Computer graphics.
Book: computer Graphics
Author: Donald Heam M Pauline baker.

What is computer Graphics?

- Use a computer to create pictures.
- Started early 60s . Ivan Sutherland (MIT)
- CG has many concept.
- Computer scientists create libraries, tools that artist /non techines can use to create pretty pictures.
- Artists use CG tools to creates pretty Pictures.

CG tools:

- CG tools include hardware and software tools.

Hardware Tools:

- Output devices video monitor , printer
- Input devices keyboard , mouse , touch panel,
- Graphics cards/ Accelerator

Software tools:

- Operating system
- Complier
- Editor
- Debuggers
- Graphics library

CG libraries:

- Functions/ routine to draw line or circle etc.
- Ellaborate : pull -down menu, 3D coordinate system etc.

Motivation of CG:

- Appealing picture produce.
- Humans respond better to pictorial information.
- Human brain recognizes visual patterns.
- “If it looks right , it is right”, Jim Blinn (CG pioneer)

Reasons to study CG:

- information presentation.
- Job in CG (Games, movies etc)
- New medium for artistic expression.
- Communicate ideas better.
- Take animation course.

Use of CG:

- Art. Entertainment, publishing
- Movies , TV, books, magazines, game
- Image processing
- Alter images, remove noise
- Processing monitoring
- Large systems or plants
- Display simulations:
- Flight simulators, virtual worlds.
- Computer- aided design, architecture, electric circuit design
- Scientific analysis and visualization.
- Molecular biology(human brain), matlab,

CG Example:

- Biggest CG consumers today are:
- Movies (Hollywood)
- Computer Games: eg Mdden NFL football 2004.

- Animated movies: E.g Toy story, e.g Finding Nemo
- Special effects: e.g spider man 2,Spider man 3,Matrix Reloaded.

Elements of CG:

- Poly lines connected st lines(edger, vertices)
- Text; font, typeface
- Filled regions; colors , patterns.
- Raster images: pixels have values (pixmax)

1.1HISTROY OF COMPUTER GRAPHICS: The fundamental principal and technique derive in the past are still applicable in today's computer graphics technology and generally will be applicable in the future too.

The history of the computer graphics can be study as a chronical development of hardware and software. The evolution of graphics under various terms are expect on following points.

- Crude plotting on hardcopy devices such as teletypes and line printers dates from the early days of computing.
- The whirlwind computer developed in 1950 at Massachusetts institute of Technology (MIT) had computer-drive CRT displays for output, both for operator use and for cameras producing hard copy.
- The SAGE air-defence system developed in the middle 1950's was the first to use command and control CRT displays console on which operators identified targets with the light pens.
- The beginning of modern interactive graphics, however were found in IVAN SUTHER LAND'S seminal doctoral work on the sketchpad drawing systems. He introduced data structures for storing symbol hierarchies built up via replication of standard components (used for drawing circuit symbols). He also developed interaction technique that use the keyboard ad light pen for making choices, pointing and drawing and formulating many other ides and technique still in use today.
- By the mid- sixties, a number of research projects and commercial products had appeared as the potentially of CAD activities in computer, automobile and aerospace grew enormously for automating drafting-insensitive activities. The general motor system for automobile design and Intek Digitek system for lens design were pioneers in showing the efforts utilizing graphics interaction in the interactive cycles common in engineering.
- Due to the high cost of graphics hardware , expensive computing resources, difficulty in wring large interactive program and due to many other reasons the human-computer interaction was still done primarily in both mode using punched cards. After the advent of graphics based personal computers such as Apple Macintosh and IBM PC, the costs of both hardware and software droven down. Millions of graphics computer were sold exclusively for offices and home. Thus the interactive graphics (GUI) as "the window on the computer" became an integral part of PC featuring graphical interaction.

The reason for making interactive graphics affordable was the advent of direct-view storage tube (DVST) which replace buffer and refresh process and eliminated all

flicker in the system. Before this, buffer memory and processors were enough only to refresh at 30HZ and only few thousand lines could be drawn without noticeable flicker.

- Another major hardware advance of late sixties was attaching a display is a minicomputer, relieving heavy demands of refreshed display devices (like user-interaction handling and updating image on the screen) with the central-time sharing computer.
- In 1968, another such devices was invented. The refresh display hardware for geometric transformations could scale, rotate and translate points and lines on the screen at real time; perform the 2D and 3D clipping and could produce parallel and perspective projections.
- The development of inexpensive raster graphics, based on television technology in early seventies contributed more to the growth of the field. The Raster displays stores displays stores display primitives (lines, characters or areas) in a refresh buffer.

The development of graphics cannot be stand-off without the study of graphics input technology. The clumsy, fragile light pen has been replaced by mouse, the tablet, touch panel, digitizers and other devices. Interaction to computer using devices require no knowledge of programming and only a little keyboard use; the use makes choices by selecting menus, icons , check options, places predefined symbols on screen and draws by indicating consecutive end points to be connected by lines or interpolated by smooth curves and fills closed areas bounded by polygons or point contours with shades of gray, colors on various patterns. Now computer graphics have become integral and infamous technology in the computers system.

Application of computer graphics:

Computer graphics started with the display of data or hard copy plotter and CRT screens had grown include the creation, storage and manipulation of mode is of images of objects. These models come from a diverse set of fields and include physical mathematical, engineering , architectural, natural phenomena and so on. There is virtually no area in which graphical displays cannot be used to some advantages.

Today al most all application programs even for manipulating text or numerical data uses graphics extensively in user interface for visualizing and manipulating the application specific objects. Graphical interaction has replaced textual interaction with alphanumeric terminal. Even people who do not use computer encounter computer graphics in TV commercial or special effects. It is an integral part of all computer user interfaces and is indepensible for visualizing 2D, 3D and higher- dimensional objects. We find computer graphics used in a diverse areas as science, engineering , medicine business, industry, government, art, entertainment , education and others.

COMPUTER AIDED DESING (CAD):

In CAD, interactive graphics is used to design components and systems of mechanical, electrical, electromechanical and electronic devices including structures such as buildings, automobile bodies, airplane, VLSI chips, optical systems and telephone and computer networks. The emphasis is on interacting with a computer-based model of the component or

system being designed in order to test, for example its structural, electrical or thermal properties. The model is interpreted by a simulator that feeds back the behavior of system to the user for further interactive design and test cycles.

Some mechanical parts are manufactured by describing how the surfaces are to be formed with machine tools. Numerically controlled machine tools are then set up to manufacture the parts according to these construction layouts.

Architects are interactive graphics method to layout floor plans that shows positioning of rooms, doors, windows, stairs, shelves and other building features. An electrical designer then try out arrangements for wiring, electrical outlets and other system to determine space utilization on a building. The realistic displays then allows architects and their clients to study appearance of building and even go for a simulated "walk" through rooms or around building.

PRESENTATION GRAPHICS:

Another major application areas of computer graphics is the "Presentation Graphics". Presentation Graphics are used to provide illustrations for reports or to generate transparencies for use with graphics.

Presentation Graphics is commonly used to summarize financial, statistical, mathematical, scientific and economic data for research reports, managerial reports and other types of reports. Typical examples are bar charts, line graphs, surface graphs, pie charts and other displays showing relationship between multiple variables.

The 3D graphics are usually used simply for effects, they can provide a more digramatic or more attractive presentation of data relationship.

Computer Art:

Computer graphics is used to generate arts. They are widely used in both fine art and commercial art applications. Fine arts is drawn by artist hand and this kind of art is perfect to the artist skill. Artist use a variety of computer methods including special-purpose hardware, artists paints brush program, other paint packages, specially developed software. Mathematics packages, CAD packages, desktop publishing software and animation packages providing facilities.

Moreover, artists uses a touchpad or a stylus or digitizer to draw pictures. The movement of object is captured by some input hardware. These arts are usually generated by using mathematical functions or algorithms.

Computer art is not as realistic as fine arts. Mostly the commercial art is used to produce animations to demonstrate or present commercial products to the public. Find artists uses a variety of computer techniques to produce images. These images are created using a combination of 3D modeling package, texture mapping, drawing programs and CAD software.

These technique for generating electronic images are also applied in commercial art for logos and other design, page layouts combining text and graphics, TV advertising sports, and other areas. Animations are also used frequently in advertising and TV commercial and produce frame by frame, where each frame of the motion is rendered and saved as an image file.

A common graphics method exployed in many commercials is morphing, where one object is transformed into another.

Education and training:

Computer graphics is used in education and training for making it more effective and more illustrative. E.g if a teacher is to teach bonding of molecules or electron jump form higher energy state to lower energy state or the structure of gene. Then he can demonstrate these concepts using computer graphics software or presentations.

Another example could be taken for surgery. A student can learn surgery using data gloves and realistic computer graphics. This way the cost of education will be low and risk of human life as well. Other examples could be flight simulator and driving simulator for pilot and driving training.

Modes of physical systems, physiological systems, population trends or equipments such as the color coded diagram helping trainees to understand the operation of the system.

Entertainment: Computer graphics methods are new commonly used in making motion pictures, music videos and TV shows. Images are drawn in wire-frame form and will be shaded with rendering methods to produce solid surfaces. Music videos use graphics in several ways. Graphics objects can be combined with the line action.

Computer graphics are also used to introduce virtual characters to movies like character in "Lord of the Rings".

Visualization: Some methods generate very large amount of data/information for example a survey of one lakh people's choice for using different toothpaste generates large amount of data. Analysing the property of the whole amount of data. Analysing the property of the whole amount of data is very difficult. If we try to locate each and every data value. Therefore to visualize large amount of information graphical computer systems are used.

Image Processing: Image can be created using simple point program or can be fed into computer by scanning the image. These picture/ images need to be changed to improve the quality.

Form image/pattern recognition systems, images need to be changed in specified format so that the system can recognize the meaning of the picture. For example scanners with OCR features must have letters similar to standard font set.

Graphical user Interface: GUIs have become key factors for the success of the software or operating system. GUI uses graphical objects called gizmos to represent certain objects or process involved in human computer communication for virtual purpose. Lots of aesthetics (colors) and psychological analysis have been done to create user friendly GUI. The most popular GUI is windows based GUI. The GUI creators are putting having emphasis on 3D GUI creation.

2.0 HARDWARE CONCEPT: Input device are used to feed data or information into a computer system. They are usually used to provide input to the computer upon which reaction, outputs are generated. Data input devices like keyboards are used to provide additional data to the computers whereas pointing and selection devices like mouse , light pens, touch panels are used to provide visual and indication-input to the application.

2.1 Keyboard, Mouse , Light pen, Touch screen and Tablet input hardware:

Keyboard: Keyboard is a primary serial input device. For each key press, a keyboard senses specific codes(American standard code for Information Interchange, ASCII) to the computes. Keyboards are also capable of sending/coding combinations of key press and special keys like function keys. The coding for the combination of key pressing is called "chording". It is useful for expert users in giving exact commands to the computer. Cursor-controlled keys and function keys are used for doing operations in a single keystroke.

Other types of cursor-pointer devices are trackball or joystick. A numeric keypad is after included on the keyboard for fast entry of numeric data.

Several different technologies are used to detect a key depression including mechanical contact closure, change in capacitance and magnetic coupling. Chording is not possible with standard "coded keyboard", which returns only an ASCII code per keystrokes and returns nothing if two keys are pressed simultaneously. An unencoded keyboard can identify of all key press simultaneously allowing chording.

Mouse (Mechanical and optical): Mouse is a serial input device used to select object movement in graphics system. It converts horizontal movement into vertical movement. Primary principle of curser movement is to detect the amount of displacement of mouse in horizontal plane. This displacement is mapped into screen. Generally the mapping ratio is 1:8 . According to their working mechanism tow types of mouse are there. They are:

- Mechanical mouse and
- Optical Mouse.

Mechanical mouse contains two rollers. One rollers count horizontal displacement and another roller counts vertical displacement. The motion of the roller in the base of mechanical mouse is converted to digital values that are used to determine the direction and magnetic of movement.

Optical mouse uses a special mouse pad with a matrix of Black and white grid of lines. The matrix reflects light from the LED on the bottom of the mouse. The intensity/quality of reflected light varies as Black and white lines reflects it alternatively. The alternate reflections are used to calculate the relative position change.

Advantages:

- None of the part of computer screen is obscure (hidden from head).
- Movement becomes easier as had rests at a desk for movement.
- Direct-point- and –click.
- Easy to use and understand.

Light Pens: Light pens are pencil shaped devices used to select screen positions by detecting the light coming from points on the CRT screen. They are sensitive to the short burst of light emitted from the phosphor coating as the instant electron beam strikes a particular point. Other light sources , such as background light in the room are usually not detected by a light pen.

An activated light pen pointed at a spot on the screen as the electron beam light. Some of that spot generates an electrical pulse that causes the co-ordinate position of the electron beam to be recorded.

Disadvantages:

- When used for long time, leads to an arm pain.
- Some part of the screen will be obscure by hand.
- Cannot read black pixel position.
- Reading might be errorous if background light has very high intensity.

Touch-Panel: Touch panels are a sensitive surface that is used to point directly. The panel can be touched by finger or any other object like stylus Transparent touch panels are integrated with computer monitor for the manipulation of information display.

A basic touch panel senses voltage drop when a user touches the panel. It knows where the voltage has dropped and accordingly calculates the touch position.

- Resistant based touch panel uses two substances of glass or plastic separates by insulators. When a user touches the panel these two substances are connected and the resistance at that point will drop down. The resistance drop is sensed and the touch position is calculated.
- In capacitance based touch panel, some charge is spread on the screen. When user touches the panel the charge is drawn by finger form each side proportionally. The sensor calculates the charge in frequency of charge/voltage and find out the touch position.
- In Acoustic(Sound) based touch panel uses very high frequency (5 Mhz). The sound waves are generated from X-axis and Y-axis sides are reflected from opposite side of either axis. When user touches the panel, the sound waves are interrupted and reflected back from their mid wave. The sensors on the X-axis and Y-axis sides measured the time, the sound waves took to get reflected back and estimate the touch position.
- An optical touch panel uses Infra red (IR) emitters and sensor on X and Y-axis sides, emitter are placed and on the apposite sides sensors are placed when a user touches the screen, the point will interrupt two IR beam from X and y-axis. This is detected by IR sensors and the touch position is calculated.

Advantages:

- Touch panel can be mounted an display screen leaving more space on desktop but mouse, joystick etc. take some space.

Tablets (Electronic, Sonic and Resistive): A common device for drawing, painting or interactively selecting co-ordinate position on an object is a digitizer. Typically a digitizer is used to scan over a drawing on object and to input a set of discrete co-ordinate position which can be joined with straight line segments to approximate the curve surface shape.

One type of digitizer is Graphics tablets (data tablets), which is used to input two dimensional (2D) co-ordinate by activity a hand curser or stylus as selected position, flat surface. A had curser contains cross hairs for sighting position, while a stylus is a pencil shaped device that is pointed to the position of the tablet. Many graphics tablets are constructed with a rectangular grid of wires embedded on the tablet surface. Electromagnetic pulses are generated in sequence along the wires and an electric signal is induced in a wire coil. In an activated stylus or hand curser to record tablet position.

Acoustic (Sonic) tablets use sound waves to detect a stylus position. Either strip microphones or point microphones are used to detect the sound emitted by an electrical spark from a stylus tip. The position of the stylus is calculated by calculating the arrival time of generated sound at different microphone position.

Application:

- Digitizing a drawing in a book.

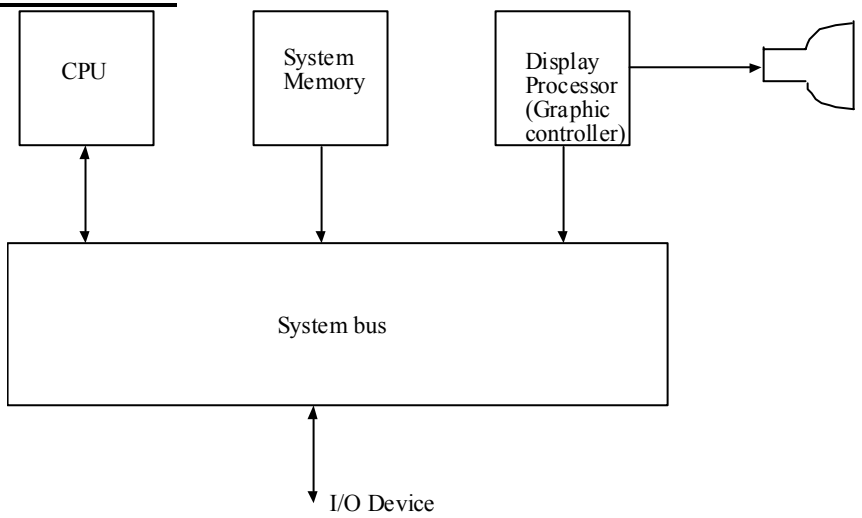
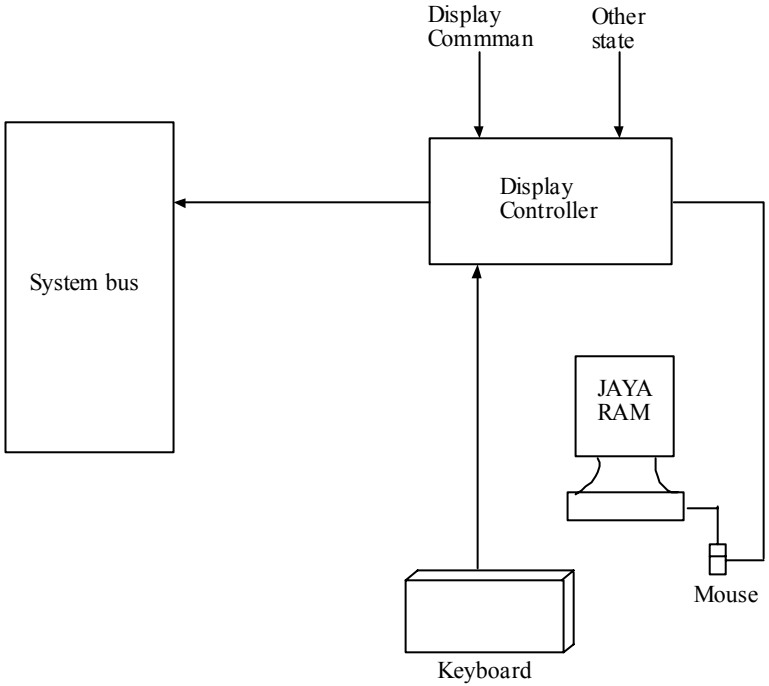


Fig. Architecture of a simple random scna system

2.2 Raster and vector display architecture:

Random Scan (or vector display system or Stoke writing or Calligraphic systems):

- In a random scan display, an image on the screen is drawn with one line at time and for this reason it is also called vector display.
- A typical organization a vector system is shown below.
- An application program is input and stored in the system memory along with the graphic package.



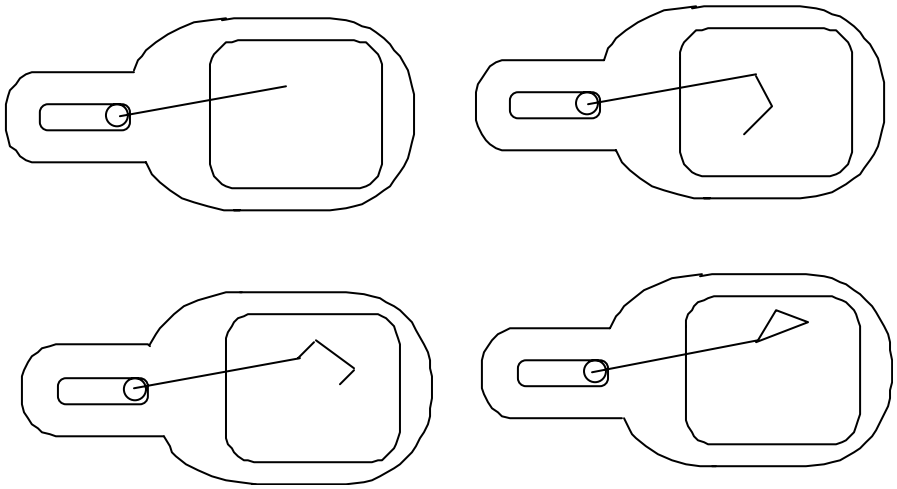


Fig. Random scan system drawing through the component lines of an object in any order specified.

- Graphics commands in the application program are translated by the graphic package in to a display file stored in the system memory.
- This display file is then accessed by the display processor to refresh the screen.
- Sometimes the display processor in a random scan is referred to as a display processing unit or a graphics controller.
- The buffer stores the computer produce display list or display program which contains points and line plotting commands with (x,y) and end point co-ordinates as well as character plotting commands.
- The commands for plotting points lines and characters are interpreted by the display processor.
- It sends digital and points co-ordinates to a vector generator that converts digital co-ordinates values to analog voltages for beam deflection circuits that display an electron beam writing on CRT phosphor coating.
- The main principal of the vector system is that the beam is deflected form end point to end point as detected by arbitrary order of the display commands term as random scan.
- Since the light out put of phosphor decays in tens or hundreds of microseconds, the display processor must cycle through the display list to refresh the phosphor at least 30 times per seconds (Hz) to have flicker hence the buffer holding display list is usually called a refresh buffer. (Note: jump instruction in the figure above in refresh buffer is to provide cyclic refresh.
- A CRT beam in this system is adjusted in such a way that electron beam only hits the spot where the graphics is to be drawn.
- Thus the refresh rate in this system depends upon the number of lines to be displayed.
- Random scan display are design to draw all the component lines of pictures 30 to 60 times per seconds.
- Vector display system are mostly used for line drawing applications and can not display realistic shaded scenes.
- It produces smooth line drawing because the CRT beam directly follows the line path definitions that are stored in the form of line drawing commands.

Disadvantages:

- When the number of command in the buffer goes high the system take long time to process and draw pictures.
- Cannot apply shading features.

Raster Display:

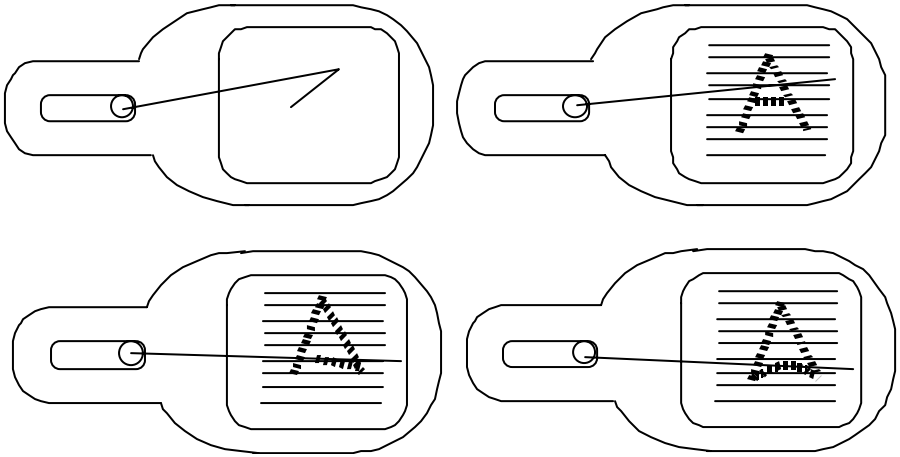


Fig. A raster scan system displays an object as a set of discrete points across each scan line.

- The raster graphics developed in early 70's.
- It is common CRT monitor.
- It is based on TV technology.
- In this system, electron beam swaps across the screen one row at a time form top to bottom.
- Beam intensity vary (on or off) by the movement of electron beam across each room.
- Picture definitions are stored in refresh buffer or frame buffer which holds the setup intensity values for all the screen point also known as pixel. (pel)
- Stored intensity values are retrieved form the refresh buffer and painted on the screen on scan line (one row at a time).
- It is suited for realistic scenes containing shading and color pattern.
- In monochrome monitor frame buffer consists of one bit per each pixel and for color monitor frame buffer consists of 24 bits for each pixel.
- The refresh rate for this system is at the rate of 60 to 80 frame per second. i.e refresh rate are described in units of cycle per second.

Raster Scan Display:

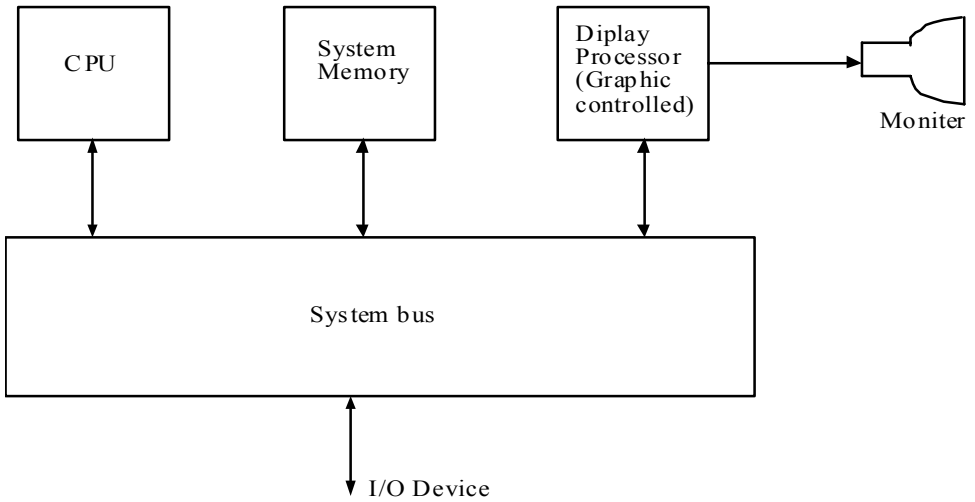


Fig: Raster scan display

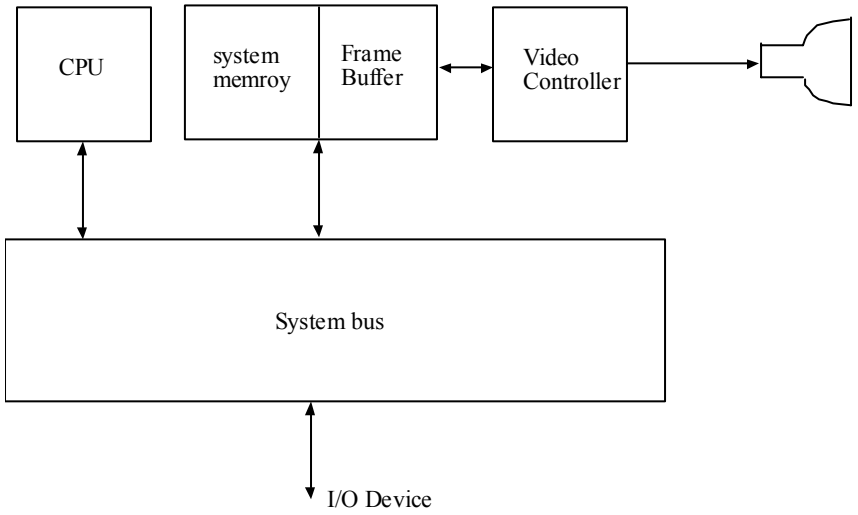


Fig. Architecture of a raster system with a fixed positon of the system memory requred for the frame buffer

- Interactive raster graphic generally employ several processing units.
- In addition to the CPU a special buffer processor called video controller or display controller is used to control the operations of display device.
- Frame buffer can be any where in the system memory, and the video controller accesses the frame buffer to refresh the screen.
- A fixed area of system memory is reserved for the frame buffer.
- Video controller direct access the frame buffer.
- Frame buffer location and corresponding screen position are reference in Cartesian co-ordinate.
- Generally the co-ordinate origin is defined at the lower left screen corner.
- Positive X value increases to the right and positive Y value increases from bottom to top.

- Scan line are labeled from Y_{max} at the top of screen to zero at the bottom.
- Scan line are labeled from 0 to X_{max} .

Video Controller:

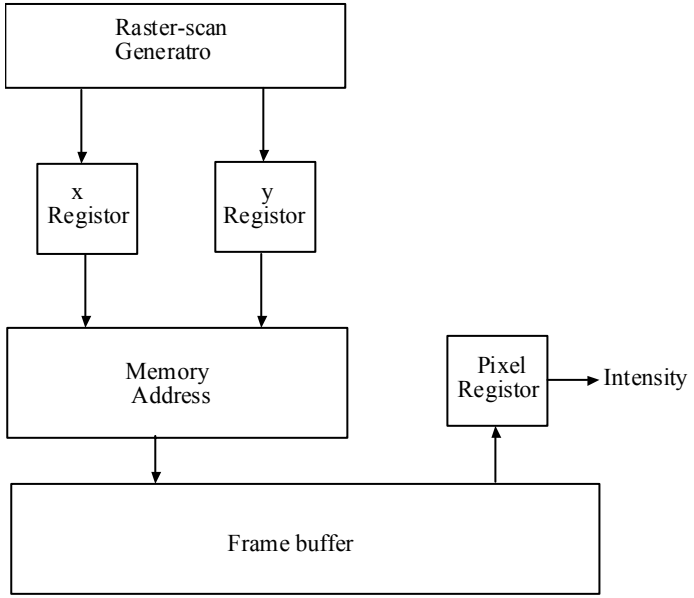


Fig. Basic video controller Refresh operation.

