pixel.

Function for 4 connected approach:

```
void boundary_fill(int x, int y, int fcolor, int bcolor)
{
```

```
   if ((getpixel(x, y) != bcolor) && (getpixel(x, y) != fcolor))
    {                delay(10);
                     putpixel(x, y, fcolor);
                     boundary_fill(x + 1, y, fcolor, bcolor);
                     boundary_fill(x - 1, y, fcolor, bcolor);
                     boundary_fill(x, y + 1, fcolor, bcolor);
                     boundary_fill(x, y - 1, fcolor, bcolor);
   }
}
```

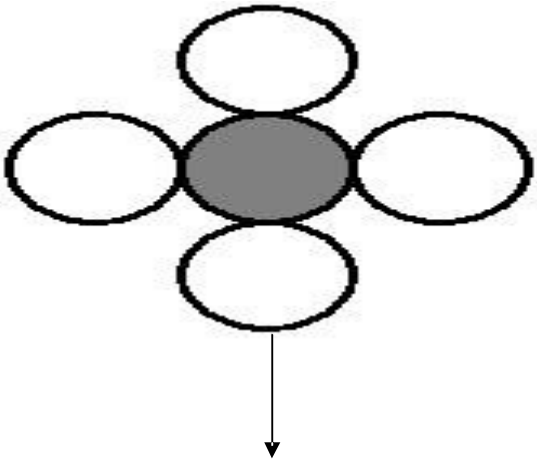
Function for 8 connected approach:

```
void boundary_fill(int x, int y, int fcolor, int bcolor)
{

   if ((getpixel(x, y) != bcolor) && (getpixel(x, y) != fcolor))
    {                        delay(10);
          putpixel(x, y, fcolor);
          boundary_fill(x + 1, y, fcolor, bcolor);
          boundary_fill(x , y+1, fcolor, bcolor);
          boundary_fill(x+1, y + 1, fcolor, bcolor);
          boundary_fill(x-1, y - 1, fcolor, bcolor);
          boundary_fill(x-1, y, fcolor, bcolor);
          boundary_fill(x , y-1, fcolor, bcolor);
          boundary_fill(x-1, y + 1, fcolor, bcolor);
          boundary_fill(x+1, y - 1, fcolor, bcolor);

     }
}
```

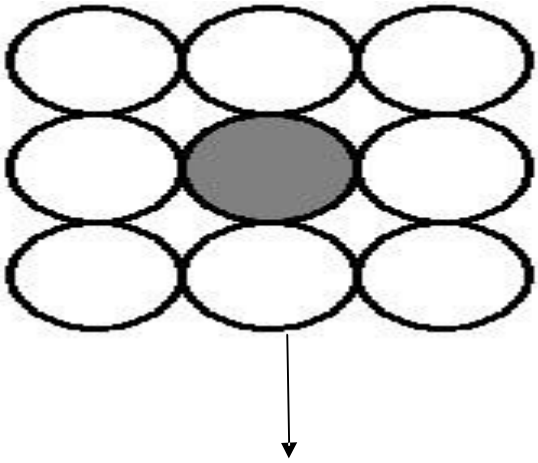
Flood Fill Algorithm:

Sometimes we want to fill in (recolor) an area that is not defined within a single color boundary. We paint such areas by replacing a specified interior color instead of searching for a boundary color value. This approach is called a flood-fill algorithm.

1. We start from a specified interior pixel (x, y) and reassign all pixel values that are currently set to a given interior color with the desired fill color.
2. If the area has more than one interior color, we can first reassign pixel values so that all interior pixels have the same color.
3. Using either 4-connected or 8-connected approach, we then step through pixel positions until all interior pixels have been repainted.



4-connected approach



8-connected approach

4-connected Flood Fill approach:

1. We can implement flood fill algorithm by using recursion.
2. First all the pixels should be reassigned to common color. here common color is black.
3. Start with a point inside given object, check the following condition:

   if(getpixel(x,y)==old_col)---old_col is common color
4. If above condition is satisfied, then following 4 steps are followed in filling the object.

   //Recursive 4-way floodfill.

```
putpixel(x,y,fill_col);
flood(x+1,y,fill_col,old_col);
flood(x-1,y,fill_col,old_col);
flood(x,y+1,fill_col,old_col);
flood(x,y-1,fill_col,old_col);
```

8-connected Flood fill approach:

1. We can implement flood fill algorithm by using recursion.
2. First all the pixels should be reassigned to common color. here common color is black.
3. Start with a point inside given object, check the following condition:

   if(getpixel(x,y)==old_col)---old_col is common color
4. If above condition is satisfied, then following 8 steps are followed in filling the object.

   //Recursive 4-way floodfill.

