Module- 1 Part 1

INTRODUCTION TO COMPUTER GRAPHICS

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.

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:

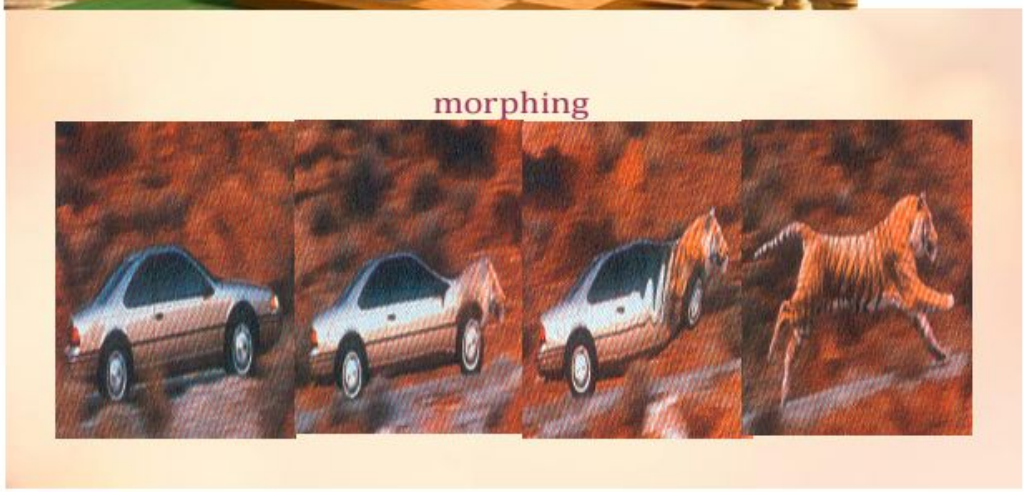
- Computer graphics methods are now commonly used in
  - making motion pictures,
  - music videos,
  - Television shows.
  - Animations

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

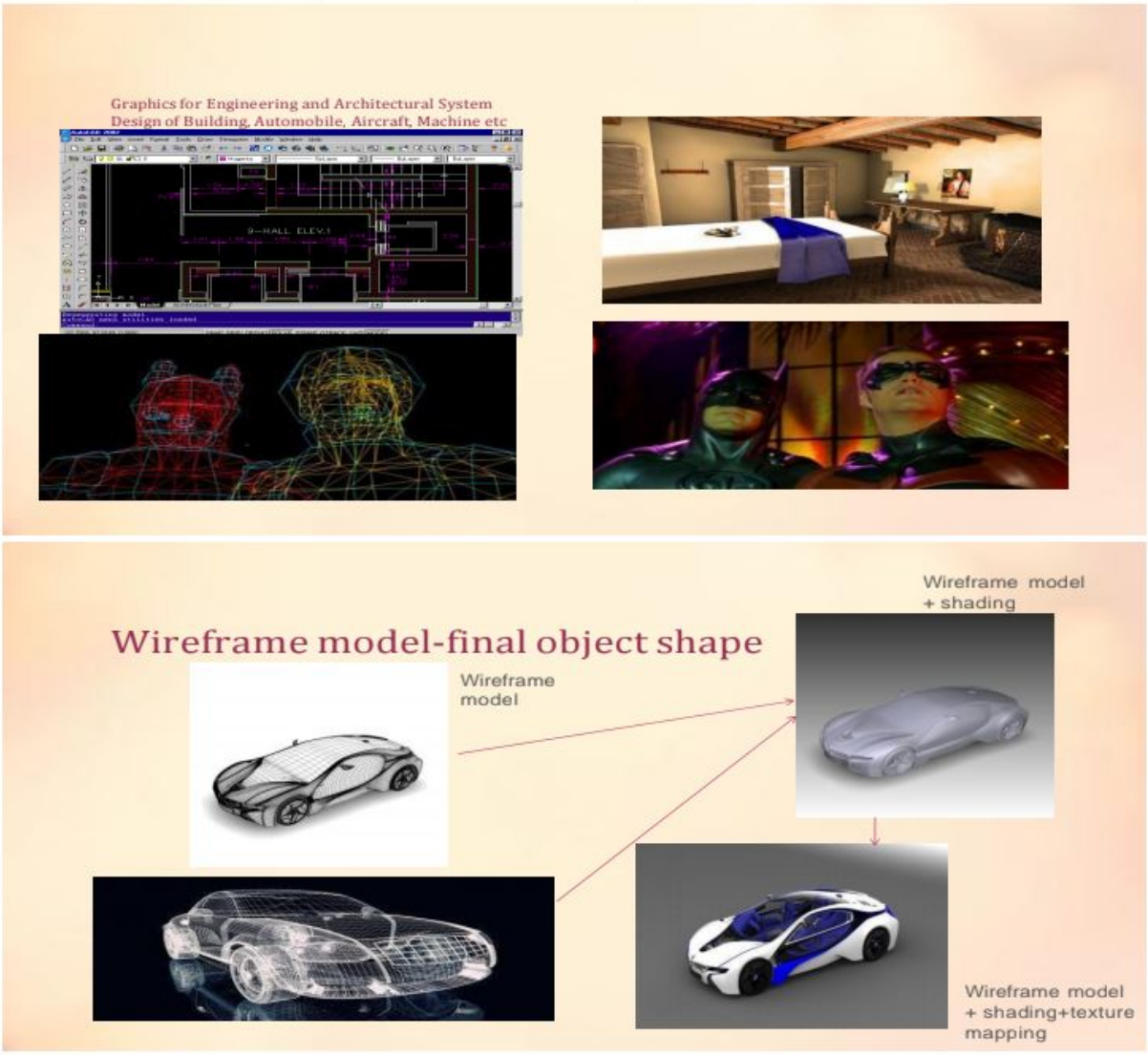


Figure: designs that can be created using CAD tool

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:

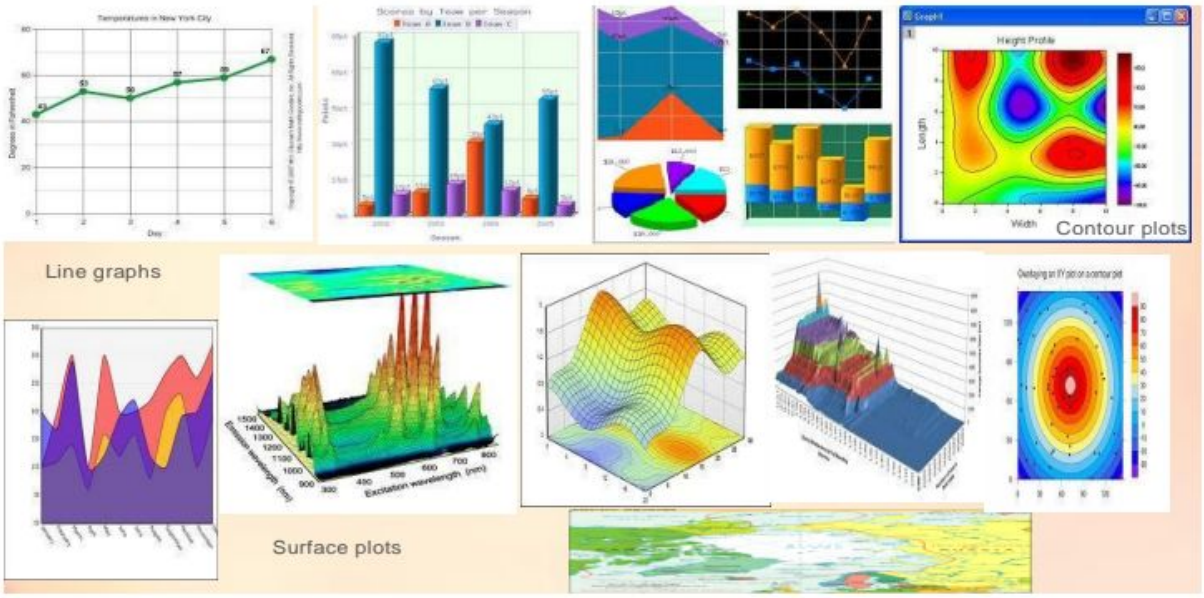
-line graphs

-bar charts

-pie charts

-surface graphs

-contour plots



5

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

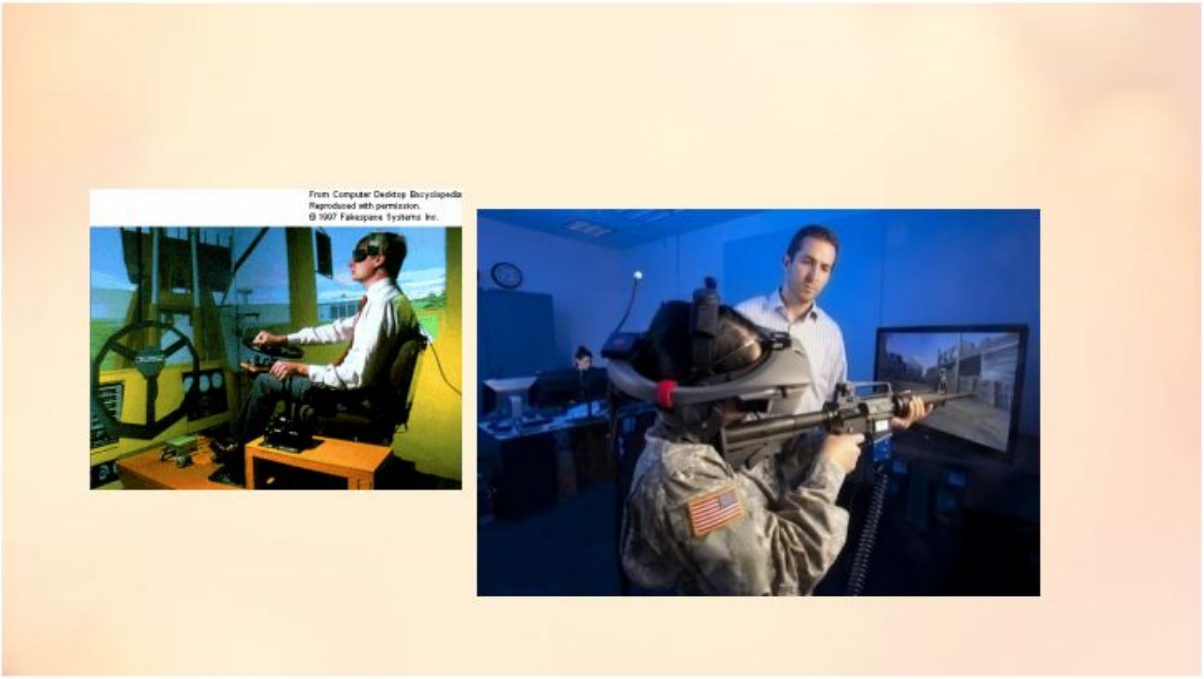


Figure: examples of virtual reality environments

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:

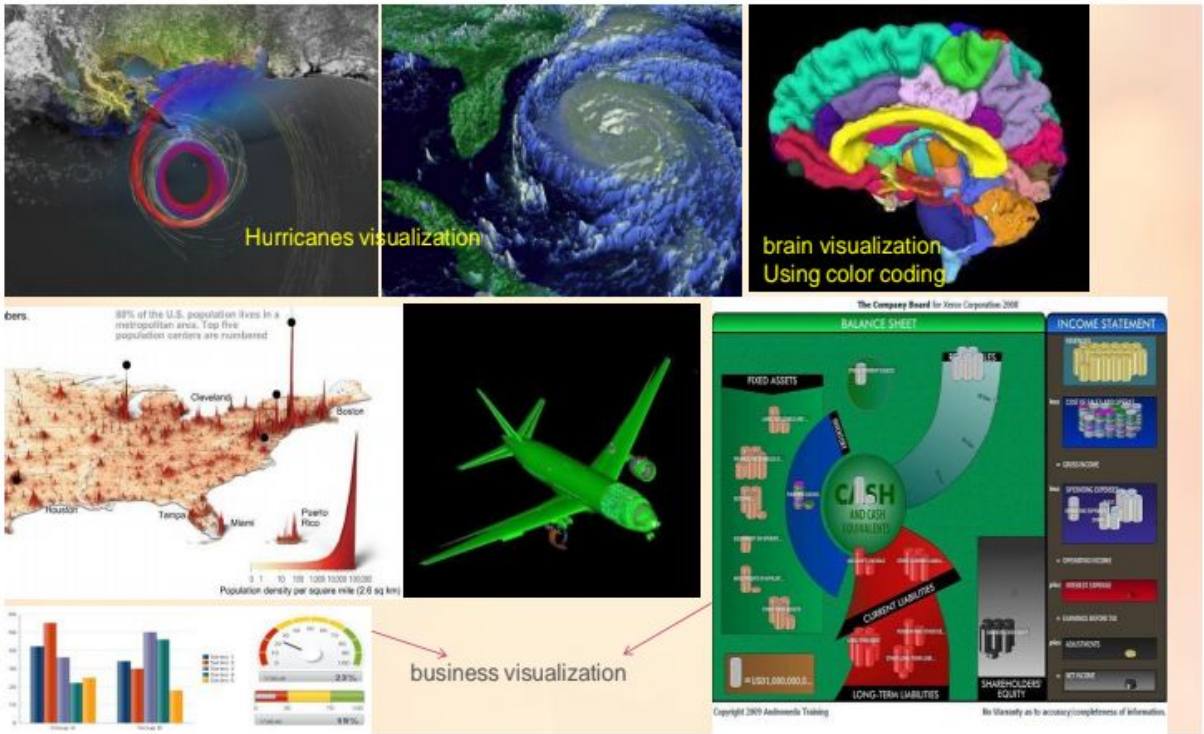
Producing graphical representations with data sets related to commerce, industry and other non scientific areas is referred to as business visualization.

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.



7

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

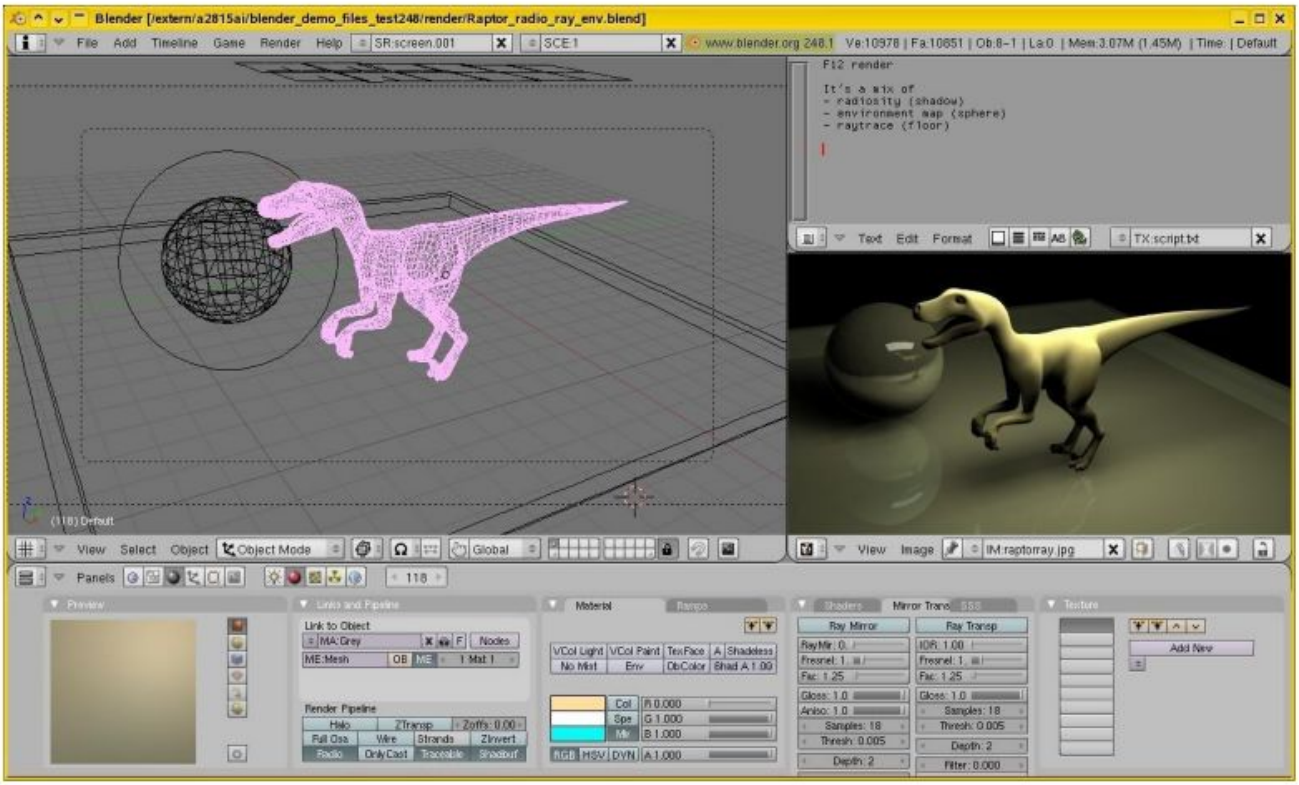


Figure: showing blender tool interface for designing a dynosarous character

9

VIDEO DISPLAY DEVICES:

1 Cathode-Ray Tube
- Raster-Scan Displays
- Random-Scan Displays

2 Color CRT Monitor
- Beam Penetration
- Shadow Mask
  - Delta-Delta
  - Inline

3 Direct-View Storage Tube

4 Flat-Panel Display
- Emissive
  - Plasma
  - Thin Film Electroluminescent
  - LED
- Non-Emissive
  - LCD

1. 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.