- Two register are used to store the co-ordinates of the screen pixel.
- Initially the X register is set 0 and Y register is set to y_{max} .
- The value stored in the frame buffer for this pixel position is then retrieved and then used to set the intensity of CRT beam.

- Then the x register is increased by one and process repeated for the next pixel in the top scan line.
- This process is repeated for each pixel along line.
- When the last pixel of the top scan line has been processed the x register is reset to zero and y register is decremented by 1 pixels on the screen lines are processed against inturn; and the process is repeated for each successive scan line.
- After cycling though each pixel along the bottom scan line (y =0) the video controller resets the register the first pixel poison on the top scan line and refresh process starts over.
- Screen must be refresh at least at the rate of 60 frames per second.
- To speed of the pixel processing video controller can retrieve multiple pixel values from the refresh buffer on each pass. The multiple pixel intensity are then stored in a separate register and used to controlled the CRT bit intensity for a group of adjacent pixel. When that group of pixel has been processes the next block of pixel values is retrieved from the frame buffer.
- Besides these refresh operation video controller also performs different operation video controller retrieved pixel intensity from different memory area on different refresh cycle.
- In high quality system, for example , two frame buffers are often provided so that gun buffer can be used for refreshing while the other is being filled with intensity values.
- This provides a fast mechanism for generating real time animations seans different views of moving object can be successively loaded in the refresh buffer.
- Video controller also contain a look up table instead of controlling CRT beam intensity directly. This provides a fast mechanism for changing screen intensity values.

Random Scan display Processor:

- Figure given below shows the architecture of raster graphic system with a display processor.
- It contains a separate display processor the purpose of the display processor is to free the CPU form graphics loads. It is also called graphic controller or display co-processor.

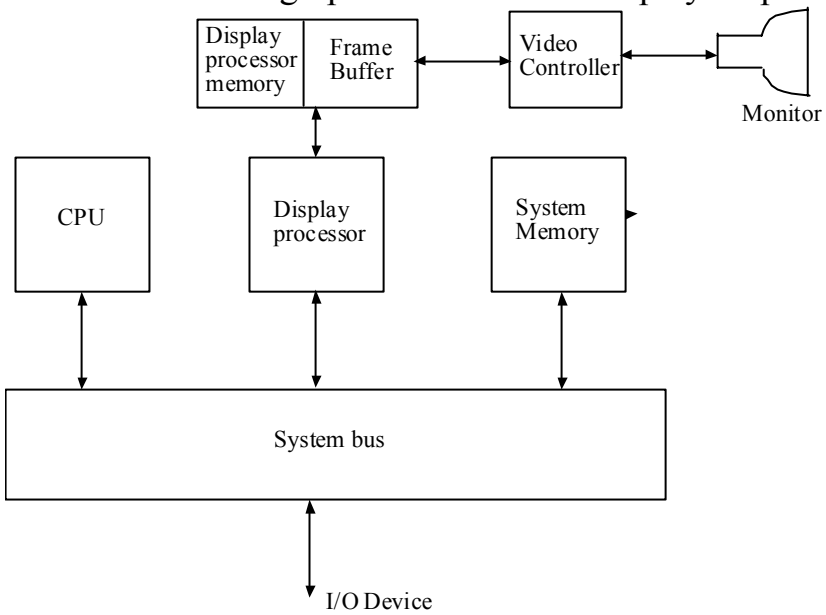


Fig. Architecture of raster graphics system with a display processor

The system memory holds the data plus those program that execute on CPU: the application program, graphics package and operation system. Similarly display processor holds data plus the programs that perform scan conversion and the raster operation.

The frame buffer contain the displayable image created by the scan conversion and raster operation.

- It has also a separate display processor or memory area in addition to system memory.
- The major task of display processor is digitizing a picture definition given in an application program into a set of pixel intensity values for storage in the frame buffer.
- This process is called scan conversion.
- Graphic commands specifying straight line and another geometric objects are scan converted into a set of discrete intensity point.
- Can converting a straight line sequence means that we have to locate the pixel position close to the line path and store the intensity for each position in a frame buffer.
- Similar methods are used for scan converting curved lines and polygon outlines.
- Character can be defined with rectangular grids or they can be defined as curved outlines.
- The array size of character grids can vary from about 5 by 7 to 9 by 12 or more for high quality display.
- Character grid is display by superimposing the rectangular grid pattern into the frame buffer at a specified co-ordinate position.
- Display processor is also design to perform the number of additional operation.
- It is used for various liens style (Dashed, dotted or solid), displaying color areas and perform certain transformation and manipulation on displayed objects.

Comparison between raster and vector graphics:

1. Since the picture definition is stored as the set of line-drawing instructions and not as a set of intensity values foe all screen points, vector displays generally have higher resolution than raster systems. For these reasons, random-scan systems are designed for line-drawing applications and cannot display realistic shaded scenes. Moreover, vector displays produce smooth line drawings because CRT beam directly follows a line path but raster system in contrast produces jagged lines that are plotted as a discrete point sets.
Figure below illustrates this:

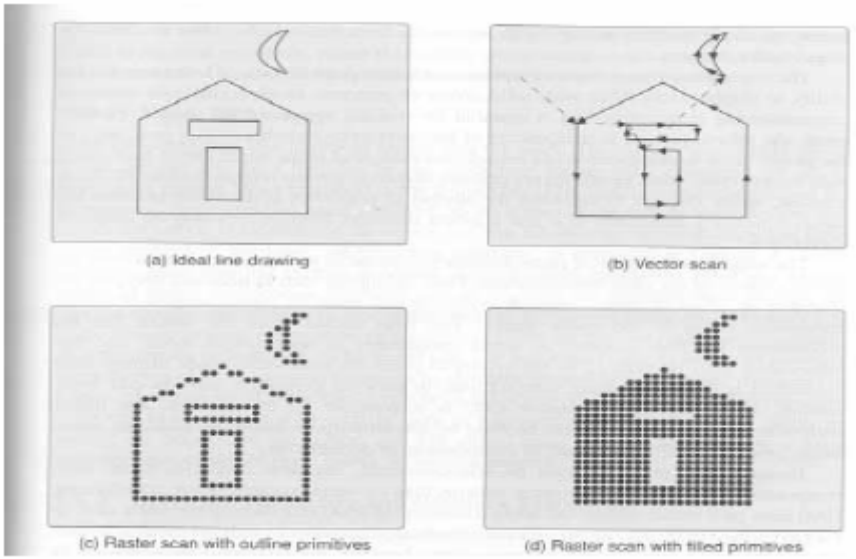


Fig.: Various Scan Methods

2. Raster scan is easier and less expensive to implement than in

random scan system, whose vector generators must be highly accurate to provide linearity and repeatability of beam's deflection.

3. One major advantages of raster graphics over vector graphics includes the ability to display areas filled with solid colors or patterns. Moreover, the refresh process is independent of complexity (i.e. no. of lines) of image and each pixel in buffer can be read out on each refresh cycles, allowing flicker free display.

4. Major disadvantage of raster system compared to vector system is the discrete nature of the pixel representation. This happens due to scan-conversion of end points (vertices) into their component pixels in frame buffer. This can be shown in figure below:

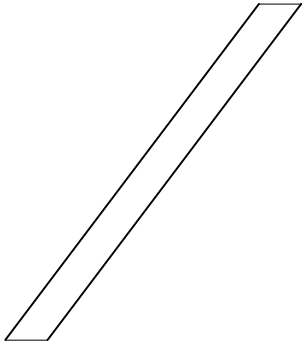


Fig (a): Actual thick

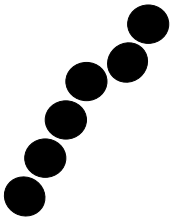


Fig (b): Scan – Converted

Fig. Discrete Pixel Representation

5. Another drawback of raster system arises from the nature of raster itself. A vector system can draw continuous, smooth line (curves) from any point on the CRT face to other, the raster system displays these lines (curves) mathematically using various algorithms causing problems of "jaggering" or "staircasing".

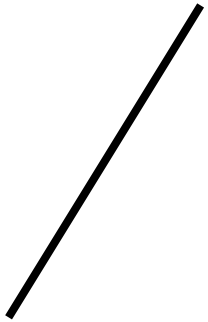


Fig. (a): Actual Line

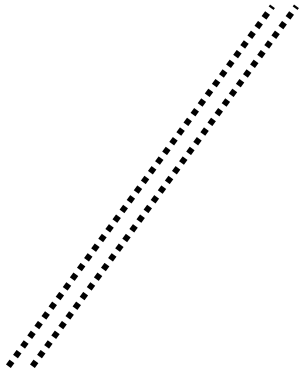


Fig. (b): Jagged Raster Line

Architecture of Graphical display terminals:
Colored CRT monitor

The CRT displays color picture by using the combination of phosphorous that emits different color light . By combining the emitted light from the different phosphorous range of color can be generate. Two basic technique for producing color display with CRT are:

1. Beam penetration method:
2. Shadow-mask Method. (Roster).

Beam penetration method: This is a random scan method in this method a line is created. CRT beam is adjusted in such a way

that electron beam only hits the spots where picture is to be drawn. Two phosphorus layer (Red and green) are adjusted such that outer layer should red and inner layer should be green. The color picture depends on how far the electron beam penetrates. The slow electron beam only excite outer red layer. The high speed electron beam excite inner green layer. At the intermediate speed, the combination of red and green lights are emitted to show two additional colors orange or yellow.

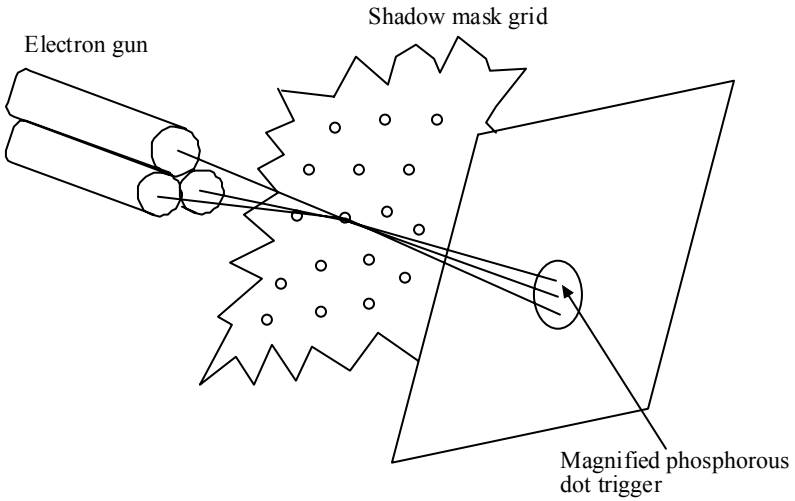


Figure: Operation of delta shadow mask method

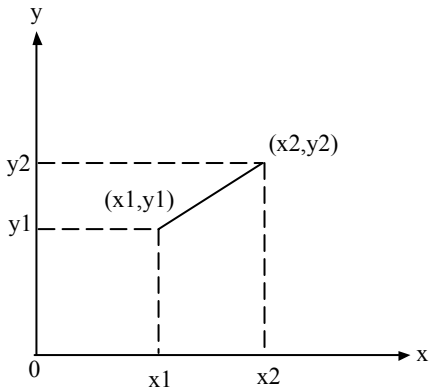
Operation of delta-delta mask CRT. These electron guns aligned with the triangular color dot patterns on the screen, are directed to each do triangle by a shadow mask.

It is used in raster scan system object is the set of discrete point across each scan line. Three types of phosphorus dots is coated in each pixel that is red, green and blue. Just behind the CRT screen shadow mask grid is placed whose number is same as a number of pixel. Three electron gun is adjusted for each phosphorous. The electron beams is detected and focus on the shadow mask hole. The Passed from the hole activate the dot triangle and produce the color spot on the screen. The color of pixel is controlled by light of intensity. By combing three color, 17 million color can be obtained. For true color each pixel has 24 bits in the frame buffer.

Chapter : 3

Line Drawing Algorithm:

Scan line Algorithm: The Cartesian slope equation of a straight line as, $y = mx+b$ ……(i)
When m represents the slope of the line and b as the y –intercept.



Suppose two end points of a lien segment at positions (x_1,y_1) and (x_2,y_2) are shown in fig.

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

b = y – mx …….(iii)

For any given x-interval △x along the line, we compute the corresponding y- intercept △y form equation (ii).

△y = m △x ……(iv)

$\triangle x = \frac{m}{\Delta y}$

These equations form the basis of determining deflection voltage in analog devices.

Case: I

For $|m| < 1$,

Then △x can be proportional to a small horizontal deflection voltage and the corresponding vertical deflection is set to △y as calculated from equation (iv).

Case: II

For $|m| > 1$

Then, △y can be set proportional to a small vertical deflection voltage with the corresponding horizontal voltage set proportional to △x calculated from equation (V)

Case III

When $|m| = 1$, then △x = △y and then horizontal and vertical voltages equal.

Comments:

1. It uses multiplication
2. It uses float operation.

DDA (Digital differential Analyzer):

This algorithm samples the line at unit interval in one-coordinate and determines corresponding integer values nearest the line path for other co-ordinates.

The equation of the line is,

Y = mx+b……(i)

$m = y_2-y_1/x_2-x_1$ ……..(ii)

For any interval △x , corresponding interval is given by △y = m △x.

Case I :

$|m| < 1$, we sample at unit x interval i.e △x = 1.

$X_{k+1} = x_k+1$ ……..(iii)

Then we compute each successive y-values,

△y = m

$Y_{k+1} = y_k + m$ ……..(iv)

Case: II

$|m| > 1$, we sample at unit y-interval i.e △y = 1 and compute each successive x-values.

Therefore, 1 = m △x

△x = 1/m

$X_{k+1}- x_k = 1/m$……..(v)

$Y_{k+1} = y_k + 1$…….(vi)

Above equation hold for the lines processed from left end to right end. For the reverse process i. e if the line is to be processed form right to left then,

For $|m|<1$, $\triangle x = -1$

$X_{k+1} = x_k - 1$

$Y_{k+1} = y_k - m$

For $|m| > 1$, $\triangle y = -1$

$Y_{k+1} = y - k_{+1}$ .....(ix)

$X_{k+1} = x_k - 1/m$.....(x)

Therefore , in general,

$Y_{k+1} = y \pm m$

$x_{k+1} = x_k \pm 1$ for $|m| < 1$

$y_{k+1} = y_k \pm 1$

$x_{k+1} = x_k \pm 1/m$ for $|m| > 1$

Eg. ***Example:*** *Digitize a line with end points (10,15) and (15, 30).*

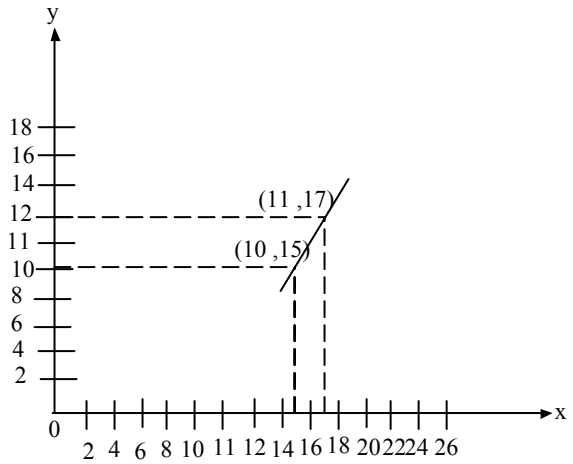
Solution:

The slope of line is

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{30 - 15}{15 - 10} = \frac{15}{3} = 3$$

$|m| > 1$

So we sample at y interval. The formula is given by $x_{k+1} = x_k + 1/m$.



S.N	x	y
1	10	15
2	10	16
3	11	17
4	11	18

Comments:

1. It is faster to calculate pixel position.
2. Due to propagation of round off errors due to successive addition the calculated pixel may shift among from the true line path.

Algorithm:

1. Declare the variables, x_1,y_1 and x_2 , y_2 dx, dy ,del x, del y as real and k as integer.
2. Perform
   dx = x2-x1
   dy = y2 – y1
3. Test if |dy|<|dx| then
   Steps = |dx|

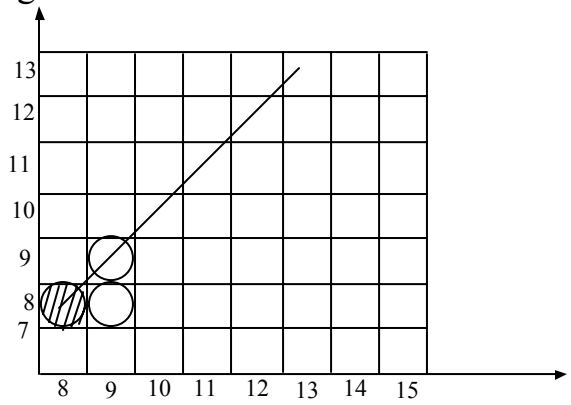
Else steps = |dy|

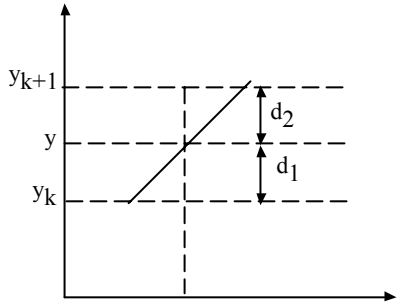
4. set del x = dx/steps
   del y = dy/steps
   x= x1
   y = y1
   5. Plot (x, y)
   6. Do for k = 1 to steps
   x = x+ delx
   y = y +del y

Plot (x,y)
End do.

3.Bresenhan's line Algorithm (BLA) :

DDA includes calculation related to m and 1/m which is little complicated. Bresenhan's improves DDA algorithm by only involving integer calculation. .





Bresenhan's Algorithm selects the best pixel co-ordinate by testing the sign of an integer parameter whose value is proportional to the difference between the separation of two pixel actual line path.

Case I:

$|m| < 1$ m >0

Let (x_k, y_k) be the pixel position determined then the next pixel to be plotted is either (x_{k+1},y_k) or (x_{k+1},y_{k+1})

Let d_1 and d_2 be the seperatio of pixel position (x_{k+1}, y_k) and (x_{k+1}, y_{k+1}) from the actual line path.

Y = mx +b

Then, at smapling position (x_{k+1})

$Y = m(x_{k+1}) + b$

From fig.

$d_1 = y - y_k$

$d_2 = y_{k+1} - y$

$d_1 - d_2 = (y - y_k) - (y_{k+1} - y)$

Let us define a decision parameter P_k for the k^{th} step by

$P_k = \Delta x (d_1 - d_2)$

$\triangle x > 0$,
Therefore, $P_k < 0$ if $d_1 < d_2$
$P_k \geq 0$, if $d_1 > d_2$

$P_k = \triangle x\{ y-y_k -(y_{k+1} - y)\}$

$= \triangle x\{ y-y_k- y_k-1+y\}$ since, $y_{k+1} = y_k+1$

$= \triangle x\{ 2\{ m(x_k+1)+b\} - 2 y_k - 1\}$

$P_k = \triangle x\{2mx_k + 2m + 2b-2y_k-1\}$ since, $x_{k+1}= x_k+1$

$P_k = 2mx_k \triangle x - 2y_k \triangle x +(2m+2b-1) \triangle x$

$P_k = 2. (\triangle y/\triangle x).x_k \triangle x -2y_k \triangle x +(2m+2b-1) \triangle x$

$P_k = 2 \triangle y x_k - 2 \triangle x y_k +c$

Where, C = $(2m+2b-1) \triangle x$ is a constant.
Now, for next step,

$P_{k+1} = 2 \triangle x x_{k+1} - 2 \triangle x y_{k+1} +c$……(ii)
From (i) and (ii)

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

i.e $p_{k+1} = p_k +2 \triangle y - 2 \triangle x (y_{k+1} -y_k)$
Where,
$Y_{k+1} - y_k$ = 'o' or '1'
If $p_k < 0$, then we plot lower pixel
$Y_{k+1} = y_k$ ……(iii)

$P_{k+1} = p_k +2 \triangle y$ ……(iv)

If $p_k \geq 0$ then we plot upper pixel.
Therefore, $y_{k+1} = y_k+1$……..(v)

$P_{k+1} = p_k + 2 \triangle y - 2 \triangle x$ ……(vi)
Therefore, Initial decision parameter.

$P_0 = 2 \triangle y x_o - 2 \triangle x y_o + c$ [from (i)]

$= 2 \triangle y x_o - 2 \triangle x y_o +(2m+ 2b-1)\triangle x$

$= 2\triangle y x_o - 2 \triangle x y_o + 2 m \triangle x + 2b \triangle x - \triangle x$

$= 2 \triangle y x_o - 2\triangle x y_o + 2. (\triangle y/\triangle x).\triangle x + 2 (y_o - mx_o)\triangle x - \triangle x$

$= 2 \triangle y x_o - 2 \triangle x y_o + 2 \triangle y + 2 \triangle x y_o + 2 \triangle y/\triangle x. x_o \triangle x - \triangle x$

$P_o = 2 \triangle y - \triangle x$

#. Digitize a line with end points (15, 18) and (10,15) using BLA
Solution:

$\triangle y = /15-18/ = 3$

$\triangle x = /10-15/ = 5$
$|m| < 1$, we sample at x direction.
First pixel ,(10, 15)

$P_0 = 2 \triangle y - \triangle x$
$= 2 \times 3- 5$
$= 1$
We know that ,
If $p_k < 0$

$P_{k+1} = p_k+2\triangle y$
$y_{k+1} = y_k$
if $p \geq 0$

$P_{k+1} = p_k + 2 \triangle y - 2 \triangle x$
$Y_{k+1} = y_k+1$

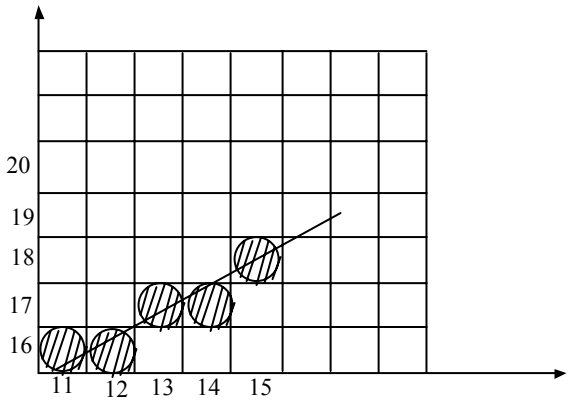
We have $p_k \geq 0$,

So, $p_1 = p_0 + 2\ \Delta y - 2\ \Delta x$
$= 1 + 2 \times 3 - 2 \times 5$
$= -3$

$P_2 = p_1 + 2\ \Delta y$
$= -3 + 2 \times 3$
$= 3$

$P_3 = p_2 + 2\Delta y - 2\ \Delta x$
$= 3 + 2 \times 3 - 2 \times 5$
$= 9 - 10$
$= -1$

K	**P_k**	**x_{k+1} , y_{k+1}**
0	1	11, 16
1	-3	12, 16
2	3	13, 17
3	-1	14, 17
4	5	15, 18



Digitize a line with end points (15, 15) and (10, 18) use BLE and DDA.

Solution:

$\Delta y = /15-18/ = 3$

$\Delta y = /\ 10-15/ = 5$

$m = 3/\ 5 = 0.6$

$|m| < 1$, we sample at x direction,

First pixel , (10, 15)

$P_0 = 2\ \Delta y - \Delta x$
$= 2 \times 3 - 5$
$= 1$

We know that,

If $p_k \geq 0$

$P_1 = p_o + 2\ \Delta y - 2\ \Delta x$
$= 1 + 2 \times 3 - 2 \times 5$
$= -3$

$P_k < 0$

$P_2 = p_k + 2\ \Delta y.$
$= -3 + 2 \times 3$
$= 3$

***Note**:*
For, $P_k \geq 0$,
$P_{k+1} = p_k + 2\Delta y - 2\Delta x$
$y_{k+1} = y_k - 1$ (when the slope is –ve)

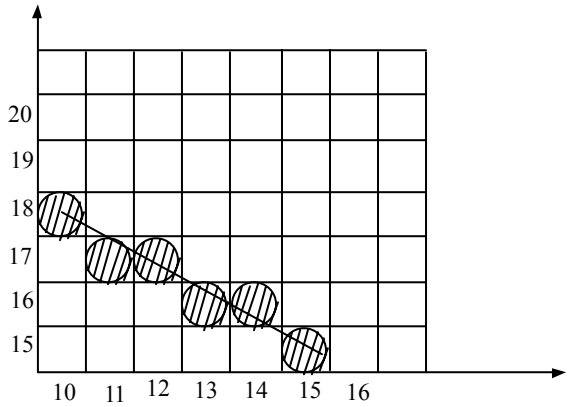
$P_k < 0$
$P_{k+1} = p_k + 2\Delta y$
$Y_{k+1} = y_k$

$P_3 = p_2 + 2\ \Delta y - 2\ \Delta x$
$= 3 + 2\times 3 - 2\times 5$
$= -1$

$P_k < 0$

$P_4 = p_3 + 2\ \Delta y$
$= -1 + 2\times 3$
$= 5$

K	p_k	x_{k+1}	y_{k+1}
0	1	11	17
1	-3	12	17
2	3	13	16
3	-1	14	16
4	5	15	15



Bresenham's line algorithm:

1. Input the tow line end points and store the end point in (x_0, y_0)
2. Load (x_0, y_0) into the frame buffer; that is plot the first point.
3. Calculate constants Δx, Δy , $2\ \Delta y$ and $2\ \Delta y - 2\ \Delta x$ and obtained the starting value for the decision parameter as ; $p_o = 2\ \Delta y - \Delta x$
4. At each x_k along the line, starting at k= 0 , perform the following test:

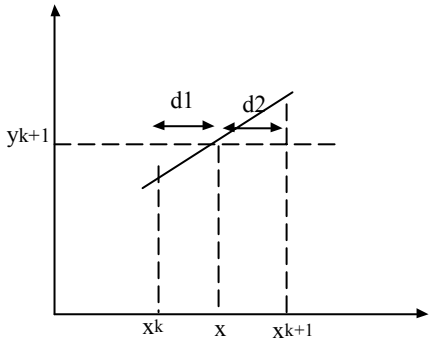
If $p_k < 0$ the next point to plot is (x_{k+1}, y_k) and $p_{k+1} = p_k + 2\ \Delta y$.