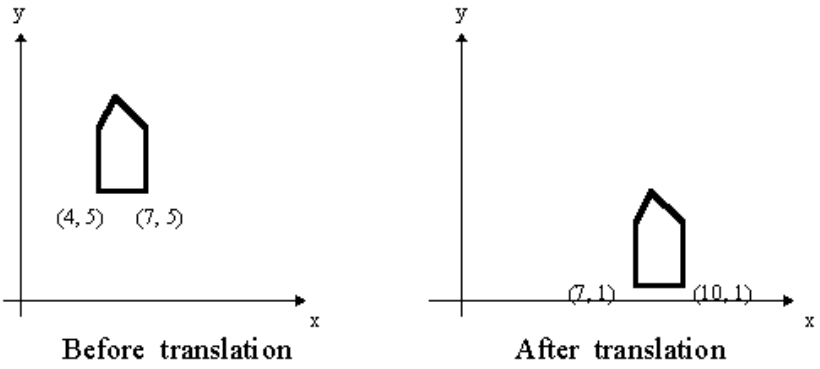
```
putpixel(x,y,fill_col);
flood(x+1,y,fill_col,old_col);
flood(x-1,y,fill_col,old_col);
flood(x,y+1,fill_col,old_col);
flood(x,y-1,fill_col,old_col);
flood(x + 1, y - 1, fill_col, old_col);
flood(x + 1, y + 1, fill_col, old_col);
flood(x - 1, y - 1, fill_col, old_col);
flood(x - 1, y + 1, fill_col, old_col);
```

2D Transformations

The 2D transformations are:

1. Translation
2. Scaling
3. Rotation
4. Reflection
5. Shear

1. **Translation:** Translation is defined as moving the object from one position to another position along straight line path.



We can move the objects based on translation distances along x and y axis. tx denotes translation distance along x-axis and ty denotes translation distance along y axis.

Translation Distance: It is nothing but by how much units we should shift the object from one location to another along x, y-axis.

Consider (x,y) are old coordinates of a point. Then the new coordinates of that same point (x',y') can be obtained as follows:

X'=x+tx

Y'=y+ty

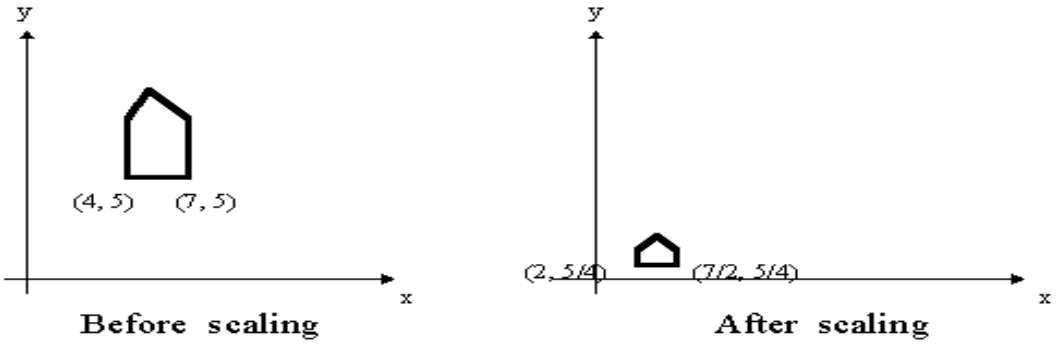
We denote translation transformation as P. we express above equations in matrix form as:

$$P' = P + T$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

x,y---old coordinates
x',y'—new coordinates after translation
tx,ty—translation distances, T is translation matrix

2. **Scaling:** scaling refers to changing the size of the object either by increasing or decreasing. We will increase or decrease the size of the object based on scaling factors along x and y-axis.



If (x, y) are old coordinates of object, then new coordinates of object after applying scaling transformation are obtained as:

x'=x*sx

y'=y*sy.

sx and sy are scaling factors along x-axis and y-axis. we express the above equations in matrix form as:

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

Scaling Matrix

3. **Rotation:** A rotation repositions all points in an object along a circular path in the plane centered at the pivot point. We rotate an object by an angle theta.

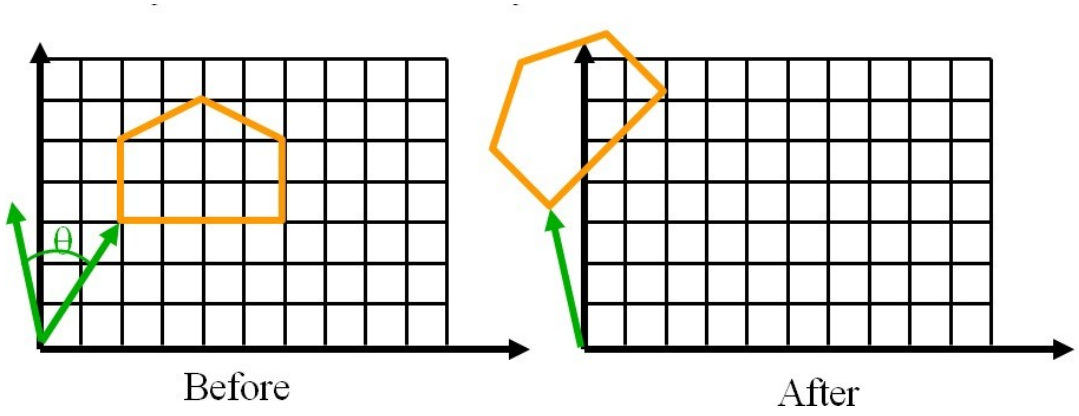
New coordinates after rotation depend on *both x* and *y*