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

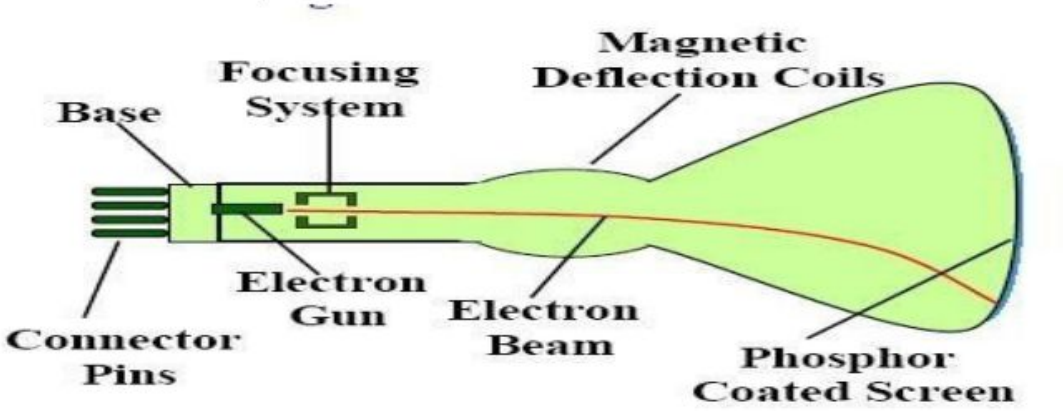


Figure: CRT design with magnetic deflection

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:

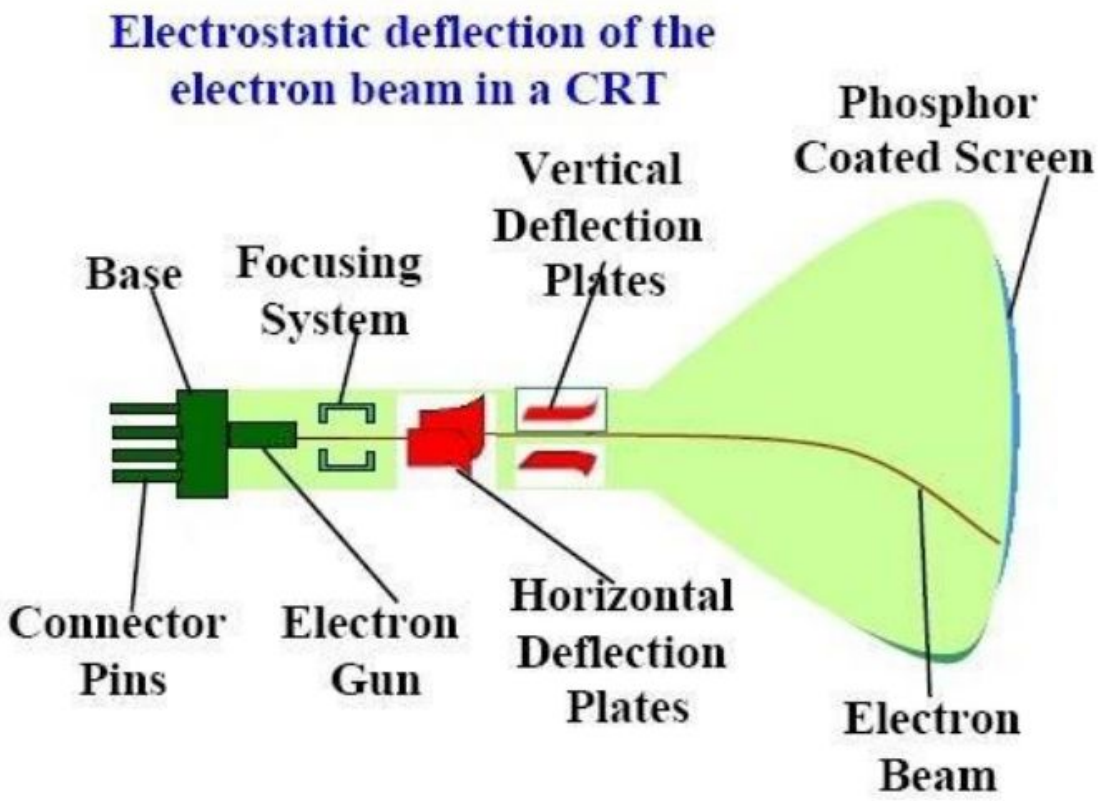
- Types of phosphorous screens:
  - High Persistence
  - Low persistence

- **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.

11

CRT design showing the operation of electron gun:



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:

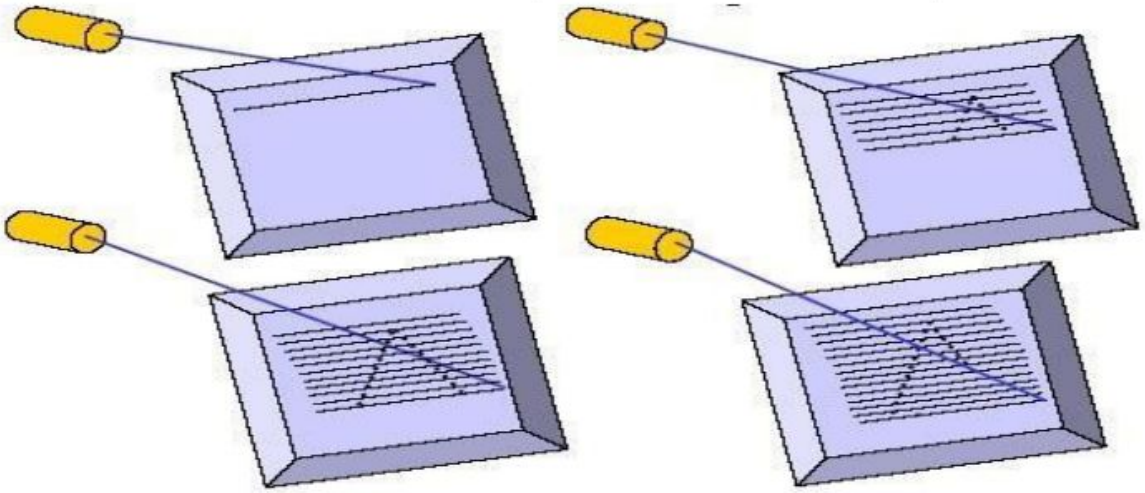
- 60 to 80 frames per second
- Represented as cycles per second/ Hertz(Hz)

8.Horizontal Retrace:

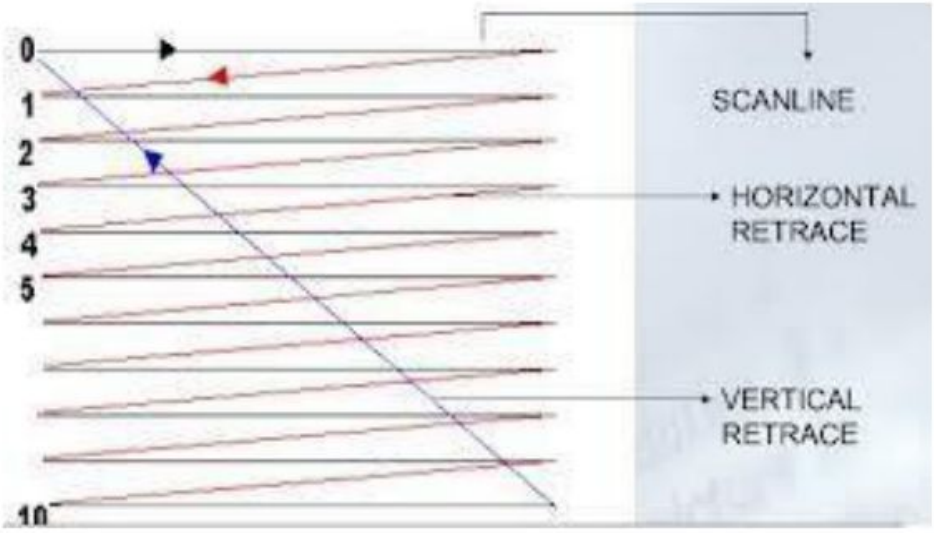
- The return of electron beam to the left of the screen after refreshing each scan line

9. Vertical Retrace:

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

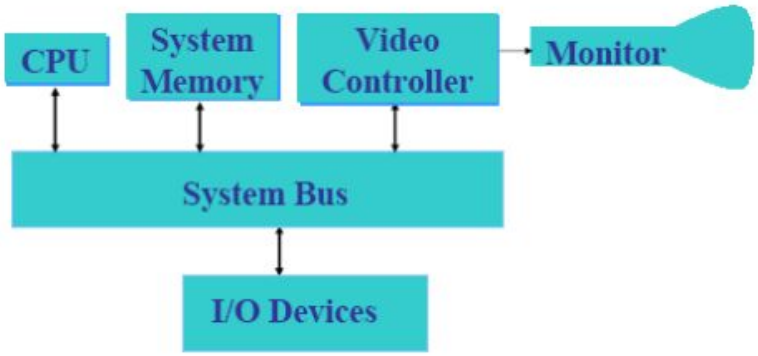


Raster Scan—Horizontal Retrace and Vertical Retrace:



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.

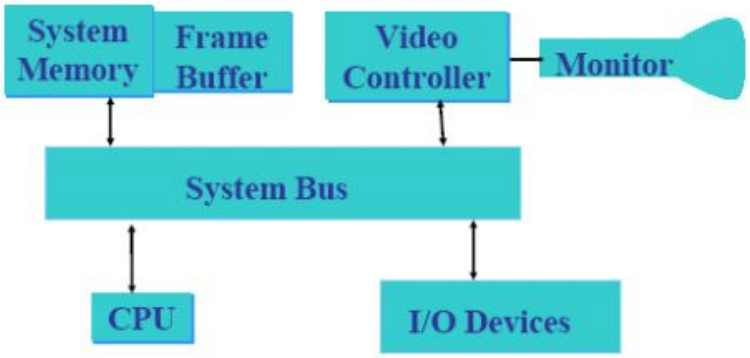


Architecture of a simple raster graphics system

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.

14

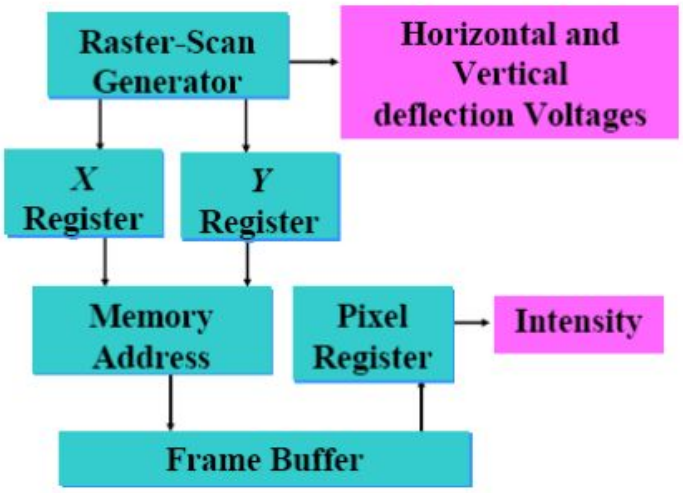
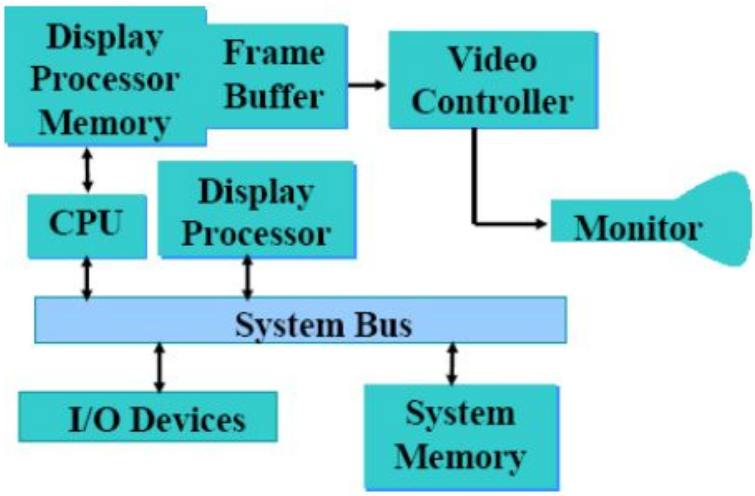


Figure: Basic Video Controller refresh operations

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:

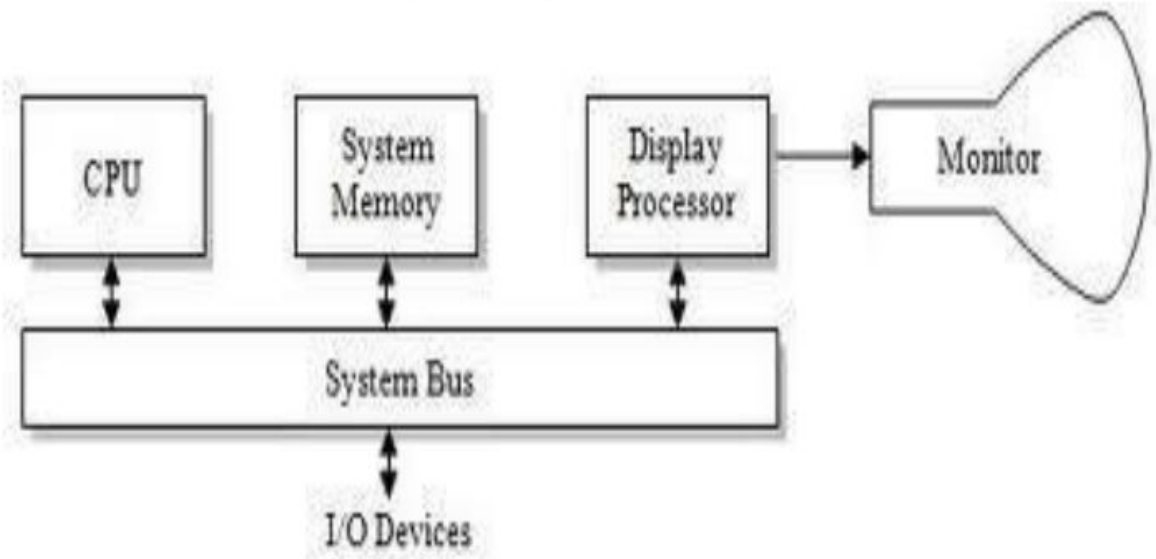
- Types of possibilities of organizing Frame buffer:
  - Frame buffer will be present somewhere inside of system memory
  - Some raster systems have a fixed portion of system memory reserved for frame buffer
  - Some raster systems have 2 frame buffers. one frame buffer will contains intensity values 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.

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.

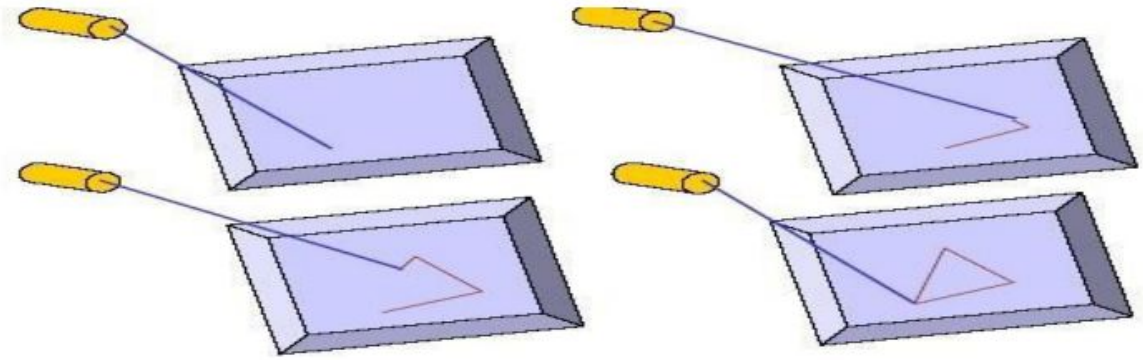
Sometimes the display processor in the random scan system is referred to as a display processing unit or a graphics controller. Graphics patterns are drawn on a random scan system by directing the electron beam along the component lines of picture. Lines are defined by the values for their coordinate 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

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

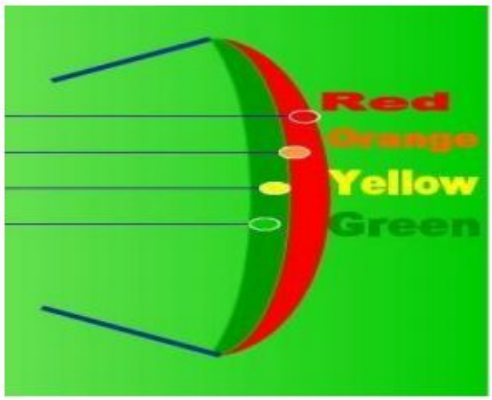
2 Two Layers of Phosphor is used Red and Green

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

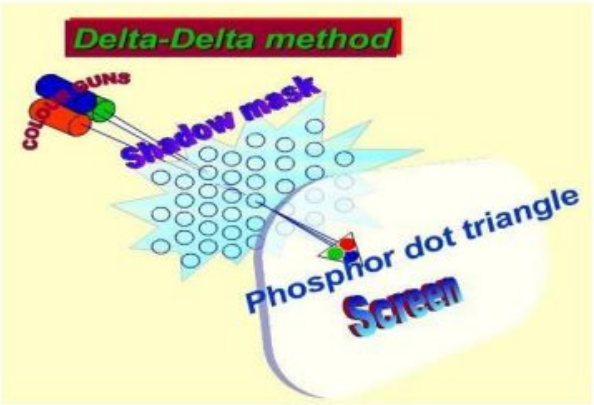
7 24 bits of storage per pixel

8 Arrangements

- Delta-Delta Method
- In-Line Method

Delta-Delta Method

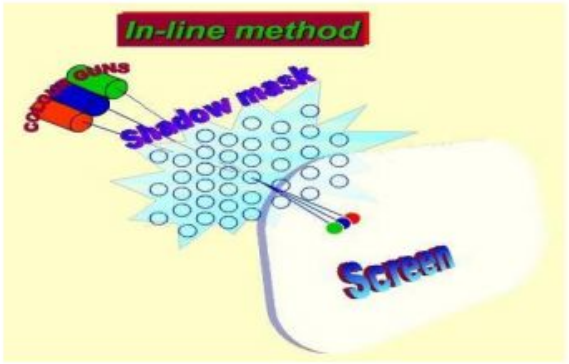
- Commonly used in color CRTs
- Mask holds the series of holes aligned with phosphor dot patterns (triangular pattern)
- Beam pass through the hole in the mask & activate the dot triangle which appears as a color spot