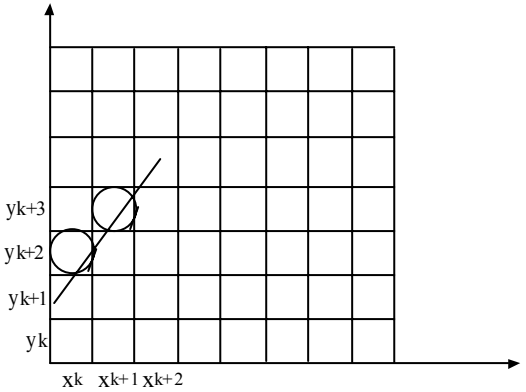
Otherwise , the next point to plot is (x_{k+1}, y_{k+1}) and $p_{k+1} = p_k + 2\ \Delta y - 2\ \Delta x$

5. Repeat step 4 Δx times.

 # Digitize a line with end point s(20, 10) and (30, 18). The slope of the line is 0.8 using BLE.

Case:II $|m| > 1$





Let (x_k, y_k) be the pixel position determined then the next pixel to the plotted is either (x_{k+1}, y_{k+1}) or (x_k, y_{k+1})
Let d1 and d2 be the separation of the pixel positons (x_k, y_{k+1}) and (x_{k+1}, y_{k+1}) for the actual line path.
$Y = mx + b$
The actual value of x is given by
$x = (y-b)/m$
Sampling position at y_{k+1}

Form figure,
$d1 = x - x_k$
$d2 = x_{k+1} - x$
Let us define a decision parameter p_k and P_k is defined by,

$P_k = \Delta y (d1- d2)$

$\Delta y > 0$
$P_k < 0$ if $d1 < d2$
$P_k \geq 0$, if $d1 \geq d2$

$P_k = \Delta y (d1 - d2)$

$= \Delta y (x- x_{k+1}+x)$

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

$= \Delta y (2x - 2x_k + 1)$ [since, $x_{k+1} = x_k+1$]

$= \Delta y\{2\left(\frac{y_{k+1} - b}{m}\right) - 2x_k - 1\}$

$= \Delta y/m\{2(y_{k+1}-b) - 2x_k m - m\}$

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

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

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

$P_k = 2 \Delta x y_k - 2 \Delta y x_k + c$ ……..(ii)

Let $C = 2 (1-b) \Delta x - \Delta y$
Now for next step,

$P_{k+1} = 2 \Delta x y_{k+1} - 2 \Delta y x_{k+1} + c$……(iii)
From equation (i) and (ii)

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

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

Where, , $x_{k+1}- x_k$ = '0' or '1'
If $p_k \geq 0$

$P_{k+1} = p_k + 2 \Delta x - 2 \Delta y$
We plot , $x_{k+1} = x_k+1$, $y_{k+1} = y_k+1$
$P< 0$

$P_{k+1} = P_k + 2 \Delta x$
$X_{k+1} = x_k$

For initial parameter,

$P_o = 2\ \triangle x\ y_o - 2\ \triangle y\ x_o + c$

$= 2\ \triangle x\ y_o - 2\ \triangle y\ x_o + 2\ (1\text{-}b)\ \triangle x - \triangle y$

$= 2\ \triangle x\ y_o - 2\ \triangle y\ x_o + 2\ \triangle x - 2b\ \triangle x - \triangle y$

$= 2\ \triangle x\ y_o - 2\ \triangle y x_o + 2\triangle x - 2\ (y_o - mx_o)\ \triangle x - \triangle y$

$= 2\ \triangle x\ y_o - 2\ \triangle y\ x_o + 2\ \triangle x - 2\triangle x\ y_o + 2.\ (\triangle y/\triangle x).x_o$

$P_o = 2\ \triangle x - \triangle y$. $\triangle x$ - $\triangle y$

Note: $|m| < 1$, x-direction sample. $x_{k+1} = x_k + 1$

$|m| > 1$, y –direction sample. $y_{k+1} = y_k +$

1

Bresenham's Complete algorithm:

1. Input the two end points (x_1,y_1) and (x_2,y_2)
2. Compute $dx = |x_2 - x_1|$ and $dy = |y_2 - y1|$
3. If $x_2 - x_1 < 0$ and $y_2 - y_1 > 0$ or $x_2 - x_1 > 0$ and $y_2 - y_1 < 0$. Then set a = -1, else a = 1
4. If $dy < dx$ then,
   i. If $x_1 > x_2$ then, $t = x_1 ; x_1 = x_2 ; x_2 = t$
      $t = y_1 ; y_1 = y_2 ; y_2 = t$
   ii. Find initial decision parameter $P = 2dy - dx$
   iii. Plot the first pixel (x_1, y_1)
   iv. Repeat the following till $/x_1/ < /x_2/$
      a. If $P < 0$ then,
         $P = P + 2dy$
         Else, $P = P + 2dy - 2dx$
         $y = y_1 + a$
      b. Increase x_1 by 1 i.e $x_1 = x_1 + 1$
      c. Plot (x_1,y_1)
5. Else /m/>1
   i. Check if $(y_1 > y_2)$ then,
      $t = x_1 ; x_1 = x_2 ; x_2 = t$
      $t = y_1 , y_1 = y_2 ; y_2 = t.$
   ii. Find initial decision parameters. $P = 2dx - dy$
   iii. Plot the first point (x_1,y_1)
   iv. Repeat the following till $y_1 < y_2$
      a. If $P < 0$ then , $P = P + 2dx$.
         Else, $P = P + 2dx - 2dy$
         $X_1 = x_1 + a$
      b. Increase y_1 by 1. i.e $y_1 = y_1 + 1$
      c. Plot (x_1,y_1)

Circle: A cicle is defined as a set of points that ar all at a given distance 'r' from the centre position (x_c, y_c). This distance relationship is expressed by the pythogorean theorem in cartesian co-ordinate as $(x - x_1)^2 + (y-y_1)^2 = r^2$

Methods to draw circle:

a. Direct method
b. Trigonometric method
c. Midpoint Circle method.

Direct method:

$X^2 + y^2 = 'r^2'$; $y = \pm\sqrt{r^2 - x^2}$

Trigonometric method:

$x = x\cos\theta$, $y = y\sin\theta$

Bresenham's mid point algorithm :

This is also a scan conveting algorithm . The equation of the circle with the centre (h,k) is given by , $(x-h)^2 + (y-k)^2 = r^2$ …….(i)

When, h = 0, x = 0 then the equation of the circle at origin

$x^2+y^2 = r^2$ ……(ii)

Differentiating both sides,

$2x+2y\ dy/dx = 0$

Or $dy/dx = -x/y$, where , dy/dx = slope

Now , if dy/dx = 0, then x = 0.

If, dy/dx = -1, then x = y

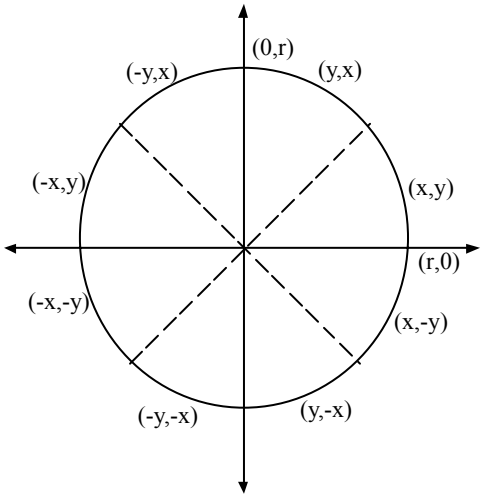


Fig. Symmetry of circle

Consider circle section for x = 0, x = y where slope of the curve varies from 0 to1.Calculation of circle point (x,y) in one octant gives the circle point shown for the other seven octants. To apply the mid point we define a circle function as,

$f_{cicle}(x,y) = x^2+y^2 - r^2$ ……..(iii)

Suppose,

f(x,y) = < 0 if (x,y) is inside the circle boundary.

= 0 if (x,y) is on the circle boundary.

>0 , if (x,y) is outside the circle boundary.

The cicle function tests are performed for the mid position between pixels near the circle path at each sampling step.

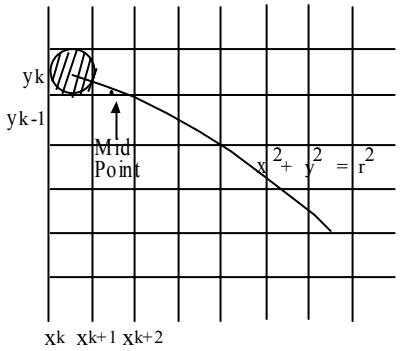


Fig. Midpoint between candidate pixels at sampling position x_{k+1} along a circular path.

Here assume that we have just plotted the pixel (x_k,y_k). We next need to determine whether the pixel at position (x_k+1,y_k) or the one at the position (x_k+1,y_k-1) is closer to the circle.

The decision parameter is the circle function which we have evaluated in equation (iii) at the mid point between these two pixel.

I,e $P_k = f_{cicle}(x_k+1, y_k-1/2)$

$= (x_k+1)^2 +(y_k -1/2)^2 - r^2$ ……..(iv)

If $p_k < 0$, this mid pixel is inside the circle and the pixel on the scan line y_k is closer to the circle boundary . Otherwise the mid point is outside or on the circle boundary , and we select the pixel y_k -1, successive decision parameters can be obtained

using incremental calculation. We obtained a recursive expression for the next decision parameter by evaluating the circle function at sampling position.

$X_{k+1} + 1 = x_k + 2$

$P_{k+1} = f_{circle}(x_{k+1}+1, y_{k+1} - ½)$

$= (x_{k+1}+1)^2 + (y_{k+1}-1/2)^2 - r^2$ ……(v)

Substituting equation (iv) from (v).

$P_{k+1} - p_k = (x_{k+1} + 1)^2 + (y_{k+1}- ½)^2 - (x_k +1)^2 - (y_k - ½)^2$ ….(VI)

If $p_k \geq 0$, the mid position lies on outside the circle hence, $x_{k+1} = x_k+1$, $y_{k+1} = y_k - 1$;
Now from equation (vi)
$x_{k+1} = x_k + 1$
$P_{k+1} = p_k + (x_k + 1+1)^2 + (y_k-1-1/2)^2 - (x_k+1)^2 - (y_k - ½)^2$
$= p_k+(x_k + 1)+ 2(x_k+1).\ 1 + 1 + (y_k -1/2)^2 -2\ (y_k -1/2).1 + 1 - (x_k +1)^2 - (y_k-1/2)^2$

$P_{k+1} = p_k + 2(x_{k+1}- y_{k+1})+1$……(vii)

Where $x_{k+1} = x_k +1$, $y_{k+1} = y_k - 1$;
If $p_k < 0$, the midpoint lies inside the circle,
$X_{k+1} = x_k + 1$, $y_{k+1} = y_k$
Substituting the value in equation (vi)
$P_{k+1} = p_k + (x_{k+1} +1)^2 + (y_k - ½)^2 - (x_k+1)^2 - (y_k - ½)^2$
$= P_k + (x_k+1)^2 + 2(x_{k+1}).1. + 1 - (x_k +1)^2$
$= P_k + 2(x_k +1)+1$
$= P_k + 2(x_{k+1})+1$
Therefore, $P_{k+1} = p_k + 2x_{k+1} + 1$ ……(viii)

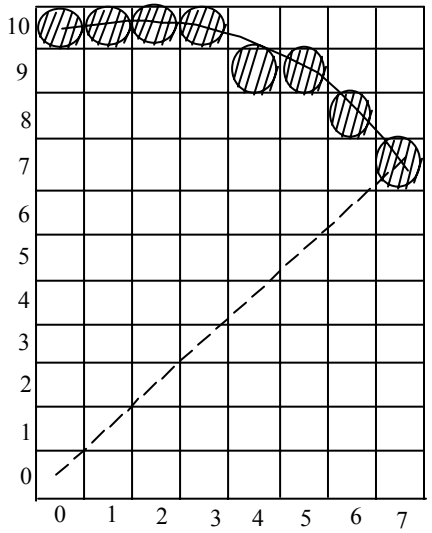
For initial decision parameter.
$(x_0,y_0) = (0,r)$
Hence , $P_0 = f(x_0+1, y_o- ½)$
$= f(1, r-1/2)$
$= 1+ (r-1/2)^2 - r^2$
$= 1 + r^2 - r + 1/4 - r^2$

$P_o = 5/4 - r$
$P_o = 1-r$
P = 1-r ……(ix) since 5 and 4 are integer values so can be rendered off.

Date: 2065/5/12

Digitize $x^2+y^2 = 100$ in first octant.
Given,
Centre = (0, 0)
Radius (r) = 10
We demonstrate the mid point circle algorithm by determining positions along the circle octant in the first quadrant from x = 0, to x = y.
The initial value of the decision parameter is
$P_o = 1-r$
$= 1 - 10$
$= -9$
Successive decision parameter values are positions along the circle path are calculated using mid-point method as,
$P_{k+1} = p_k + 2x_{k+1} + 1$ ($P_k<0$)
Set , $x_{k+1} = x_k +1$, $y_{k+1} = y_k$
$P_{k+1} = p_k + 2x_{k+1} - 2y_{k+1} + 1$ ($P_k > 0$)
Set $x_{k+1} = x_k + 1$, $y_{k+1} = y_k - 1$

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



Digitize a circle $(x-2)^2 + (y-3)^2 = 25$

Solution:

Center C (h,k) = (2,3)

Radius (r) = 5

1st pixel (0,5)

$P_o = 1-r$

$= 1-5$

$= -4$

$P_{k+1} = p_k + 2x_{k+1} + 1 \ (p_k < 0)$

$Y_{k+1} = y_k$

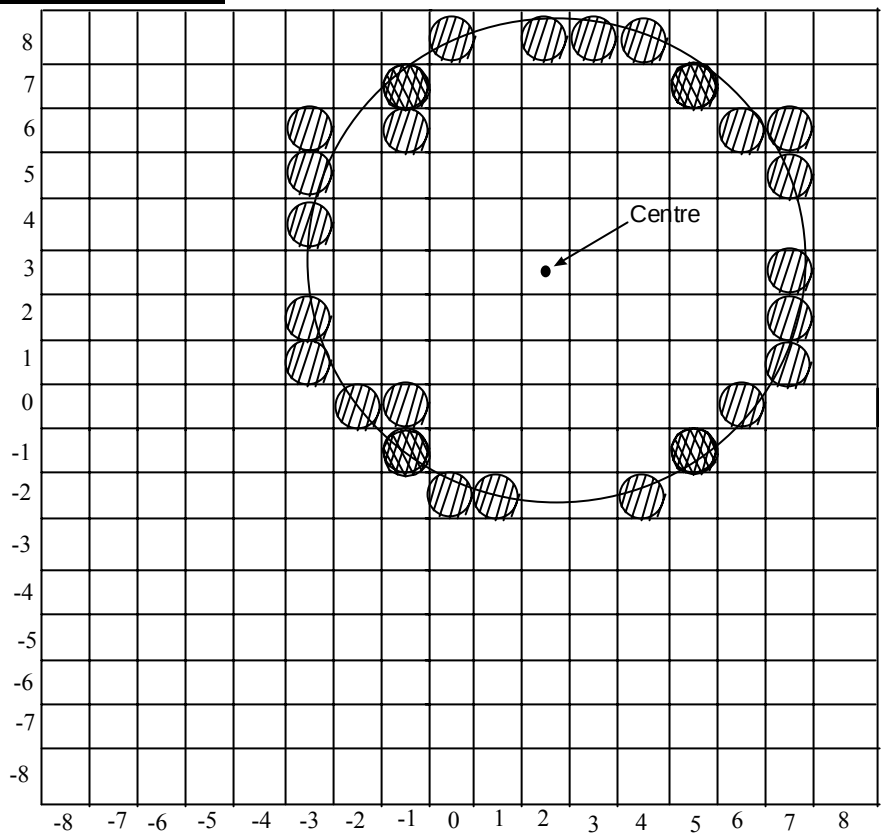
$P_{k+1} = p_k + 2x_{k+1} - 2y_{k+1} + 1 \ (p_k \geq 0)$

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

K	**p_k**	**x_{k+1} y_{k+1}**	**$2x_{k+1}$**	**$2y_{k+1}$**
0	-4	(1,5)	2	10
1	-1	(2,5)	4	10
2	4	(3,4)	6	8
3	3	(4,3)	8	6

Other pixel		Actual pixel	
(1,5)	(1,5)	(3,8)	(x,y)
	(-1,5)	(1,8)	(-x,y)
	(-1,-5)	(1,-2)	(-x,-y)
	(1,-5)	(3,-2)	(x,-y)
	(5,1)	(7,4)	(y,x)
	(5,-1)	(7,2)	(y,-x)
	(-5,-1)	(-3,2)	(-y,-x)
	(-5,1)	(-3,4)	(-y,x)
(2,5)	(2,5)	(4,8)	
	(-2,5)	(0,8)	
	(-2,-5)	(0,-2)	
	(2,-5)	(4,-2)	
	(5,2)	(7,5)	
	(5,-2)	(7,1)	
	(-5,-2)	(-3,1)	
	(-5,2)	(-3,5)	

(3,4)	(3,4)	(5,7)	
	(-3,4)	(-1,7)	
	(-3,-4)	(-1,-1)	
	(3,-4)	(5,-1)	
	(4,3)	(6,6)	
	(4,-3)	(6,0)	
	(-4,-3)	(-2,0)	
	(-4,3)	(-2,6)	
(4,3)	(4,3)	(6,6)	
	(-4,3)	(-2,6)	
	(-4,-3)	(-2,0)	
	(4,-3)	(6,0)	
	(3,4)	(5,7)	
	(3,-4)	(5,-1)	
	(-3,-4)	(-1,-1)	
	(-3,4)	(-1,7)	



Complete गर्न बाँकी छ।

Algorithm:

1. Get the input centre (x_c, y_c) and radius of the circle.
2. Set x = 0, and y = r.
3. Plot the first point and its symmetry at appropriate positions by $x = x+x_c$, $y = y + y_c$
4. Compute the initial decision parameter. P = 1- r
5. Repeat the following till x < y.
   i. Set x = x+1.
   ii. Check if P< 0 then.

P = P+2x + 1

Else,

y = y-1

p = p+2x – 2y +1

iii. Shift the point (x, y) and its symmetry points by x = $x+x_c$, y = $y+y_c$ and plot.

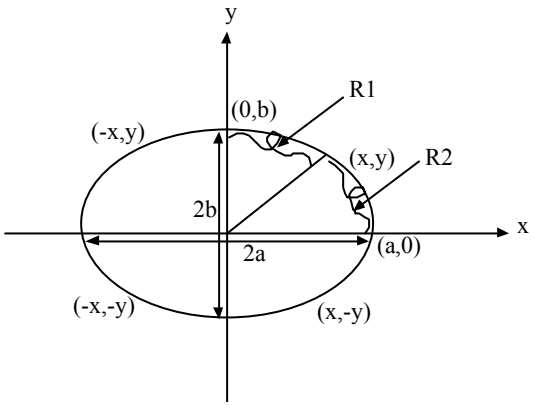
iv. sdf

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

The equation of ellipse in standard form is given by ,

$x^2/a^2 + y^2/b^2 = 1$

Therefore, $y = +- b \sqrt{(1-x^2/a^2)}$



This algorithm is used to display the ellipse in standard form. The algorithm is applied throughout the 1st quadrant according to the slope of the ellipse. For the region (R_1) where the slope of the curve is less then one. We process by taking unit steps in x-direction and for the region (R_2) we take unit steps in y-direction.

Let us define an ellipse function centered at origin by

$f(x, y)= b^2x^2 + a^2y^2 – a^2b^2$.

If f(x,y) < 0 , the point lies inside the ellipse.

$= 0$, the point lies on the ellipse.

> 0, the point lies outside the ellipse.

Starting at (0,b) we take unit steps in x-direction until we reach the boundary between region 1 region 2 (R_1 , R_2) . Then we switch to unit steps in y-direction over the remainder of curve in first quadrant. At each step, we need to test the value of the slope of the curve .The ellipse slope is calculated from equation :

$x^2/a^2 +y^2/b^2 = 1$

Differentiating both sides w.r to x

$2x/a^2 + 2y/b^2. dy/dx = 0$

$Dy/dx = -2b^2x/2a^2y$

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

$2b^2x= 2a^2y$..

Therefore , we move out of region 1 (R_1) when $2b^2x \geq 2a^2y$. **For region R_1.**

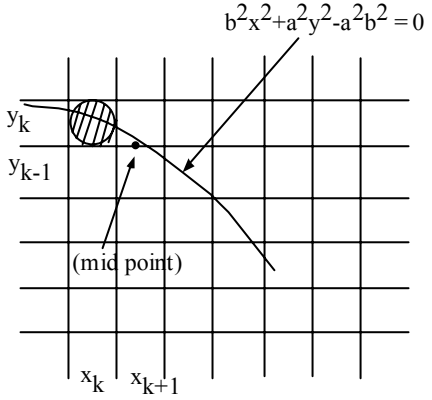


Fig. Mid point between candidate pixel at sampling position x_k+1 along in elliptical path.

If (x_k, y_k) is the pixel first plotted then the candidate pixel are (x_k+1,y_k) and (x_k+1, y_k-1). Let us define decision parameter at the midpoint $(y_k-1/2)$ at sampling position x_k+1 by

$P_k = f(x_k+1, y_k-1/2)$

$= b^2 (x_k+1)^2 + a^2 (y_k-1/2)^2 - a^2b^2$ ……..(i)

$P_{k+1} = b^2(x_{k+1}+1)^2 + a^2 (y_{k+1}-1/2)^2 - a^2b^2$ …….(ii)

Substituting (i) from (ii)

$P_{k+1} - p_k = b^2 \{ (x_{k+1}+1)^2 - (x_k+1)^2 \} + a^2 \{(y_{k+1}-1/2)^2 - (y_k - ½)^2 \}$

If $P_k < 0$, mid point lies inside the ellipse, So we plot, (x_{k+1}, y_k) i.e $x_{k+1} = x_k + 1$. $y_{k+1} = y_k$

Therefore, $P_{k+1} = p_k + b^2 \{ (x_k+1)^2 + 2x_{k+1} + 1 - (x_k + 1)^2\} + a^2\{(y_k - ½)^2 - (y_k - ½)^2 \}$

$p_{k+1} = p_k + 2b^2x_{k+1} + b^2$ …….(iii) Where,

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

If $P_k \geq 0$, the mid point lies outside the ellipse so we plot (x_k+1, y_k-1). i.e $x_{k+1} = x_k + 1$, $y_{k+1} = y_k - 1$.

Therefore, $P_{k+1} = p_k + b^2 \{(x_k+1)^2 + 2 (x_k+1)+1\} + a^2 \{(y_k-1-1/2)^2 - (y_k -1/2)^2 \}$

$= p_k + b^2 \{2x_{k+1}+1\} + a^2 \{(y_k-1/2)^2 - 2(y_k - ½) + 1 - (y_k - ½)^2\}$

$= p_k + 2b^2x_{k+1} + b^2 - 2a^2y_{k+1}$

Therefore, $P_{k+1} = p_k + 2b^2x_{k+1} - 2a^2y_{k+1} + b^2$

Where,

$X_{k+1} = x_k + 1$

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

Now initial decision parameter is given by is given by ,

$P_0 = f(x_0+1, y_0 - 1/2)$

$= f(0+1, b-1/2)$

$= b^2 1^2 + a^2(b-1/2)^2 - a^2b^2$

$= b^2 + a^2(b^2-b+1/4) - a^2b^2$

$= b^2 + a^2b^2 - a^2b + 1/4a^2 - a^2b^2$

$P_o = b^2 - a^2b + a^2/4$ …….(v)

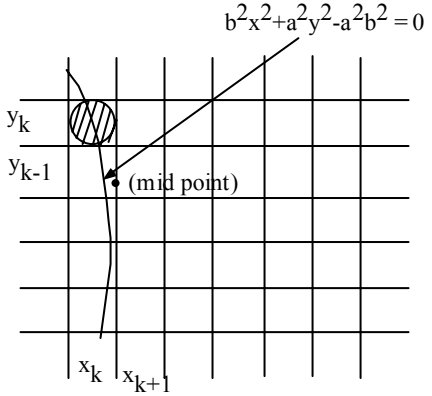


Fig. Mid point between candidate pixel at sampling position y_k-1 along an elliptical path.

For region R_2 (/m/>1)

If (x_k,y_k) be the pixel just plotted then the candidate pixel at (x_k,y_k-1) and $(x_k+1, y_k -1)$. So let us define the decision parameter at the midpoint $x_k+1/2$ at sampling position y_k-1 by

$P_k = f(x_k+1/2, y_k-1)$

$= b^2(x_k+1/2)^2 + a^2 (y_k -1)^2 - a^2b^2$ ……(i)

$P_{k+1} = b^2(x_{k+1}+1/2)^2 + a^2 (y_{k+1}-1)^2 - a^2b^2$ ……(ii)

Substituting (i) from (ii)

$P_{k+1} - p_k = b^2 \{ (x_{k+1}+1/2)^2 - (x_k+1/2)^2\} + a^2 \{(y_{k+1}-1)^2 - (y_k - 1)^2 \}$ ……..(iii)

If $p_k > 0$, the mid point lies outside the ellipse, so we plot (x_k,y_{k-1})

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

$P_{k+1} = p_k + b^2\{(x_k+1/2)^2 - (x_k+1/2)^2\} + a^2 \{ (y_k -1 - 1)^2 - (y_k -1)^2\}$

$= p_k + a^2\{(y_k+1)^2 - 2(y_k-1)+1 - (y_k-1)^2\}$

$= p_k + a^2 \{-2y_{k+1} + 1\}$

$P_{k+1} = p_k - 2a^2y_{k+1} + a^2$ ……(iv) Where $x_{k+1} = x_k$, $y_{k+1} = y_k - 1$

If $p_k \leq 0$, the midpoint lies on/inside the ellipse, so we plot $(x_k+1, y_k - 1)$ i. e

$X_{k+1} = x_k + 1$

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

$P_{k+1} = p_k + b^2 \{ x_k + 1 + ½)^2 - (x_k+1/2)^2\} + a^2 \{(y_k-1-1)^2 - (y_k - 1)^2 \}$

$= p_k + b^2\{(x_k+1/2)^2 + 2(x_k + 1/2)1 + 1-(x_k + 1/2)^2\} + a^2 \{(y_k - 1)^2 - 2(y_k - 1).1 + 1.-(y_k - 1)^2 \}$

$= p_k + b^2 \{2x_k + 1+1\} + a^2 \{-2y + 2 + 1\}$

Therefore, $P_{k+1} = p_k + 2b^2x_{k+1} - 2a^2y_{k+1} + a^2$ ……...(v)

Where, $x_{k+1} = x_k+1$

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

The initial value of initial decision parameter for region 2 is

$P_0 = f(x_0+1/2, y_o-1)$

$P_0 = b^2 (x_0+1/2)^2 + a^2 (y_o - 1)^2 - a^2b^2$ …….(vi)

[**Note:** (x_o,y_o) for region 2 in the last point for region 1. The pixel for other quadrant are determined by symmetry]

Date: 2065/5/16

Algorithm:

1. Obtain the centre x_c , y_c semi-major and semi-minor axis length as a and b.
2. Set $x = 0, y = b$
3. plot the point (x,y) and its symmetry points at appropriate positions by $x = x+x_c$, $y = y+y_c$
4. Compute initial decision parameter for R_1 , $P_1 = b^2 - a^2b^2 + a^2/4$
5. Repeat the following till $2b^2x < 2a^2y$
   i. $x = x+1$
   ii. Test if $P_1 < 0$ then
   $P_1 = p_1 + 2b^2x + b^2$
   Else,
   $y = y-1$
   $P_1 = p_1 + 2b^2x - 2a^2y = b^2$
   iii. Plot the points (x,y) and its symmetry points at approximate position by
   $x = x+x_c$ $y = y+y_c$
6. Compute initial decision parameter for R_2
   $P_2 = b^2(x+0.5)^2 + a^2 (y-1)^2 - a^2b^2$
7. Repeat the following till $y>0$
   i. $y = y-1$
   ii. Test if $P_2 > 0$ then
   $P_2 = p_2 - 2a^2y + a^2$
   Else,
   $X = x+1$
   $P_2 = p_2 + 2b^2x - 2a^2y + a^2$
   iii. Plot the points (x,y) and its symmetry points at approximate points by $x = x+x_c$, $y = y + y_c$

 # Digitize an ellipse $(x-2)^2/64 + (y+5)^2/36 = 1$. Using mid point algorithm.

Solution:

Center = (2,-5)

a = 8

b = 6

For region R_1

First pixel is (0,6)

$P_o = b^2 - a^2b + a^2/4$

$= 36 - 64 \times 6 + 64/4$

$= -332$

Successive decision parameter values ad positions long the ellipse path are calculated using the midpoint method as

If $p_k < 0$, $p_{k+1} = p_k + 2b^2x_{k+1} + b^2$

If $p \geq 0$,

$P_{k+1} = p_k + 2b^2x_{k+1} - 2a^2y_{k+1} + b^2$ where $x_{k+1} = x_k + 1$; $y_{k+1} = y_k - 1$

k	p_k	(x_{k+1}, y_{k+1})	$2b^2x_{k+1}$	$2a^2y_{k+1}$
0	-332	(1,6)	72	768
1	-224	(2,6)	144	768
2	-44	(3,6)	216	768
3	-208	(4,5)	288	640
4	-108	(5,5)	360	640
5	288	(6,4)	432	512
6	244	(7,3)	504	384

We now move out of region 1, since $2b^2x > 2a^2y$

For region 2, the initial point is $(x_0,y_0) = (7,3)$ and the initial decision parameter is

$P_0 = b^2(x_0+1/2) + a^2(y-1)^2 - a^2b^2$

$= 36(7+1/2)^2 + 64(3-1)^2 - 36 \times 64$

$= 36 \times 225/4 + 64 \times 4 - 36 \times 64$

$= -151$

The remaining positions along the ellipse path in the first quadrant are then calculated as

If $p_k > 0$

$P_{k+1} = p_k - 2a^2y_{k+1} + a^2$ Where, $x_{k+1} = x_k$; $y_{k+1} = y_k - 1$

If $p_k \leq 0$

$P_{k+1} = p_k + 2b^2x_{k+1} - 2a^2y_{k+1} + a^2$ Where, $x_{k+1} = x_k + 1$; $y_{k+1} = y_k - 1$

K	**p_k**	**(x_{k+1},y_{k+1})**	**$2b^2x_{k+1}$**	**$2a^2y_{k+1}$**
0	-151	(8,2)	576	256
1	233	(8,1)	576	128
2	745	(8,0)	-	-

A plot of the selected positions around the ellipse boundary within the first quadrant is shown in fig.

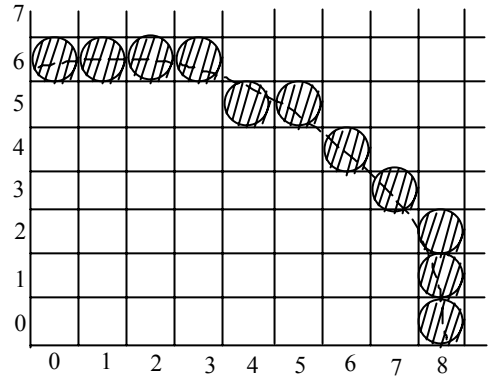


Fig. Positions along an elliptical path centerted on the origin with a=8 and b = 6 using the midpoint algorithm to calculate pixel addresses in first quadrant.