- $x' = x\cos\theta - y\sin\theta$
- $y' = x\sin\theta + y\cos\theta$
- or in matrix form:

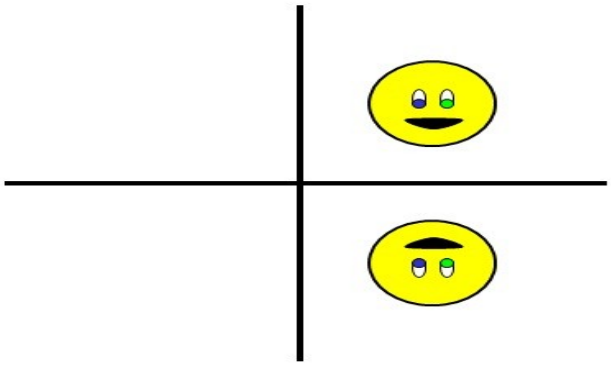
P' = R • P,

R-rotation matrix.

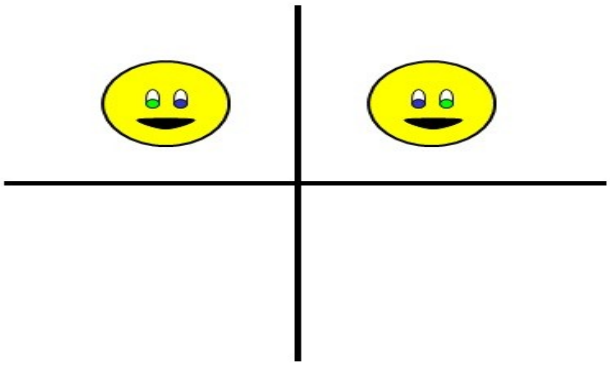
$$R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$



4. **Reflection:** Reflection is nothing but producing mirror image of an object. Reflection can be done just by rotating the object about given axis of reflection with an angle of 180 degrees.

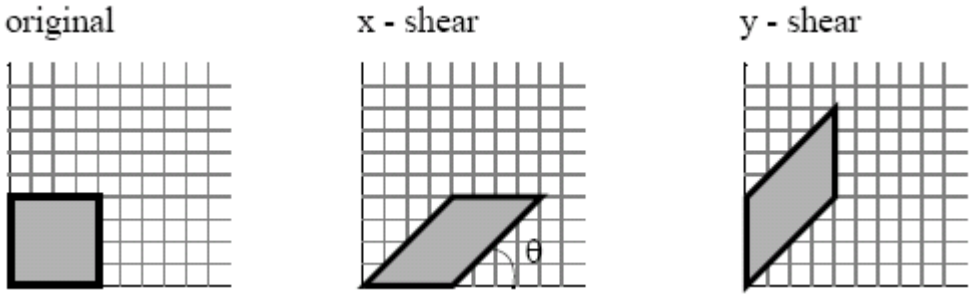


X-axis reflection



Y-axis reflection

5. **Shear:**
    1. Shear is the translation along an axis by an amount that increases linearly with another axis (Y). It produces shape distortions as if objects were composed of layers that are caused to slide over each other.
    2. Shear transformations are very useful in creating italic letters and slanted letters from regular letters.
    3. Shear transformation changes the shape of the object to a slant position.



    4. Shear transformation is of 2 types:
        a. X-shear: changing x-coordinate value and keeping y constant

            x'=x+shx*y

            y'=y

        b. Y-shear: changing y coordinates value and keeping x constant

            x'=x

            y'=y+shy*x

            shx and shy are shear factors along x and y-axis.

Matrix representation of Transformations

The matrix representation of different transformations are given below

Translation

$$P = \begin{bmatrix} x \\ y \end{bmatrix}$$

$$P' = \begin{bmatrix} x' \\ y' \end{bmatrix} and\ T = \begin{bmatrix} d_x \\ d_y \end{bmatrix}$$

$$so\ \ P' = P + T$$

Scaling

$$x' = s_x x$$

$$y' = s_y y$$

$$which\ \ has\ \ a\ \ matrix\ \ form:$$

$$P' = \begin{bmatrix} x' \\ y' \end{bmatrix},\ S = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix},\ P = \begin{bmatrix} x \\ y \end{bmatrix}$$

Rotation

$$P' = RP\ \ where$$

$$P' = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

$$P = \begin{bmatrix} x \\ y \end{bmatrix}$$

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

Homogeneous Coordinates:

Homogeneous coordinates are another way to represent points to simplify the way in which we express affine transformations. Normally, bookkeeping would become tedious when affine transformations of the form $A\bar{p} + \vec{t}$ are composed. With homogeneous coordinates, affine transformations become matrices, and composition of transformations is as simple as matrix multiplication. In future sections of the course we exploit this in much more powerful ways.

With homogeneous coordinates, a point $\bar{p}$ is augmented with a 1, to form $\hat{p} = \begin{bmatrix} \bar{p} \\ 1 \end{bmatrix}$.

All points $(\alpha\bar{p}, \alpha)$ represent the same point $\bar{p}$ for real $\alpha \neq 0$.