18

In-Line Method

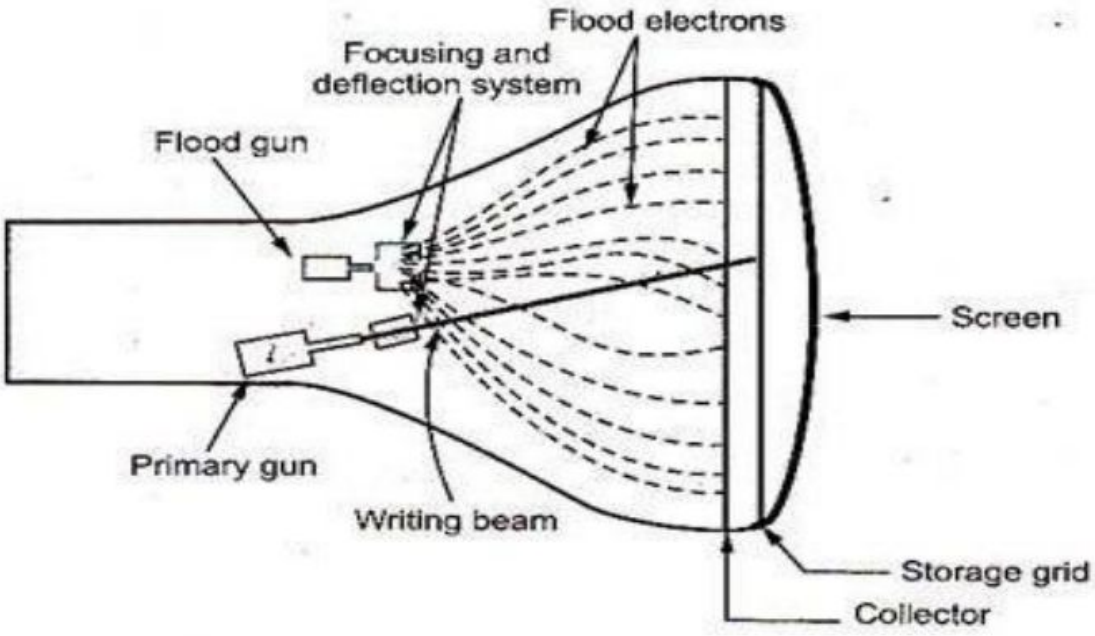
- Three Electron guns, the tri-colour dots in the screen are aligned along one scan line instead oftriangular shape
- Commonly used in high resolution color CRTs



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.



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.

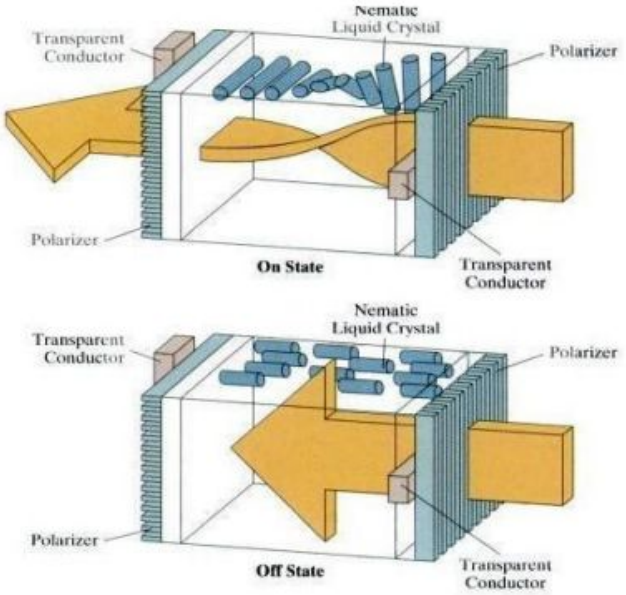


FIGURE 2-15 The light-twisting, shutter effect used in the design of most liquid-crystal display devices.

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.

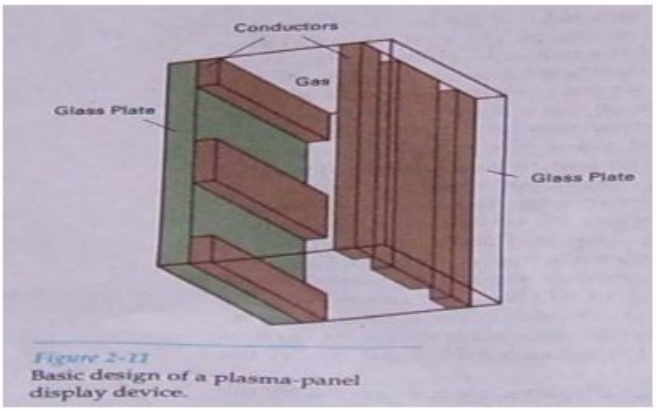
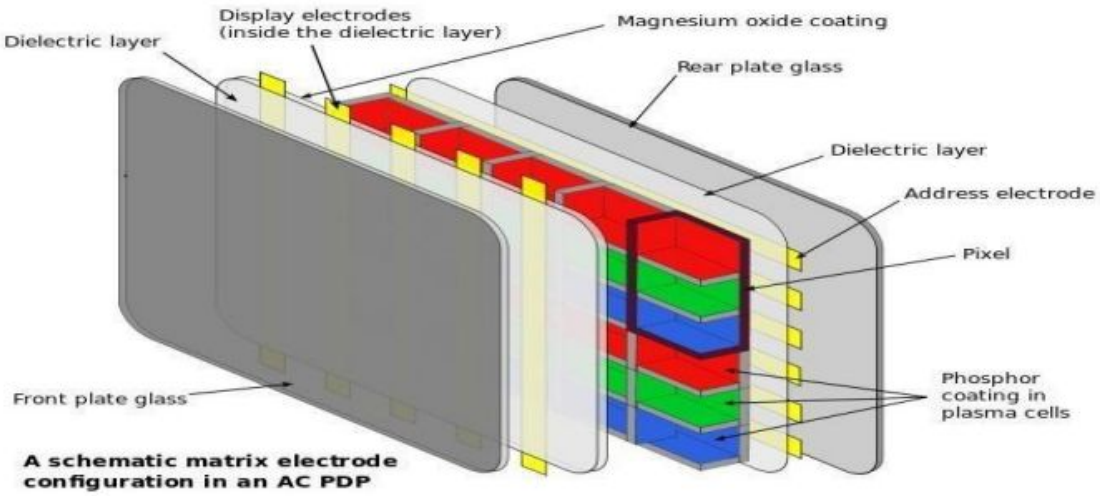


Figure 2-11
Basic design of a plasma-panel display device.

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.

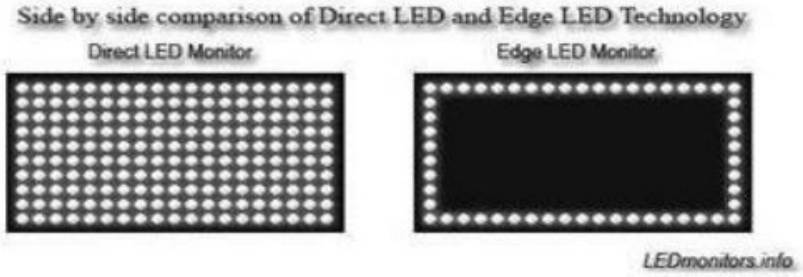
6 A plasma display consists of fluorescent lights which causes the formation of an image on screen. Like a CRT TV, each pixel has the three composite fluorescent color lights. These fluorescent lights are illuminated and the different colors are formed by combining the composite colors.

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.



22

II. 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.*)

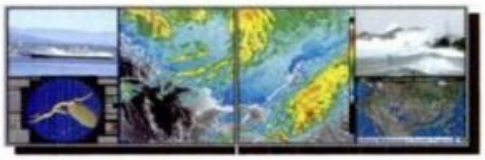


FIGURE 2-35 A multi-panel display system called the "Super Wall". (*Courtesy of RGB Spectrum.*)

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.*)

23

III. 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.*)

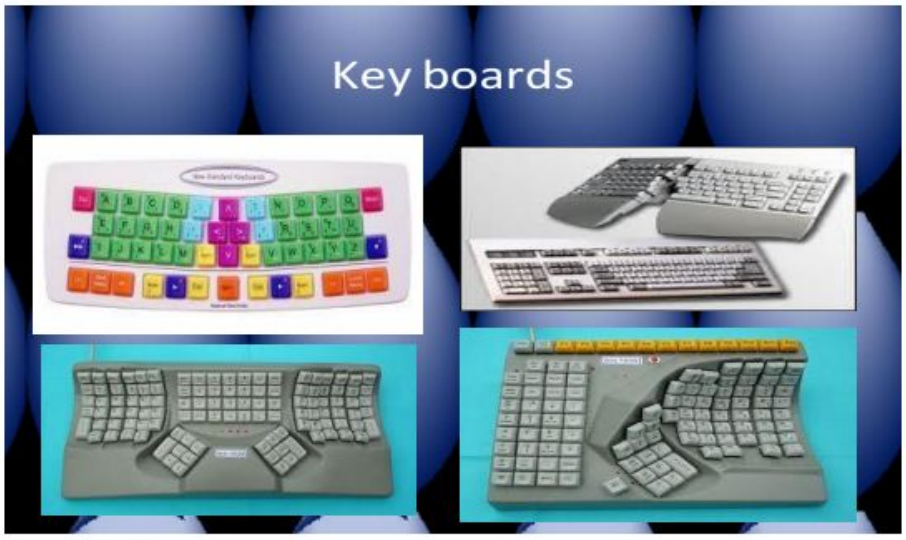
24

INPUT DEVICES:

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

Key Board:

- For entering text strings
- For Menu selections, or graphics functions



Key board consists of:

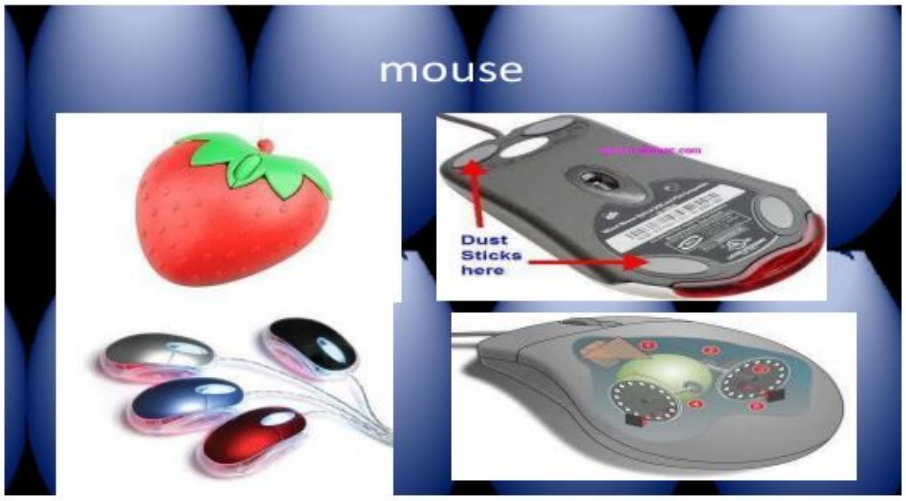
- Cursor-control keys
- function keys
- numeric keypad

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.

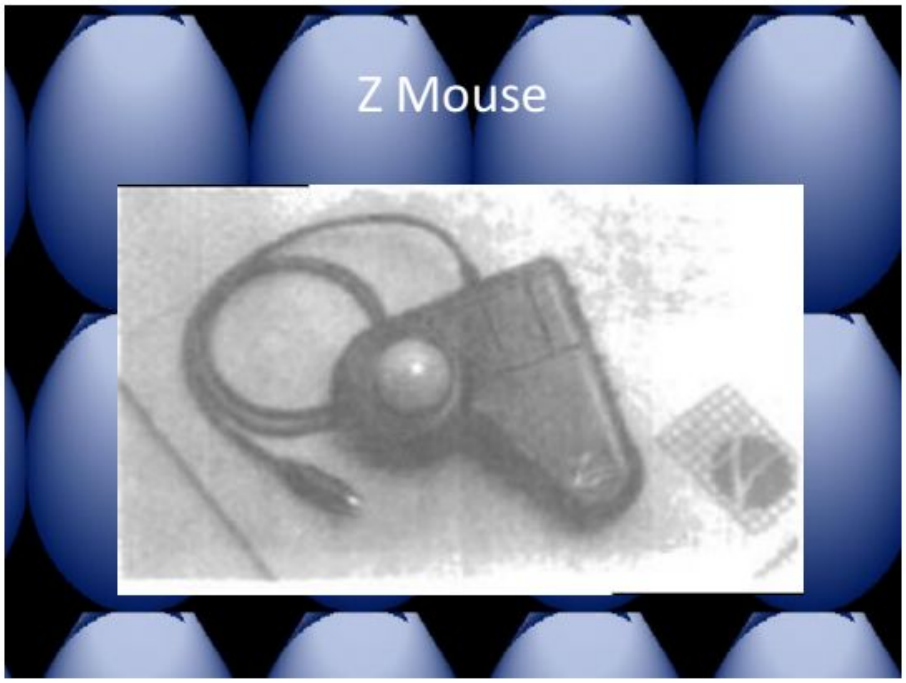
Mouse:

- 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 Mouse:

- 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 parameters.
- With the Z mouse, we can pick up an object, rotate it, and move it in any direction, or we can navigate our viewing position and orientation through a three dimensional scene.
- Applications of the Z mouse include virtual reality systems, CAD, and animation.



26

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 often mounted on keyboards

Space ball:

1 Space ball provides six degrees of freedom.

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

3 Strain gauges measure the amount of pressure applied to the space ball to provide input for spatial positioning and orientation as the ball is pushed or pulled in various directions.

4 Space balls are used for three-dimensional positioning and selection operations in virtual-reality systems, 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

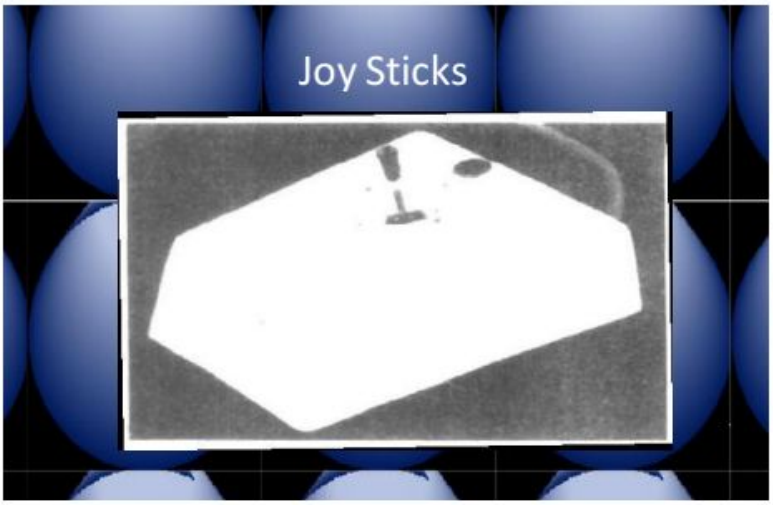
5 Potentiometers mounted at the base of the joystick measure the amount of movement

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.

8 Pressure sensitive joysticks, also called isometric joysticks, have a non movable stick.

9 Pressure on the stick is measured with strain gauges and converted to movement of the cursor in the direction specified.



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 information about the position and orientation of the hand.
- 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:

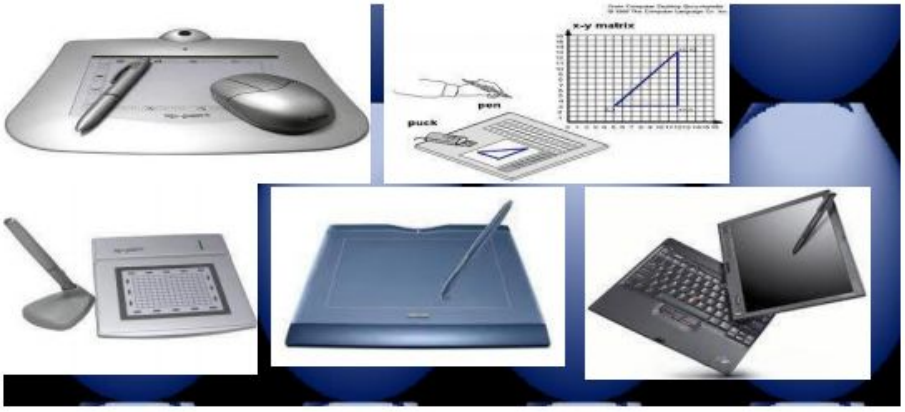
a. Graphics tablet (based on hand cursor)

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.

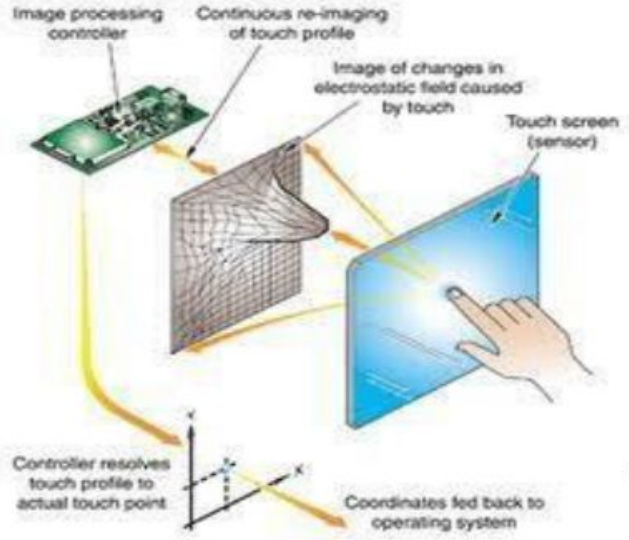
4 Three-dimensional digitizers use sonic or electromagnetic transmissions to word positions.