In the following procedure, the mid point algorithm is used to display an ellipse with input parameters a,b , x center and y center. Position along the curve in the first quadrant are generated and then shifted to their proper screen positions. Intensities for these positions and the symmetry positions in the

other three quadrants are loaded into the frame buffer using the set pixel routine.

Transformation: Changing co-ordinate description of an object is called transformation.

Types:

- Rigid body transformation (transformation without defomation in shape.)
- Non rigid body transformation (transformation with change in shape)

Basic Transformations:	**Other Transformation**
1. Translation	1.reflection
2.Rotation	2.Shearing
3.Scaling	

Two dimensional transformation:

1. Translation: Changing the co-ordinate position along a st. line is called translation.

If P(x,y) is taranslated to a position p'(x',y') by t_x units parallel to x-axis and t_y units parallel to y axis then,

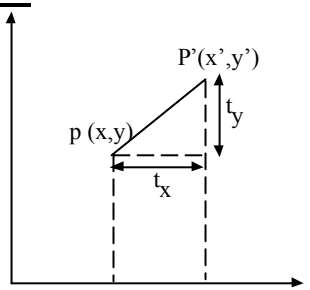
$X' = x + t_x$

$Y' = y + t_y$

In matrix form:

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

i.e $p' = p+T$ where T is transformation matrix.



Translate the given points (2,5) by the translating value (3,3)

Solution:

(x,y) = (2,5)

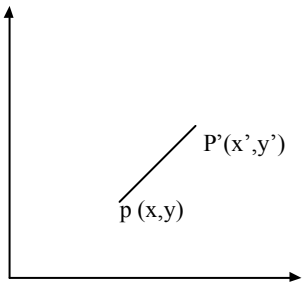
$T_x = 3$

$T_y = 3$

$X' = x + t_x$

$Y' = y + t_y$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} 2 \\ 5 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \end{bmatrix} = \begin{bmatrix} 5 \\ 8 \end{bmatrix}$$



Scaling: The co-ordinate position of an object by multiplying a constant factor (scaling factor) is called scaling.

Scaling Techniques:

a. About origin
b. About fixed point

About origin: If p(x,y) be scaled to a point p'(x',y') by s_x times in x-unit and s_y times in y-units then,

$x' = x.\ s_x$

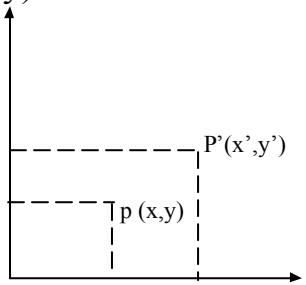
$y' = y.s_y$

In matrix form:

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

i.e

$p' = S\ (s_x, s_y).P(x,y)$



Where

$S\ (s_x, s_y)$ is a scaling matrix

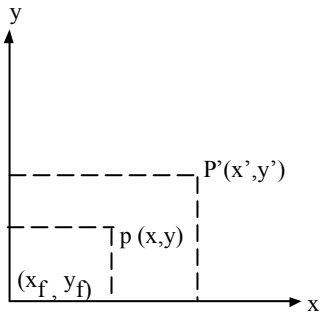
If $s_x = s_y$ scaling type is uniform

If s_x is not equal to s_y scaling type is differential.

(b) About fixed point: If p(x,y) be scaled to a point p'(x',y') by s_x times in x-unit. and s_y time in y-unit about (x_f,y_f) then

$x' - x_f = s_x(x-x_f)$

$y' - y_f = s_y\ (y-y_f)$



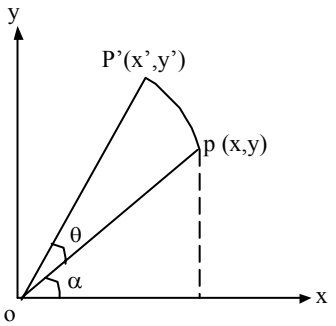
In matrix form,

…….

Scaling:

Scale the given triangle whose co-ordinate values are (2,3), (1,2) , (3,4) where the scaling factor is (2,3) about (i) origin (ii) about fixed point (1,2)

Rotation: Changing the co-ordinate position along a circular path is called rotation.



Types of rotation:

a. About origin
b. About any point

About origin: If p(x,y) is rotated to a new position p'(x',y') in anti-clockwise direction by an angle θ , then (x',y') can be calculated as following.

Let the angle made by line OP with x-axis is α and the radius of circulation path is r then,

$x = r\cos\alpha$

$y = r\sin\alpha$

Also,

$x' = r\cos(\theta + \alpha)$

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

$x' = r(\cos\theta.\cos\alpha - \sin\theta.\sin\alpha)$

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

$y' = r(\sin\theta.\cos\alpha + \cos\theta.\sin\alpha)$

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

In matrix form:

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

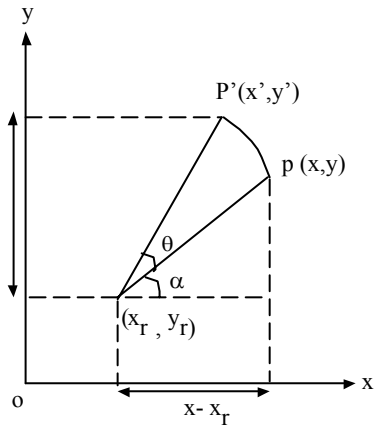
Note: θ +ve (anticlock wise)

θ –ve (clock wise)

b. About any point: if (x,y) is rotated to a new position p'(x',y') by an angle θ, then (x', y') can be calculated as following.

Now $x - x_r = r\cos\alpha$

$y - y_r = r\sin(\theta + \alpha)$



Do yourself .

 # Rotate the triangle having co-ordinates (1,2) , (2,3) (4,5) by 60 ° about origin.

Date: 2065/5/18

Matrix Representation and Homogeneous Co-ordinates:

To treat all the transformation in the same manner, homogeneous co-ordinate are used for reapresentation. Each points (x,y) is represented by triple (x,y,h).

(x,y,h) and (x',y',h') represent same point if on one is multiple of other.

Ie x'/x = y'/y = h'/h

So, (x,y,h) and (x/h, y/h,1) also represent same point, for simplicity we use h = 1.

Therefore, (x,y) in Cartesian system is represented as (x,y,1) in homogeneous co-ordinate system.

Now (2×2) matrix is changed into 3× 3 matrix,

For translation:

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

Therefore, p' = $T(t_x,t_y)$.p

For Rotation (about origin)

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

P' = $R(\theta)$. P

For Scaling : (about origin)

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

Inverse Transformation:

For inverse translation:

$T^{-1}(t_x,t_y) = T(-t_x, -t_y)$

For inverse Rotation:

$R^{-1}(\theta = R(-\theta)$

For inverse scaling:

$s^{-1}(s_x,s_y) = S(1/s_x, 1/s_y)$

Prove that two successive translations are additive.

i.e $T(t_x, t_y).\ T(t_{x2},t_{y2}) = T(t_{x1} + t_{x2},\ t_{y1} + t_{y2})$.

Solution:

Let p be the point translated by $T(t_{x1},t_{y1})$ to point p' then,

P' = $T(t_{x1},t_{y1})$. P

Let p' be the point translated by $T(t_{x2},t_{y2})$ to point p'' then,

P'' = $T(t_{x2}, t_{y2})$. P'

= $T(t_{x2},t_{y2}).T(t_{x1},t_{y1}).p$

Net transformation = $T(t_{x2},t_{y2}).\ T(t_{x1},t_{y1})$

$$= \begin{bmatrix} 1 & 0 & t_{x1} \\ 0 & 1 & t_{y1} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & t_{x1} \\ 0 & 1 & t_{y1} \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & tx_1 + tx_2 \\ 0 & 1 & ty_1 + ty_2 \\ 0 & 0 & 1 \end{bmatrix}$$

= $T(tx_1+ tx_2,\ ty_1+ty_2)$

$\therefore\ T(tx_1,ty_2).\ T(tx_1,ty_1) = T(tx_1+ tx_2,\ ty_1+ty_2)$

Which demonstrates that two successive translations are additive.

Prove that two successive rotations are additative. i.e $R(\theta_2)\ .\ R(\theta_1) = R(\theta_1 + \theta_2)$

Solution:

Let p be the point rotated by $R(\theta_1)$ to point p' then p' = $R(\theta_1)$. p

Let p' the point rotated by $R(\theta_2)$ to point p'' then

P'' = $R(\theta_2)$. P'

Net transformation = $R(\theta_2).\ R(\theta_1)$

P' = $R(\theta_1+ \theta_2)$ p

Prove that two successive scalings are multiplicative.

i.e $S(s_{x2}, s_{y2}).\ S(s_{x1},s_{y1}) = S(s_{x1}, s_{x2}, s_{y1}, s_{y2})$

Solution:

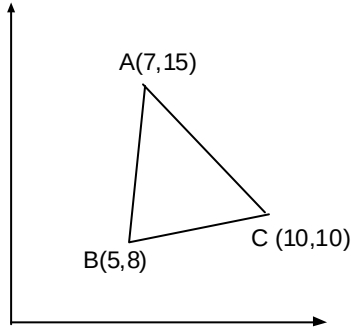
Concatenating transformation matrices for two successive scaling operations produces the following composite scaling matrix.

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

Or $s(s_{x2},s_{y2})\ S(s_x,s_y) = S(s_{x1}\, s_{x2}, S_{y1}\, s_{y2})$

Rotate the $\triangle ABC$ by 45° clock wise about origin and scale it by (2,3) about origin.

Solution:



Step 1: Rotation by 45° (clock wise)

Step-II: Scaling by (2,3)

S(2,3)

Net transformation: = S(2,3). R(-45)

$$= \begin{bmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} \cos 45 & Sin45 & 0 \\ -Sin45 & \cos 45 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \\ -1/\sqrt{2} & 1/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 2/\sqrt{2} & 2/\sqrt{2} & 0 \\ -3/\sqrt{2} & 3/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The transformation points are ,

A' = TA

$$\begin{bmatrix} 2/\sqrt{2} & 2/\sqrt{2} & 0 \\ -3/\sqrt{2} & 3/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 7 \\ 15 \\ 1 \end{bmatrix} =$$

B'=TB

$$\begin{bmatrix} 2/\sqrt{2} & 2/\sqrt{2} & 0 \\ -3/\sqrt{2} & 3/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 5 \\ 8 \\ 1 \end{bmatrix} =$$

C' = TC

$$\begin{bmatrix} 2/\sqrt{2} & 2/\sqrt{2} & 0 \\ -3/\sqrt{2} & 3/\sqrt{2} & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 10 \\ 10 \\ 1 \end{bmatrix} = \begin{bmatrix} 20/\sqrt{2}+20/\sqrt{2}+0 \\ -30/\sqrt{2}+30/\sqrt{2}+0 \\ 0+0+0+1 \end{bmatrix} = \begin{bmatrix} 40/\sqrt{2} \\ 0 \\ 1 \end{bmatrix}$$

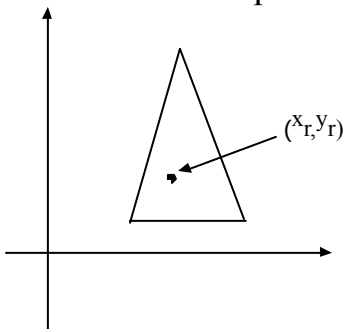
General pivot point rotation: We can rotate any selected point (x_r,y_r) by following the sequence of translate – rotate –translate operations.

Step 1: Translate the object such that fixed pont moves to origin i.e $T(-x_r,-y_r)$.

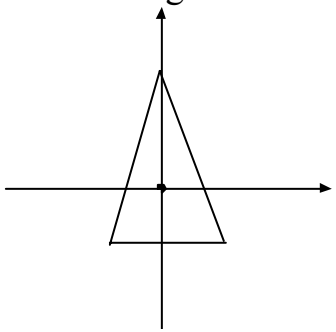
Step 2: Rotate the object coordinate origin i.e $R(\theta)$.

Step 3: Translate back the object so that the pivot point is returned to its original position $T(x_r,y_r)$.

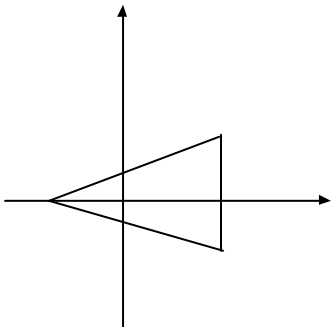
This transformation sequence is illustrated in fig below.



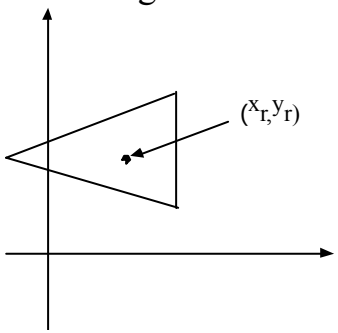
(a) Original position of object And pivot point



(b) Translation of object so that povit point (x_r,y_r) is at origin.



(c) Rotation about origin



(d) Translation of object so That the pivot point is Returned to position (x_r, y_r)

Net Transformation (T) = $T(x_r,y_r)$. $R(\theta)$. $T(-x_r,-y_r)$

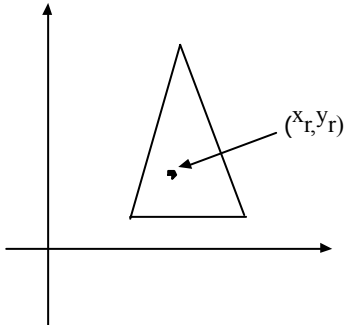
$$= \begin{bmatrix} 1 & 0 & x_r \\ 0 & 1 & y_r \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_r \\ 0 & 1 & -y_r \\ 0 & 0 & 1 \end{bmatrix}$$

General Fixed point Scaling:

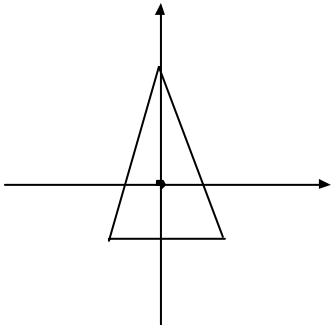
A transformation sequence to produce scaling with respect to a selected fixed position (x_f, y_f) using a scaling function is illustrated below:

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

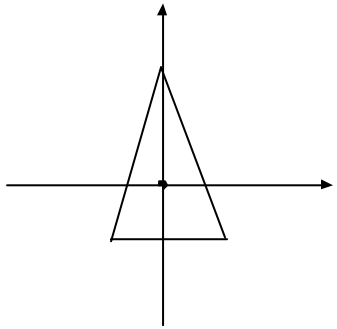
The transformation sequence is illustrated in fig below:



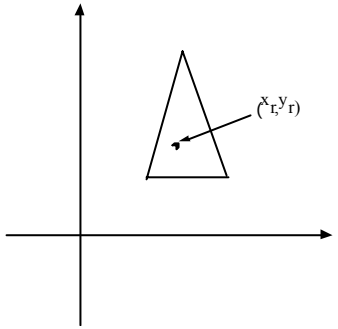
(a) Original position of object And fixed point



(b) Translate object so that fixed point (x_r, y_r) is at origin.



(c) Scale object with respect To origin



(d) Translate object so the fixed point is Returned to position (x_f, y_f)

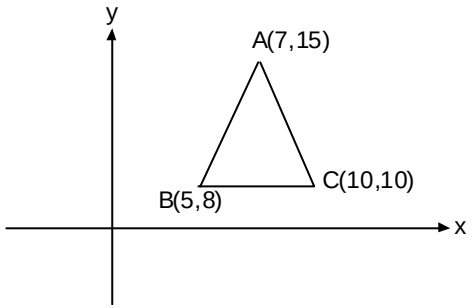
Net Transformation T = $\begin{bmatrix} 1 & 0 & x_f \\ 0 & 1 & y_f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_f \\ 0 & 1 & -y_f \\ 0 & 0 & 1 \end{bmatrix}$

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

Date: 2065/6/5

Rotate the △ABC by 90° anticlock wise about (5,8) and scale it by (2,2) about (10,10)

Solution:



Step:1
T(-5,-8)
Step: 2
R(90°)
Step: 3
T(5,8)
Step: 4
T(-10,-10)
Step:5
S(2,2)
Step: 6
T(10,10)
Net transformation:
T(10,10). S(2,2). T(-10,-10).T(5,8).R(90) . T(-5,-8)

$$= \begin{bmatrix} 1 & 0 & 10 \\ 0 & 1 & 10 \\ 0 & 01 & \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -10 \\ 0 & 1 & -10 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 5 \\ 0 & 1 & 8 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 90 & -\sin 90 & 0 \\ \sin 90 & \cos 90 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & -5 \\ 0 & 1 & -8 \\ 0 & 0 & 1 \end{bmatrix}$$

Other transformation:

1. Reflection

2. Shearing.

Reflection: Providing a mirror image about an axis of an object is called reflection.

(i) about x-axis (y =0)

x’ = x

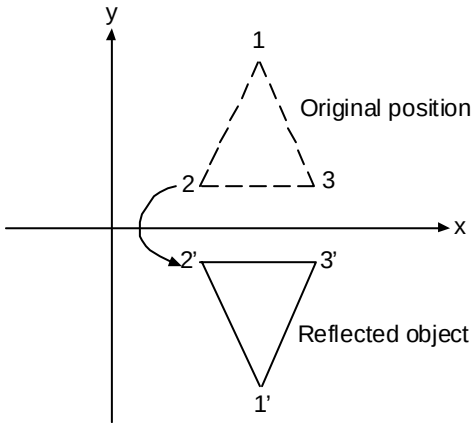
y’ = -y

In matrix from,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$P' = R_{fx}.P$

R_{fx} = Reflection matrix about x-axis



(ii) about y-axis (x=0)

x’ = -x

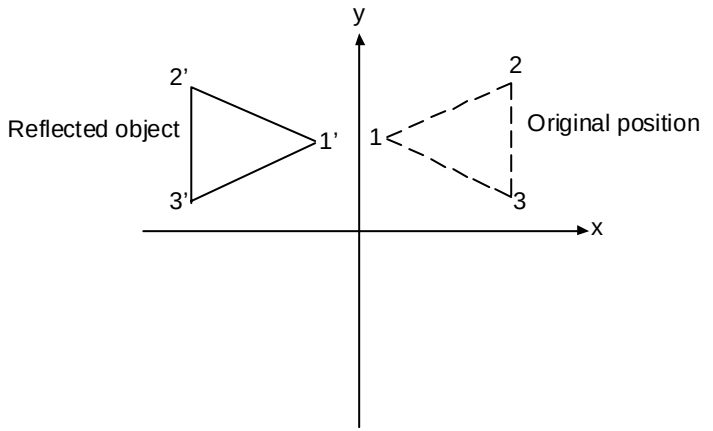
y’ = y

In matrix from,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$P' = R_{fy}.P$

R_{fy} = Reflection matrix about y-axis



Date: 2065/6/6

Alternative way for reflection:

Step:1

Rotate the object so that y axis concise x-axis. i.e R(-90)

Step: 2

Reflection about x-axis. R_{fx}

Step: 3

Rotate back the object so that y-axis move to original position . R(-90)

Net transformation:

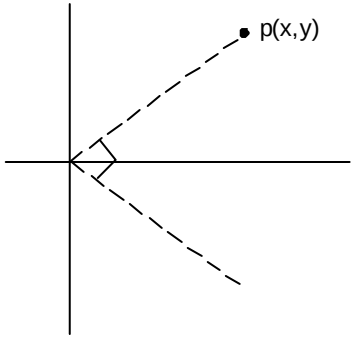
i.e $R_{fy} = R(90) \cdot R_{fx} \cdot R(-90)$

$$= \begin{bmatrix} \cos 90 & -\sin 90 & 0 \\ \sin 90 & \cos 90 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 90 & \sin 90 & 0 \\ -\sin 90 & \cos 90 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

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

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

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



Determine transformation matrix responsible for refection of object about the line y = x

(1) R(-45)
(2) R_{fx}
(3) R(45 °)

Net transformation ,

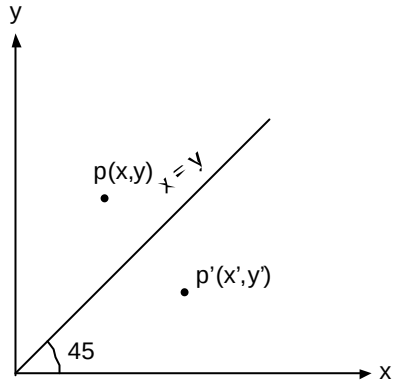
R(45). R_{fx} R(-45)

$$= \begin{bmatrix} \cos 45 & -\sin 45 & 0 \\ \sin 45 & \cos 45 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 45 & -\sin 45 & 0 \\ -\sin 45 & \cos 45 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

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

.i.e x' = y

y' = x



Determine transformation matrix responsible for reflection of about the line y+x = 0 .

Solution:

(i) R(45)
(ii) R_{fx}
(iii) R(-45)

Net transformation,

R(-45) R_{fx} . R(+45)

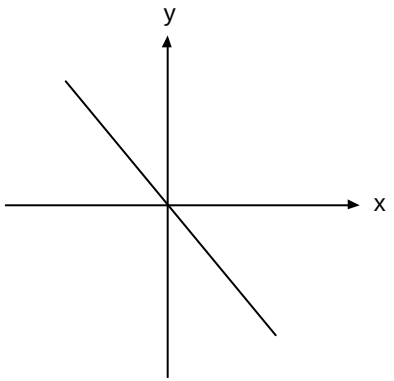
$$= \begin{bmatrix} \cos 45 & \sin 45 & 0 \\ -\sin 45 & \cos 45 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos 45 & -\sin 45 & 0 \\ \sin 45 & \cos 45 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ -\frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

i. e $x' = -y$

$y' = -x$



Determine transformation responsible for reflection of object about the line $y + x = 2$.

(i) T(0,-2)

(ii) R(45 °)

(iii) R_{fx}

(iv) R(-45)

(v) R(0,2)

Determine transformation responsible for reflection of object the line y+x = 2.

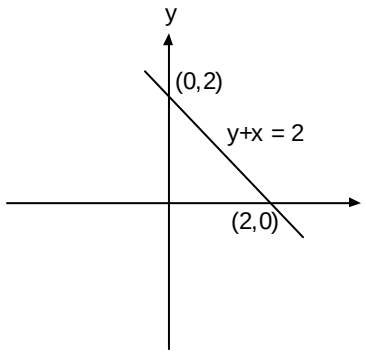
(1) T(0,-2)

(2) R(45)

(3) R_{fx}

(4) R(-45)

(5) R(0,2)



Determine transformation matrix responsible for reflection of object about the line y = mx+b

(i) T(0,-b)

(ii) R(-θ)

(iii) R_{fx}

(iv) R(θ)

(v) T(0,b)

Net transformation = T(0,b). R(θ) . R_{fx}.R(-θ). T(0,-b)

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & b \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

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

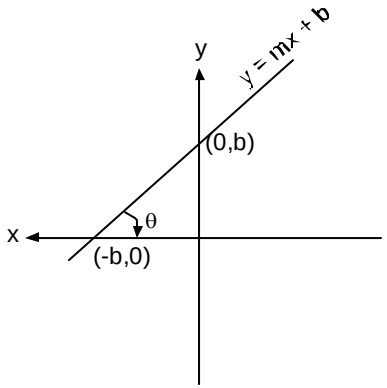
$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & b \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & \sin\theta & -b\sin\theta \\ \sin\theta & -\cos\theta & b\cos\theta \\ 0 & 0 & 1 \end{bmatrix}$$

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

$$\begin{bmatrix} \frac{1}{\sqrt{1+m^2}} & -\frac{m}{\sqrt{1+m^2}} & 0 \\ \frac{m}{\sqrt{1+m^2}} & \frac{1}{\sqrt{1+m^2}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{1+m^2}} & \frac{m}{\sqrt{1+m^2}} & -\frac{b_m}{\sqrt{1+m^2}} \\ \frac{m}{\sqrt{1+m^2}} & -\frac{1}{\sqrt{1+m^2}} & \frac{1}{\sqrt{1+m^2}} \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & b \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1-m^2}{1+m^2} & -\frac{2m}{1+m^2} & \frac{-2bm}{1+m^2} \\ \frac{2m}{1+m^2} & \frac{m^2-1}{1+m^2} & \frac{-bm^2+b}{1+m^2} \\ 0 & 0 & 1 \end{bmatrix}$$

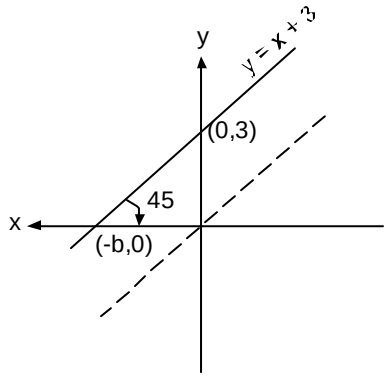
$$= \begin{bmatrix} \frac{1-m^2}{1+m^2} & \frac{2m}{1+m^2} & \frac{-2bm}{1+m^2} \\ \frac{2m}{1+m^2} & \frac{m^2-1}{1+m^2} & \frac{2b}{1+m^2} \\ 0 & 0 & 1 \end{bmatrix}$$



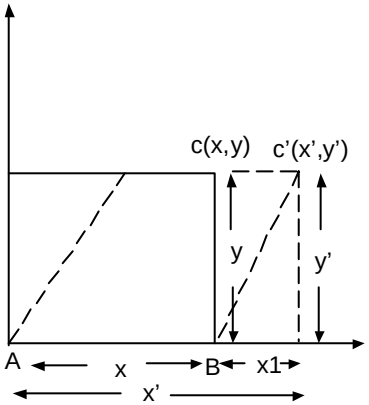
Reflection:

Determine the X'-formation matrix responsible for reflection about y = x+3 .

(1) T(0,-3)
(2) R(-45)
(3) R_{fx}
(4) R(45°)
(5) T(0,3)



Shearing: Transformation that causes deformation in the original object by supposing as if the object is composed of internal layers and these layers are caused to slide over is shearing.



About x-axis

$y' = y$

$x' = x + x_1$ where,

$x_1 \alpha\ y$

$x_1 = Sh_x.y$

where, $_{Shx}$ is shearing constant.

$y' = y$

$x' = x + sh_x.y$

In matrix form,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & sh_x & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Similarly about y-axis

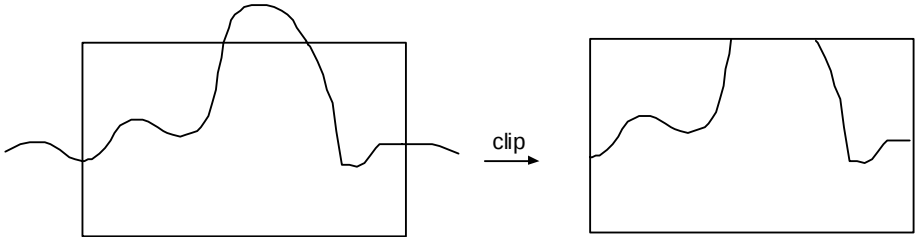
$x' = x$

$y' = y + sh_y.x$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ sh_y & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

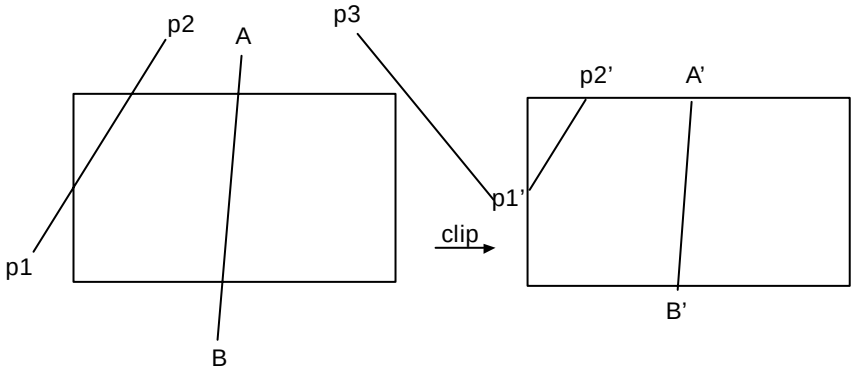
$P' = SH_y.p$

2D window clipping:



The procedure that identifies these portions of the picture that are either inside or outside of the specified region of span is clipping .

Line clipping:



Cohen Sutherland line clipping:

In this method, every line endpoint is assigned a four digit binary code(region code) that identifies the location of the point relative to the boundary.