Given $\hat{p}$ in homogeneous coordinates, to get $\bar{p}$, we divide $\hat{p}$ by its last component and discard the last component.

Example:
The homogeneous points $(2, 4, 2)$ and $(1, 2, 1)$ both represent the Cartesian point $(1, 2)$. It's the orientation of $\hat{p}$ that matters, not its length.

Many transformations become linear in homogeneous coordinates, including affine transformations:

$$\begin{aligned} \begin{bmatrix} q_x \\ q_y \end{bmatrix} &= \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \\ &= \begin{bmatrix} a & b & t_x \\ c & d & t_y \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} A & \vec{t} \end{bmatrix} \hat{p} \end{aligned}$$

USES and ABUSES of homogeneous coordinates:

Homogeneous coordinates provide a different representation for Cartesian coordinates, and cannot be treated in quite the same way. For example, consider the midpoint between two points $\bar{p}_1 = (1, 1)$ and $\bar{p}_2 = (5, 5)$. The midpoint is $(\bar{p}_1 + \bar{p}_2)/2 = (3, 3)$. We can represent these points in homogeneous coordinates as $\hat{p}_1 = (1, 1, 1)$ and $\hat{p}_2 = (5, 5, 1)$. Directly applying the same computation as above gives the same resulting point: $(3, 3, 1)$. However, we can *also* represent these points as $\hat{p}'_1 = (2, 2, 2)$ and $\hat{p}'_2 = (5, 5, 1)$. We then have $(\hat{p}'_1 + \hat{p}'_2)/2 = (7/2, 7/2, 3/2)$, which cooresponds to the Cartesian point $(7/3, 7/3)$. This is a different point, and illustrates that we cannot blindly apply geometric operations to homogeneous coordinates. The simplest solution is to **always convert homogeneous coordinates to Cartesian coordinates. That said, there are several important operations that can be performed correctly in terms of homogeneous coordinates, as follows.**

Note: The above points regarding homogeneous coordinates Referred from David Fleet and Aaron Hertzmann Notes

Composite Transformations:
These are nothing but a sequence of any transformations. We obtain composite matrix just by multiplying the transformation matrices of 2 or more transformations in sequence.

The composite transformations examples are:

1. Fixed point scaling
2. Pivot point rotation

Note: we can express shear and reflection transformations also as composite transformations.

Fixed point scaling:

It is nothing but applying scaling transformations on a object with respect to a fixed point.
Thus the sequence of transformations needed to do fixed point scaling are:

1. Translate the object so that the fixed point position is moved to the coordinate origin
2. scale the object about the coordinate origin
3. Inverse Translation the object so that fixed point returns to its original position

The composite matrix for above fixed point scaling is obtained as product of transformation matrices of individual transformations.

Composite matrix for Fixed point scaling= (**translation matrix** * **scaling matrix** * **inverse translation matrix**)

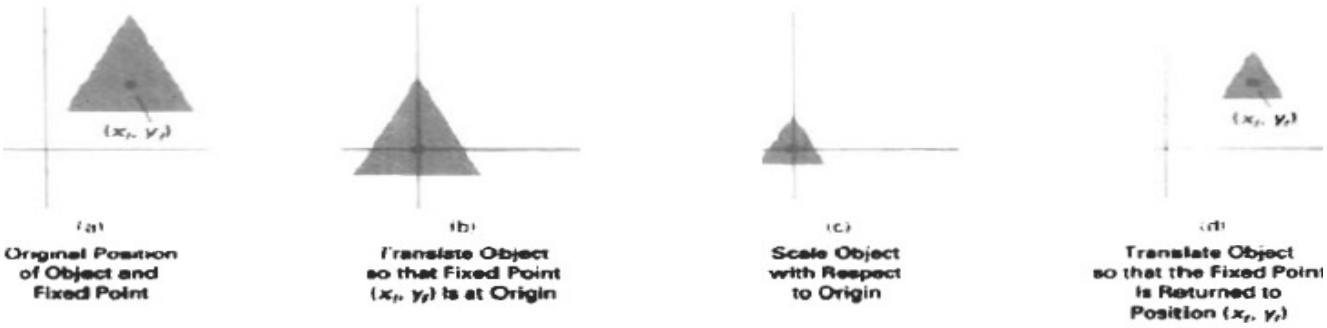


Fig: Fixed point scaling

The composite matrix is obtained as follows:

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

$$\begin{bmatrix} 1 & 0 & x_f \\ 0 & 1 & y_f \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & -x_f \\ 0 & 1 & -y_f \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & x_f(1-s_x) \\ 0 & s_y & y_f(1-s_y) \\ 0 & 0 & 1 \end{bmatrix} \quad (5\text{-}33)$$

or

$$\mathbf{T}(x_f, y_f) \cdot \mathbf{S}(s_x, s_y) \cdot \mathbf{T}(-x_f, -y_f) = \mathbf{S}(x_f, y_f, s_x, s_y) \quad (5\text{-}34)$$

Pivot point rotation:

It is nothing but applying rotation transformations on a object with respect to a fixed or pivot point.
Thus the sequence of transformations needed to do pivot point rotation are:

4. Translate the object so that the pivot point position is moved to the coordinate origin
5. Rotate the object about the coordinate origin
6. Inverse Translation the object so that pivot point returns to its original position

The composite matrix for above pivot point rotation is obtained as product of transformation matrices of individual transformations.