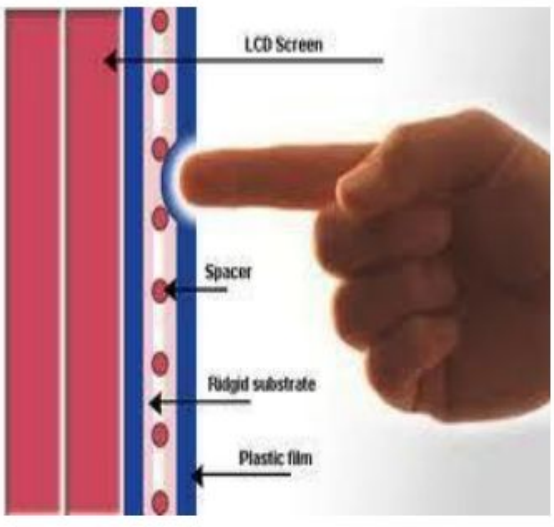
5 One electromagnetic transmission method is similar to that used in the data glove: A coupling between the transmitter and receiver is used to compute the location of a stylus as it moves over the surface of an object



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.
- It allows the user to point to displayed objects or draw on the screen in a similar way to a touch screen butwith greater positional accuracy.
- Since light pens operate by detecting light emitted by the screen phosphors, some nonzero 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)

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

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

32

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

Module 1 Part 2

Output Primitives

- Description of objects in terms of primitives and attributes and converts them to the pixels on the screen.
  - Primitives – what is to be generated
  - Attributes – how primitives are to be generated
- Each of the output primitives has its own set of *attributes*.
- Examples: point, line, text, filled region, images, quadric surfaces, curves.
- A point is a location in space.
- A line is a connection between 2 points
- By using these points, line, curved lines and filled regions, etc, basic geometrical structures or graphical objects can be created.

Points:

- The electron beam is turned on to illuminate the phosphor at the selected location (x, y) where
  $0 \le x \le$ Xmax
  $0 \le y \le$ Ymax
- setPixel(x, y, intensity) – loads an intensity value into the frame-buffer at (x, y).
- getPixel(x, y) – retrieves the current frame-buffer intensity setting at position (x, y).
- X-pixel column number
- Y-scan line number
- Analog devices, such as a random-scan display or a vector plotter, display a straight line smoothly from one endpoint to another.
- Linearly varying horizontal and vertical deflection

Line Drawing Algorithms:

- Digital Differential Analyzer (DDA).
- Bresenham's Line Algorithm.

DDA:

Line Drawing Algorithms:

DDA: ~~Right~~ left → Right either Δy or Δx

* DDA based on calculating
* A line with +ve slope, with less than or equal to 1, sample x interval $\Delta x = 1$ & compute each successive y

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

∵ y is rounded

* For ~~exact~~ line with +ve slope > 1, y intervals $\Delta y = 1$ and, calculate succeeding x value.

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

* Right → left

So starting endpoint is at right, we have

$$y_{k+1} = y_k - m \quad ; \quad \Delta x = -1$$

[∵ slope <= 1]

* Slope > 1, we have

$$\Delta y = -1;$$
$$x_{k+1} = x_k - 1/m.$$

$\Delta x = x_2 - x_1$

$\Delta y =$

* For -ve values of Slope,

Left → Right

slope < 1	$\Delta x = 1$
Slope > 1	$\Delta y = -1$

Right → left

$\Delta x = -1$

$\Delta y = 1$

```
void LineDDA(int x0, int y0, int x1, int y1)
{
  int dx = x1 – x0, dy = y1 – y0, steps;

  if (abs(dx)>abs(dy)) steps = abs(dx);
  else steps = abs(dy);

  // one of these will be 1 or -1
  double xIncrement = (double)dx / (double )steps;
  double yIncrement = (double)dy / (double )steps;

  double x = x0;
  double y = y0;
  setPixel(round(x), round(y));

  for (int i=0; i<steps; i++) {
  {
     x += xIncrement;
     y += yIncrement;
     setPixel(round(x), round(y));
  }
}
```

Problem:

(2,2), (7,5)

x_1 y_1 x_2 y_2

$x_a = 2$ $x_b = 7$

$y_a = 2$ $y_b = 5$

$dX = X_b - X_a$

$= 7 - 2 = 5$

$dY = Y_b - Y_a$

$= 5 - 2$

$= 3$

5 steps

$m = \frac{\Delta Y}{\Delta X} = \frac{Y_b - Y_a}{X_b - X_a}$

$= \frac{3}{5} = 0.6$

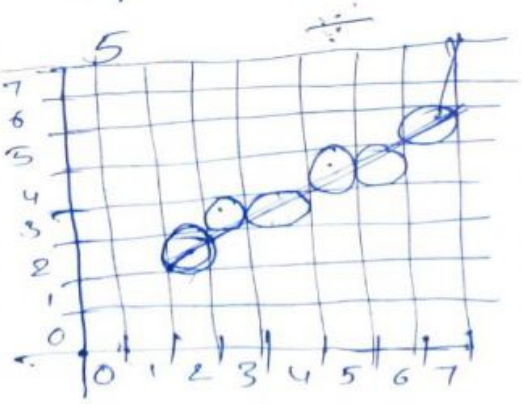
Rank: 1 - 4 = Down

5 - 9 = Up

$x_{inc} = 5/5 = 1$

$y_{inc} = 3/5 = 0.6$

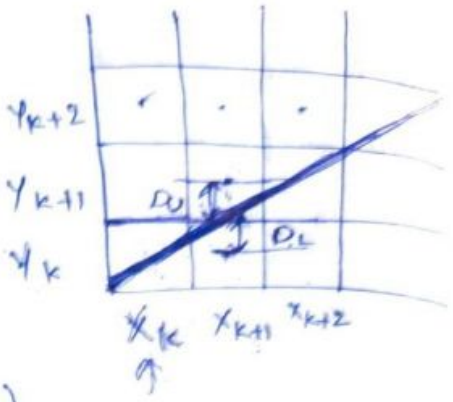
K	X	Y	X_{K+1}	Y_{K+1}
0	2	2	2+1=3	2+0.6=2.6
1	3	3	3+1=4	2.6+0.6=3.2
2	4	3	4+1=5	3.2+0.6=3.8
3	5	4	5+1=6	3.8+0.6=4.4
4	6	4	6+1=7	4.4+0.6=5.0
5	7	5		



Bresenham's Line Algorithm:

$|m| < 1$

$$y = mx + b \rightarrow (1)$$



For Y coordinate at $x_k + 1$ is

$$Y = m(x_k + 1) + b$$

> Current position (x_k, y_k)

> Next pixel choices $(x_k + 1, y_k)$ & $(x_k + 1, y_k + 1)$

> To choose the pixel, we take help of Decision parameter (D) or Error (e).

$$e = D_B - D_A \quad (\text{or}) \quad D = D_L - D_U$$

> To determine D_U & D_L

$$D_L = y_k - y_k$$
$$= m(x_k + 1) + b - y_k \rightarrow (2)$$

$$D_U = (y_k + 1) - y$$
$$= y_k + 1 - m(x_k + 1) + b$$
$$= y_k + 1 - m(x_k + 1) - b \rightarrow (3)$$

> Decision parameter D is $(2) - (3)$

$$D_L - D_U = m(x_k + 1) + b - y_k - y_k + 1 + m(x_k + 1) + b$$

$$D_L - D_U = 2m(x_k+1) + 2b - 2y_k - 1$$
$$= 2m(x_k+1) - 2y_k + 2b - 1 \rightarrow (4)$$

> Substitute m with $\Delta y/\Delta x$

$$\Delta x(D_L - D_U) = \Delta x\left(2\frac{\Delta y}{\Delta x}(x_k+1)\right) - 2y_k + 2b - 1$$
$$= 2\Delta y.x_k + 2\Delta y - 2\Delta x.y_k + \Delta x(2b-1)$$
$$= 2\Delta y.x_k - 2\Delta x.y_k + \underbrace{2\Delta y + \Delta x(2b-1)}_{C}$$
$$= 2\Delta y.x_k - 2\Delta x.y_k + C$$

So decision parameter (p) at k^{th} position is

$$\boxed{p_k = 2\Delta y.x_k - 2\Delta x.y_k + C} \rightarrow (5)$$

> To calculate next decision parameter ie (p_{k+1})

$$p_{k+1} = 2\Delta y.x_{k+1} - 2\Delta x.y_{k+1} + C \rightarrow (6)$$

> Subtract p_k from p_{k+1}

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

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

$\because \boxed{x_{k+1} = x_k + 1}$

$$P_{K+1} = P_K + 2\Delta Y(x_{K+1} - x_K) - 2\Delta x(y_{K+1} - y_K)$$

$$\boxed{P_{K+1} = P_K + 2\Delta Y - 2\Delta x(y_{K+1} - y_K)} \rightarrow (7)$$

> where $(y_{K+1} - y_K)$ is either 0 or 1 depending on sign of P_K

> Initial value of Decision parameter P_0 is

$$\boxed{P_0 = 2\Delta Y - \Delta X}$$

To derive initial value of DP, P_0,

$$P_K = 2\Delta Y x_K + 2\Delta Y - 2\Delta x y_K + \Delta x(2(Y - mx) - 1)$$

$$= 2\Delta Y x_K + 2\Delta Y - 2\Delta x y_K + \Delta x\left(2Y - 2\frac{\Delta Y}{\Delta x} \cdot x - 1\right)$$

$$= 2\Delta Y x_K + 2\Delta Y - 2\Delta x y_K + 2\Delta x Y - 2\Delta Y x - \Delta x$$

> Initial values are (x_K, y_K) so replace Y with y_K

$$= 2\Delta Y x_K + 2\Delta Y - 2\Delta x y_K + 2\Delta x y_K - 2\Delta Y x_K - \Delta x$$

$$\boxed{P_0 = 2\Delta Y - \Delta X}$$

P_K is -ve, choose lower pixel

P_K is +ve, choose upper pixel.

BRESENHAM'S LINE DRAWING ALGORITHM
(for $|m| < 1.0$)

1. Input the two line end-points, storing the left end-point in (x_0, y_0)
2. Plot the point (x_0, y_0)
3. Calculate the constants Δx, Δy, $2\Delta y$, and $(2\Delta y - 2\Delta x)$ and get the first value for the decision parameter as:
4. At each x_k along the line, starting at $k = 0$, perform the following test. If $p_k < 0$, the next point to plot is (x_k+1, y_k) and:

$$p_{k+1} = p_k + 2\Delta y$$

Otherwise, the next point to plot is (x_k+1, y_k+1) and:

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

5. Repeat step 4 $(\Delta x - 1)$ times

① Given (0,0) and (−8,−4)

$x_a = 0 \quad y_a = 0 \quad m = \frac{\Delta y}{\Delta x} = \frac{4}{8} = 0.5$

$x_b = -8 \quad y_b = -4$

$|\Delta x| = 8 - 0 = 8 \quad 2\Delta x = 16$

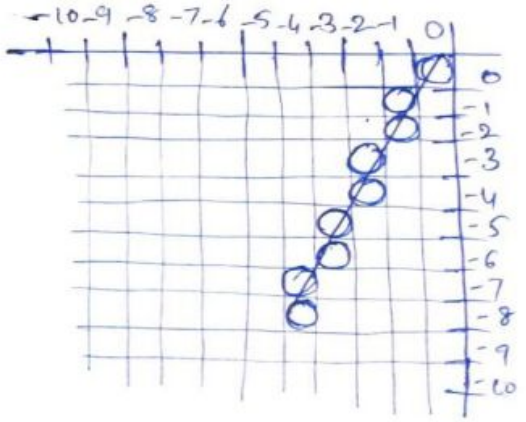
$|\Delta y| = 4 - 0 = 4 \quad 2\Delta y = 8$

$P < 0$ → Y: y_k, $P_k + 2\Delta y$; N: y_{k+1}, $P_k + 2\Delta y - 2\Delta x$

$P_0 = 2\Delta y - \Delta x = 8 - 8 = 0$

Initial values $(x, y) = (0,0)$

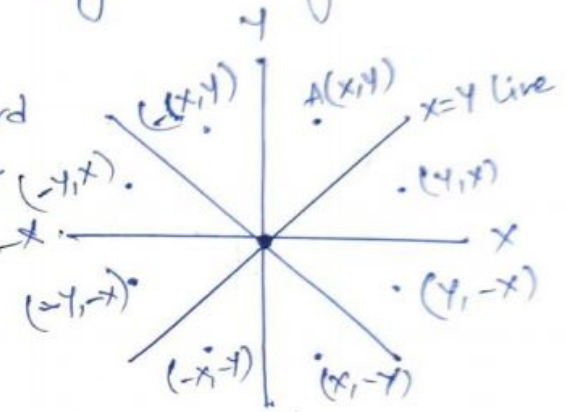
	x_{k+1}	y_{k+1}	
$P_0 = 0$	−1	−1	
$P_1 = -8$	−2	−1	$P_1 = 0 + 8 - 16 = -8$
$P_2 = 0$	−3	−2	$P_2 = -8 + 8 = 0$
$P_3 = -8$	−4	−2	$P_3 = 0 + 8 - 16 = -8$
$P_4 = 0$	−5	−3	$P_4 = -8 + 8 = 0$
$P_5 = -8$	−6	−3	$P_5 = 0 + 8 - 16 = -8$
$P_6 = 0$	−7	−4	$P_6 = -8 + 8 = 0$
$P_7 = -8$	−8	−4	$P_7 = 0 + 8 - 16 = -8$



Circle Generating Algorithms:

* Circle : A set of points that are all at a given distance r from center position (x_c, y_c).

* Circle have 8 way symmetry. Thus any circle generating algorithm can take advantage of it & plot 8 points by calculating coordinates of any one point.



* There are 2 standard methods of mathematically representing a circle centered at origin.

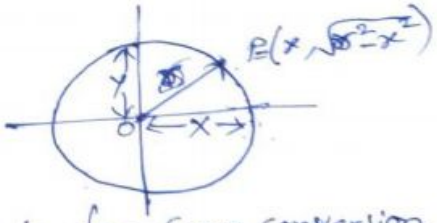
(i) Polynomial method

- In this method, circle is represented by a polynomial equation

$$x^2 + y^2 = r^2$$

$$y = \sqrt{r^2 - x^2}$$

↳ for each step of x



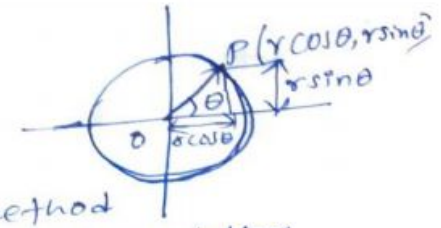
- But this method is inefficient for scan conversion

(ii) Trignometric method

θ – current angle

$x = r\cos\theta$

$y = r\sin\theta$

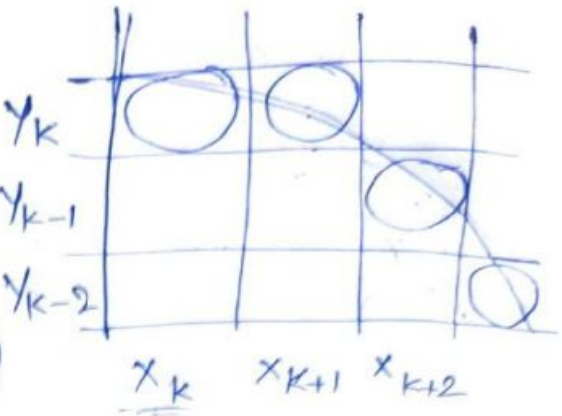


- This is even more inefficient method due to $\cos\theta, \sin\theta$ value computations

Mid Point Circle drawing Algorithm:

Current coordinate
(x_k, y_k)

Choose next coordinate
y_k or $y_k - 1$
(x_k+1, y_k) or (x_k+1, y_k-1)



Finding mid point

$$\left(\frac{x_k+1+x_k+1}{2}, \frac{y_k+y_k-1}{2}\right)$$

Midpoint $= (x_k+1, y_k-1/2) \rightarrow (1)$

$$r^2 = x^2 + y^2 \rightarrow (2)$$

Apply (1) in (2) to get decision parameter

$$P_k = (x_k+1)^2 + (y_k - 1/2)^2 - r^2$$

$$P_{k+1} = (x_{k+1}+1)^2 + (y_{k+1} - \tfrac{1}{2})^2 - r^2$$

$$P_{k+1} - P_k = (x_{k+1}+1)^2 + (y_{k+1} - 1/2)^2 - r^2 - (x_k+1)^2 - (y_k - 1/2)^2 + r^2$$

x_{k+1} can be written as $x_k + 1$ as x moves increments in unit units.

$$= \underbrace{((x_k+1)}_{a} \underbrace{+1)}_{b}^2 + (y_{k+1} - \tfrac{1}{2})^2 - (x_k+1)^2 - (y_k - \tfrac{1}{2})^2$$

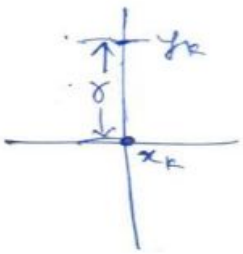
$$= (x_k+1)^2 + 1 + 2(x_k+1) + y_{k+1}^2 + \tfrac{1}{4} - y_{k+1} - (x_k+1)^2 - y_k^2 - \tfrac{1}{4} + y_k$$

$$= 2(x_k+1) + (y_{k+1}^2 - y_k^2) - (y_{k+1} - y_k) + 1$$

$$\boxed{P_{k+1} = P_k + 2(x_k+1) + (y_{k+1}^2 - y_k^2) - (y_{k+1} - y_k) + 1}$$

Initial decision parameter:

Starting point is $(0, r)$ (x_k, y_k)



we (2) & (1)

$$P_0 = (0+1)^2 + (r - \tfrac{1}{2})^2 - r^2$$

$$= 1 + r^2 + \tfrac{1}{4} - r - r^2$$

$$= \frac{5}{4} - r \qquad 5/4 \simeq 1$$

$$\boxed{P_0 = 1 - r}$$

$P_k \geq 0$	$P_k < 0$
$y_{k+1} = y_k - 1$	$y_{k+1} = y_k$

MID-POINT CIRCLE ALGORITHM

1. Input radius r and circle centre (x_c, y_c), then set the coordinates for the first point on the circumference of a circle centred on the origin as:

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

2. Calculate the initial value of the decision parameter as:

$$p_0 = 5/4 - r$$

3. Starting with $k = 0$ at each position x_k, perform the following test. If $p_k < 0$, the next point along the circle centred on $(0, 0)$ is (x_k+1, y_k) and:

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

Otherwise the next point along the circle is (x_k+1, y_k-1) and:

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

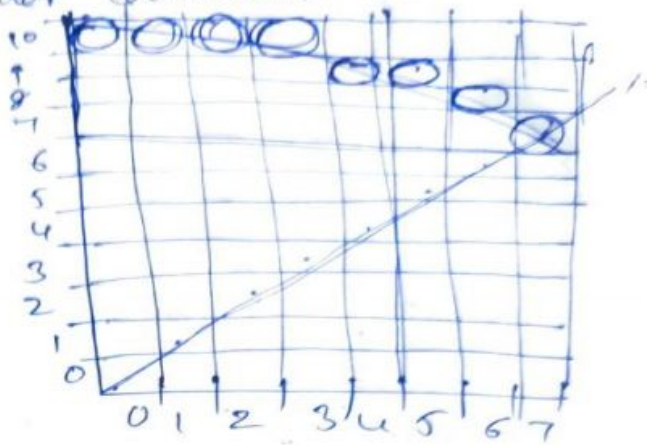
4. Determine symmetry points in the other seven octants

5. Move each calculated pixel position (x, y) onto the circular path centred at (x_c, y_c) to plot the coordinate values:

$$x = x + x_c \qquad y = y + y_c$$

6. Repeat steps 3 to 5 until $x >= y$

* Calculate the values upto $X=Y$ line in Quadrant 1 and by using symmetry, coordinates in other Quadrants can be obtained.