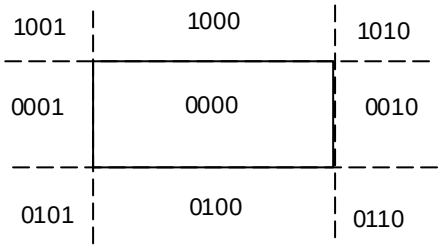
For,

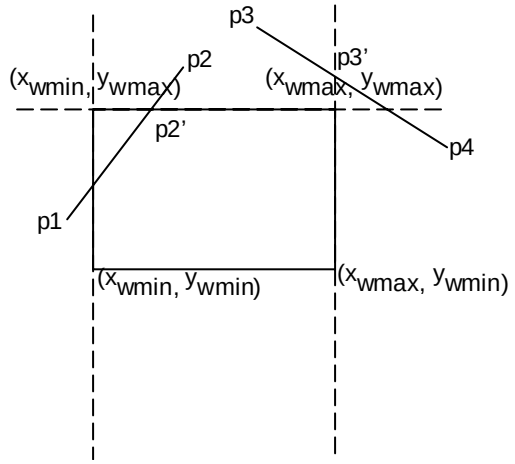
b_1 : left

b_2 : right

b_3 : below

b_4 : above





The value 1 indicates its relative position. If a point is within clipping rectangle then region code is 0000. So ,

If $x - x_{wmin} < 0$, $b_1 = 1$

If $x_{wmax} - x < 0$, $b_2 = 1$

If $y - y_{wmin} < 0$, $b_3 = 1$

If $y_{wmin} - y < 0$, $b_4 = 1$

If the region codes of both end points are 0000 then we accept the line.

Any line that have one in the same bit position is rejected i.e if R_A AND $R_B \neq 0$

Line is completely outside .

The lines which con not be identified as completely inside or outside a window by these tests are checked for intersection with the window boundary. Such lines may or may not cross into the window interior.

In the fig. aside , region code of

$P_1 = 0001$

$P_2 = 1000$

P_1 AND $P_2 = 0$

So we need further calculation.

Starting form p_1,

Intersection of P_1 with left boundary is calculated.

Region code of P_1' = 0000

P_1' AND $P_2 = 0$.

Intersecting of P_2 with above boundary is calculated region code of p_2' = 0000

Since both end points have region codes (0000) .So P_1', P_2' portion of the line is saved.

Similarly,

For P_3 , P_4.

$P_3 = 1000$

$P_4 = 0010$

P_3 AND $P_4 = 0$

So we need further calculations; starting from P_3 region code of P_3 is 1000, i.e b_4 is high, so intersection of P_3 with upper boundary which yields P_3' having region code 1010.

Again p_3' AND $P_4 \neq 0$

So P_3P_4 is totally clipped.

The intersection point with vertical boundary can be obtained by

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

Where (x_1,y_1) and (x_2,y_2) are end points of line and y is the co-ordinate value of intersection point where x value is either x_{wmin} or x_{wmax} and $m = y_2 - y_1 / x_2 - x_1$.
Similarly , intersection point with horizontal boundary
$x = x_1 + (y-y_1)/m$
Where , $y = y_{wmin}$ or y_{wmax}

Use cohen Sutherland algorithm to clip the line for the following dimensions. Line endpoints (15, 45) and (25,15)
Window:
Lower left: (10, 20)
Upper right : (30, 40)

Date: 2065/6/7

Two dimensional object to screen viewing Transformation:

Mechanism for displaying view of a picture on an output device:

A graphical package allows a user to specify which part of the defined picture is to be displayed and where that part is to be placed on the display device. Any convenient Cartesian co-ordinate system; referred to as the world co ordinate reference frame can be used to define the picture.

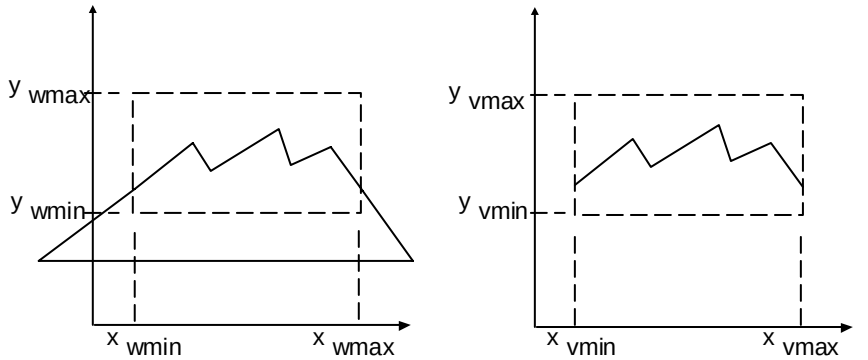
For a 2-D picture, a view is selected by specifying a subarea of the total picture area. A user can select single area for displaying or several areas could be selected for simultaneous display. The picture parts within the selected areas are then mapped onto specified areas of the device co-ordinates. When the multiple view areas are selected, these areas can be placed in separate display locations or some areas could be inserted into other, larger display area. Transformations from world to device co-ordinates involve translation rotation and scaling operations, as well as procedures for deleting those parts of the picture that are outside the limits of a selected display area.

A world co-ordinate area selected for display is called a window. An area on the display device to which a window is mapped is called a view port. The window defines what is to be viewed. The viewport defines where it is to be displayed.

Often, windows and viewports are rectangles in standard positions, if the rectangle edges are parallel to the co-ordinate axis.

" The mapping of a port of a world co-ordinate scene to a device co-ordinates is referred to as viewing transformation." Sometimes , the 2D viewing transformation is simply referred to as the window – to – viewport transformation or the windowing transformation.

Fig. given below illustrates the mapping of a picture selections that falls within a rectangular window onto a designated rectangular viewpoint.



World Co-ordinate Device co-ordinate

Fig. A viewing transformation using standard rectangles for the window and viewpoint.

Steps to be followed for window to viewpoint transformations area:

- Generate world co-ordinate
- Convert world co-ordinate to view co-ordinate.
- Map the view co-ordinate to normalized viewing co-ordinate.
- Map the normalized viewing co-ordinate to device co-ordinate system.

These steps can be illustrated through a pipeline.

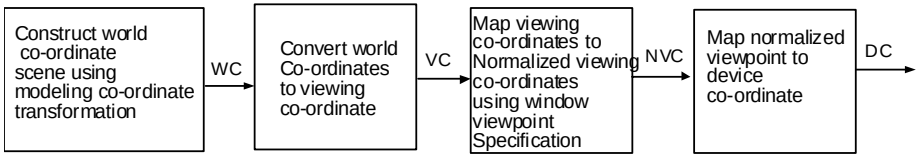
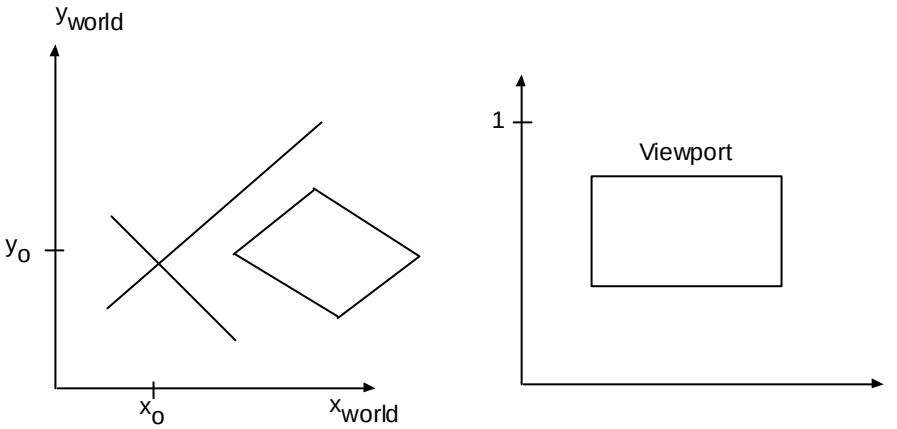


Fig. The two dimensional viewing transformation pipeline

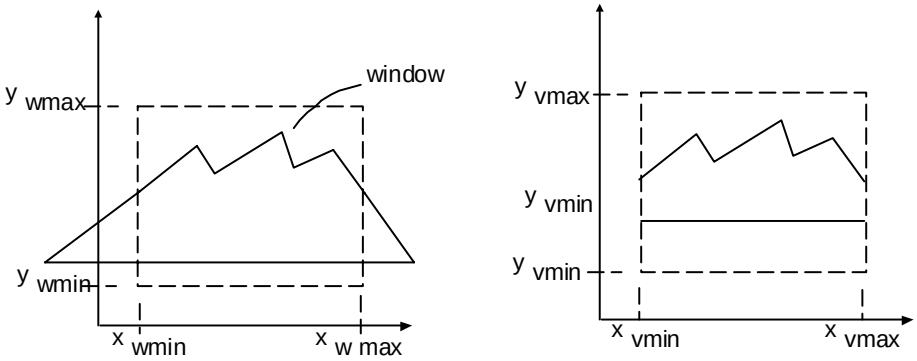
Fig. given below show a rotating view co-ordinate reference frame and the mapping to normalized co-ordinates.



World co-ordinate Normalized device co-ordinate

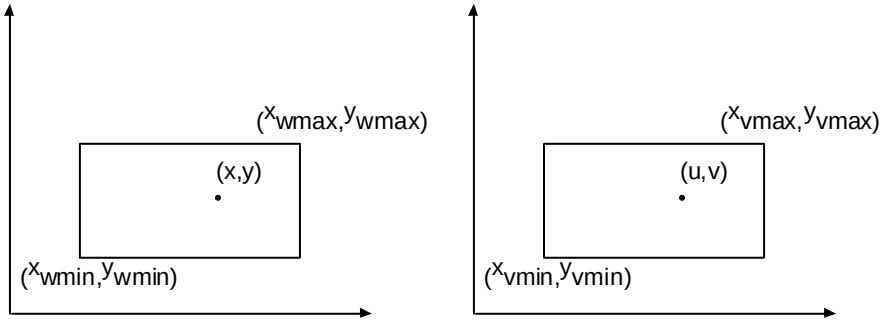
By changing the position of the viewport we can view objects at different positions on the display area of an output device. Also by varying size of the viewport we can change the size and proportions of displayed objects.

Window to viewport transformation:



World co-ordinate Device co-ordinate

It is the mechanism for displaying view of a picture on an output device. The world co-ordinate selected for display is called window. The area on the display device to which window is mapped is called viewport. So, window defines what is to be viewport defines where it is to be displayed. The mapping of part of world co-ordinate scence to device co-ordinate is called viewing transformation or window-to-viewport transformation.



Window – to – viewport transformation can be explained as:

- Choose the world co-ordinate in a rectangle.
- Transform it to origin.
- Scale it with appropriate value.
- Transform it to the relative position in viewport.

Step 1: T ($-x_{wmin}$, $-y_{wmin}$)
Step:2: S (s_x, s_y)
Step:3: T(x_{vmin}, y_{vmin})

∴ Net transformation,
T_{wv} = T (x_{vmin}, y_{vmin}). S(s_x, s_y).T($-x_{wmin}$, $-y_{wmin}$)

The value of s_x and s_y is found by calculating the position of (x,y) in the window to the corresponding position of point (u,v) in the viewpoint.

To maintain the same relative position, so

$$\frac{x - x_{w\min}}{x_{w\max} - x_{w\min}} = \frac{u - x_{v\min}}{x_{v\max} - x_{v\min}}$$

Or, $$\frac{y - y_{w\min}}{y_{w\max} - y_{w\min}} = \frac{u - y_{v\min}}{y_{v\max} - y_{v\min}}$$

$$S_x = \frac{x_{v\max} - x_{w\min}}{x_{w\max} - x_{w\min}}$$

$$S_y = \frac{y_{v\max} - y_{w\min}}{y_{w\max} - y_{w\min}}$$

∴ T_{wv} =

$$\begin{bmatrix} 1 & 0 & x_{v\min} \\ 0 & 1 & y_{v\min} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{x_{v\max} - x_{w\min}}{x_{w\max} - x_{w\min}} & 0 & 0 \\ 0 & \frac{y_{v\max} - y_{w\min}}{y_{w\max} - y_{w\min}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_{w\min} \\ 0 & 1 & -y_{w\min} \\ 0 & 0 & 1 \end{bmatrix}$$

Is the required window- to – viewpoint transformation.

Determine window to view port transformation for the following dimensions of windows and viewpoint.

	Window	viewpoint
Lower left corner	(5,10)	(8,12)
Upper right corner	(15,20)	(12,18)

Different types of Graphical package are:

1. GKS (Graphical kernel system)

2. PHIGS(programmers Hierarchical interactive graphics standard)
3. GL(Graphics Library)
4. Open GL.

Data Structural Concept:

Introduction:

Data may be organized in different ways; the logical or mathematical model of a particular organization of data is called data structure. In other words, data structure is a collection of data elements whose organization is characterized by accessing operations that are used to store and retrieve individual data elements.

Data structure is of two types:

1. Linear data structure. E.g Array, stack, Queue etc.
2. Non linear data structure. E.g Trees.

(i) Array: An array is collection of similar elements all elements of any given array must be of the same type.

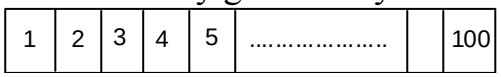


Fig. Array of 100 students

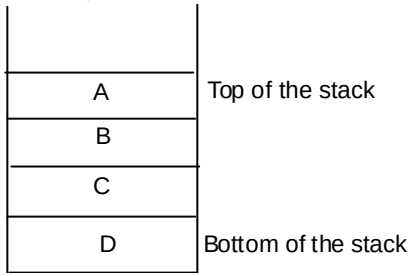
(ii) Stack: A stack is defined formally as a list (a linear data structure) in which all insertion and deletions are made at one end called the top of the stack(TOS). The fundamental operations which are possible on a stack are:

- push - insertion
- pop - deletion
- peep – extract information
- Update – Update information associated at some location in the stack.

The most recently pushed element can be checked prior to performing a pop operation.

A stack is based on the last in first out algorithm (LIFO) in which insertion and deletion operations are performed at one end of the stack. The most accessible information in a stack is at the top of the stack and least accessible information is at the bottom of the stack.

If someone wants to delete on item from an empty stack, it causes underflow and if someone wants to add an element to the stack (which is already full) then it is the case of overflow.



Implementation of Stack:

Stack can be implemented in two ways:

1. pointer (Linked list)
2. Array

Push operation:

Step 1: Check for the overflow
If TOS ≥ = size
Output: "stack overflow and exit".

Step 2: Increment the pointer value by one
TOS = TOS +1

Step 3: Perform insertion

S[TOS] = value

Step 4: Exit

POP operation:

Step 1: Check for the underflow
If TOS = 0
Output: "Stack underflow and exit"

Step 2: Decrement the pointer value by one
Value = S[TOS]
TOS = TOS-1

Step 3: Return the former information of the stack
Return [value]

Step 4: Exit.

PEEP operation: If one interested only about an information stored at some location in a stack, then peep operation is required. In this operation we simply move the pointer to the desired location and then fetch the information associated with the location.

Step 1: [Check for the stack underflow]
If TOS-i+1 < 0
Output : " Stack underflow" and exit.

Step 2: [Return the i^{th} element from the top of the stack]
Return[S(TOS –i+1)]

Update Operation: It is required when the content of some location in a stack is to be changed. Suppose one wants to update information at the i^{th} location in the stack 's'. We move TOS pointer to the i^{th} location from the top of the stack and input the new value of that location.

Step 1: [Check for underflow]
If TOS-i+1<0
Output "stack underflow" and exit.

Step 2: [Change the element]
S(TOS-i+1) = value

Step 3: Return

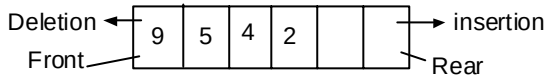
Polish Notation: The process of writing the operator of an expression either before their operands or after them is called the polish notation in honor of its discover, the "Polish mathematician Jan Luksiewicz.".

The polish notations are classified into three categories: these are:

- Infix
- Postfix
- Prefix

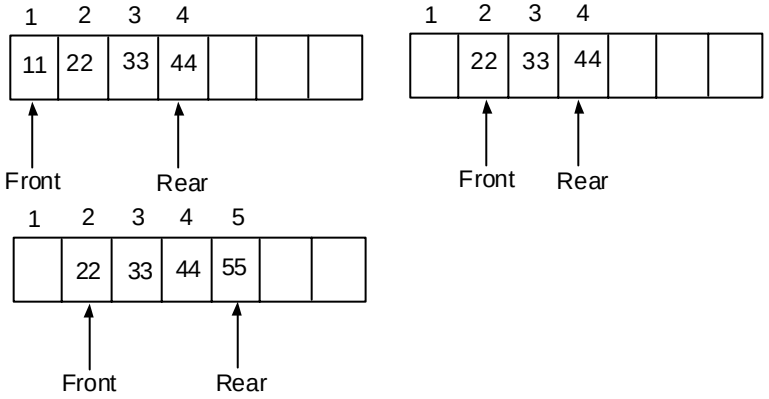
For example, take an expression a+b, for infix +sign is placed between two operands 'a+b'. For postfix, we can write the same expression as 'ab+' i.e operator is placed after the operands.
For prefix the same expression can be written as +ab.

Queue:



This is a linear data structure used to represent a linear list and permits deletion to be performed at one end of the list and insertion on the other end of the list. The information is such a list is processed in the same order as it was received, i.e on first come first serve basis.(FCFS)

A queue has two pointes; front and rear , pointing to the front and rear elements of the queue respectively. Consider a queue consisting of 'n' elements and an element value which we have to insert into the queue. The value NULL (0) of front pointer employees an empty queue. Figure given above illustrates a queue.



Algorithm to insert an elements into queue:

Step 1: [check overflow condition]
If rear ≥ size
o/p : "overflow" and return

Step 2: [Increment rear poi
nter]
Rear = rear+1

Step 3: [Insert an item]
Q[rear] = value

Step 4: [Set the front pointer]
If front = 0
Front = 1

Step 5: Return

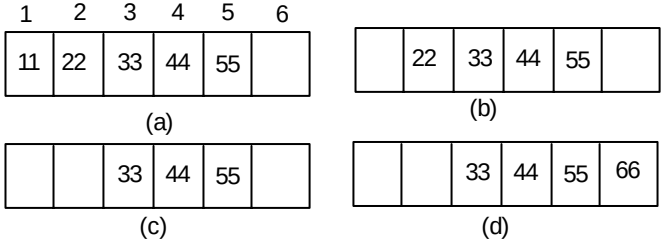
Algorithm to Delete an element form queue:

Step 1: [check the underflow condition]
If front = 0
Output: " Underflow" and return.

Step 2: [Remove an element]
Value = Q[front]

Step 3: [check for empty queue]
If front = rear
Front = 0
Rear = 0
Else
Front = front + 1

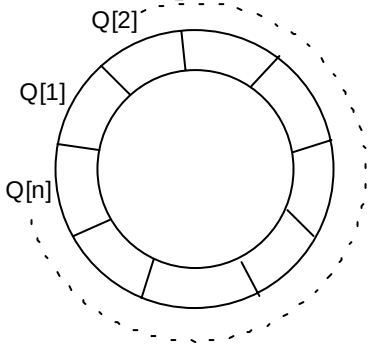
Step 4: Return (Value)

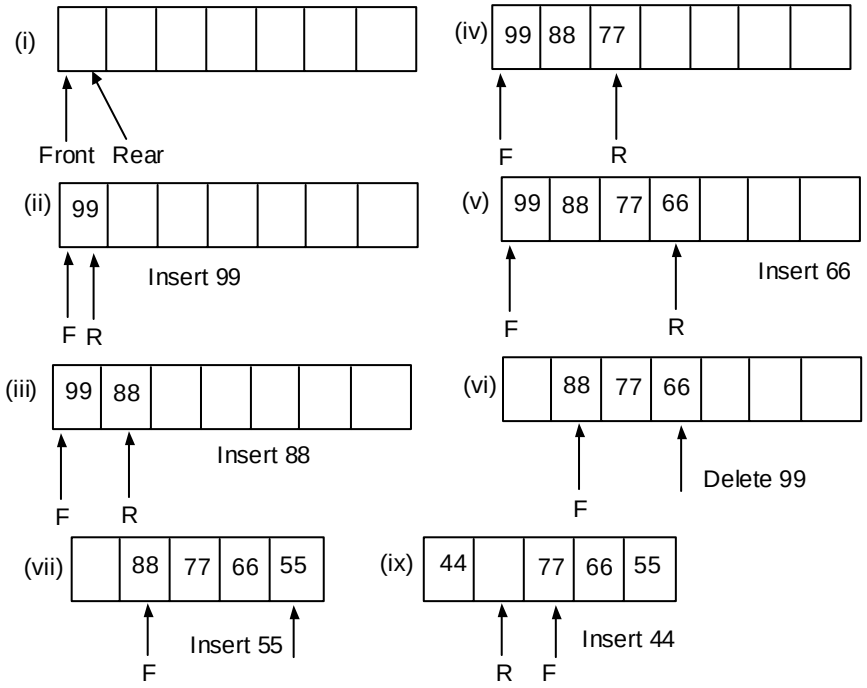


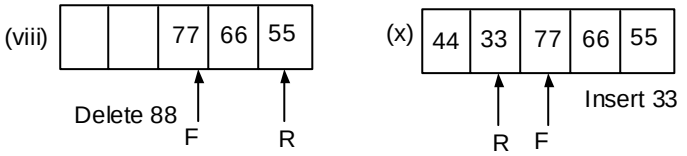
Circular queue:

Suppose we have an array Q that contains n element in which Q_1 comes after Q_n in the array. When this technique is

used to construct a queue then the queue is called the circular queue. In other words we can say that a queue is called circular when the last room comes just before the first room. Fig given below show the circular queue .

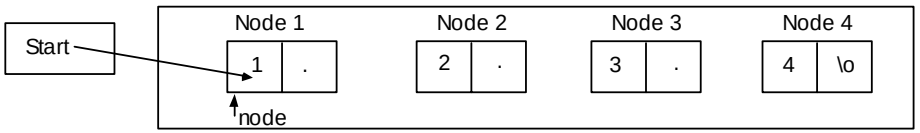






Linked List :

A linked list is defined as the collection of nodes.



Each node has two port.

- information
- Pointer to the next node.

Information part may consist of one or more than one fields. In other words a linked list consist of a series of a structure contains one or more then one contiguous information fields and a pointer to a structure containing its successor. The last node of the list contains NULL (‘\0’).

The pointer contains the address of the location where the next information is stored. A linked list also contains a list pointer variable called start, which contains the address of the first node is the list. Hence , there is an arrow drawn from start to the node in the linked list. If there is no node in the list, the is called NULL list or EMPTY list.

The different operations performed on the linked list are:

(1) Insertion:

a. Inset a node at the beginning of the list.
b. Inset a node at the end of the list.

c. Inset a node between the nodes.

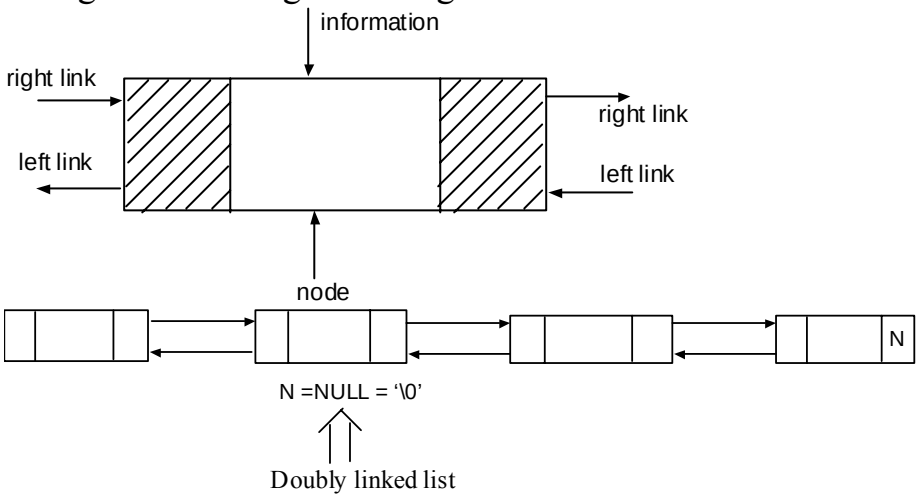
(2) Deletion

a. Delete a node at the beginning of the list.
b. Delete a node at the end of the list.
c. Delete a node between the nodes.

(3) Traversing . : Traveling from one node to another node.

Doubly linked list:

A singly linked list is so named because each list element contain a pointer to the next element . In that list traveling is possible only in one direction. Some times it is required to transverse the list in either direction forward or backward. This involves the performance and efficiency of algorithms. So traversing of linked list in both the directions requires nodes having the links as given in figure below:



The links are used to denote the predecessor and successor of a node. The link denoting the predecessor of a node is called the left link and that denoting its successor is right link. A list with this type of arrangement is called Doubly Linked List as shown in the figure above. Now it is possible to define a doubly linked as a collection of nodes, each nodes having three fields.

- pointer to previous node (pointer to predecessor)
- Information field
- Pointer to next node (pointer to successor)

Different operations performed on doubly linked list are:

1. Insertion 2. Deletion 3. Traversing.

Non linear data structure:

(1) Trees

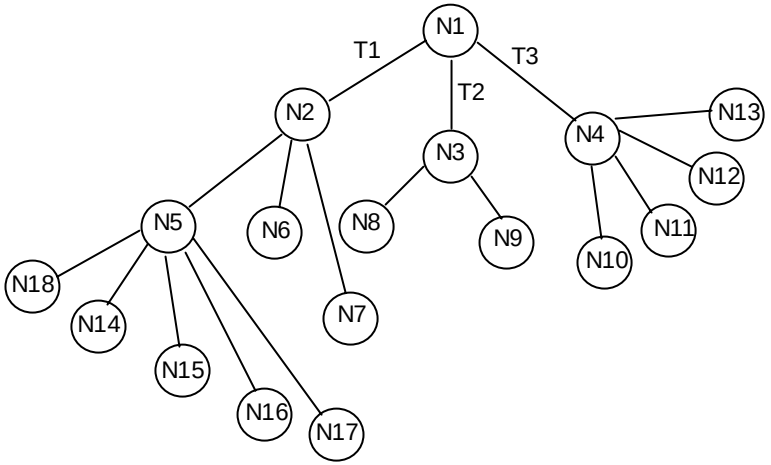


Fig. Tree T having 18 nodes

A tree T is defined as a set of finite set of one or more nodes such that.

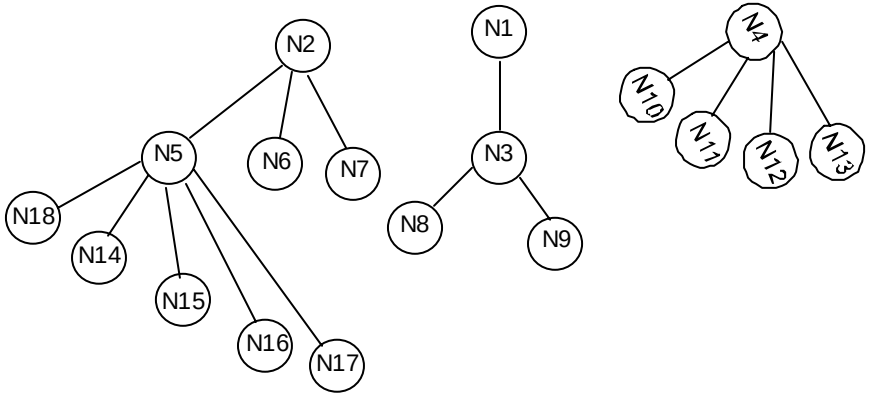
- There is a special node called the root R.
- The remaining nodes are divided into $n \geq 0$

Disjoint sets T1,T2,T3,……Tn , where each of these set is a tree . T1, T2, T3,……Tn is called the subtree of the root.

The root node N1 has three subtrees T1,T2,T3 and the roots of these subtrees are N2,N3,N4 respectively . The roots of these

subtrees of a node is called its degree, root N1, has degree 3. Similarly node N5 has degree 5 and node N14 has degree 0. A node with degree 0 is called leaf. N6,N7,N8,N9,N10,N11,N12,……N18 are called leaf nodes. The leaf nodes are also called Terminal nodes and rest of nodes in a tree is called non-terminal.

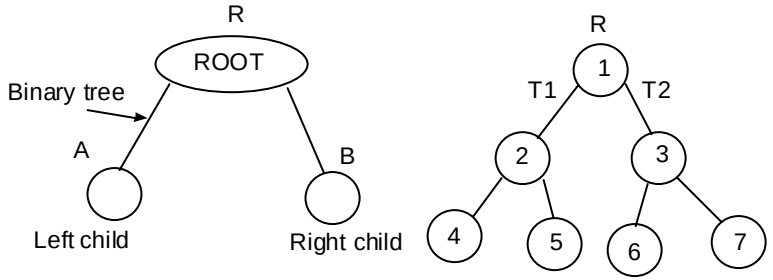
Forest:



A forest is a set of n ≥ 0 disjoint trees, where n represents no of nodes in the tree. The meaning of forest is very close to a tree because if we remove the root of a tree, we get a forest. For e.g in the fig. above we remove N1, we get the forest of three trees. It is shown in the figure above.