Composite matrix for pivot point rotation= (**translation matrix** * **rotation matrix** * **inverse translation matrix**)

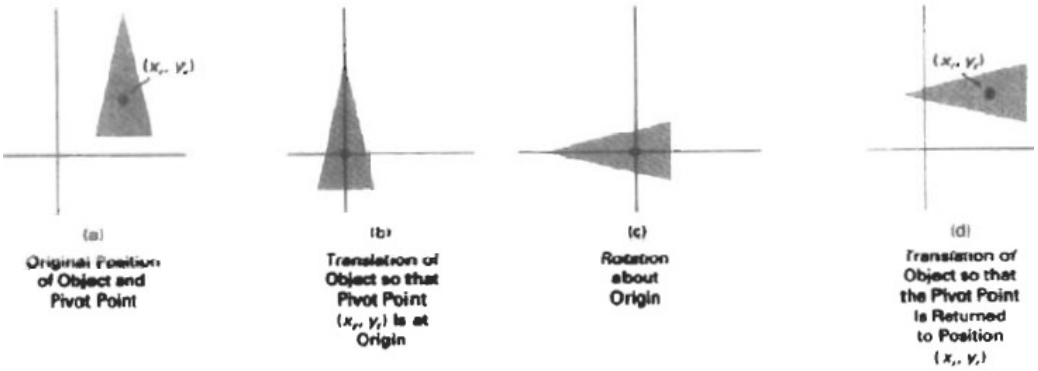


Fig: pivot point rotation

The composite matrix is obtained as follows:

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

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

which can be expressed in the form

$$\mathbf{T}(x_r, y_r) \cdot \mathbf{R}(\theta) \cdot \mathbf{T}(-x_r, -y_r) = \mathbf{R}(x_r, y_r, \theta)$$

3D Transformations

The 3D transformations are:

Translation

Scaling

Rotation

Translation: Translation is defined as moving the object from one position to another position along straight line path. We can move the objects based on translation distances along x and y axis. tx denotes translation distance along x-axis and ty denotes translation distance along y axis.

Translation Distance: It is nothing but by how much units we should shift the object from one location to another along x, y-axis.

Consider (x,y) are old coordinates of a point. Then the new coordinates of that same point (x',y') can be obtained as follows:

X'=x+tx

Y'=y+ty

Z'=z+tz

We denote translation transformation as P.

2. Scaling: scaling refers to changing the size of the object either by increasing or decreasing. We will increase or decrease the size of the object based on scaling factors along x and y-axis.

If (x, y) are old coordinates of object, then new coordinates of object after applying scaling transformation are obtained as:

x'=x*sx

y'=y*sy.

Z'=z*sz.

sx ,sy and sz are scaling factors along x-axis, y-axis and z-axis. we express the above equations in matrix form as:

3. **Rotation:** A rotation repositions all points in an object along a circular path in the plane centered at the pivot point.

We rotate an object by an angle theta.

The Transformation matrices for above 3D transformations are given below:

X-Rotation in 3D

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi & 0 \\ 0 & \sin\phi & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Z-Rotation in 3D

$$\begin{bmatrix} \cos\phi & -\sin\phi & 0 & 0 \\ \sin\phi & \cos\phi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Scale in 3D

$$\begin{bmatrix} Sx & 0 & 0 & 0 \\ 0 & Sy & 0 & 0 \\ 0 & 0 & Sz & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(4x4)*(4x1) = (4x1)

Y-Rotation in 3D

$$\begin{bmatrix} \cos\phi & 0 & \sin\phi & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\phi & 0 & \cos\phi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Translation in 3D

$$\begin{bmatrix} 1 & 0 & 0 & Tx \\ 0 & 1 & 0 & Ty \\ 0 & 0 & 1 & Tz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Matrix Multiplication

$$\begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \\ q \end{bmatrix}$$

Reflection: Reflection is nothing but producing mirror image of an object. Reflection can be done just by rotating the object about given axis of reflection with an angle of 180 degrees.

The matrix representation for this reflection of points relative to the **xy** plane is

$$RF_z = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Shear:

1. Shear is the translation along an axis by an amount that increases linearly with another axis (Y). It produces shape distortions as if objects were composed of layers that are caused to slide over each other.
2. Shear transformations are very useful in creating italic letters and slanted letters from regular letters.
3. Shear transformation changes the shape of the object to a slant position
4. Shearing transformations can he used to modify ohject shapes. They are also useful in three-dimensional viewing for obtaining general projection transformations. In three dimensions, we can also generate shears relative to the *z* axis.As an example of three-dimensional shearing. the following transformation produces a z-axis shear:

$$SH_z = \begin{bmatrix} 1 & 0 & a & 0 \\ 0 & 1 & b & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Parameters a and *b* can be assigned any real values. The effect of this transformation matrix is to alter x- and y-coordinate values hy an amount that is proportional to the 2 value, while leaving the *z* coordinate unchanged. Boundaries of planes that are perpendicular to the *z* axis are thus shifted by an amount proportional to *z*.