Example: ① Exercise Prob $r=6$

② $r=10$

$(x_0, y_0) = (0,10)$

$P_0 = 1-10 = -9$

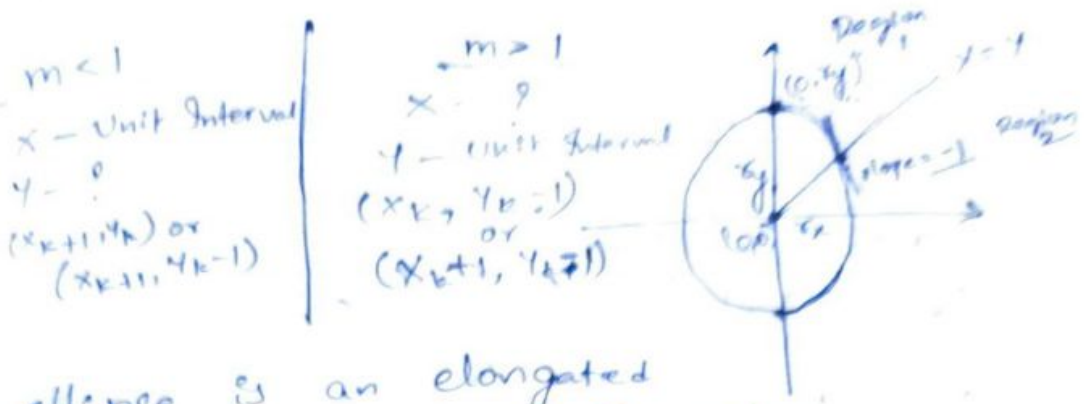
Initial incremental terms for DP,

$2x_0 = 0$; $2y_0 = 20$

K	P_K	(x_{k+1}, y_{k+1})		$2x_{k+1}$	$2y_{k+1}$
Initial value	-9	0	10	0	20
1	$-9+2+1$ $=-6$	1	10	2	20
2.	$-6+4+1$ $=-1$	2	10	4	20
3.	$-1+6+1$ $=6$	3	10	6	20
4.	$6+8+1-18$ $=-3$	4	9	8	18
5.	$-3+10+1$ $=8$	5	9	10	18
6.	$8+12+1-16$ $=5$	6	8	12	16
		7	7	14	14

Ellipse Drawing Algorithm:

Ellipse Drawing Algorithm:



- ellipse is an elongated circle. It has 2 way symmetry
- This is an ellipse with slope $r_x < r_y$.
- An ellipse is drawn from 90° to 0° where in x moves in +ve direction & y in -ve direction.

ellipse can Expressed mathematically as

$$\frac{x^2}{r_x^2} + \frac{y^2}{r_y^2} = 1$$

(or)

$$x^2 r_y^2 + y^2 r_x^2 - r_x^2 r_y^2 = 0 \longrightarrow (1)$$

$$\text{slope } |m| = \frac{dy}{dx} \Rightarrow -\frac{2r_y^2 x}{2r_x^2 y}$$

$$\Rightarrow \boxed{2r_y^2 x \geq 2r_x^2 y} \rightarrow \text{Boundary blw region 1 \& region 2}$$

Move out of region 1.

$$\therefore \frac{dy}{dx} = -1 \; ; \; 2r_y^2 x = 2r_x^2 y.$$

Region - 1

$m < 1$; x - moves unit intervals & fixed

$y - y_k / y_k - 1$

Calculate mid point

$(x_k + 1, y_k)$ $(x_k + 1, y_k - 1)$

Mid point = $(x_k + 1, y_k - \frac{1}{2})$

Apply mid point in Eq - (1)

$$P_{1k} = (x_k + 1)^2 r_y^2 + (y_k - \tfrac{1}{2})^2 r_x^2 - r_x^2 r_y^2$$

$$P_{1k+1} = (x_{k+1} + 1)^2 r_y^2 + (y_{k+1} - \tfrac{1}{2})^2 r_x^2 - r_x^2 r_y^2$$

(x_{k+1}) can be written as $(x_k + 1)$ as x moves in unit intervals.

Region 1

$$P1_{K+1} - P1_K = ((x_K+1)+1)^2 r_y^2 + (y_{K+1} - \tfrac{1}{2})^2 r_x^2 - (x_K+1)^2 r_y^2 - (y_K - \tfrac{1}{2})^2 r_x^2$$

$$P1_{K+1} = P1_K + (x_K+1)^2 r_y^2 + r_y^2 + 2(x_K+1) r_y^2 + (y_{K+1} - \tfrac{1}{2})^2 r_x^2 - (x_K+1)^2 r_y^2 - (y_K - \tfrac{1}{2})^2 r_x^2$$

$$\boxed{P1_{K+1} = P_K + 2(x_K+1) r_y^2 + r_y^2 + r_x^2 \left((y_{K+1} - \tfrac{1}{2})^2 - (y_K - \tfrac{1}{2})^2\right)}$$

Initial decision parameter can be calculated using initial point i.e., $(0, r_y)$

$$P1_K = 1^2 r_y^2 + (r_y - \tfrac{1}{2})^2 r_x^2 - r_x^2 r_y^2$$

$$= r_y^2 + r_y^2 r_x^2 + \frac{r_x^2}{4} - r_y r_x^2 - r_x^2 r_y^2$$

$$\boxed{P1_K = r_y^2 + \frac{r_x^2}{4} - r_y r_x^2}$$

Region – 1 ($m < 1$)

$P_{1k} \geq 0$
$\Rightarrow (x_k + 1, y_k - 1)$

$P_{1k} < 0$
$\Rightarrow (x_k + 1, y_k)$

Region – 2 ($m > 1$)

$P_{2k} \geq 0 \Rightarrow (x_k, y_k - 1)$

$P_{2k} < 0 \Rightarrow (x_k + 1, y_k - 1)$

$2 r_y^2 x_{k+1} \geq 2 r_x^2 y_{k+1}$

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

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

(y_{k+1}) can be written as $(y_k - 1)$; as y moves in unit intervals in region 2.

Region 2

$$P_{2k+1} - P_{2k} = (x_{k+1} + \tfrac{1}{2})^2 r_y^2 + (y_{k+1} - 1)^2 r_x^2 - (x_k + \tfrac{1}{2})^2 r_y^2 + (y_k - 1)^2 r_x^2$$

$$P_{2k+1} = P_{2k} + (x_{k+1} + \tfrac{1}{2})^2 r_y^2 + ((y_k - 1) - 1)^2 r_x^2 - (x_k + \tfrac{1}{2})^2 r_y^2 - (y_k - 1)^2 r_x^2$$

$$= P_{2k} - (x_{k+1} + \tfrac{1}{2})^2 r_y^2 + (y_k - 1)^2 r_x^2 + r_x^2 - 2(y_k - 1) r_x^2 - (x_k + \tfrac{1}{2})^2 r_y^2 - (y_k - 1)$$

$$P_{2\,k+1} = P_{2k} - 2(y_k - 1) r_x^2 + r_x^2 + r_y^2 ((x_{k+1} + \tfrac{1}{2})^2 - (x_k + \tfrac{1}{2})$$

Mid point Ellipse Algorithm:

1. Input r_x, r_y & ellipse center (x_c, y_c) and obtain the first point on an ellipse centered on the origin as

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

2. Calculate the initial value of decision parameter in region 1 as

$$P1_0 = r_y^2 - r_x^2 r_y + \frac{1}{4} r_x^2$$

3. At each x_k position in region 1, starting at $k=0$, Perform the following test:

If $P1_k < 0$, the next point along the ellipse centered ~~from~~ on $(0,0)$ is (x_{k+1}, y_k) and $P1_{k+1} = P1_k + 2r_y^2 x_{k+1} + r_y^2$

Otherwise, the next point along circle is

$(x_{k+1}, y_k - 1)$ and

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

with

$2r_y^2 x_{k+1} = 2r_y^2 x_k + 2r_y^2$; $2r_x^2 y_{k+1} = 2r_x^2 y_k - 2r_x^2$

and continue until $2r_y^2 x \geq 2r_x^2 y$.

4. Calculate the initial value of decision parameter in region 2 using the last point (x_0, y_0) calculated in region 1 as

$$p2_0 = r_y^2 (x_0 + 1/2)^2 + r_x^2 (y_0 - 1)^2 - r_x^2 r_y^2$$

5. At each y_k position in region 2, starting at $k = 0$, Perform the following test:

If $p2_k > 0$, the next position along the ellipse is $(x_k, y_k - 1)$ and

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

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

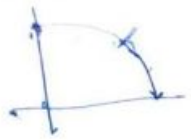
$$p2_{k+1} = p2_k + 2r_y^2 x_{k+1} - 2r_x^2 y_{k+1} + r_x^2$$

use same incremental calculations as region 1

continue until $y = 0$.

6. Determine symmetry points for other 3 quadrants.

7. Plot the coordinate values.



Example:

1. $r_x = 8$ and $r_y = 6$.

Sol:

1. Given

$r_x = 8 \quad r_x^2 = 64$

$r_y = 6. \quad r_y^2 = 36$

Initial point $(x_0, y_0) = (0, r_y) = (0, 6)$

Region-1

2. $P_{10} = r_y^2 - r_x^2 r_y + \frac{1}{4} r_x^2$

$= 36 - 64 \times 6 + \frac{1}{4} \times 64\ 16$

$= 36 - 384 + 16$

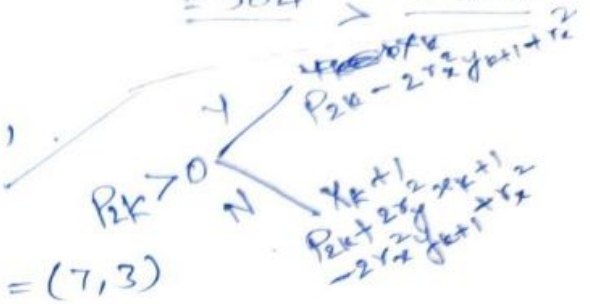
$P_{10} = -332$

K	P_K	(x_{K+1}, y_{K+1})		$2r_y^2 x_{K+1}$	$2r_x^2 y_{K+1}$
0	-332	(1, 6)			
1	$-332 + 2\times36\times1 + 36 = -296 + 72 = -224$	2	6	$2\times36\times1 = 72$	$2\times64\times6 = 768$
2	$-224 + 72\times2 + 36 = -44$	3	6	$72\times2 = 144$	$128\times6 = 768$
3	$-44 + 72\times3 + 36 = 208$	4	5	$72\times3 = 216$	$128\times5 = 640$
4	$208 + 288 - 640 + 36 = -108$	5	5	$72\times4 = 288$	
				$72\times5 = 360$	$128\times5 = 640$

5. $-108 + 360 + 36 = 288$ | 6 | 4 | $72 \times 6 = 432$ | $128 \times 4 = 512$

6. $288 + 432 - 512 + 36 = 244$ | 7 | 3 | $72 \times 7 = 504$ | $128 \times 3 = 384$

Moving out of region 1.



Region 2

Initial point $(x_0, y_0) = (7, 3)$

Initial decision parameter

$$P2_0 = r_y^2 (x_0 + 1/2)^2 + r_x^2 (y_0 - 1)^2 - r_x^2 r_y^2$$

$$= 36 \left(7 + \frac{1}{2}\right)^2 + 64(3-1)^2 - 36 \times 64$$

$$= 36 \times 49 + \frac{36}{4} + 2 \times 7 \times \frac{1}{2} \times 36 + 64 \times 9 + 64 - 2304$$

$$= 1764 + 9 + 252 + 640 - 384 - 2304$$

$$= 2665 - 2688$$

$$= -23$$

k	P_k	(x_{k+1}, y_{k+1}) = (7, 3)		$2r_y^2 x_{k+1}$	$2r_x^2 y_{k+1}$
0	-23	8	2	$72 \times 8 = 576$	$128 \times 2 = 256$
1	$-23 + 576 - 256 + 64 = +361$	8	1	$72 \times 8 = 576$	$128 \times 1 = 128$
2.	$361 - 128 + 64 = 297$	8	0	—	—

15/2

Part-1 2D Geometric Transformations

In many applications, there is need for altering or manipulating object displays. Changes in orientation, size & shape are accomplished with Geometric transformation that alter the coordinate discription of objects.

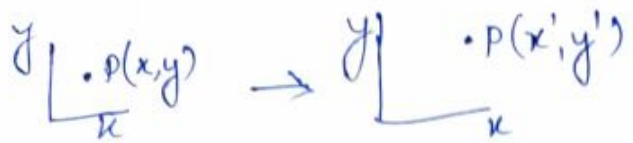
Basic Geometric Transformations are as follows:

(i) Translation – Changing Location
(ii) Rotation – rotation with angle θ
(iii) Scaling – Size is increased/decreased
(iv) Reflection – Mirror Image
(v) Shear – Slants the position

Don't disturb size & Shape of an obj (or) Rigid body transformation

(1) Translation:

* Translation is defined as proces of moving an obj from one location to the another.
* A translation is applied to an obj by repositioning it along a straight line path from one coordinate location to another.
* We translate a 2D point by adding translation distances i.e, t_x & t_y, to original coordinate position (x, y) to move the point to new position (x', y').



P from (x, y) to (x', y')

here $x' = x + tx$

$y' = y + ty$

where x & y are old coordinates

x' & y' are new coordinates

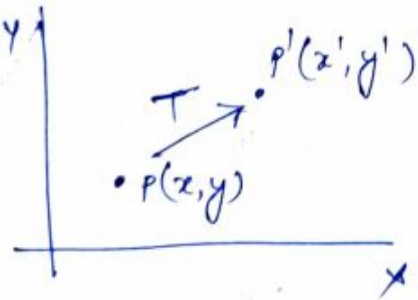
tx & ty are translation distance along x & y respectively

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

* Expressing translation Equation in column vectors.

$$P = \begin{bmatrix} x \\ y \end{bmatrix} \quad P' = \begin{bmatrix} x' \\ y' \end{bmatrix} \quad T = \begin{bmatrix} tx \\ ty \end{bmatrix}$$

$$\boxed{P' = P + T} \qquad T_{(tx,ty)} = \begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} \Rightarrow \text{Homogeneous coordinate}$$



T – Translation

P – Point before translation

P' – point after translation

* Translation is a rigid body transformation that moves objects without deformation.

* Every point on the obj is translated by the same amount.

* Translation can be done by redrawing the line between the new endpoint positions.

Examples:

1. Translate the position of object with coordinates A(3,6), B(8,9) & C(10,8). Translation distances along x axis is 3 & y-axis is 4. Find the new coordinates.