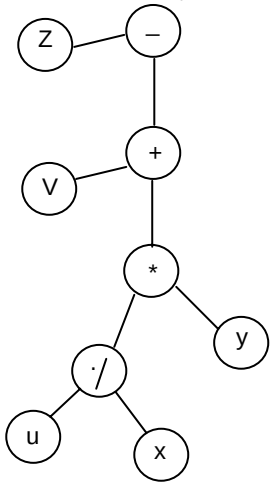
Binary tree:

A binary tree T is a finite set of nodes, which is either empty or consists of special nodes called R and two disjoint binary tree T1 and T2 (Which are called the left subtree and right subtree respectively). If T1 is not empty then the root of T1 is called the left successor of R. If T2 is not empty then the root of T2 is known as right successor of R.



Expression :

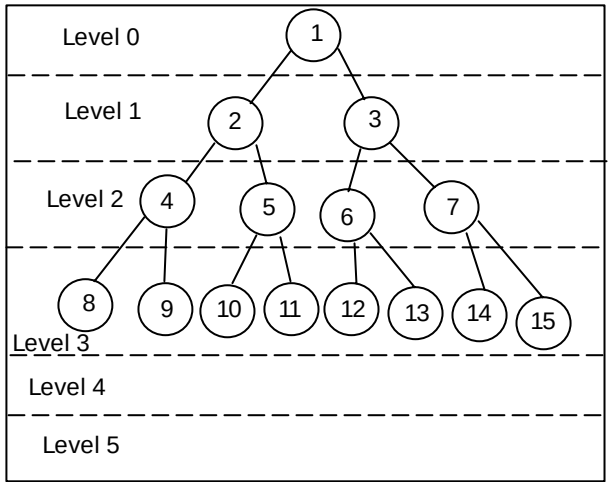
E = v + u/x* y-z



Complete Binary Tree: A binary tree T is called complete if each node of T can have at the most two children. A binary tree at level L can have at the most 2^L nodes.

If L = 0 $2^0 = 1$
If L = 1 $2^1 = 2$
If L = 2 $2^2 = 4$
If L = 3 $2^3 = 8$

Implies that there is only one node in the tree that is root.



Traversing Binary Tree:

1. Pre order	Traversal	RAB
2. In order	Traversal	ARB
3. Post order	Traversal	ABR

2. Graphs: graphs G is defined as set of two tuples G = (V,E). Where ,

V represents set of vertices of G and E represents the set of edges of G. There exists a mapping from the set of edges to a set of pairs of element of V. Fog eg. given below shows the different types of graphs.

Fig.(a)

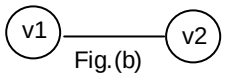


Fig.(b)

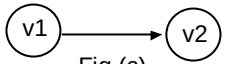


Fig.(c)

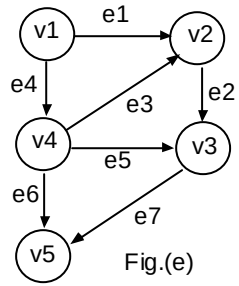


Fig.(e)

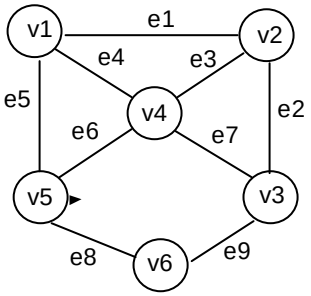


Fig.(d)

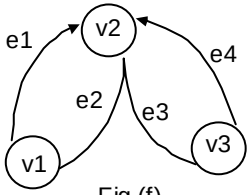
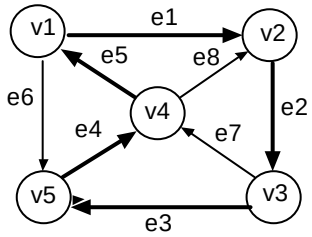


Fig.(f)

From the figure it is clear that fig. (a) consists of two vertices each one is stand alone. Fig (b) contains an undirected graph having one edge and two vertices. Fig. (c) consists of two vertices and one edge and it is called directed graph. Fig. (d) illustrates an undirected graph with six vertices and nine edges. Fig (e) is a directed graph with five vertices and seven edges. Fig (f) consists of three vertices and four edged in which two edges are directed and two are undirected known as mixed graph.

Path: A sequence of edges of a digraph (directed graph) such that the terminal vertex of an edge sequence in the initial vertex of the next if exits is called the path. For example:



E = {(v1,v2), (v2,v3), (v3,v4),(v4,v5),(v5,v1)}

If we consider the element of E, we find that E is a set of five tuples, i.e for edge.

e_1 = (v1,v2)

e_2 = (v2,v3)

e_3 = (v3,v4)

e_4 = (v4,v5)

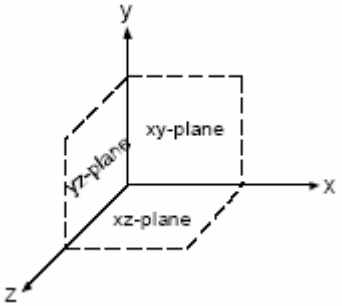
e_5 = (v5,v1)

From e_1 and e_2 we find that v_2 is common in both. In the e_1 , v2 is the terminal vertex and it is initial vertex of e_2. Similarly in e_2 and e_3 , v3 is corner and so on. Thus we can say that there is cycle and there exist a circular path that is shown by the dark line in the above fig.

Date: 2065/7/24

Three dimensional Graphics:

6.1 Three Dimensional object to screen perspective viewing Transfomation:



The 3D viewing process is inherently more complex then is the 2D viewing process. In 2D we simply specify specify the window on 2D world and a viewpoint on the 2D-view surface. Conceptually objects in the world are clipped against the window and are then transformed into the viewpoint for display. The extra complexicity 3D-viewing is caused in the part by the added dimensions and in part by the fact that display device are only 2D

The solution is mismatch between 3D object and 2D displays is accomplished by introducing projections which transforms 3D objects onto a 2D-projection plane.

In 3D-viewing , we specify a new volume in world a projection onto a projection plane and a viewport on the view surface. Conceptually objects in the 3D-new volume and are then projected. The content of the projection of the new volume onto the projection plane, called the window, are then displayed (transformed into the view port for display). Figure below shows the conceptual model of the 3D-viewing process.

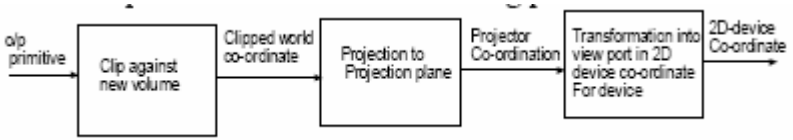


Fig. Conceptual model of 3D-viewing process.

Projections: Projections transform points in a co-ordinate system of dimension 'n' into points in a co-ordinate system of dimension less than 'n' projection of 3D-object is defied by straight projection rays (Called projectors) emanding from a centre of projection, passing through each point of the object and intersecting a projection plane to form the projected object. If the projection is onto a plane (rather than some curved surface) and uses straight objectors (rather then curved) then we call it planargeometric projection.

Types of Geometric Projection:

1. Parallel projection
2. Perpesctive projection.

Parallel projection:

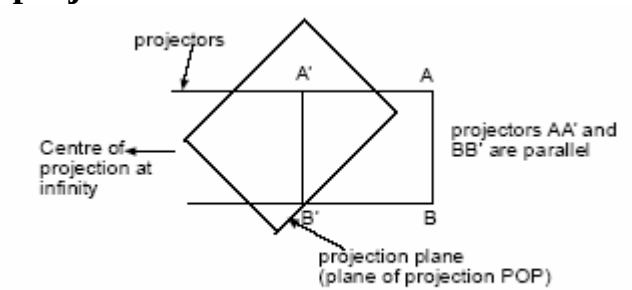


Fig. line AB and its parallel projection A'B'

If the distance between COP and POP is infinite then projection is called parallel projection, i.e the rays from COP are paralleled. So it maintains relative proportion of the object. In this projection we extend parallel lines from each vertex plane of the screen.

Equation of line in 2D is

$$(y-y_1)/(x-x_1) = (y_2-y_1)/(x_2-x_1)$$

In parametric form,

$$x = x_1+(x_2-x_1)u$$
$$y = y_1+(y_2-y_1)u \quad u \in [0,1]$$

Similarly in 3D,

$$x = x_1+(x_2-x_1)u$$
$$y = y_1+(y_2-y_1)u$$
$$y = z_1+(z_2-z_1)u$$

Let direction of projection is given by vector $[x_p,y_p,z_p]$ and the image is to be projected on the xy-plane. If we have a point on the object at (x_1,y_1,z_1) and the projected point be (x_2,y_2) then the equation of the line is ,

$$x = x_1+x_pu$$
$$y = y_1+y_pu$$
$$z = z_1+z_pu$$

But on the xy plane

$$0 = z_1+z_pu$$
$$\therefore u = -z_1/z_p$$
$$x_2 = x_1-z_1(x_p/z_p)$$
$$y_2 = y_1-z_1(y_p/z_p)$$

$$\text{i.e} \begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & -x_p/z_p & 0 \\ 0 & 1 & -y_p/z_p & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$$

However, while drawing the projected image we just ignore the z plane.

(2) Perspective projection: If the distance between POP and COP is finite and converges to a single point (Vanishing point) then the projection is called perspective projection i.e it

transforms points of 3D object along projection lines that meet at vanishing point. The size of the object doesn't remains same and varies with the distance of the object from COP. Objects further from viewing position is displayed smaller than objects of same size that are nearer to viewing positions. Perpesctive projection is more realistic than parallel.

If the COP at [x_c,y_c,z_c] and the point on the object is [x,y,z] then projection ray will be the line joining these points. Equation of the line is

$x = x_c+(x_1-x_c)u$

$y = y_c+(y_1-y_c)u$

$z = z_c+(z_1-y_c)u$

The projected point on xy plane i.e (x_2,y_2) will be the point where the line intersects the xy plane.

i.e $z_c = 0$, $z_c+(z_1-z_c)u = 0$

i.e $u = -z_c/(z_1-z_c)$

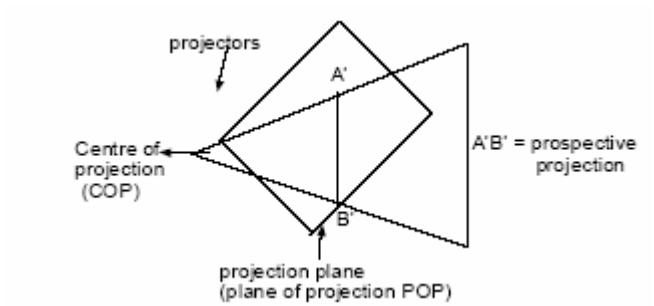
$x_2 = (x_cz_1-x_1z_c)/(z_1-z_c)$

$y_2 = (y_cz_1 – y_1z_c)/(z_1-z_c)$

$[x_2,y_2,z_2] = [(x_cz_1-x_1z_c)/(z_1-z_c), (y_cz_1 – y_1z_c)/(z_1 – z_c), 0]$

In homogenous form (Homogenous co-ordinate system)

$[x_2,y_2,z_2,1] = [x_cz_1-x_1z_c, y_cz_1-y_1z_c, 0, z_1-z_c]$



A'B' = perspective projection of AB

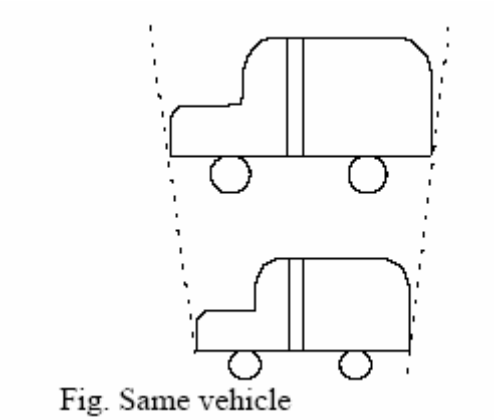


Fig. Same vehicle

In Matrix form ,

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} -z_c & 0 & x_c & 0 \\ 0 & -z_c & y_c & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -z_c \end{bmatrix} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$$

$$\therefore T_{persp} = \begin{bmatrix} -z_c & 0 & x_c & 0 \\ 0 & -z_c & y_c & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -z_c \end{bmatrix}$$

However z value is needed for hidden surface detection, so transformation matrix changes to

$$\begin{bmatrix} -z_c & 0 & x_c & 0 \\ 0 & -z_c & y_c & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & -z_c \end{bmatrix}$$

Date: 2065/7/25

Extension of two-dimensional to three dimensional transformation:

Just as 2D-transfromtion can be represented by 3x3 matrices using homogeneous co-ordinate can be represented by 4x4 matrices, provided we use homogenous co-ordinate representation of points in 3D space as well

Translation:

- Translation in 3D is similar to translation in the 2D except that there is one more direction parallel to the z-axis.
- If , t_x,t_y,and t_z are used to represent the separate shifts , then the translation of the position (x,y,z) into the point (x',y',z') is done by

$x' = x + t_x$

$y' = y + t_y$

$z' = z + t_z$

- In matrix notation using normalized homogeneous coordinate this is performed by the matrix multiplication.

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

Inverse translation,

$T^{-1}(t_x,t_y,t_z) = T(-t_x,-t_y,-t_z)$

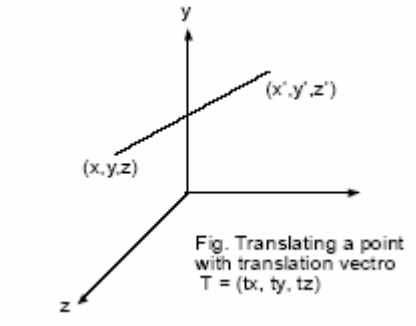


Fig. Translating a point with translation vectro T = (tx, ty, tz)

Scaling:

(i) About origin:

Explicit expression:

$X' = x * s_x$

$Y' = y * s_y$

$Z' = z * s_z$

- The matrix representation for the scaling x'-formation of a position p(x,y,z) relative to the co-ordinate origin.

Matrix form:

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

Where scaling parameter s_x,s_y,s_z are assigned positive values.

(iii) About any fixed point: (x_f, y_f, z_f)

- Scaling w.r.t to a selected fixed position (x_f,y_f,z_f) can be represented with the following transformation.

Step:1 Translate the fixed point with the origin $T(-x_f,-y_f,-z_f)$

Step:2 Scale the object relative to the co-ordinate origin.

$S(s_x,s_y,s_z)$

Step:3 Translate the fixed point back to its origin position.

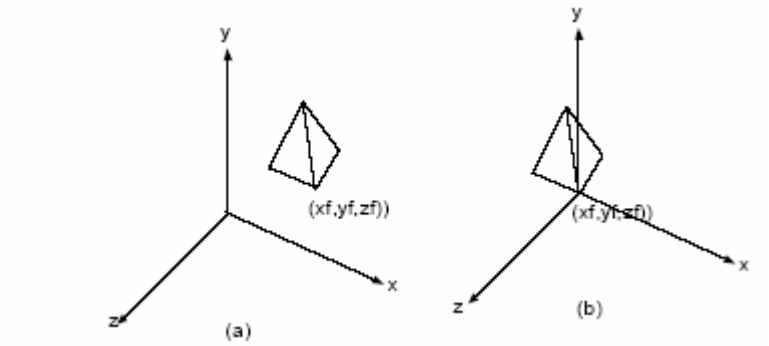
$T(x_f, y_f,z_f)$

Net transformation

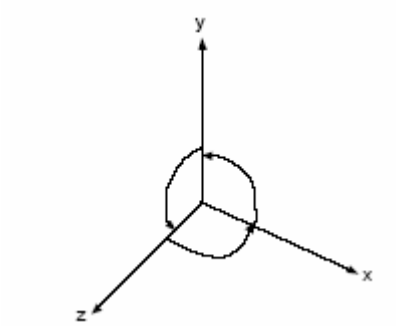
$T = T(x_f,y_f,z_f).\ S(s_x,s_y,s_z).\ T(-x_f,-y_f,-z_f)$

$$= \begin{bmatrix} 1 & 0 & 0 & x_f \\ 0 & 1 & 0 & y_f \\ 0 & 0 & 1 & z_f \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -x_f \\ 0 & 1 & 0 & -y_f \\ 0 & 0 & 1 & -z_f \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} s_x & 0 & 0 & (1-s_x)x_f \\ 0 & s_y & 0 & (1-s_y)y_f \\ 0 & 0 & s_z & (1-s_z)z_f \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Rotation:



Axis of rotation	Direction of +ve rotation
X	y to z
Y	z to x
Z	x to y

Rotation About z-axis:

Z-component does not change. The rotation of a point about zaxis is equivalent to rotation of point projected on xy plane (keeping z-value constant) which is equivalent to 2D rotation i.e rotation angle is equal to the angle between initial and final projected points on xy-plane

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

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

$z' = z$

$$R_z = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

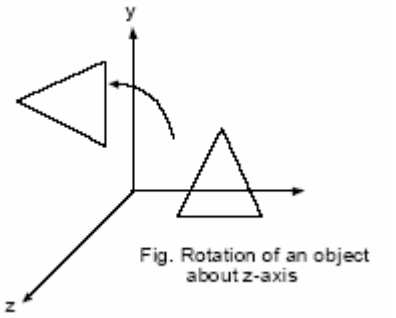


Fig. Rotation of an object about z-axis

Rotation about x-axis:

$x' = x$

$y' = y\cos\theta + z\sin\theta$

$z' = y\sin\theta + z\sin\theta$

$$\therefore R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

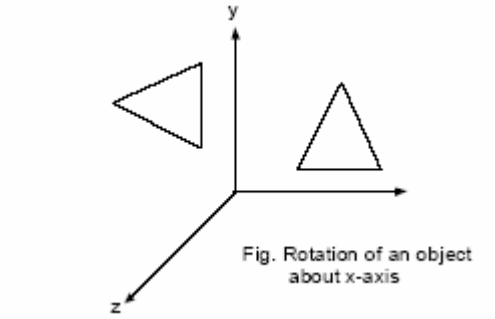


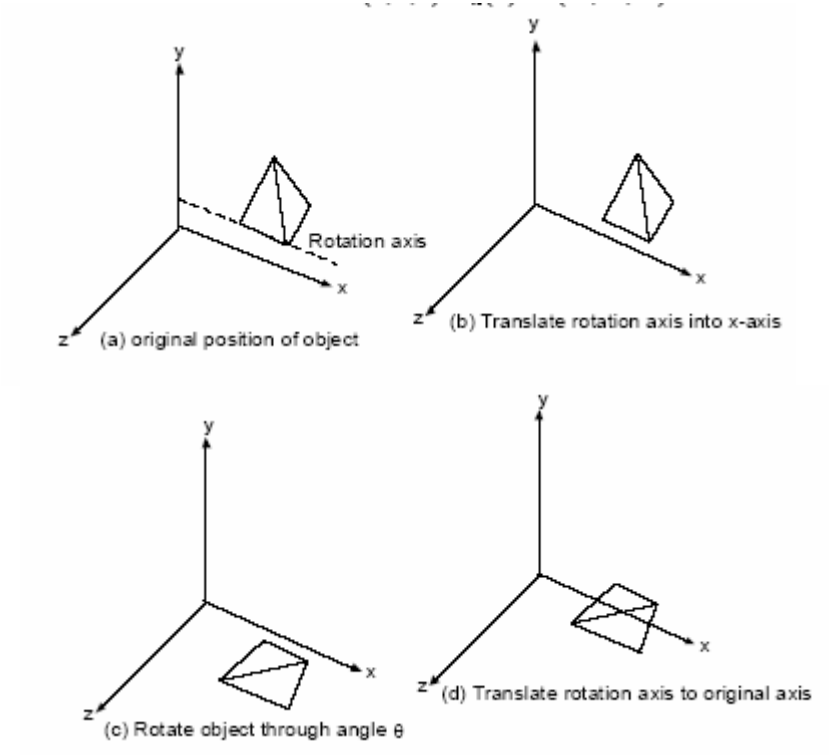
Fig. Rotation of an object about x-axis

About y-axis:

Y' = y

z = zcosθ – xsinθ

x = zsinθ+ x cosθ



Rotation about any co-axis:

(a) Parallel to any of the co-axis: When an object is to be rotated about an axis that is parallel to one of the co-ordinate axis, we need to perform some series of transformation.

1. Translate the object so that the rotation axis coincides with the parallel co-ordinate axis. T(-a,-b,-c) where, (a,b,c) is any point on the rotation axis.
2. Performed the specified rotation about the axis. $R_x(\theta)$
3. Translate the object so that the rotation axis is moved to its original position. T(a,b,c)

Net transformation,

= T(a,b,c) $R_x(\theta)$. T(-a,-b,-c)

(b) Not parallel to any of the co-axis:

When an object is to be rotated about an axis that is not parallel to one of the co-ordinate axes, we need to perform some series of transformation.

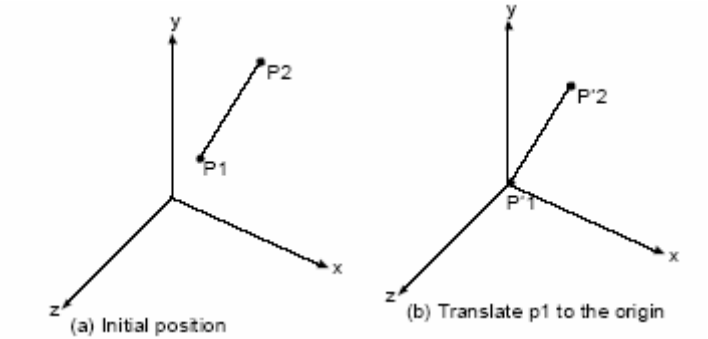
(i) Translate the object such that rotation axis passes through co-ordinate origin.

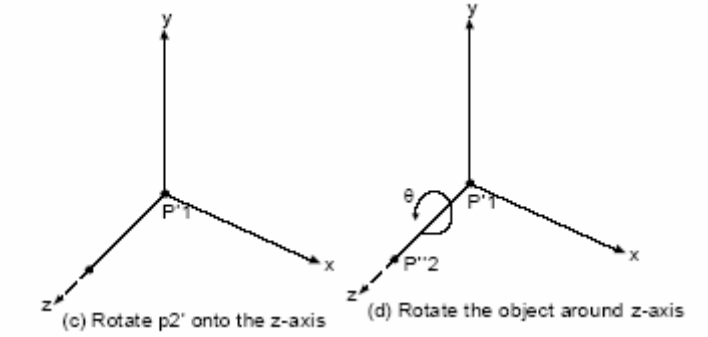
(ii) Rotate the axis such that axis of rotation coincides with one of the co-ordinate axis.

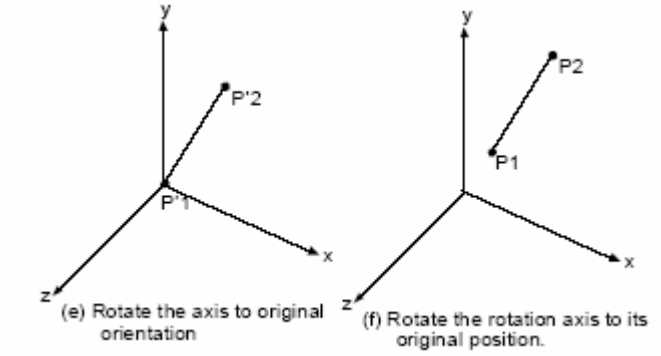
(iii) Perform the specific rotation about the ordinate axis.

(iv) Apply inverse rotation to bring the rotation axis back to its original orientation.

(v) Apply inverse translation to bring the rotation axis back to its original position.







Let p_1, p_2 be the two end points of axis of rotation then the axis vector is given by:

$$\triangle = p_2 - p_1 = (x_2,y_2,z_2) - (x_1,y_1,z_1)$$
$$= (x_2-x_1,\ y_2-y_1,\ z_2-z_1)$$

Unit vector along rotation axis is

$$u = \frac{\vec{v}}{|\vec{v}|} = \frac{(x_2-x_1, y_2-y_1, z_2-z_1)}{\sqrt{(x_2-x_1)^2+(y_2-y_1)^2+(z_2-z_1)^2}}$$

$= (a,b,c)$ [let]

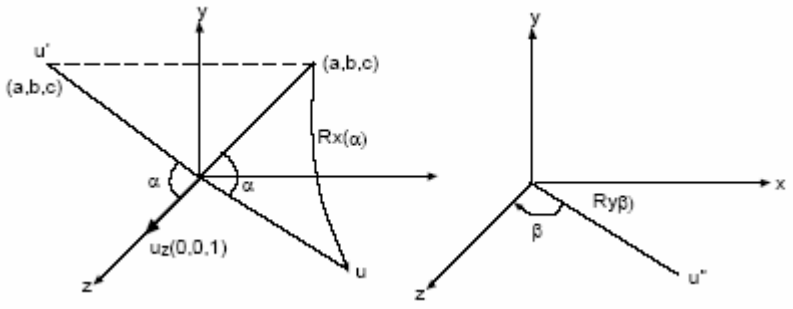
Where (a,b,c) are direction cosines of rotation axis i.e $a_2+b_2+c_2 = 1$.

(i) $$T(-x_1,-y_1,-z_1) = \begin{bmatrix} 1 & 0 & 0 & -x_1 \\ 0 & 1 & 0 & -y_1 \\ 0 & 0 & 1 & -z_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(ii) For step II rotation can not be performed directly since the rotation is not about any of the co-ordinate axis hence. It is done in two steps.

(a) Rotation by angle α about x axis.

(b) Rotate by angle β about y axis.



Step (a)

Angle for rotation of u around x-axis into xz plane is equal to angle between u' (projection of u on yz plane and z-axis).

$$\text{Cos}\alpha = \frac{\vec{u}'.\vec{u}_z}{|\vec{u}'||\vec{u}_z|} = \frac{(0,b,c).(0,0,1)}{\sqrt{b^2+c^2}} = \frac{c}{d}$$

$$\text{Sin}\alpha = \frac{\vec{u}'\times\vec{u}_z}{|\vec{u}'||\vec{u}_z|} = \frac{(0,b,c)(0,0,1)}{\sqrt{b^2+c^2}} = \frac{b}{d} \qquad \text{Where, } d = \sqrt{b^2+c^2}$$

$$\therefore\ R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha & 0 \\ 0 & \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c/d & -b/d & 0 \\ 0 & b/d & c/d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Steps (b)

U moves to the position u'' after being rotated by α about x axis so the position of u'' is given by

$u'' = R_x(\alpha).u$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c/d & -b/d & 0 \\ 0 & b/d & c/d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ 1 \end{bmatrix} = \begin{bmatrix} a \\ 0 \\ d \\ 1 \end{bmatrix}$$

β = angle between $\vec{u}''$ and $\vec{u}_z$

$$\cos\beta = \frac{\vec{u}''\times\vec{u}_z}{|\vec{u}''||\vec{u}_z|} = \frac{(a,0,d)(0,0,1)}{\sqrt{a^2+d^2}} = d$$

$$\text{Sin}\alpha = \frac{\vec{u}''\times\vec{u}_z}{|\vec{u}''||\vec{u}_z|} = \frac{(a,0,d)\times(0,0,1)}{\sqrt{a^2+d^2}} = -a$$

$$R_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} d & 0 & -a & 0 \\ 0 & 1 & 0 & 0 \\ a & 0 & d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Step: 3

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

Step: 4

(a) $R^{-1}{}_y(\beta)$

(b) $R^{-1}{}_x(\alpha)$

Step: 5

$T(x_1, y_1, z_1)$

$\therefore$ Net transformation,

$= T(x_1,y_1,z_1).\ R^{-1}{}_x(\alpha)\ R^{-1}{}_y(\beta)\ R_2(\theta).\ R_y(\beta).R_x(\alpha)T(-x,-y,-z)$

Reflection:

(i) About Axis: Reflection about axis is equivalent to 180° rotation about the axis.

About z-axis:

$$R_{fz} - R_z(180^\circ) = \begin{bmatrix} \cos 180 & -\sin 180 & 0 & 0 \\ \sin 180 & \cos 180 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

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

About x-axis:

$$R_{fx} = R_x(180)$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos 180 & -\sin 180 & 0 \\ 0 & \sin 180 & \cos 180 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

About y-axis:

$$R_{fy} = R_y(180)$$

$$= \begin{bmatrix} \cos 180 & 0 & \sin 180 & 0 \\ 0 & 1 & 0 & 0 \\ -\sin 180 & 0 & \cos 180 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(ii) About any axis:

Net transformation:

$T(x_1,y_1,z_1)\ R_x(-\alpha)\ R_y(-\beta)\ .\ R_{fz}\ R_y(\beta).\ R_x(\alpha)\ T(-x_1,-y_1,-z_1)$

(iii) About plane:

X' = x ,

Y' = y

Z' = z

(a) about xy plane (z-axis)

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

(b) about xz plane (y-axis):

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

(c) about yz-plane (x-axis):

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

About any plane:

Step:1: Translate the plane such that it passes through origin i.e normal vector passes through origin.

Step:2: Rotate the plane such that normal vector lies on one of the co-ordinate axis.

Step:3: Perform reflection about the plane whose normal vector is one of the co-ordinate axis.

Step:4: Rotate back the plane such that normal vector takes its original orientation.