Composite Transformations:
These are nothing but a sequence of any transformations. We obtain composite matrix just by multiplying the transformation matrices of 2 or more transformations in sequence.

The composite transformations examples are:

1. Fixed point scaling
2. Pivot point rotation

Note: we can express shear and reflection transformations also as composite transformations.

Fixed point scaling:

It is nothing but applying scaling transformations on a object with respect to a fixed point.
Thus the sequence of transformations needed to do fixed point scaling are:

1. Translate the object so that the fixed point position is moved to the coordinate origin
2. scale the object about the coordinate origin
3. Inverse Translation the object so that fixed point returns to its original position

The composite matrix for above fixed point scaling is obtained as product of transformation matrices of individual transformations.

Composite matrix for Fixed point scaling= (**translation matrix** * **scaling matrix** * **inverse translation matrix**)

Pivot point rotation:

It is nothing but applying rotation transformations on a object with respect to a fixed or pivot point.
Thus the sequence of transformations needed to do pivot point rotation are:

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. Inverse Translation the object so that pivot point returns to its original position

The composite matrix for above pivot point rotation is obtained as product of transformation matrices of individual transformations.

Composite matrix for pivot point rotation= (**translation matrix** * **rotation matrix** * **inverse translation matrix**)

Computer Animation:

- Computer Animation refers to any time sequence of visual changes in a scene.
- A computer generated animation scene could display time variations in object size, color, transparency or surface texture.
- Advertising animations –morphing
- Computer animations can be generated by changing camera parameters such as position, orientation and focal length.
- Visual Effects—producing exaggerated shapes and unrealistic motions and transformations

Steps for Design of Animation Sequences:

1. Storyboard layout
2. Object definition
3. Key - frame specifications
4. Generation of in- between frames

The ***storyboard*** is an outline of the action. It defines the motion sequence as a set of basic events that are to take place .Depending on the type of animation to be produced, the storyboard could consist of a set of rough sketches or it could be a list of the basic ideas for the motion.

An ***object definition*** is given for each participant in the action. Object can be defined in terms of basic shapes, such as polygons or splines In addition, the associated movements for each object are specified along with shape.

A ***key*** **frame** is detailed drawing of the scene at a certain time in the animation sequence. Within each key frame, each object is positioned according to the time for that frame. Some key frames are chosen at extreme positions in the action; others are spaced so that the time interval between key frames is not too great.

In-*between*s are the intermediate frames between the key frames. The number of in - betweens needed is determined by the media to be used to display the animation .Film requires 24 frames per second, and graphics terminals are refreshed at the rate of 30 to 60 frames per second .

General Computer Animation Functions:

1. store and manage the object database
2. functions for motion generation and for object rendering
3. functions that simulates camera movements
4. zooming, panning, and tilting

Some steps in development of animation sequence are well - suited to computer solution. These include object manipulation and rendering camera motions, and the generation of in - betweens. One function available in animation packages is provided to **store and manage the object database.** Object shapes and associated parameters are stores and updated in the database. Other object functions include those for **motion generation** and those for **object rendering**. Motion

can be generated according to specified constraints using two dimensional or three dimensional transformations.
Another typical function **simulates camera movements**. Standard motions are **zooming, panning, and tilting.**

Computer Animation Languages:

1. C
2. Lisp
3. Pascal
4. FORTRAN
5. Special animation packages like Adobe Flash , autodesk maya etc.

Design and control of animation sequences are handled with a set of animation outlines. A general purpose language, such as **C, Lisp, Pascal or FORTRAN** is often used to program the animation functions, but several **specialized animation languages** have been developed. Animation functions include a **graphics editor**, **a key frame generator, an in-between generator and standard graphics routines.** The graphics editor allows us to design and modify object shapes, using spline surface, 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, and setting the camera parameters (position, orientation, and less characteristics). Another standard function is **action specification**. This involves the layout of motion paths for the object and camera. And we need the usual graphics routines: viewing and perspective transformations, geometric transformations to generate object movements as a function of accelerations or kinematic path specification, visible-surface identification and the surface rendering operations.

key-frame systems are specialized animation languages designed simply to generate the in betweens 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 a limited number of degrees of freedom.

Parameterized systems allow object motion characteristics to be specified as part of the object definitions. The adjustable parameters control such object characteristics as degree of freedom, motion limitations, and allowable shape changes.

Scripting systems allows object specifications and animation sequences to be define with a user-input script. From the script, a library of various objects and motions can be constructed.