Sol: Given coordinates

$A(3,6)$, $B(8,9)$, $C(10,8)$

$t_x = 3$, $t_y = 4$

For $A(3,6)$:

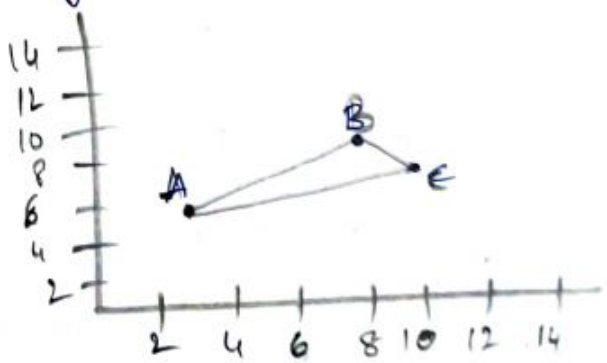
$x' = x + t_x$ & $y' = y + t_y$

Here $x = 3$, $y = 6$, $t_x = 3$, $t_y = 4$

$x' = 3 + 3 = 6$

$y' = 6 + 4 = 10$

$\therefore A' = (6, 10)$



For $B(8,9)$

$x' = 8 + 3 = 11$

$y' = 9 + 4 = 13$

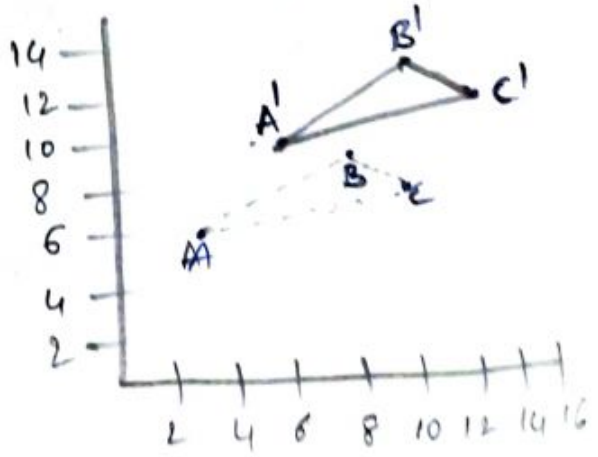
$\therefore B' = (11, 13)$

For $C(10,8)$

$x' = 10 + 3 = 13$

$y' = 8 + 4 = 12$

$C' = (13, 12)$



Problems:

① Translate a point P(10,13) with translation distances along x axis by -1 & y axis by -2.

② Translate the position of obj with (8,7), (9,10), (10,8) by translation distances along x axis by 3 & y axis by 4.

③. Translate the object with coordinates A(0,0), B(1,0), C(1,1) & D(0,1) by distance along x axis 5 & y axis 3.

② Rotation:

> Rotation is defined as rotating an obj with an angle θ in xy plane with respect to a pivot point (x,y) or origin (0,0).

> 2D rotation is applied to an obj by repositioning it along a circular path in XY Plane.

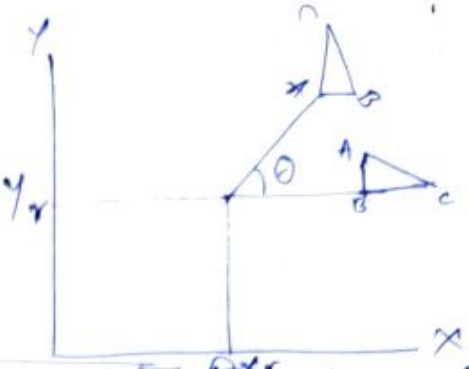
> To rotate an obj, rotation angle θ & rotation position (x_r, y_r) rotation point are needed.

> +ve value of θ → clockwise rotation
-ve value of θ → Anti clockwise rotation

> This transformation can also be described as a rotation about a rotation axis that is ⊥ to XY plane and passes through pivot point.

For rotation about origin

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



For transformed rotation:

$x' = r\cos(\phi+\theta)$ ~~cancel~~

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

Rotation in reference to rotation point

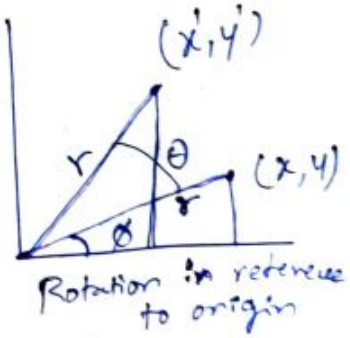
$$x' = r\cos(\phi+\theta) = r\cos\phi\cos\theta - r\sin\phi\sin\theta$$
$$y' = r\sin(\phi+\theta) = r\cos\phi\sin\theta + r\sin\phi\cos\theta$$

But original coordinates of point in polar coordinates are

$$x = r\cos\phi$$
$$y = r\sin\phi$$

Substitute (x,y) in (x',y').

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



Rotation in reference to origin

* In representation of row matrix

$$P' = R \cdot P$$

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

$$[x'\ y'] = [x\ y]\begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}$$

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

* If a point (x,y) is rotated to position (x',y') with angle θ about rotation point (x_r, y_r) otherwise origin (x',y'), then

For rotation in reference to rotation point

$$x' = x_r + (x - x_r)\cos\theta - (y - y_r)\sin\theta$$
$$y' = y_r + (x - x_r)\sin\theta + (y - y_r)\cos\theta$$

Example:

① Rotate an obj (4,3) with angle 45°

Sol: Given point (4,3)

$\theta = 45°$

$$x' = x\cos\theta - y\sin\theta$$
$$= 4.\cos 45° - 3.\sin 45°$$
$$= 4.\frac{1}{\sqrt{2}} - 3.\frac{1}{\sqrt{2}}$$
$$= 1/\sqrt{2}$$

$$y' = x\sin\theta + y\cos\theta$$
$$= 4\sin 45° + 3\cos 45°$$
$$= 4\,1/\sqrt{2} + 3\,1/\sqrt{2}$$
$$= 7/\sqrt{2}$$

New point after rotation is $(x', y') = (1/\sqrt{2}, 7/\sqrt{2})$

Problems:

① Rotate a point (7,8) with an angle of rotation 90°.

② Rotate a line segment A(2,2), B(7,5) with angle −45°

③ Rotate an object with coordinates A(2,4), B(8,9), C(10,11) with reference to P(1,2) with angle 60°.

(3) Scaling:

> Scaling is defined as increasing or decreasing the size of an obj.

> A scaling transformation alters size of an obj

> This operation can be carried out for polygons by multiplying the coordinates (x, y) of each vertex by scaling factors (S_x, S_y) to produce transformated coordinates (x', y').

$$x' = x \cdot S_x \quad \& \quad y' = y \cdot S_y$$

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

$$P' = S \cdot P \longrightarrow \text{Scaling with reference to origin.}$$

> For scaling an obj w.r.t a fixed point (x_f, y_f)

$$x' = x \cdot S_x + x_f (1 - S_x)$$
$$y' = y \cdot S_y + y_f (1 - S_y)$$

where, $x_f(1-S_x)$ & $y_f(1-S_y)$ are constants for all points in the obj

> If S, $\left(S = \begin{bmatrix} S_x \\ S_y \end{bmatrix}\right)$, values are < 1, reduces the size of the obj.

> If S values are > 1, Enlarges the sizes of obj

> If S values are $= 1$, remains same.

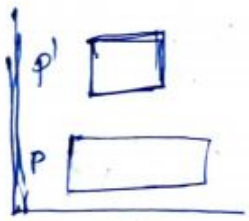
> If S_x & S_y are same values, a uniform scaling is produced.

> If S_x & S_y are different values, differential scaling is obtained.

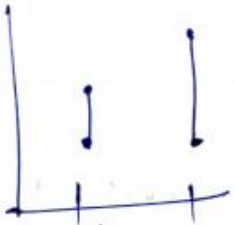
> objs transformed are both scaled & repositioned

> Scaling factors with values <1 moves objs closer to coordinate origin & values >1 moves coordinate position farther from origin.

> we can control the location of scaled obj by choosing a position called "fixed point".



$S_x = 2, S_y = 1$



$S_x = S_y = 0.5$

Example

(1) Scale the obj with coordinates A(2,5), B(7,10), C(10,2) The scaling factor along x & y is 2 with reference to the point P(3,3)

Sol: Given A(2,5), B(7,10), C(10,2)

$S_x = S_y = 2$

$x' = x \cdot S_x \quad y' = y \cdot S_y$

$x_f(1 - S_x)$

$3(1-2) = -3$

$x_f = y_f = -3.$

for A(2,5):

$x' = 2 * 2 = 4$

$y' = 5 * 2 = 10$

A'(4,10)

for B(7,10)

$x' = 7 * 2 = 14$

$y' = 10 * 2 = 20$

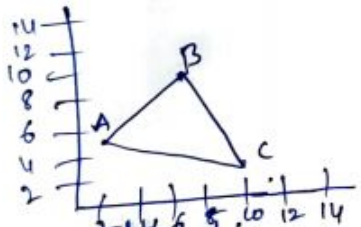
B'(14,20)

for C(10,2)

$x' = 10 * 2 = 20$

$y' = 2 * 2 = 4$

C'(20,4)



Original obj

$x' = x \cdot S_x + x_f(1 - S_x)$

$y' = x \cdot S_y + y_f(1 - S_y)$

A(2,5)

$x' = 4 - 3 = 1$

$y' = 10 - 3 = 7$ A'(1,7)

B(7,10)

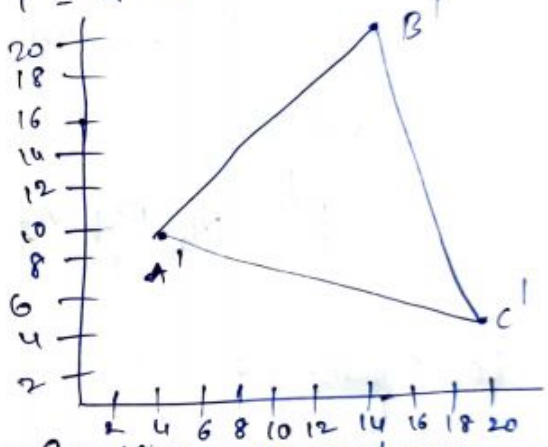
$x' = 14 - 3 = 11$

$y' = 20 - 3 = 17$ B'(11,17)

C(10,2)

$x' = 20 - 3 = 17$ C'(17,1)

$y' = 4 - 3 = 1$

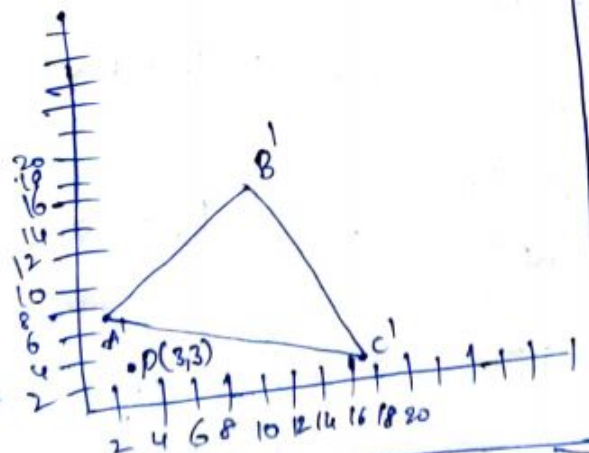


Scaling with reference to origin

Practice Problems:

1. A(0,3), B(3,3), C(3,0)
P(0,0) : Scaling parameter along x is 2 & y is 3.

2. P(1,4), Q(4,4), R(4,1), T(1,1)
Scaling factor along x is 3 & Y is 0.5



3. Magnify a triangle placed at A(0,0), B(1,1) & C(5,2) to twice its size wrt R(1,1)

Matrix Representations & Homogeneous Coordinates:

Use of matrix representations:

Some graphics applications involves sequences of geometric transformation i.e, 2 or more transformations are applied to an object. For Effecient processing the sequence of transformations, we use matrix representation.

> Each basic transformations can be expressed in the general matrix form

$$P' = M_1 \cdot P + M_2$$

For translation,

$$P' = P M_1 + M_2$$

where M_1 - Identity matric & M_2 - translation matrix

For rotation, $P' = P M_1 + M_2$

where M_1 - Identity & M_2 - Scaling matrix

For Scaling, $P' = P M_1 + M_2$

where M_1 - Identity & M_2 - Rotation matrix

Homogeneous Coordinates:

→ Normal Cartesian coordinate point - (x, y)

→ Homogeneous coordinate triple point representation

$(x_h, y_h, h) \rightarrow (x, y, 1)$

→ For 2D transformations, we use $h=1$.

$$x = x_h/h \; ; \; y = y_h/h$$

→ we use homogeneous coordinates to produce a sequence of transformations.

→ Each transformation applied at a time.

Sequence for applying transformations:

Scaling — we get Scaling matrix

↓

Translation — by using Scaling matrix, we translate here o/p is translation coordinates.

↓

Rotation — translation coordinates are used to get rotation coordinates.

→ In the above process, 3 intermediate calculations are used for deriving final coordinates.

(i) Combine all the above 3 transformations so that final coordinates are obtained directly.

(ii) we eliminate intermediate calculations.

→ For implementing above 2 steps,

(a) matrix additions are eliminated

(b) Only matrix multiplications are used

(c) 2×2 matrices are converted to 3×3 matrices

(d) To express any 2D transformation, we convert them to homogeneous coordinates.

i.e., (x, y) – 2D ; $(x, y, 1)$ – homogeneous.

Homogeneous matrices for

(i) Translation:

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

$$\boxed{P' = T(t_x, t_y) \cdot P}$$

(ii) Scaling:

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

$$\boxed{P' = S(S_x, S_y) \cdot P}$$

(iii) Rotation:

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

$$\boxed{P' = R(\theta) \cdot P}$$

Composite transformations:

> With matrix representations of transformations, we can setup a matrix for any sequence of transformations as a composite transformation matrix by calculating the matrix product of individual calculation.

Translations:

If 2 successive translation vectors (t_{x_1}, t_{y_1}) & (t_{x_2}, t_{y_2}) are applied to a coordinate position P, then final transformation location P' is calculated as

$$P' = \{T(t_{x_2}, t_{y_2}) \cdot T(t_{x_1}, t_{y_1})\} \cdot P$$

$$T(t_{x_2}, t_{y_2}) \cdot T(t_{x_1}, t_{y_1}) = \begin{bmatrix} 1 & 0 & t_{x_2} \\ 0 & 1 & t_{y_2} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_{x_1} \\ 0 & 1 & t_{y_1} \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & t_{x_1} + t_{x_2} \\ 0 & 1 & t_{y_1} + t_{y_2} \\ 0 & 0 & 1 \end{bmatrix}$$

∴ 2 successive translations product is additive

Rotations:

Two successive rotations applied to a point P produce the transformation position

$$P' = \{R(\theta_2) \cdot R(\theta_1)\} \cdot P$$

By multiplying rotation matrices, we get

$$R(\theta_2) \cdot R(\theta_1) = R(\theta_1 + \theta_2)$$

∴ 2 successive rotations product is additive.

Scaling:

2 Successive scaling operations produce

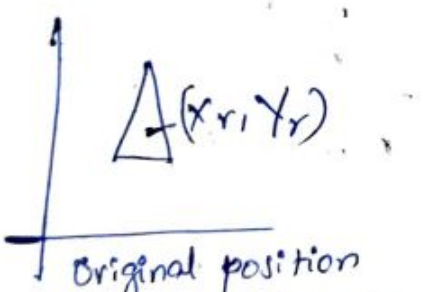
$$\begin{bmatrix} S_{x_2} & 0 & 0 \\ 0 & S_{y_2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} S_{x_1} & 0 & 0 \\ 0 & S_{y_1} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} S_{x_1} \cdot S_{x_2} & 0 & 0 \\ 0 & S_{y_1} \cdot S_{y_2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$S(S_{x_2}, S_{y_2}) \cdot S(S_{x_1}, S_{y_1}) = S(S_{x_1} \cdot S_{x_2}, S_{y_1} \cdot S_{y_2})$$

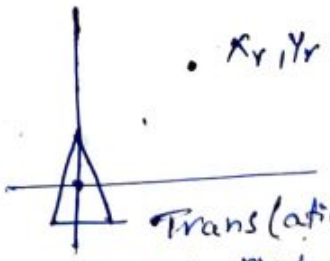
General Pivot Point Rotation:

We can generate rotations about any selected Pivot point (x_r, y_r) by performing the following sequence of translate – rotate – Inverse translate operations.

(i) Translate the obj, so that the pivot point position is moved to the coordinate origin.

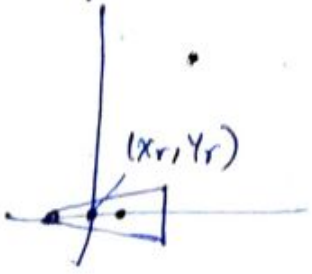


Original position of obj & Pivot point

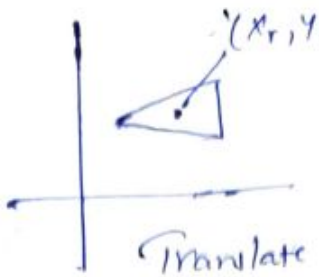


Translation of obj so that pivot point (x_r, y_r) is at origin.

(ii) Rotate the obj about the coordinate origin



(iii) Inverse translate the obj to its original position



Transformation sequence as follows:

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

Translate · Rotate · Translate (Inverse)

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

$$\therefore T(x_r, y_r) \cdot R(\theta) \cdot T^{-1}(x_r, y_r) = R(x_r, y_r, \theta)$$

where $T^{-1}(x_r, y_r) = T(-x_r, -y_r)$

General Pivot Fixed Point Scaling:

The transformations sequence to produce scaling wrt a selected fixed position (x_f, y_f) using a scaling function that can only scale relative to coordinate origin

(i) Translate obj so that the fixed point coincides with origin.

(ii) Scale the obj with respect to the origin.