Step5: Translate back the plane to its original position.

Shearing:

In 2D, shearing about x-axis means x-values changes by amount proportional to y-values and y-values remains same.

However in 3D, z-axis means x and y value change by amount proportional to z-value and z-value remains same. i.e $x' = x + S_{hx}.z$

$y' = y + S_{hy}.z$

$z' = z$

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & S_{hx} & 0 \\ 0 & 1 & S_{hy} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Similarly,

$$SH_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ S_{hx} & 1 & 0 & 0 \\ S_{hx} & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$SH_y = \begin{bmatrix} 1 & S_{hx} & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & S_{hx} & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

#1. Find the transformation matrix for the reflection of 3D object about plane $2x+y-4z+5 = 0$.

#2. Find the transformation matrix for the reflection of 3D abject about plane passing through origin whose normal vector is (i+j +k) (all are vector).

#3. Reflection of 3D object about axis joining the points (1,2,3) and (4,5,6).

4. Rotation of 3D object by 60° about axis joining the points (1,2,3) and (6,5,7).

Realization In 3-D Graphics:

Many computer graphic application involve the display of 3-D objects and scenes. For examples CAD systems allow the users to manipulate models of machine components, automobile bodies and aircraft parts. Simulation system present a continuously moving picture of a 3-D world to the pilot of ship or aircraft. These applications differ from 2-D applications not only in the added dimension: they also requires concern for realism in the display of objects.

Producing a realistic image of a 3-D scene on a 2D-scene on a 2D-display present many problem. How is depth, the 3rd dimension to be displayed on the scene? How are pars of the object that are hidden by other object to be modified and removed form the image? How is a 3-D world to be modulated in a computer in a computer so that images can be generated? There are a no. of techniques for achieving realism.

The basic problem addressed by visualization technique is sometimes called depth cueing. When a 3-D scene is projected onto a 2-D display scene , information about the depth of objects in the image tends to be reduced or loosed entirely. Techniques that provide depth cues are designated to restore or enhance the communication of depths to the observer. The different technique for achieving realism are:

- Parallel projection.
- Perspective projection
- Intensity cues.
- Stereoscopic views.
- Kinetic depth effect.
- Hidden line elimination
- Shadding with hidden surface removed
- 3-D images.

Modelling 3-D scenes: The techniques used to generate different kinds of 3-D scenes are start from a model of the scene. The model is needed for two purpose.

1. It is used by viewing algorithm together with information about the location of the viewer.
2. It is used to modify and analyze the objects in the scene, activities usually considered part of the application program.

The information in a model of a 3-D scene can be divided into two important classes: geometry and topology. Geometry is concerned with measurements , such as the location of a point or the dimension of the object. Topological information records the structure s of a scene: how points are combined to form polygons, how polygons form object and how object from scenes.

3-D objects Representation:

Polygon surface:

- The most commonly used boundary representation for a 3-D graphic object is a set of surface polygon that enclose the object interior. Many graphic system stored all object description a sets of surface polygon. This simplifies and speed of the surface rendering and display a object scene all the surface describe with linear equation.
- A polygon representation of a polyhedron precisely defines the surface of a objects.

Polygon Table Representation:

Polygon surface is generally used to represent 3-D object. It consists of two parts:

(1) Geometric table.

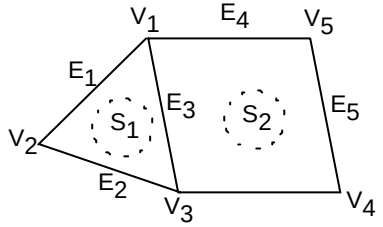
(2) Attribute Table.

(a) Geometric table: It contains vertex co-ordinates and parameter to identify the spatial orientation of polygon surface.

(b) Attribute table: It gives attribute information for an object (degree of transparency, surface reflectivity etc)

Geometric data consists of three tables:

(i) Vertex table: It stores co-ordinate values for each vertex of the object.



Vertex	coordinate
V_1	(x_1,y_1,z_1)
V_2	(x_2,y_2,z_2)
V_3	(x_3,y_3,z_3)
V_4	$(x_4,y4,z_4)$
V_5	(x_5,y_5,z_5)

(ii) Edge Table:

It contains pointers back into vertex to identify the edges for each polygon.

Edge	Vertex
E_1	(v_1,v_2)
E_2	(v_2,v_3)
E_3	(v_3,v_1)
E_4	(v_1,v_5)
E_5	(v_3,v_4)

Surface table:

Surface	Edge
S_1	(E_1,E_2,E_3)
S_2	(E_3,E_4,E_5,E_6)

Rues for creating Geometric table:

1. Every vertex in listed as an end point for at least tow edges.
2. Every edge is par of at least one polygon
3. Each polygon has at least one shared edge.
4. Every surface is close.

Plane Equation:

It is used to determine the spatial orientation of the individual surface component of the object.

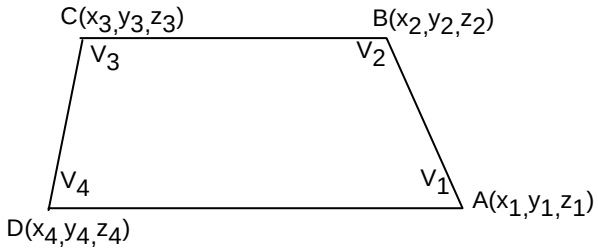
The equation of the plane is

$Ax+By+Cz+D = 0$

Solution of plane Co-efficient:

1. Algebraic Approach.
2. Vertex Approach.

(1) Algebraic Approach:



If (x_1,y_1,z_1) , (x_2,y_2,z_2) and (x_3,y_3,z_3) are three successive vertex of the plogygon then

$Ax + By + Cz = -D$

$(A/D)x + (B/D)y + (C/D)z = -1$

$(A/D)x_1 + (B/D)y_1 + (C/D)z_1 = -1$ …….(i)

$(A/D)x_2 + (B/D)y_2 + (C/D)z_2 = -1$ …….(ii)

$(A/D)x_3 + (B/D)y_3 + (C/D)z_3 = -1$ …….(iii)

By cramers rule:

$$A = \begin{vmatrix} 1 & y_1 & z_1 \\ 1 & y_2 & z_2 \\ 1 & y_3 & z_3 \end{vmatrix}$$

$$B = \begin{vmatrix} x_1 & 1 & z_1 \\ x_2 & 1 & z_2 \\ x_3 & 1 & z_3 \end{vmatrix}$$

$$C = \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix}$$

$$D = \begin{vmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{vmatrix}$$

Expanding the determinant we can write that,

$A = (y_3z_3 - y_3z_2) - y_1(z_3-z_2) + z_1(y_3-y_2)$

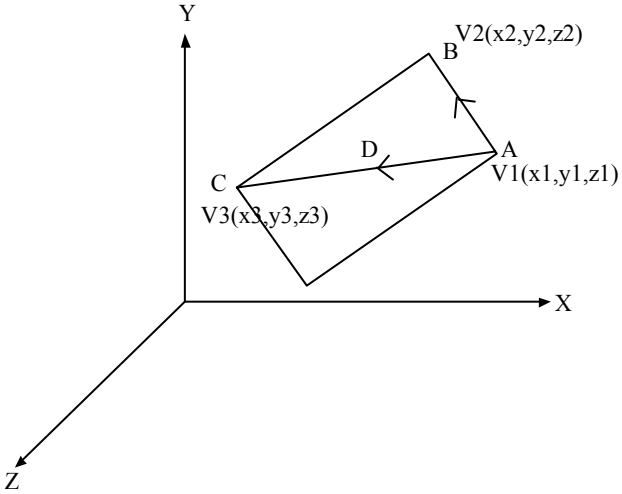
$B = (z_3-z_2) - (x_2z_3 - x_3z_2) + (x_2-x_3)$

$C = (y_2 - y_3) - y_1(x_2 - x_3) + (x_2y_3 - x_3y_2)$

$D = -x_2(y_2z_3-y_3z_2) + y_1(x_2z_3-x_3z_2) - z_1(x_2y_3 - x_3y_2)$

As vertex values and other information are entered into the polygon data structure, values for A, B, C and D are computed for each polygon and store with other polygon data.

Vector Approach:



The position vector of v_1,v_2, and v_3 are

$\mathbf{V_1} = \mathbf{OA} = (x_1,y_1,z_1)$

$\mathbf{V_2} = \mathbf{OB} = (x_2,y_2,z_2)$

$\mathbf{V_3} = \mathbf{OC} = (x_3,y_3,z_3)$

Taking cross product $\mathbf{AB} \times \mathbf{AC}$

$\mathbf{AB} = \mathbf{OB} - \mathbf{OA}$

$= (x_2-x_1, y_2-y_1, z_2-z_1)$

$\mathbf{AC} = \mathbf{OC} - \mathbf{OA}$

$= (x_3-x_1, y_3-y_1, z_3-z_1)$

$$\therefore \ \mathbf{AB} \times \mathbf{AC} = \begin{bmatrix} i & j & k \\ x_2 - x_1 & y_2 - y_1 & z_2 - z_1 \\ x_3 - x_1 & y_3 - y_1 & z_3 - z_1 \end{bmatrix}$$

$\therefore$

$= i\ \{(z_3\text{-}z_1).(y_2\text{-}y_1) - (z_2\text{-}z_1).(y_3\text{-}y_1)\}$
$= -j\ \{(x_2 - x_1)(z_3 - z_1) - (z_2 - z_1)(x_3\text{-}x_1)\}$
$= k\ \{(x_2\text{-}x_1)(y_3 - y_1) - (y_2\text{-}y_1)(x_3\text{-}x_1)\}$

Component of **i**= $\begin{bmatrix} y_2 - y_1 & z_2 - z_1 \\ y_3 - y_1 & z_3 - z_1 \end{bmatrix}$

$$= \begin{bmatrix} 1 & y_1 & z_1 \\ 0 & y_2 - y_1 & z_2 - z_1 \\ 0 & y_3 - y_1 & z_3 - z_1 \end{bmatrix}$$

$R_2 \rightarrow R_1 + R_2$
$R_3 \rightarrow R_1 + R_3$

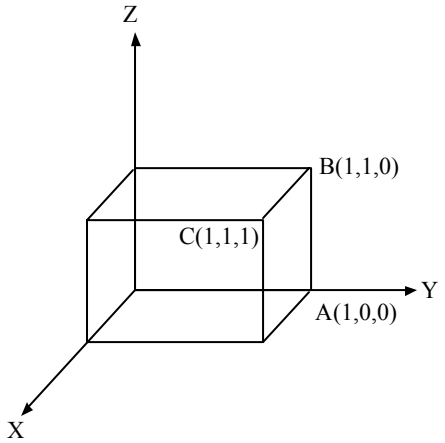
Note: The bold alphabets are vector notation.

Similarly,
B = Component of **j**
C = component of **k**

Hence , the plane coefficients (A,B,C) gives normal vector to the plane. Further , if P(x,y,z) be any point on the plane then,

N.P = (A, B, C). (x,y,z)
= Ax+By+Cz
$\therefore$ D = -N.P

Eg.



$\vec{N} = \overrightarrow{AB} \times \overrightarrow{AC}$
$\overrightarrow{AB} = \overrightarrow{OB} - \overrightarrow{OA}$
= (1-1,1-0,0-0)
= (0,1,0)
$\overrightarrow{AC} = \overrightarrow{OC} - \overrightarrow{OA}$
=(1-1,1-0,1-0)
= (0,1,1)

$\vec{N} = \overrightarrow{AB} \times \overrightarrow{AC}$
= (0,1,0)×(0,1,1)
= **j**×(**j**+**k**)
= 0+**i**
= i
= (1,0,0)

Therefore, A = 1, B= 0, C=0
$\vec{D} = -\vec{N}.\vec{P}$
= -(1,0,0).(1,0,0)
= -1
$\therefore \ \vec{D} = 1$
$\therefore$ equation of plane is

Ax+By+Cz+D = 0
Or , x-1 = 0
The plane equation are used ot determine the positon of spatial points relative to the palne surface of an object .

If p(x,y,z) is any point on the plane then the distance between the point and the plane is

$$d = \frac{Ax + By + Cz}{\sqrt{A^2 + B^2 + C^2}}$$

If d = 0 , Ax+By+Cz+D = 0 i.e point is on the plane.
If d<0, Ax+By+Cz+D = 0 i.,e point is inside the plane.
If d>0, Ax+By+Cz+D = 0 i.e point is outside the plane.

The side of the plane that faces the object interior is 'inside' face and visible face is 'outside' face.

If polygon vertices are specified in counter clockwise direction when viewing the outer side of the plane in right handed co-ordinate system the direction of normal vector will be from inside to outside.

Date: 2065/8/16

Polygon Meshes: It is a collection of edges , vertices and surfaces such that each edge is shered by at most

1. Two polygons. An edge connects two vertices and a polygon
2. is a closed sequence of edge. An edge can be shared by two adjacent polygon and vertices is shared at least two edges. \

Figure:

Visible Surface detection Method:

- The method that are used for identifying those of a scene that are visible from a chosen viewing position are referred to as visible surface detection methods.
- Sometimes these methods are also known as hidden surface elimination method.

Classification of visible surface Detection Algorithm:

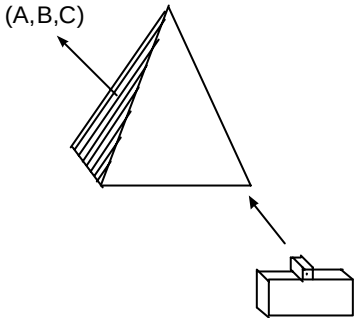
- Visible surface detection algorithms are broadly classified according to whether they deal with object definitions directly or with their projected images . These tow approaches are called
- Object space Method.
- Image Space Method.
- An object space method compares objects and parts of objects to each other within scene definition to determine which surfaces are visible.
- An image space method visibility is decided point by point at each pixel position on the projection plane.
- Most visible surface algorithms use image-space method although object space method can be used effectively to locate visible surfaces in some cases.
- Line display algorithm generally use object space method to identify visible lines in wire-frame display.

Date: 2065/8/17

Back Face detection:

- A fast and simple object space method for identifying the back-faces of a poly hedron is based on the "inside-outside" tests.

- A point (x,y,z) is "inside" a polygon surface with plane parameters A, B, C and D if,
- $Ax + By + Cz + D < 0$
- When an inside point is along the line of sight to the surface , the polygon must be a back face (we are inside that face cannot see the front of it form our viewing position).
- We can simplify this tests by considering the normal vector N to a polygon surface, which has certain components (A,B,C).
- In general if V is a vector in the viewing direction from the eye (or "camera") position, as shown in fig. then the polygon is back face if,
- $V.N > 0$



- Furthermore , if object description have been converted to projection co-ordinate and our viewing direction is parallel to the viewing z_v axis then,
- $V = (0,0,v_z)$ and $V.N = v_z C$

So that we only need to consider the sign of C (i.e z-component of the normal vector N)

- In a right handed viewing system with viewing direction along the negative $z_v$axis , the polygon is back face if $C < 0$ and also if $C = 0$.
- Thus, in general ,we can label any polygon as back face if its normal vector has z-component value.

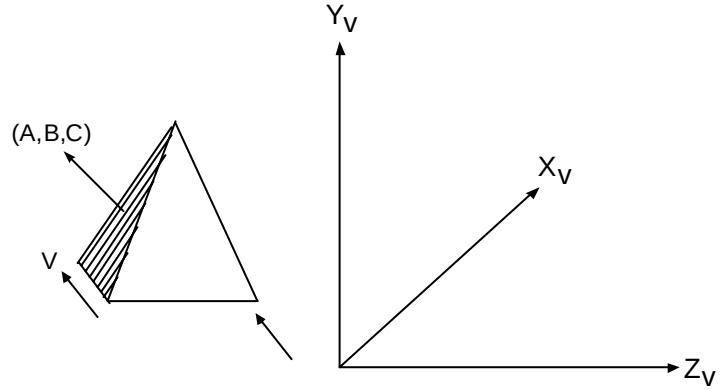
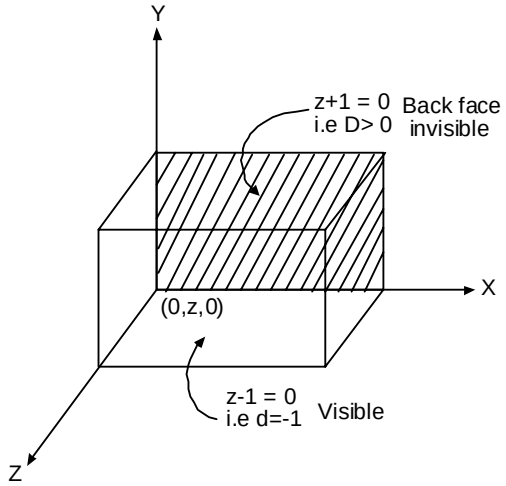


Fig. A polygon surface with plane parameter $C<0$ in a right-handed co-ordinate system is identified as a back-face when the viewing direction is along the –ve v_z axis.

Object is centered at origin if $D > 0$ (back face) else front face.



Depth Buffer Method (z-Buffer Method):

- A commonly used image-space approach for detecting visible surface is the depth buffer method which compares surface depth at each pixel position on the projection plane.

- This process is also referred to as the z-buffer method, since object depth is usually measured from plane along the z-axis of a viewing system.
- This method is usually applied to scenes containing only polygon surface because depth values can be computed very quickly and the method is easy to implement.
- With object description converted to projection co-ordinate each (x,y) position on a polygon surface corresponds to the orthographic projection point (x,y) on the view plane.
- Therefore for each pixel position (x,y) on the view plane object depth can be compared by comparing z-values.

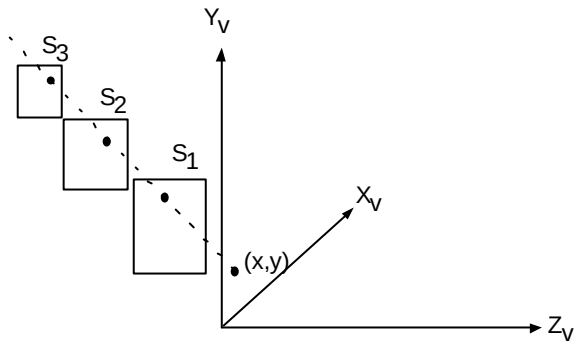


Fig. At view plane position (x,y) surface s_1 has the smallest depth form the view plane and so its surface intensity value (x,y) is saved.

- -We can implement the depth buffer algorithm in normalize co-ordinates so that the z-values range form zero at the back clipping plane to z_{max} at the front clipping plane.
- The value of z_{max} can be set to either one (for a unit cube) or to the largest value that can be stored in the system.

In this method two buffer areas are required. One is the Depth buffer: To sore the depth value for each (x,y) position as surface S are processed.

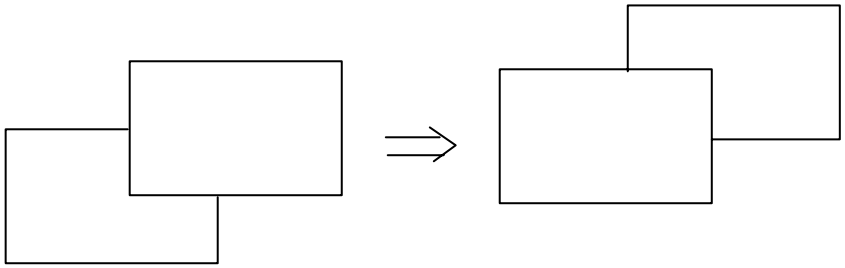
A Refresh buffer: To store the intensity value for each position.

- Initially , all positions in the depth buffer are set to 0 (minimum depth) and the refresh buffer is initialized to the back-ground intensity.

- Each surface listed in the plygo tables is then processed. One scan line at a times calculaign the depth (z-value) at each (x,y) pixel position.
- The calculated depth is compared to the value previously stored in the depth buffer at that position.
- The calculated depth is greater than the value stored in the depth buffer, the new depth value is stored and the surface intensity at that position is determined and place in the same x-y location in the refresh buffer.

Depth buffer Algorithm:

1. Initialize the depth buffer and refresh buffer as depth (x,y) $=z_{min}$ or 0
2. For each pixel on each polygon surface compare depth value (z-value) to previously stored value in depth buffer (x,y).
   If z > depth (x,y) then set
   Depth(x,y) = z , refresh(x,y) = I_{suface}(x,y)
3. Plot the point(x,y) and z-value at each depth buffer with corresponding values in refresh buffer.



After all surfaces have been processed the depth buffer contins depth values for visible surfaces and refresh buffer contains corresponding intensity values for those surface.

Where,

$I_{surface}(x,y)$ is the projected intensity value for the surface at pixel position (x,y).

The mathematics involved (Calculation of z-value):

Depth values for a surface position (x,y) are calculated from the plane equation for each surface.

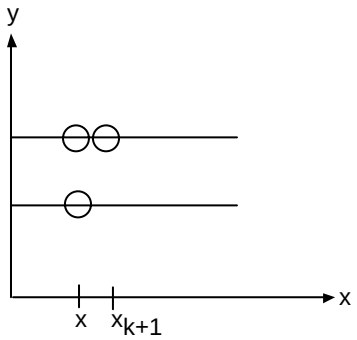
$Z = (-Ax – By – D)/C$

From position (x,y) on a scene line the next position across the line the next position across the line has co-ordinates(x+1,y) and the position immediately below on the next line co-ordinates (x,y-1).

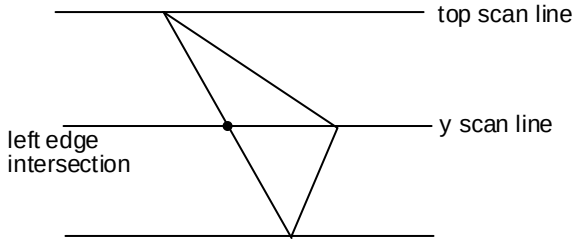
Now we determine the depth of the next position (x+1,y) as Depth (x+1,y).

$Z' = [-A(x-1)-By-D]/C$

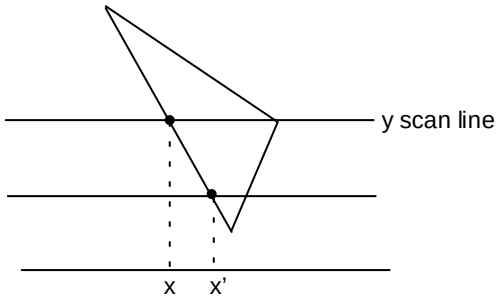
Or, $z' = z – A/C$



- The ratio –A/C is constant for each surface, so succeeding depth values across a scan line are obtained from processing values with a single addition.
- On each scan line, we start by calculating the depth on a left edge of the polygon that intersects the scan lines as shown in the fig. below.



- Depth values at each successive position across the scan line are then calculated from the above equation for z'.
- We first determine the y-coordinate extents of each polygon and process the surface form the top-most scan line as shown in the above figure.
- Starting at a top vertex, we recursively calculate n positions down a left edge of the polygon as:
- $x' = x – 1/m$
- where, m is the slope of the edge of the polygon.



Depth values down the edge are then obtained recursively as,

$Z = [-Ax-By-D]/C$

$y' = y - 1$ and $x' = x - 1/m$

$z' = z + (A/m+B)/c$

If we are processing down a vertical edge, the slope is infinite and the recursive calculations reduces to

$Z' = z + B/C$.

What are the advantages and disadvantages of z-buffer algorithm?

Advantages:

- No presorting and object-object comparisons are required .
- Time taken by visible surface calculation is constant .
- It is simple and easy implementation hardware.
- If memory is at permission scan converted in strips.
- Good for animation.

Disadvantage:

- Requires amount of memory for z and frame buffer.
- It is subject to aliasing.

Note from miss:

iii) Scan line Method:

- The scan line method solve the hidden surface problem one scan line at a time.
- Processing of the scan line start from the top to the bottom of the display.
- This method calculates, the z-value for only the overlapping surface which is tested by scan line.
- It request edge table, polygon table/surface table active edge list and flag.

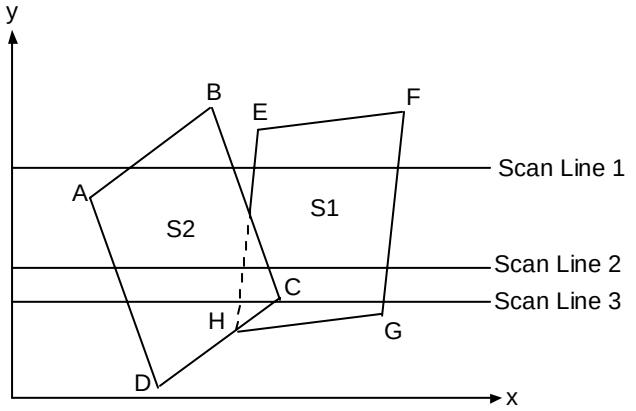


Fig. Scan lines crossing the projection of tow surfaces, s_1 and s_2 in the view plane. Dashed line indicate the boundaries of hidden surfaces.

Edge table contents:

i) Coordinate end points for each scan line.
ii) Pointers in the polygon table to identify the surfaces bounded by each line.
iii) Inverse slope of each line.

Polygon table contents:

i) Coefficients of plane equations for each surfaces.
ii) Pointers into edge table.
iii) Intensity information of the surfaces.

Active edge list contains edges that cross the current scan line, shorted in order of increasing x.

Flag is defined for each surface that is set 'ON' or 'OFF' to indicate if a position alone a scan line is inside or outside of the surface at left most boundary surface flag is 'ON' and at right most flag is 'OFF'.

Above figure illustrate the scan line method for locating visible portion of surfaces for pixel position along the line.

The active list for scan line one contains information form the edge table for edges AB , BC, EH and FG.

For position along this scan line between edges AB and BC, flag (S_1) = ON and flag (S_2) = OFF.

Therefore no depth calculation are necessary and intensity information for surface S_1 is entered from the polygon table into the refresh buffer.

Similarly , between edge EH and FG

Flag (S_1) = OFF and flag (S_2) = ON

On other position along scan line 1intersect surfaces, so the intensity values in the other areas are set to the background intensity. The background intensity can be loaded throughout the buffer in an initialization routine.

For scan line 2 and 3 figure, the active edge list contains edges AD, EH, BC and FG.

Along the scan line 2, from edge AD to edge EH, flag (S_1)= ON and flag (S_2)= OFF

Between edges EH and BC,

Flag (S_1)= ON and flag (S_2) = ON

Therefore depth calculation must be made using the plane coefficient for the two surfaces.

For this example the depth of surface S_1 is assumed to be less than that of S_2, So intensity for surface S_1 are loaded into the refresh buffer until boundary BC is encountered. Then the flag for the surface S_1 goes off. And intensity for surface S_2 are stored until edge FG is passed.

For the scan line 3 the active edge list contains the same edge as scan line 2.

Since no changes have occurred in line intersection, it is unnecessary again to make depth calculation between edges EH and BC. The surfaces must be in the same orientation as determined on scan line 2. So the intensity for the surface S_1 can be entered without further calculation.

Any number of overlapping polygon surfaces can be processed with this scan line method. Flag for the surfaces are set to indicate whether a position is inside or outside, and depth calculation are performed when surfaces overlap.

Algorithm:

1. Established data structure.
   I. Edge table with line endpoints , inverse slope to polygon pointers.
   II. Polygon table with plane coefficient, colour and pointer to edge table.
   III. Active edge list sorted in increasing order of x.
   IV. A flag for each surface.

2. Repeat for each scan line.
   I. Update AEL for scan line y,

II. Scan line across light using background color until flag goes ON.
III. When one flag is ON , enter intensity of that surface to refresh buffer.
IV. When two or more flags are ON, do depth sorting and stored the intensity of pixel nearest to the view plane.

Need for Shading in Engineering Data visualization: The realism of a raster scan image of a 3D scene depends upon the successful stimulation of shading effects.

Once visible surface has been identified by hidden surface algorithm , a shading model is used to compute the intensities and color to display for the surface.

The Shading model doesnot precisely stimulate the behaviour of light and surfaces in the real world but only approximation actual condition.

The shading model has two components.

1. Properties of surfaces
2. Properties of illumination falling on it.

The principle surface property is its reflectance which determine how much of the incident light is reflected.

- If a surface has different reactance for light of different wavelength, it will appear to be color.
- If a surface is texture or has a patterned printed on it, the reflectance will vary with position on the surface.
- Another surface property that play a role in shaded picture is transparency.