Raster Animations:

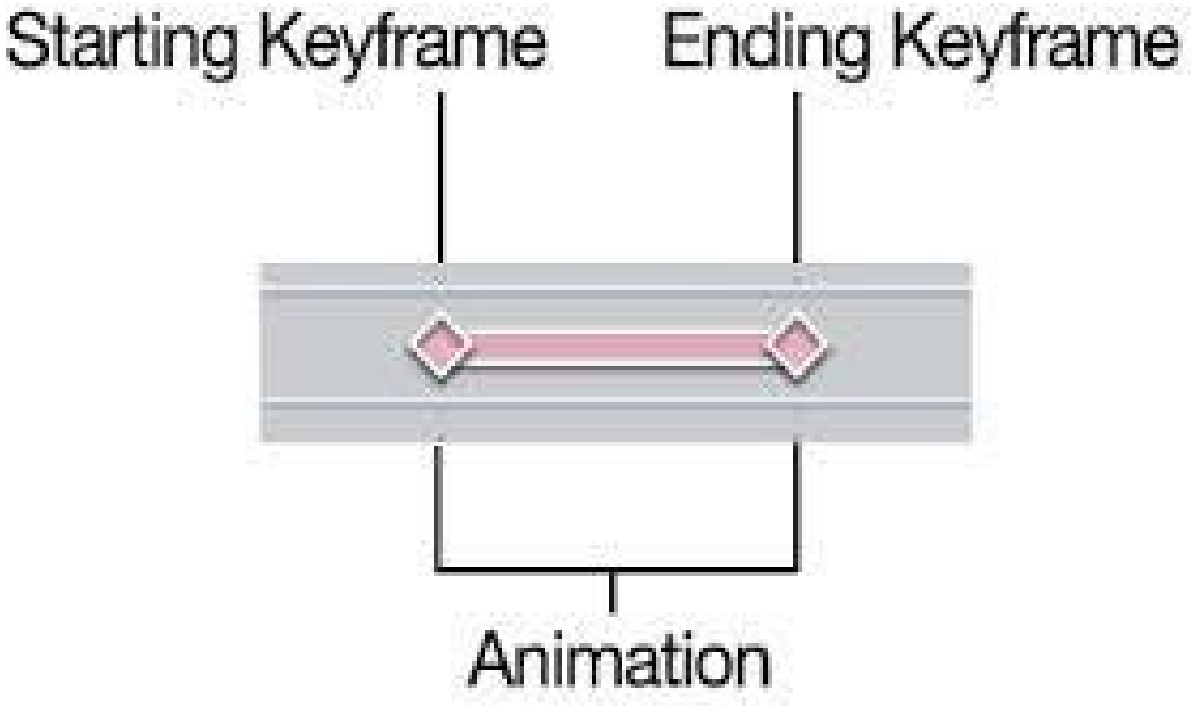
On raster scan systems we can generate real time animation in limited applications using **raster operations**. A simple method for translation in the xy plane is to transfer a rectangular blocks of pixels through arbitrary angles using anti-aliasing procedures. To rotate a block of pixels we need to determine the percent of area coverage for those pixels that overlap the rotated block. Sequences of raster operations can be executed to produce real time animation of either two-dimensional or three-dimensional objects, as long as we restrict the animation to motions in the projection plane. Then no viewing or visible surface algorithms need be invoked.

Color table transformations also produce animation. Just by changing the intensity values of pixels to on and off, we can produce animation on raster systems.

Key Frame Systems:

- Generate set of in-betweens from specification of two key frames
- Motion paths can be given with kinematic or dynamic descriptions
- Cels
- Animation paths—interpolation
- Morphing(transforming shape from one form to another form).
- Linear Interpolation
- Simulating Accelerations

A key frame is a detailed drawing of the scene at a certain time in the animation sequence. Within each key frame each object is positioned according to the time for that frame.In-betweens are intermediate frames between the key frames.

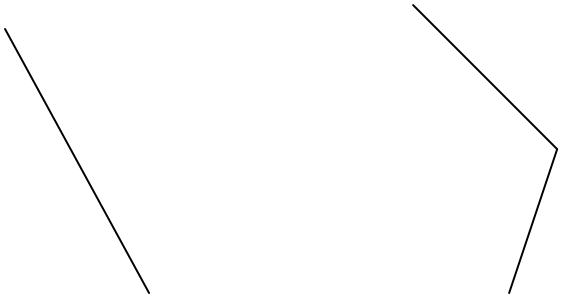


Morphing

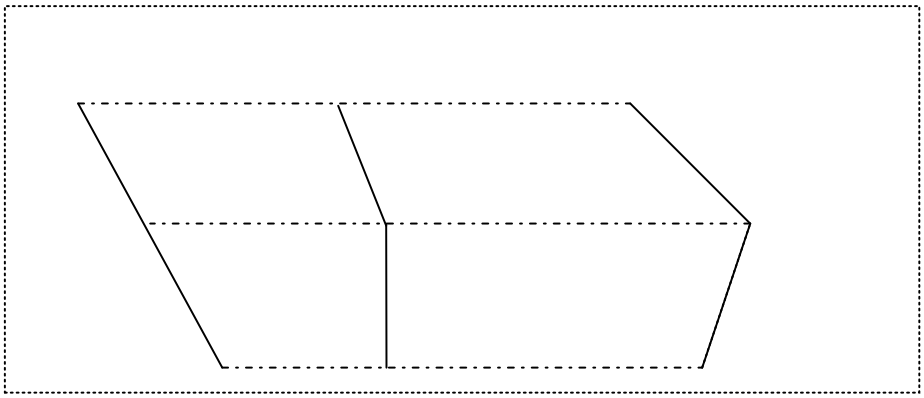
Transformation of object shapes from one form to another is called morphing. We generate set of in-betweens from the specification of two or more key frames. Given the animation paths we can interpolate the positions of individual objects between any two times or key frames. With complex object transformations the shapes of the object may change over time. If all surfaces are described with polygon meshes then the number of edges per polygon can change from one frame to the next. Thus the total number of line segments can be different in different frames.