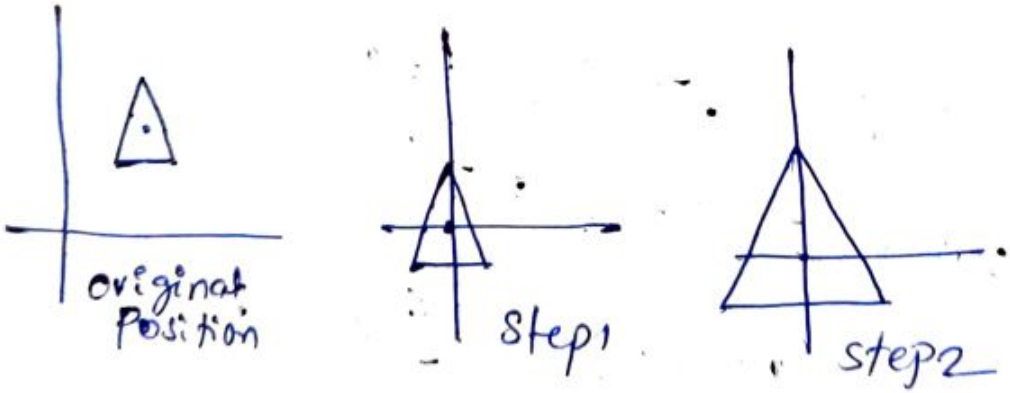
(iii) Use inverse translation to return the obj to its original position

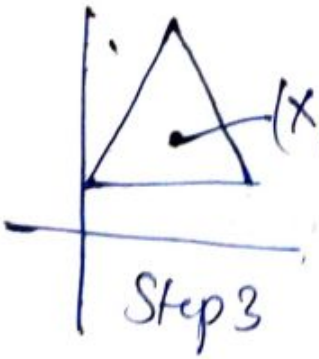
Transformation sequence as follows:

$$\begin{bmatrix} 1 & 0 & x_f \\ 0 & 1 & y_f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_f \\ 0 & 1 & -y_f \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} S_x & 0 & x_f(1-S_x) \\ 0 & S_y & y_f(1-S_y) \\ 0 & 0 & 1 \end{bmatrix}$$

$$T(x_f, y_f) \cdot S(S_x, S_y) \cdot T^{-1}(x_f, y_f) = S(x_f, y_f, S_x, S_y)$$





Composite transformations:

General Scaling Directions:

> Parameters s_x & s_y scale objects along the x & y directions.

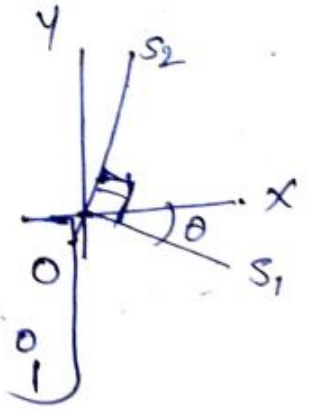
> we can scale an obj in other directions by rotating the object to align the desired directions with the coordinate axes before applying the scaling transformation.

> To accomplish scaling without changing the orientation of obj, rotate x & y axis for s_1 & s_2 respectively.

(i) Rotate the origin so that directions for s_1 & s_2 coincide x & y axes.

(ii) Apply Scaling

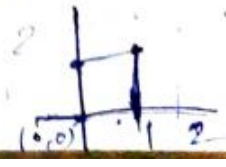
(iii) Rotate back the origin.

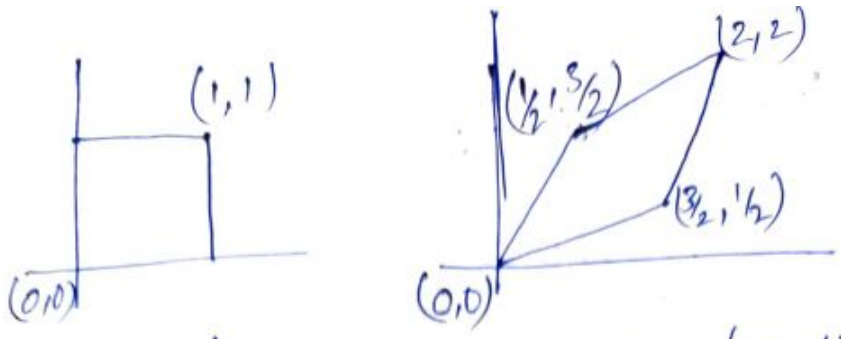


$$R^{-1}(\theta) \cdot S(s_1, s_2) \cdot R(\theta)$$

$$= \begin{bmatrix} s_1\cos^2\theta + s_2\sin^2\theta & (s_2 - s_1)\cos\theta\sin\theta & 0 \\ (s_2 - s_1)\cos\theta\sin\theta & s_1\sin^2\theta + s_2\cos^2\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

> Turn a unit square into a parallelogram by stretching it along diagonal from (0,0) to (1,1) with parameters $\theta = 45°$, $s_1 = 1$ & $s_2 = 2$.





□ to ◇ using composite transformation matrix with $S_1 = 1$, $S_2 = 2$ & $\theta = 45°$

Concatenation properties & Sequence of Operations:

$$A \cdot B \cdot C = (A \cdot B) \cdot C = A \cdot (B \cdot C)$$

> ∴ Transformation products may not be commutative.

> The product $A \cdot B$ is not equal to $B \cdot A$.

> Careful about the order in which the composite matrix is evaluated.

General composite transformations & Efficiency:

> A general 2D transformation representing a combination of translation, rotation & scaling can be expressed as

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} rs_{xx} & rs_{xy} & trs_x \\ rs_{yx} & rs_{yy} & trs_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

> For pivot & fixed point translations, matrix can be given as

$$T(t_x, t_y) \cdot R(x_c, y_c, \theta) \cdot S(x_c, y_c, s_x, s_y)$$

$$= \begin{bmatrix} s_x\cos\theta & -s_y\sin\theta & x_c(1-s_x\cos\theta) + y_c s_y \sin\theta + t_x \\ s_x\sin\theta & s_y\cos\theta & y_c(1-s_y\cos\theta) + x_c s_x \sin\theta + t_y \\ 0 & 0 & 1 \end{bmatrix}$$

> After 9 multiplications & 6 additions, the explicit calculation for transformed coordinates are

$$x' = x \cdot rs_{xx} + y \cdot rs_{xy} + trs_x$$

$$y' = x \cdot rs_{yx} + y \cdot rs_{yy} + trs_y$$

> A general rigid transformation matrix, involving only translations & rotations,

$$\begin{bmatrix} r_{xx} & r_{xy} & tx_x \\ r_{yx} & r_{yy} & t_{ry} \\ 0 & 0 & 1 \end{bmatrix}$$

> A rigid body transformation say first rotates ~~about~~ translates, ~~given as~~ with θ about a Pivot point (x_r, y_r), & translates given as:

$$T(t_x, t_y) \cdot R(x_r, y_r, \theta) =$$

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

2. A unit square is transformed by 2×2 transformation matrix. The following matrix is resulting position vector are $\begin{bmatrix} 0 & 2 & 8 & 6 \\ 0 & 3 & 4 & 1 \end{bmatrix}$, what is transformation matrix?

Sol: Suppose the unit square ~~matrix~~ have coordinates

(x,y), $(x+1,y)$, $(x+1, y+1)$, $(x, y+1)$

and let transformation matrix be $\begin{bmatrix} a & c \\ b & d \end{bmatrix}$

So, $\begin{bmatrix} 0 & 2 & 8 & 6 \\ 0 & 3 & 4 & 1 \end{bmatrix} = \begin{bmatrix} a & c \\ b & d \end{bmatrix} \begin{bmatrix} x & x+1 & x+1 & x \\ y & y & y+1 & y+1 \end{bmatrix}$

$= \begin{bmatrix} ax+cy & a(x+1)+cy & a(x+1)+c(y+1) & ax+c(y+1) \\ bx+dy & b(x+1)+dy & b(x+1)+d(y+1) & bx+d(y+1) \end{bmatrix}$

Now, $ax+cy=0$ & $bx+cy=0$

$a(x+1)+cy=2$ & $b(x+1)+dy=3$

$a(x+1)+c(y+1)=8$ & $b(x+1)+d(y+1)=4$

$ax+c(y+1)=6$ & $bx+d(y+1)=1$

for this we get $a=2$, $b=3$, $c=6$ & $d=1$.

∴ Transformation matrix is $\begin{bmatrix} 2 & 6 \\ 3 & 1 \end{bmatrix}$.

Other transformations:

(4) Shear transformation:

A shear transformation can distorts the shape of an obj.

2 types:

(i) X-shear

x value changes; y is constant

(ii) Y-Shear

y value changes; x is constant.

(i) X Shear: x coordinates are changed.

$$x' = x + sh_x \cdot y$$
$$y' = y$$

$$\begin{bmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

A coordinate position (x, y) is then shifted horizontally by an amount proportional to its distance from x axis.

X-shear wrt referal lines;

$$x' = x + sh_x(y - y_{ref})$$
$$y' = y$$

$$\begin{bmatrix} 1 & sh_x & -sh_x \cdot y_{ref} \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

(ii) Y Shear: x is constant and y value is changed

$$x' = x$$
$$y' = y + Sh_y \cdot x$$

$$\begin{bmatrix} 1 & 0 & 0 \\ Sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Y Shear wrt referal line,

$$x' = x$$
$$y' = Sh_y(x - x_{ref}) + y$$

$$\begin{bmatrix} 1 & 0 & 0 \\ Sh_y & 1 & -Sh_y x_{ref} \\ 0 & 0 & 1 \end{bmatrix}$$

Practice Problems:

1. Triangle with points (1,1) (0,0) & (1,0). Shear the given triangle along
   (i) X axis with Shear parameter 2
   (ii) Y axis with Shear parameter 2
   (iii) X axis Sp=3 with referal Y=1
   (iv) Y axis Sp=3 with referal X=-1

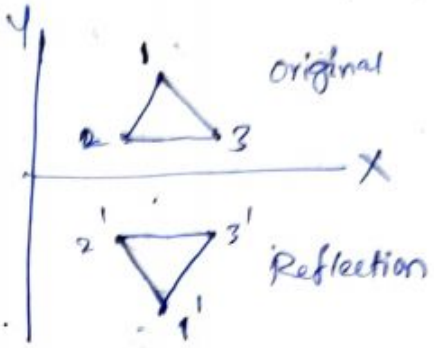
(5) Reflection:

A reflection produces a mirror image.

Mirror image for 2D reflection is generated relative to an axis of reflection, by rotating the obj 180°.

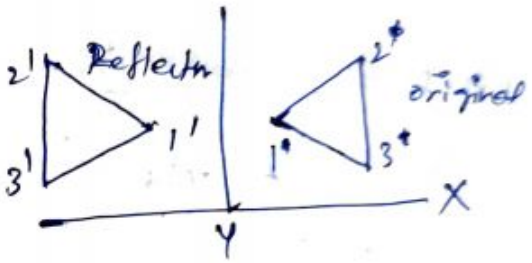
Reflection X axis:

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



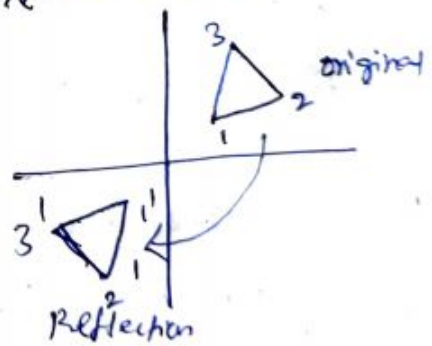
Reflection Y axis:

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



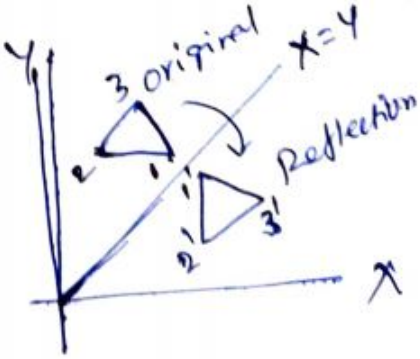
Reflection about XY plane

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



Reflection about X=Y diagonal line

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



For counter clockwise rotation about X=Y line,

~~T ⇒ Reflect ⇒~~

(i) clockwise rotation by 45°
(ii) Reflection about Y axis
(iii) counter clockwise rotation by 45°

Resulting transformation matrix,

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

Practice Problems:

1. Reflect the triangle with points A(3,4), B(6,4) & C(5,6) on the X axis.
2. Reflect the object, w.r.t Y axis, with coordinates A(3,4), B(6,4) & C(5,6).
3. Reflect ~~the along XY~~ the object (2,0), (4,2) & (5,0) along X=Y line.
4. Reflect the object (2,0) (4,2) & (5,0) along XY plane.

Part - 2

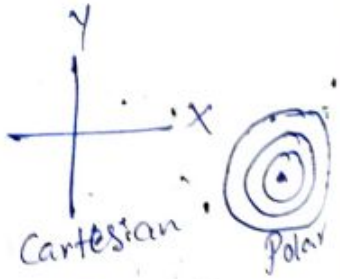
2D Viewing

The viewing pipeline:

* A world coordinate area selected for display is called a "window".
* An area on a display device to which a window is mapped is called a "viewport".
* Window defines what is to be viewed
* Viewport defines where it is to be displayed
* Mapping of a part of world coordinate scene to device coordinates is referred to as viewing transformation/ window-to-viewport transformation

Terminology:

1. Cartesian coordinate sys.
2. Polar coordinate sys



3. Modeling coordinates/Local coordinates:
   Each obj has its own origin
4. World coordinates:
   It describes the relative positions and orientations of every generated obj.

5. Normalized device coordinates:

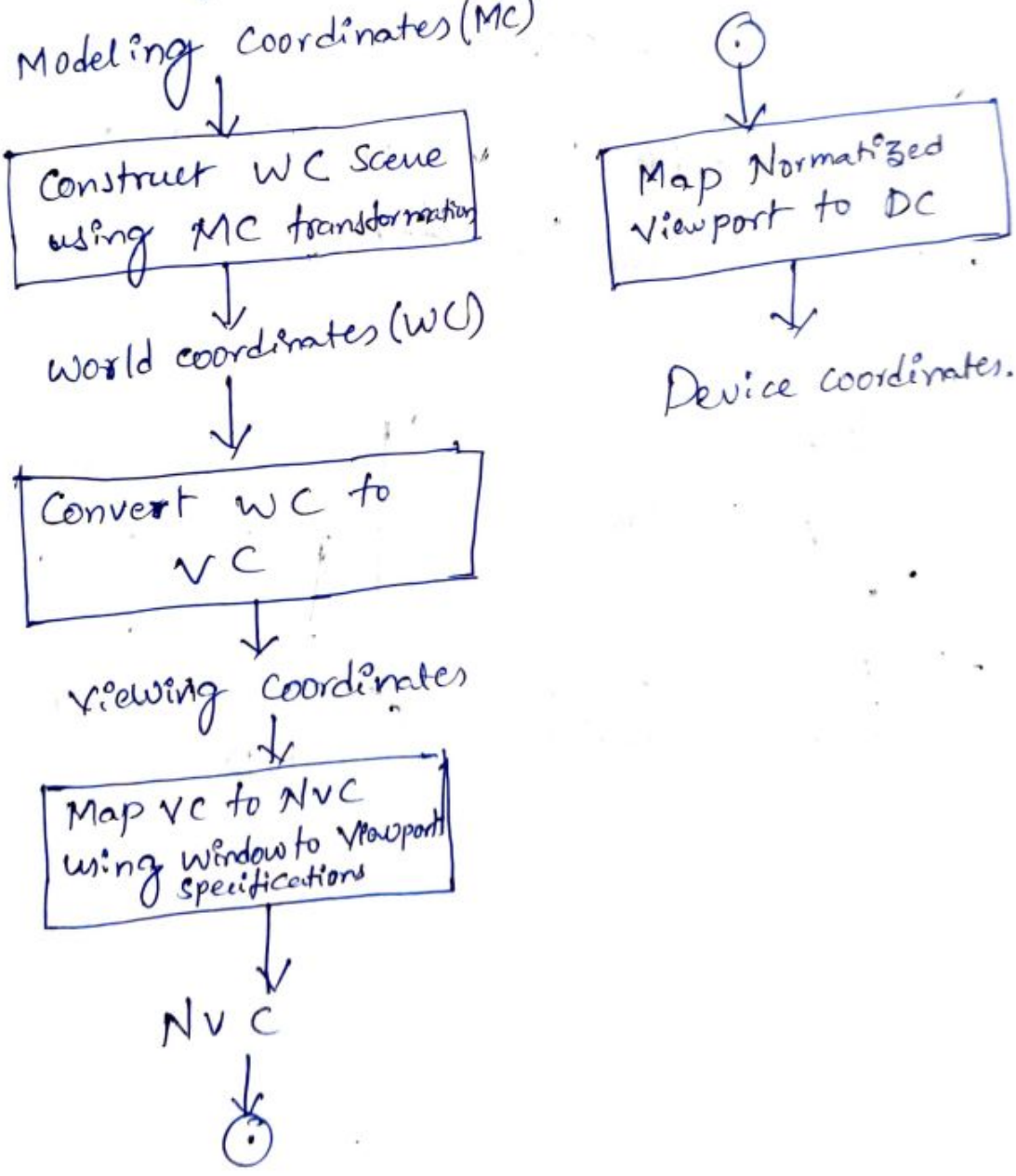
O/p devices have their own coordinates, ie., X & Y axis range from 0 to 1.

6. Device coordinates:

Specific coordinates used by a device. The transformation based on individual device is handeled by computer sln with out user concern.

eg: pixels on monitor, mm on plotter.

2D Viewing transformation pipeline:



> Construct Scene in wc using o/p primitives etc. from MC.

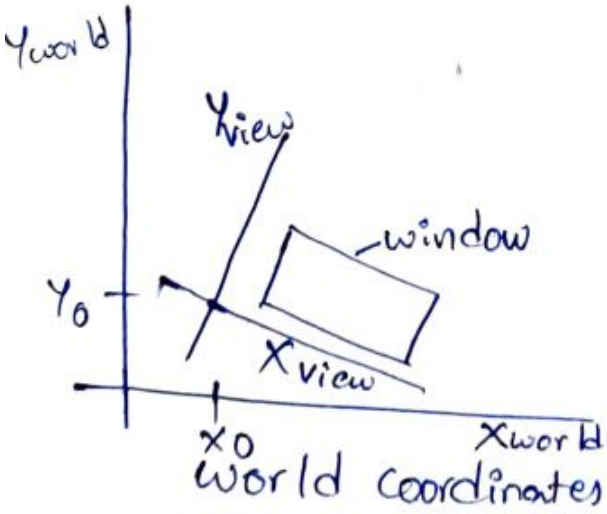
> To obtain particular orientation for window, Set up 2D viewing coordinate s/m. (window in VC)

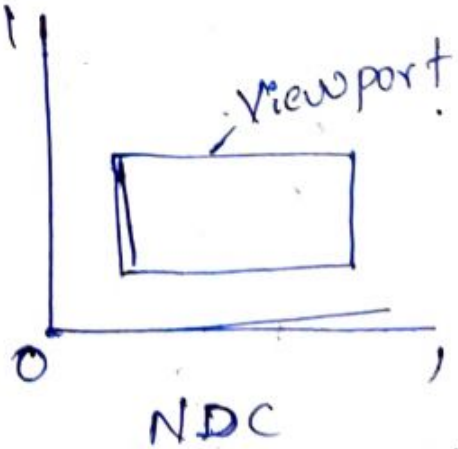
> The viewing coordinate reference frame is used to Provide a method for setting up arbitrary orientation for rectangular windows.