Transformation of object shapes from one form to another is called morphing. Morphing methods can be applied to any motion or transition involving a change of shape.

Given two key frames for an object transformation we first adjust the object specification in one if the frames so that the number of polygon edges (or the number of vertices) is the same for the two frames. This is illustrated below



Frame K Frame K+1



Frame K Halfway Frame Frame K+1

A straight-line segment in key frame k is transformed into two line segments in key frame k+1. Since key frame k+1 has an extra vertex, we add a vertex between 1 and 2 in key frame K to balance the number of vertices and edges in the two key frames. Using linear interpolation to generate the in betweens we transition the added vertex in key frame k into vertex 3’ along the straight-line path as shown.We can state general preprocessing rules for equalizing key frames in terms of either the number of edges or the number of vertices to be added to a key frame.

Case 1: Equalizing edge count.

Let the parameters L_k and L_{k+1} denote the number of line segments in two consecutive frames. We then define

$$L_{max} = \max(L_k, L_{k+1})$$

$$L_{min} = \min(L_k, L_{k+1})$$

$$N_e = L_{max} \bmod L_{min}$$

$$N_s = \text{int}(L_{max} / L_{min})$$

Then the preprocessing is accomplished by

1. Dividing the N_e edges of keyframe_{min} into N_s sections.
2. Dividing the remaining lines of keyframe_{min} into N_s sections

Case 2: Equalizing vertex count

Let the parameters be V_k and V_{k+1} denote the number of vertices in the two consecutive frames. We define

$$V_{max} = \max(V_k, V_{k+1})$$

$$V_{min} = \min(V_k, V_{k+1})$$

$$N_{ls} = (V_{max}-1) \bmod (V_{min}-1)$$

$$N_p = \text{int}((V_{max}-1) / (V_{min}-1))$$

Preprocessing using vertex count is performed by

1. Adding N_p points to N_{ls} line sections of keyframe_{min}

Adding N_p-1 points to the remaining edges of keyframe_{min}

Motion Specification in Computer Animation

There are several ways in which the motions of objects can be specified in an animation system. We can define motion in very explicit terms or We can use more abstract or more general approaches.

Ways for defining motion of objects are:

- **Direct motion specification**
- **Goal-directed systems**
- **Kinematics, Inverse kinematics and dynamics, Inverse Dynamics**

Direct motion specification : -

The most straightforward method for defining a motion sequence is direct specification of the motion paremeters. Here, We explicitly give the rotation angles and translation vectors. Then the geometric transformation matrices are applied to transform co-ordinate positions. Alternatively, We could use an approximating equation to specify certain kinds of motions. These methods can be used for simple user programmed animation sequencs.

Goal-directed systems : -

At the opposite extreme, We can specify the motions that are to take place in general terms that abstractly describe the actions. these systems are referred to as goal directed because they determine specific motion parameters given the goals of the animation. For example, We could specify that we want an object to "walk " or to "run" to a particular destination. Or We could state that we want an object to "pick up " some other specified object. The inpute directive are then interpreted in term of component motions that will accomplish the selected task. Human motion, for instance, can be defined as a heirarchical structure of sub motion for the toros, limbs,and so forth.

Kinematics and dynamics : -

We can also construct animation sequences using kinematic or dynemic descriptions. With a kinematic description, we specify the animation by giving motion parameters position, velocity, and acceleration) without reference to the forces that cause the motion. for constant velocity (zero acceleration), we designate the motions of rigid bodies in a scene by giving an initial position and velocity vector for each objects.

An alternate apporach is to use inverse kinematics. Here, we specify the initial and final positions of objects at specified times and the motion parameters are computed by the system . For example, assuming zero acceleration , we can determine the constant velocity that will accomplish the movement of an object from the initial position to the final position.

Dynamic descriptions on the other hand, require the specification of the forces that produce the velocities and acceleration. Descriptions of object behavior under the are generally referred to as a physically based modeling. Example of forces affecting object motion include electromagnetic,

gravitational, friction, and other mechanical forces.

Object motion are obtained from the forces equations describing physical laws, such as newton's law of motion for gravitational ang friction processes, euler or navier-stokes equations describing fluid flow, and maxwell 's equations for electromagnetic forces. For example, the general form os newton's second law for a particle of mass m is
$F = d(mv)/dt$

with F as the force vector, and v as the velocity vector. If mass is constant, we solve the equation F=ma, where a is the acceleration vector. otherwise, mass is a function of time, as in relativistic motions of space vehicles that consume measurable amounts of fuel per unit time. We can also use inverse dynamics to obtain the forces, given the initial and final positions of objects and the type of motion.

Application of physically based modeling include complex rigid-body systems and such non rigid systems as cloth and plastic materials. Typically, numerical methods are used to obtain the motion parameters incrementally from the dynamical equations using initial conditions or boundary values.