> Once the viewing reference frame is established, transform descriptions from wc to VC.

> Then define a viewport in NVC, ranging 0 to 1, & map VC to NVC

> All parts outside the VC are clipped and contents of VC transfered to DC.





Viewing coordinate reference frame:

> This coordinate sys provides the reference frame for specifying the WC window.

> Viewing coordinate origin is selected at some world position $P_0 = (x_0, y_0)$.

> Then for orientation, rotate the frame. One way is to do this is specify a world vector V called "viewup" vector for defining viewing y_v direction.

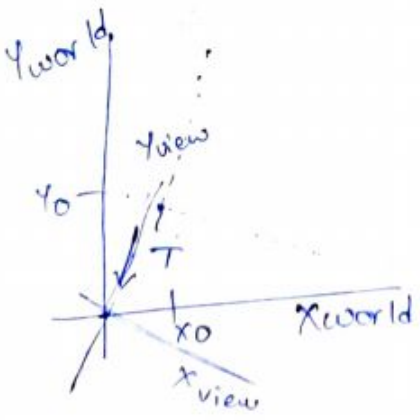
> Steps for defining viewing coordinates

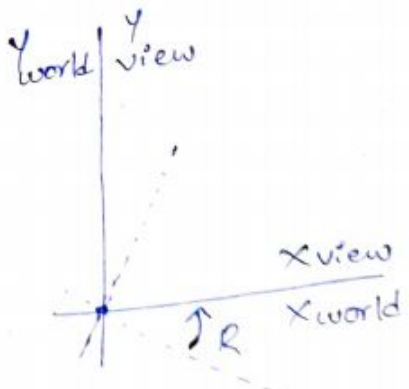
(i) Translate VC origin to the world origin

(ii) Rotate to align 2 coordinate reference frames.

> Composite 2D transformation to convert ~~world~~ WC to VC

$M_{WC,VC} = R \cdot T$ where R is rotation matrix
T is translation matrix





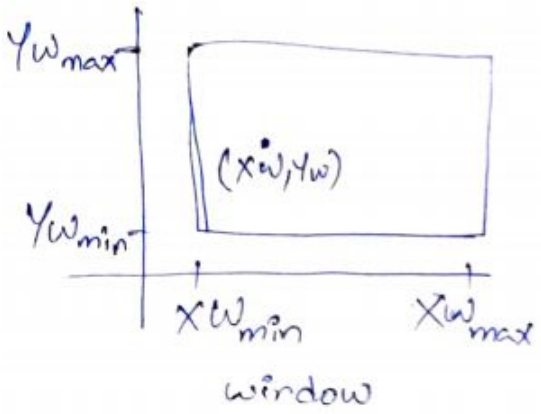
Window-to-Viewport coordinate transformation:

* Objects that are in WC are transfered to VC using transformations that maintain same relative placement in Normalized Space.

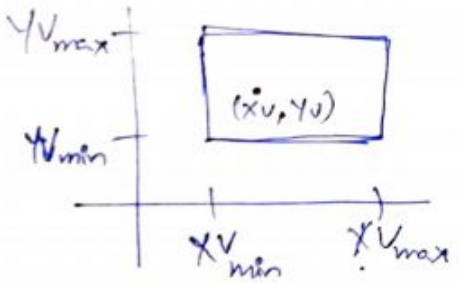
* To maintain the same relative placement in viewport as in window,

$$\frac{XV - XV_{min}}{XV_{max} - XV_{min}} = \frac{XW - XW_{min}}{XW_{max} - XW_{min}}$$

$$\frac{YV - YV_{min}}{YV_{max} - YV_{min}} = \frac{YW - YW_{min}}{YW_{max} - YW_{min}}$$



window



Viewport

* Solving above Expression, viewport position (X_V, Y_V)

$$XV = XV_{min} + (XW - XW_{min})\, SX$$

$$YV = YV_{min} + (YW - YW_{min})\, SY$$

Scaling factors:-

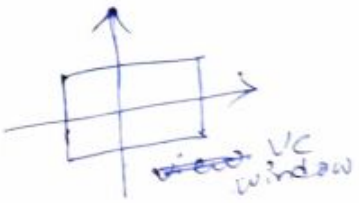
$$SX = \frac{XV_{max} - XV_{min}}{XW_{max} - XW_{min}} \quad ; \quad SY = \frac{YV_{max} - YV_{min}}{YW_{max} - YW_{min}}$$

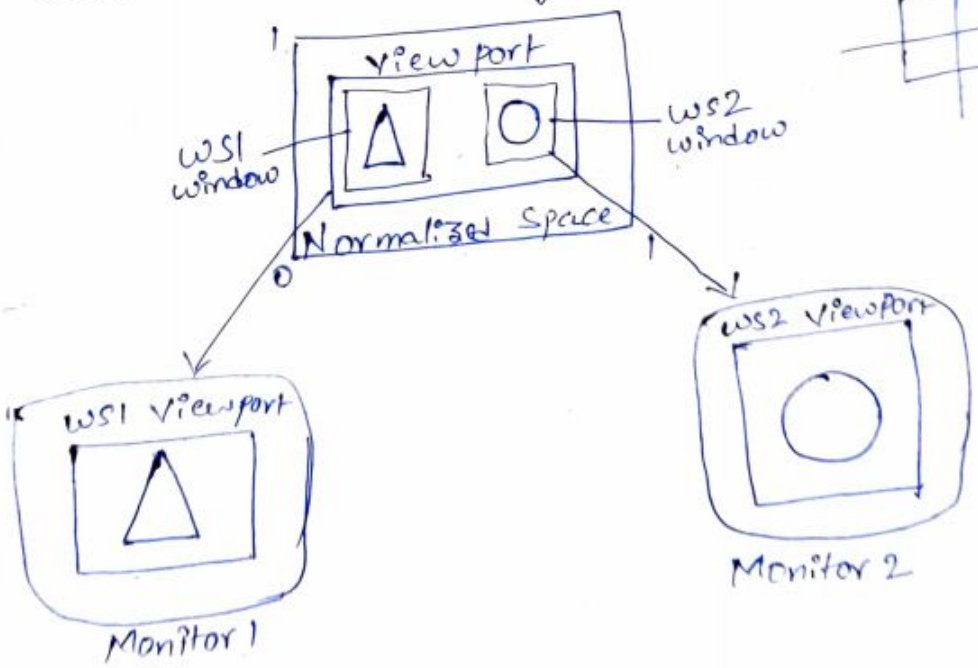
* This convertion can be performed by
  (i) perform a scaling using fixed point position of (Xw_{min}, Yw_{min}) that scales the window area to the size of viewport.
  (ii) Translate the scaled window area to position of viewport.

* Relative proportions of obj's are maintained if the scaling factors are same $(sx = sy)$.

* Sometimes, from NVC, objects are mapped to the various display devices. This is called "workstation transformation"

* This is accomplished by selecting a window area in normalized space and viewport area in the coordinates of display device





Clipping Operations:

* Clipping algorithm is a procedure that identifies those positions of a picture that are either inside or outside of a specified region of space.

* The region against which an obj is clipped is called Clip window.

* Types of clipping
  - > Point
  - > Line
  - > Area
  - > Curve
  - > Text

Line clipping:

~~A line clipping method involves Several parts.~~

A line clipping process have Several steps.

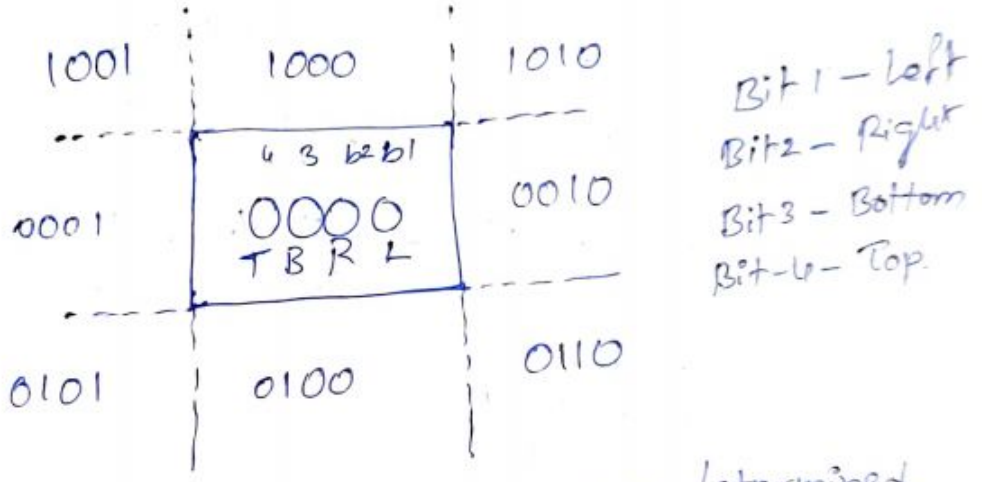
(i) Check if a given line lies inside the clipping window. — accept

(ii) Or check if it is lies completely outside the clipping window — reject.

(iii) If a line lies partially inside & outside, then perform intersection calculations — clip line.

Cohen-Sutherland Line clipping Algorithm:

* The algorithm uses divide and conquer strategy.
* This is one of the oldest & most popular line clipping procedure.
* Every End point in a picture is assigned a 4-bit binary code. called region code.
* Region code identifies the location of relative to boundaries.



* Bit values in the region code are determined by comparing end point coordinate values (x,y) to the clip boundaries.
* Bit 1 is set to 1 if $x < xw_{min}$. Determine other bits similarly
* To determine region codes,
  (i) Calculate Calculate differences between end point coordinates & clipping boundaries
  (ii) Use the resultant Sign bit of each difference calculation to set the coordinate value in region code.

Bit 1 is sign bit of $x - xw_{min}$

Bit 2 is " " " $xw_{max} - x$

Bit 3 is " " " $x - yw_{min}$

Bit 4 " " " " $y_{max} - y$.

* Once ~~determined~~ established region codes for all line endpoints, it is clear that which lines are completely inside, outside & partially inside the window.

* Intersection points with a clipping boundary can be calculated using slope-intersection of line Equ.

$$y = y_1 + m(x - x_1)$$ — (x_1, y_1) & (x_2, y_2) line endpoints (for y intersection)

For x intersection

$$x = x_1 + \frac{y - y_1}{m}$$

Algorithm:

1. ~~Determine interval from~~
2. Given line segment with end point $p1(x_1, y_1)$ & $p2(x_2, y_2)$
3. Compute the 4 bit codes for each end point.
4. If both codes are 0000, (OR is 0000) line lies completely inside the window i pass the end points to draw routine
5. If both codes have a 1 in same bit position (bitwise AND of code is not 0000), the line lies out side the window - trivially reject

4. If a line cannot be trivially accepted or rejected. This line must be clipped.

5. Examine one of the endpoints say $P1 = (X_1, Y_1)$ when a set bit (1) is found, compute the intersection P_1' of corresponding window edge with the line from P1 to P2. Replace P_1 with P_1' & repeat the algorithm.

Polygon clipping

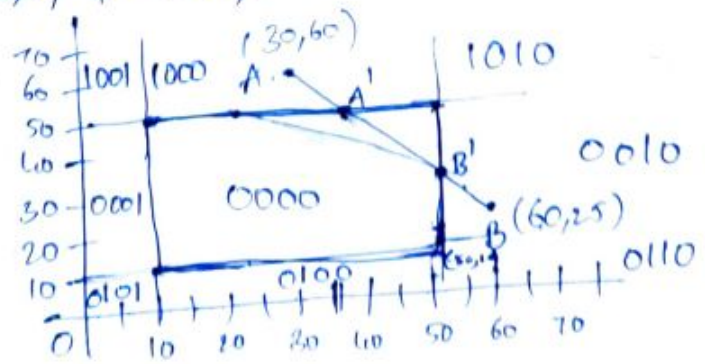
example problems:

1. Apply the cohen Sutherland line clipping algorithm to clip the line segment with coordinates (30, 60) & (60, 25) against the window with $(Xw_{min}, Yw_{min}) = (10,10)$ & $(Xw_{max}, Yw_{max}) = (50,50)$.

Sol: Given line AB is (30,60) and (60,25)

$(Xw_{min}, Yw_{min}) = (10,10)$

$(Xw_{max}, Yw_{max}) = (50,50)$



B'(50, ?)

A'(? 50)

Point A Region code 1000

Point B Region code 0010

0000 → AND operation

Partially Inside

Line Slope $|m| = \frac{|y_2 - y_1|}{|x_2 - x_1|} = \frac{25-60}{60-30} = \frac{35}{30} = 7/6$

Finding A'; As A is across X axis,

$$x = x_1 + \frac{y - y_1}{m}$$

$$= 30 + \frac{50-60}{(7/6)}$$ [∵ y = 50; as y value at clipping window is 50]

$$= 30 - 10 \times (6/7)$$

$$= 30 - \frac{60}{7}$$

$$= \frac{210-60}{7} = \frac{150}{7}$$

A' = 21.2

Finding B'; As B is across y axis,

$$y = y_1 + m(x - x_1)$$

$$= 25 + \frac{7}{6}(50-60)$$ [X = 50 wrt clipping window]

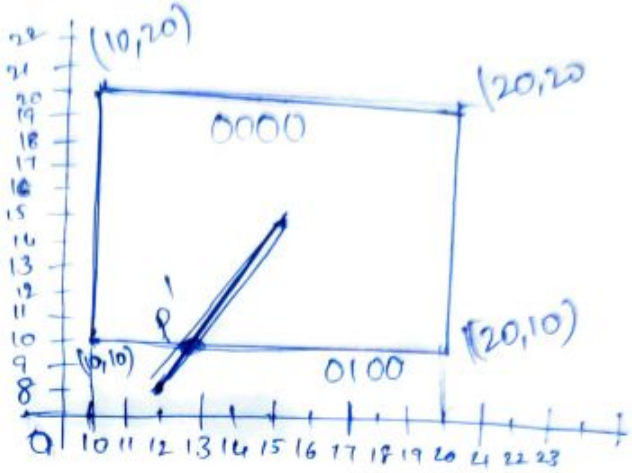
$$= 25 - \frac{70}{6}$$

$$= \frac{150-70}{6}$$

$$= \frac{80}{6}$$

B' = 13.3

(2) Find visible portion of the line between points (10,10), (10,20), (20,20), (20,10) for line (12,8) and (15,15)



$x_1 = 12 \quad y_1 = 8$

$x_2 = 15 \quad y_2 = 15$

$|m| = \frac{15-8}{15-12} = 7/3$

First point region 0100

Second point region 0000

AND- 0000 $\Rightarrow$ Partially Inside

Finding P'; as line is across x axis,

$x = x_1 + (y - y_1)\frac{1}{m}$

$= 12 + (10-8)\ 3/7$ [$\because y = 10$; as clipping window y value is 10.]

$= 12 - 2(3/7)$

$= 12 - \frac{6}{7}$

$= \frac{84-6}{7}$

$= 78/7$

$P' = 11.14$

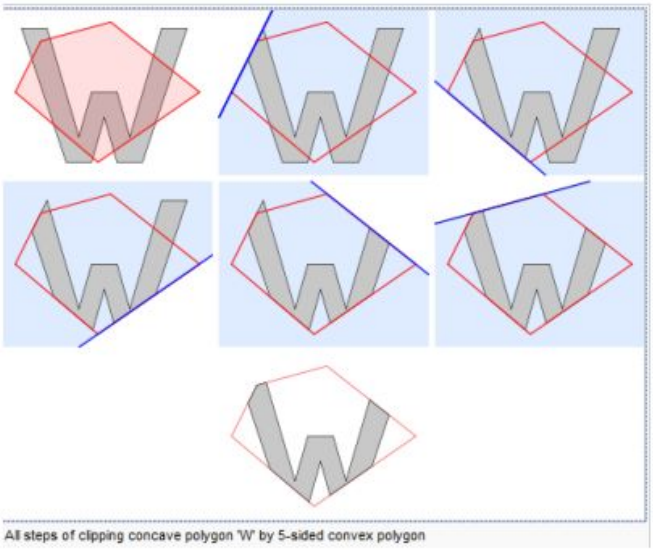
POLYGON CLIPPING METHODS

There are various polygon clipping methods
1. Sutherland-Hodgeman polygon clipping
2. Weiler-Atherton polygon clipping etc.

Sutherland-Hodgeman polygon clipping

The Sutherland–Hodgman algorithm is used for clipping polygons. It works by extending each line of the convex *clip polygon* in turn and selecting only vertices from the *subject polygon* those are on the visible side.

We can correctly clip a polygon by processing the polygon boundary as a whole against each window edge. First clip the polygon against the left rectangle boundary to produce a new sequence of vertices. The new set of vertices then successively passed to a right boundary clipper, a bottom boundary clipper and a top boundary clipper.



All steps of clipping concave polygon 'W' by 5-sided convex polygon

Steps:
1. Clip left – Left Clipper
2. Clip right – Right Clipper
3. Clip bottom – Bottom Clipper
4. Clip top – Top Clipper

The 4 main points are,

1. if the first vertex Is outside the window and the second is inside, both the intersection point on the polygon edge with the window boundary and the second vertex are added to the output vertex list.
2. if both the vertex are inside the window boundary ,then second only added to the output window boundary.
3. if the first vertex is inside and the second is outside ,then only the edge intersection is added to the output vertex list.
4. if both are outside the window ,nothing is added to the output window boundary.