Light Source:

When we view a opaque non-luminous object, we see reflected light from the surface of the object. The total reflected light is the sum of the contribution from the light source and other reflective surfaces as shown in the figure:

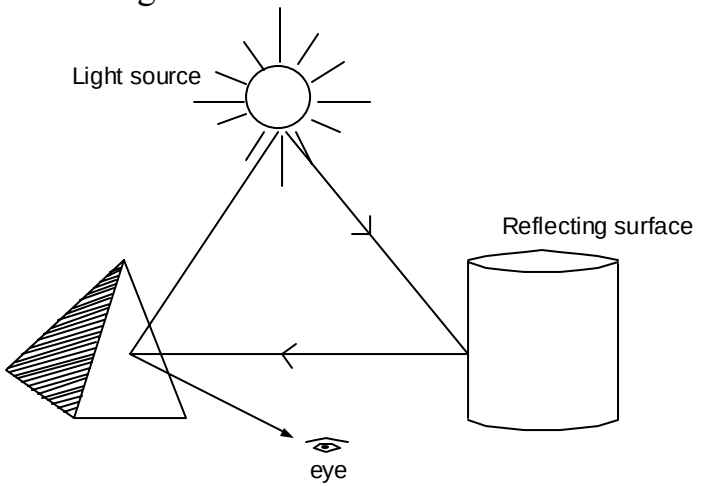


Fig. Light viewed from an opaque non luminous surface is in general a combination of reflected light from a light source and reflection of light reflection from other surfaces.

- Thus a surface that is not directly exposed to light sources may still be visible if near by objects are eliminated.
- Sometimes light sources are referred to as light emitting source and reflecting surface such as walls of a room are termed light reflecting source.
- A luminous object in general can be both light source and a light reflector. For example: A plastic globe with a light bulb inside both emits and reflects light from the surface of globe.
- A simplest model for a light emitter is a point source.
- A near by source such as long fluorescent light is more accurately modelled as a distributed light source.

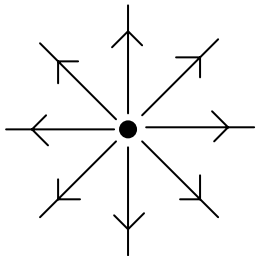


Fig. Diverging ray paths From a point light Source.

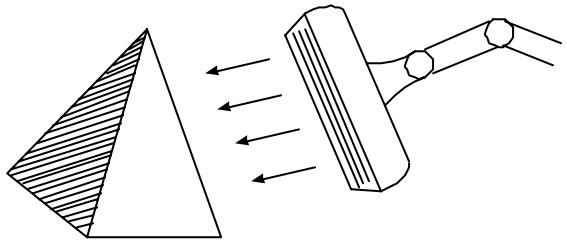
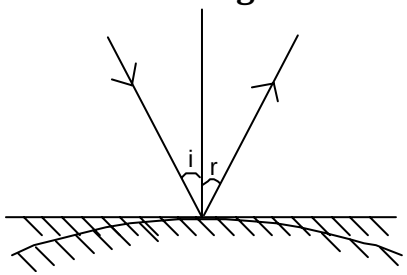
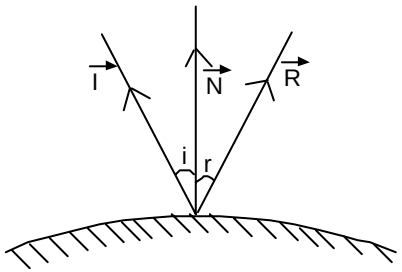


Fig. An object illuminated with a distributed light source.

- When light source is incident on a opaque surface, part of it is reflected and part of it is absorbed.
- The amount of incident light reflected by a surface depends on the type of material.
- Shining material reflects more incident light and dull surface absorbs more of the incident light.
- Surface to produce a uniform illumination is called the ambient light or background.

Reflection of light:





In ideal reflection i = r

Let $\vec{L}$ be the unit vector along incident ray. $\vec{N}$ be the unit vector along normal ray. $\vec{R}$ be the unit vector along reflected ray then,

i = r

or cos i = cos r

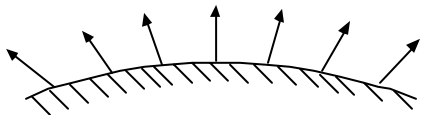
or $\vec{L} . \vec{N} = \vec{N} . \vec{R}$

But practically when light is incident on opaque surface part of it is reflected and part of it is absorbed.

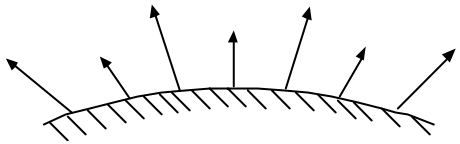
$\therefore$ I = A+R

Shining material reflects more and dull surface absorb more of incident light. Rough surface tends to scatter the reflected light in all direction. The scattered light is called diffuse reflection.

So surface appears equally bright from all viewing directions.

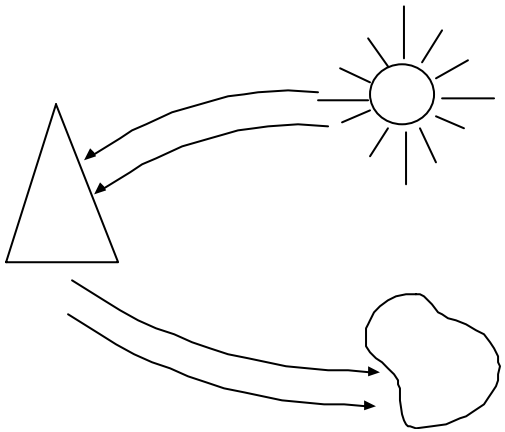


In addition to diffuse refection, light source creates high lights or bright spots called <u>specular reflection.</u> How ever this effect is seen more on shiny surface then dull surfaces. Examples: Persons forehead.



Basic illumination models:

Ambient light: Surface that is not exposed directly to a light source still will be visible if near by objects are illuminated. This light is called ambient light.



Characteristic of Ambient light:

- It provide general level of brightness for all surfaces.
- It has no spatial or directional characteristics.
- It has same intensity for each surface regardless of viewing direction.

If intensity of ambient light = I_a

Then $I_{amb} = I_a$

If we take surface reflectivity then,

$I_{amb} = I_a$

If we take surface reflectivity then

$I_{amb} = k_a I_a$

Where k_a is co-efficient of ambient reflection.

Diffuse Reflection: Diffuse reflection is constant over each surface in a scene independent of viewing direction. The intensity of diffuse reflection due to ambient light is

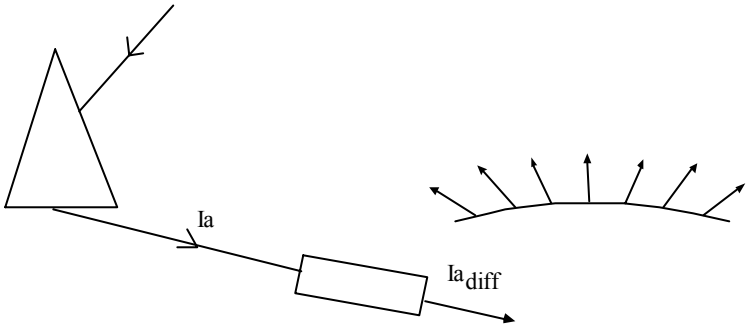
$I_{adiff} = k_d . I_a$

Where,

K_d = diffuse reflection co-efficient

For highly reflective surface $k_d = 1$

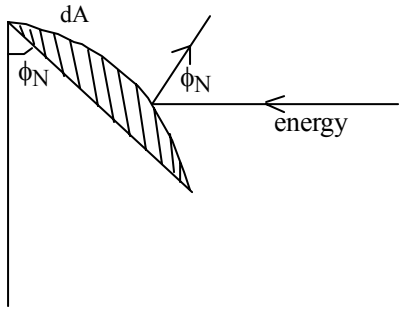
For low reflective surface $k_d = 0$



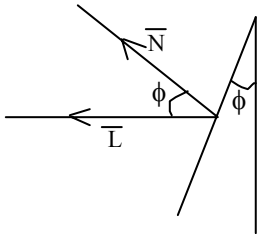
If surface is exposed to a point source we assume diffuse reflections from this surface are scattered with equal intensity in all directions. Such surfaces are also called ideal diffuse reflectors which follows Lambert's Cosine Law.

Lambert's Cosine Law: The radiant energy force from any small surface dA in and direction ϕ_N relative to surface normal is proportional to $\cos\phi_N$.

$\therefore$ Light intensity α $\cos\phi_N$



Although there is equal light scattering in all directions from an ideal diffuse reflection, the brightness of the surface depends on the orientation of the surface related to the light source. If the surface is perpendicular to source it appears brighter.



$\therefore I_{diff} = k_d I_l \cos\phi$

$= k_d I_l (\vec{L}.\vec{N})$

$\therefore$ Total diffuse reflection = Diff (due to amb.)+Diff. due to pt.source.

$= k_a I_a + k_d I_L (\vec{L}.\vec{N})$

Specular Reflection:

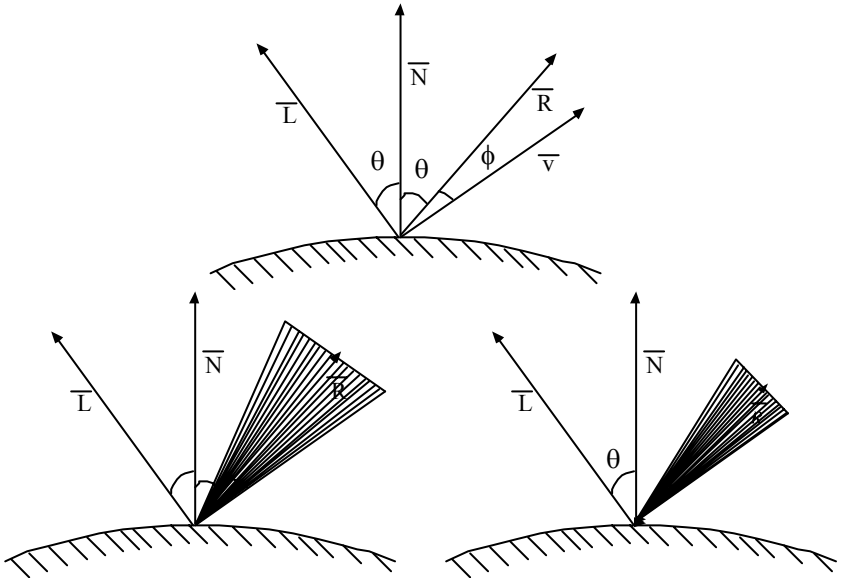
In shiny surface we see high light or bright spot from certain viewing directions called specular reflection. This is due to total or nearly total reflection of light.

For ideal reflector,

(perfect mirror) = $\phi = 0$

i.e incident light is reflected only in specular reflection direction. So $\vec{V}$ and $\vec{R}$ coincides.

Objects other than ideal reflection for a finite range of viewing positions around $\vec{R}$. Shiny surfaces have narrow specular reflection range and dull surfaces have a wider range.



Dull surface (small n_s)

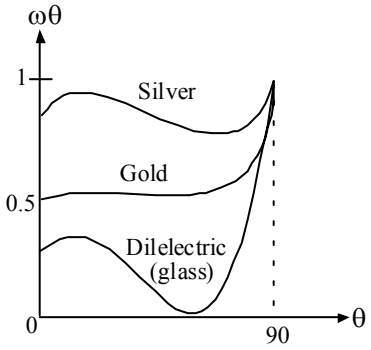
Shiny surface (large n_s)

The empirical formula for calculating the specular reflection range is given by phong Model.

Phong Model: It sets the intensity of the specular reflection proportional to $\cos^{\eta s}\phi$, where η_s is specular reflection parameter and is determined by the type of surface that we want to display.

For very shiny surface η_s is set 100 and for dull surface η_s is set 1. The intensity of specular reflection can be modeled using specular reflection coefficient $\omega(\theta)$

$\therefore I_{spec} = \omega(\theta) I_L \cos^{\eta s}\phi$

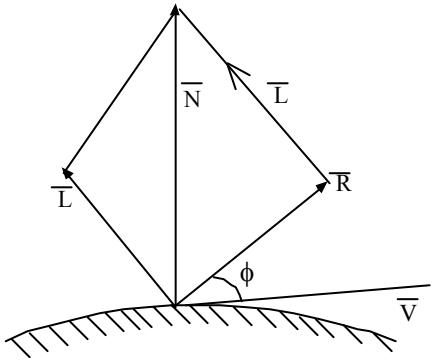


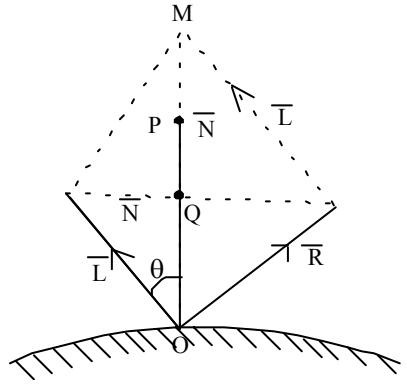
The glass exhibits appreciable specular reflection when.
$\theta \to 90°$
At $\theta = 0°$ only 4% of incident light is specularly reflected.
Similarly for gold at $\theta = 0°$, almost 50% of light is specularly reflected and at $\theta = 90°$ almost 100% incident light is specularly reflected.

However ,for many opaque materials specular reflection is nearly constant for all incident angles. So, we replace $\omega(\theta)$ by a constant k_s (specular reflection co-efficient).

$\therefore I_{spec} = k_s I_L \cos^{\eta s}\phi$

$= k_s I_L (\vec{V}\ \vec{R})^{\eta s}$





Expanding R in terms of $\vec{L}$ and $\vec{N}$

From fig. aside,

$\vec{R} + \vec{L} = \overrightarrow{OM}$

$= 2$ OQ (diagonal of ||gm bisect each other)

$= 2|\overrightarrow{OQ}|.\vec{N}$

$= 2$ ox $\cos\theta.\vec{N}$

$= 2(\vec{L}.\vec{N}).\vec{N}$

$\therefore \vec{R} = 2(\vec{L}.\vec{N}).\vec{N} - \vec{L}$

$\therefore$ Total intensity is,

$I = I_{amb} + I_{diff} + I_{spec}$

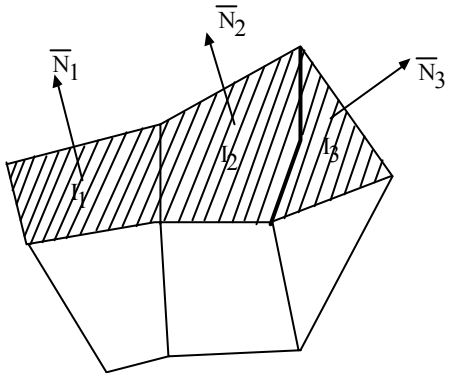
$= K_a I_a + k_d I_L (\vec{N}.\vec{L}) + K_s I_L (\vec{V}.\vec{R})^{\eta s}$

Where, $\vec{R} = 2(\vec{N}.\vec{L}).\vec{N} - \vec{L}$

Polygon Rendering Methods:

(i) Constant Intensity shading Method.
(ii) Gouraud Shading method(Intensity Interpolation)
(iii) Phong Shading Method (Normal Vector Interpolation).

(i) Constant Intensity Shading Method:



The intensity is calculated for a point for each surface (usually at the center) and that intensity is applied to all the points of that surface. Hence, all the points over the surface of the polygon are displayed with same intensity value given by

$I = k_a I_a + k_a I_a (\vec{N} . \vec{L}) + k_s I_L (\vec{V} \vec{R})^{\eta s}$

Where, $\vec{R} = 2(\vec{N} . \vec{L}) . \vec{N} - \vec{L}$

This method is faster and simple but not realistic and holds following assumptions:

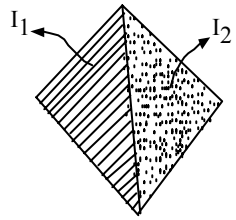
(a) Object is polyhedron and is not an approximation of an object with a curved surface.

(b) All light sources are sufficiently far from the object so that $\vec{N} . \vec{L}$ is constant.

(c) The viewing position is sufficiently far so that $\vec{V} . \vec{R}$ is constant.

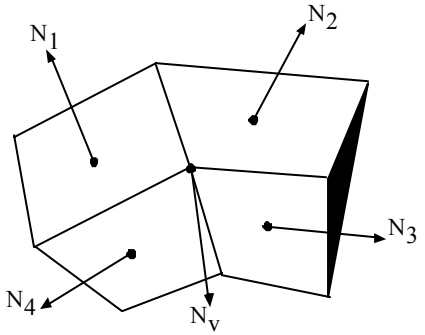
Disadvantage:

- The intensity discontinuity occurs at the border of the two surfaces.



(ii) Gouraud Shading:

It renders the polygon surface by linearly interpolating intensity values across the surface.



Steps:

(i) Determine the average unit normal vector at each polygon vertex by

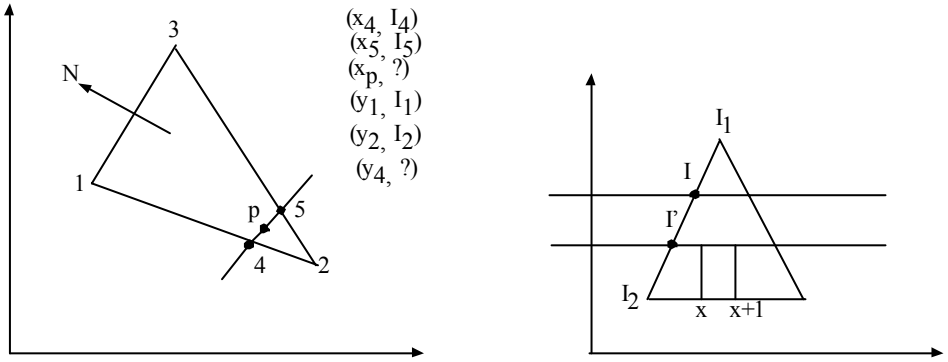
$$N_{avg} = \frac{\sum N_i}{\left|\sum N_i\right|} = \frac{\vec{N}_1 + \vec{N}_2 + \vec{N}_3 + \vec{N}_4}{\left|\vec{N}_1 + \vec{N}_2 + \vec{N}_3 + \vec{N}_4\right|}$$

(ii) Calculate the intensity at each vertex by

$I = k_a I_a + k_a I_a (\vec{N} . \vec{L}) + k_s I_l (\vec{V} . \vec{R})^{ns}$

(iii) Linearly interpolate the vertex intensities over the surface.

$$I_4 = \frac{y_4 - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y_4}{y_1 - y_2} I_2$$



Similary,

$$I_P = \frac{x_5 - x_p}{x_5 - x_4} I_4 + \frac{x_p - x_4}{x_5 - x_4} I_5$$

In fig. aside, I can be calculated by interpolation between I_1 and I_2.

$$\therefore \quad I = \frac{y - y_2}{y_1 - y_2} I_1 + \frac{y_1 - y}{y_1 - y_2} I_2$$

$$\therefore \quad I' = \frac{(y-1) - y_2}{y_1 - y_2} I_1 + \frac{y_1 - (y-1)}{y_1 - y_2} I_2$$

$$\therefore \quad I' = I + \frac{I_2 - I_1}{y_1 - y_2}$$

Similarly , intensity at x sampling position is

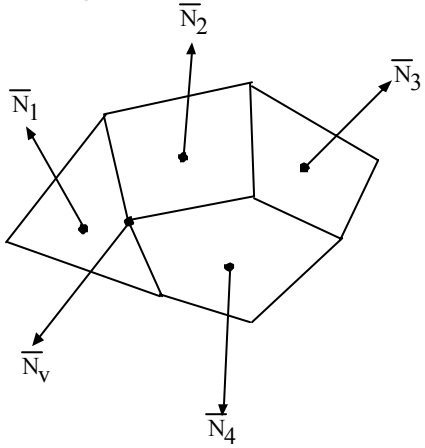
$$I = \frac{x_5 - x}{x_5 - x_4} I_4 + \frac{x - x_4}{x_5 - x_4} I_5$$

At next sampling position (x+1)

$$\frac{x_5 - (x+1)}{x_5 - x_4} I_4 + \frac{(x+1) - x_4}{x_5 - x_4} I_5$$

$$\therefore \quad I' = I + \frac{I_5 - I_4}{x_5 - x_4}$$

(iii) Phong Shading: It renders the polygon surface by linearly interpolating normal vector across the surface.



Steps:

a) Determine the average unit normal vector at each polygon vertex.

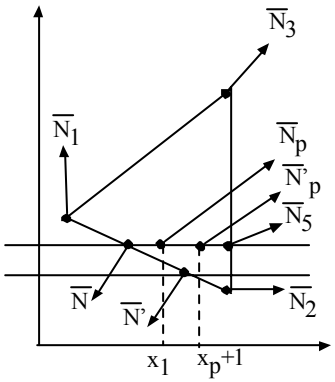
$$\overline{N}_{avg} = \frac{\sum \overline{N}_i}{\left|\sum \overline{N}_i\right|}$$

b) Linearly interpolate the vertex normal over the surface of the polygon.

c) Apply an illumination model along each scan line.

$$I = k_a I_a + k_a I_a (\vec{N} . \vec{L}) + k_s I_L (\vec{V}\ \vec{R})^{ns}$$

Calculation for step (b).



$\overline{N}$ can be obtained by interpolating $\overline{N}_1$ and $\overline{N}_2$

$$\overline{N} = \frac{y - y_2}{y_1 - y_2}.\overline{N}_1 + \frac{y_1 - y}{y_1 - y_2}.\overline{N}_2$$

Similarly for next scan line.

$$\overline{N}' = \frac{(y-1) - y_2}{y_1 - y_2}.\overline{N}_1 + \frac{y_1 - (y-1)}{y_1 - y_2}.\overline{N}_2$$

$$\overline{N}' = \overline{N} + \frac{\overline{N}_2 - \overline{N}_1}{y_1 - y_2}$$

Similarly, for horizontal scan, $\overline{N}_1$ can be obtained from interpolation between $\overline{N}$ and $\overline{N}_5$ given by

$$\overline{N}_p = \frac{x_5 - x_p}{x_5 - x}.\overline{N} + \frac{x_P - x}{x_5 - x}\overline{N}_5$$

Similarly normal vector at next sampling position x_{p+1} is

$$\overline{N}_P' = \overline{N}_p + \frac{\overline{N}_5 - \overline{N}}{x_5 - x}$$

Comments:

(i) Requires more calculation and hence time consuming.

(ii) Produces more realistic.

Chapter

NEED FOR MACHINE INDEPENDENT GRAPHICAL LANGUAGE

GRAPHICS SOFTWARE

There are two general classifications for graphics:

1. General Programming Packages:

- A general programming package provides extensive set of graphics functions that can be used I n high level programming language such as C or FORTRAN.
- An example of a general graphics programming package is the GL (Graphics Library) system on Silicon Graphics equipment.
- Basic functions in a general package include those for generating picture components (St. Line, polygon, circles and other figures) setting color and intensity values, selecting views and BC applying transformations.

2. Special Purpose Application Packages:

- Application graphics packages are designed for non-programmers, so that users can generate displays without worrying about how graphics operations work.
- The interface to the graphics routines in such packages allows users to communicate with the programs in their own terms. Example: CAD

SOFTWARE STANDARDS

- The primary goal of standardized graphics software is portability.
- When packages are designed with standard graphics functions, software's can be moved easily from one hardware system to another.
- Without standards, programs designed for one hardware often can't be transformed to another system without extensive rewriting of the programs. Thus we require machine independent graphical language.
- International and national standards planning organizations in many countries have co-operated in an effort to develop generally accepted standard for computer graphics.
- Graphical Kernel System (GKS) was adopted as the first graphics software standard by the International standard Organization (ISO) and by various national standards organizations, including the American National Standards Institute (ANSI).
- Although GKS was originally designed as a 2D graphics, a 3D GKS extension was subsequently developed.
- The second software standard to be developed and approved by the standards organization was PHIGS (Programmer's Hierarchical Interactive Graphics Standard) which is an extension of GKS.
- Increased capabilities for object modeling, color specifications and picture manipulations are provided in PHIGS.
- Standard Graphics functions are defined as a set of specifications that is independent of any programming language. A language binding is then defined as a particular high level programming language. This language gives the syntax for accessing the various standard graphics functions from this language.

Discussion of available languages and file formats:

File Format: A file format is a standard method of encoding data for storage. There are many types of file formats, they are important because they tell programs what kind of data is contained in the file and how data is organized.

File formats are of two types:

(2) Proprietary
(3) Universal

Proprietary and universal

The structure of proprietary is under the sole control of the software developer who invented the format. This file can be opened or saved without the use of an import or export filter by the program that created them.

Universal file formats are based on openly published specification and are commonly used by many different program and on different operating system.

For example: Adobe photoshop by default saves images in its proprietary format, but it can also save file in several universal formats such as TIFF , GIF , JPEG, PICT and TGA. A word processing program can read and save files in specific formats such as DOC and TXT.

All the bitmap based to graphics programs can use any of the file formats listed in the fig. below:

Format	Description
1. BMP	A graphics format which is used in windows BMP is used for wall paper.
2. PICT	(picture). It is generally used on MACS computers.
3. TIFF	(Tagged Image file format).A bit map format defined in 1986. by Microsoft and Aldus and widely used in both Macs and PCs. This format is usually the best to use when exchanging bit map files that will be printed or edited further.
4.JPEG	(Joint photographic expert group).This bit map format is Common on the world wide web and is often used for photos and high resolution images.(24-bit or millions of colors)that will be viewed on the screen.
5. GIF	(Graphic Interchange Format).Like JPEG images GIF images are often found on world wide web pages. Unlike JPEG images, GIF images are reduced to 256 or fewer unique colors.
6. PNG	(portable, network graphics). This format was developed as an alternative to GIF and JPEG.PNG, like JPEG, can store the color images in a small amount of space, but PNG file can also store transparency information the way GIF files do. The PNG format was initially designed for use in web pages.

Most vector base programs create and save file in a proprietary file format. These formats are either incompatible with other programs not totally supported by other programs. In the later case, other programs may use import filters to use these file formats. Import filters never work perfectly because program using the import filter will not have the same exact features as the program that created the file. This lack of commonality has forced the software developer to create universal files formats, which enables user of one program to work with files created-in other programs. Only a handful of common file formats such as date exchange formats (D×F) and initial graphics exchange specifications (IGES), exists for vector graphics. This universal format should enable you to create a vector file in one program , such as AUTOCAD and use it in another program such as corel DRAW or VISIO.

Graphics software:

1. **Paint program:** Paint program works with bit map images.
2. **Photo manipulation program:** It works with bit map images and are widely used to edit digitized photographs.

3. **Computer Aided Design program:** CAD software is used in technical design fields to create models of objects that will be built or manufactured. CAD software allows user to design objects in 3 dimensions and can produce 3-D frames and solid models.
4. **3-D Modelling Programs:** It is used to create visual effects. 3-D modeling program works by creating objects like surface , solid, polygon etc.
5. **Animation:** Computer are used to create animation for use in various fields, including games, and movies composing to allow game makers and film makers to add characters and objects to scenes that did not originally contain them.

Error Diagnostics: Programming errors often remain undetected until an attempt is made to compile the program once the compile command has been issued. However, the presence of certain errors will become readily apparent, since these errors will prevent the program from being compile successfully, some particular common errors of this types are declaring constants and variables improperly , a reference to an undeclared variable and incorrect punctuation. Such errors are referred as syntactical error or grammatical error. Most version of C program will generate a diagnostic message when a syntactical error has been detected (the compiler usually come to an interrupt when this happens). This diagnostic messages are not always completely straight forward in their meaning, but they are nevertheless helpful in identifying the nature and location of the error.

Logical Debugging: Syntactical errors and certain types of logical errors will cause diagnostic messages to be generated when compiling or executing a program. Errors of this types are easy to find and correct. Some types of logical errors can be much more difficult to detect, however, since the output resulting from a logically incorrect program may appear to be error free. Moreover, logical errors are often hard to find even when they are known to exits (as for example, when the computed output is obviously incorrect)

Chapter:

PROJECT MANAGEMENT

A project is an inter-related set of activities that has definite starting and ending point that result in a unique product or service. Each project is unique, even if it is routined. Uncertainties such as the advent of new technologies or the timing of certain events can change the character of projects. Projects are temporary activities as personal materials and

facilities are assembled to accomplish a goal within a specified time frame.

Projects are common in everyday life. Planning weddings, remodeling rooms, writing term papers and organizing surprise birthday parties are examples of small projects.

ELEMENTS OF PROJECT MANAGEMENT

Successful project management involves the coordination of tasks, people, organization and other resources to achieve a goal. Three major elements are the

1. Project Manager
2. Project Team
3. Project Management System

1. PROJECT MANAGER(PM)

- Project manager integrate people from various functional areas to achieve a specified project goals.
- Traditional organization hierarchy tends to slow progress on project because of lack of communication, coordination and some time motivation.
- A project manager can overcome these road blocks to project compilation.
- A project manager is responsible for establishing the project goals and providing the means to achieve them.
- The project manager must also specify how the work will be done and ensure that the appropriate hiring is done and any necessary training is conducted.
- Project manager must demonstrate leadership and provide the motivation necessary to accomplish the task required.
- Project manager also evaluates progress and tasks and appropriate action is taken when the project schedules are in jeopardy (trouble).

2. PROJECT TEAM

- The project team is a group of people often differential functional areas or organizations, led by the PM.
- Members of the project team may be the represent the supplier of essential materials, components or services and may be actively involved in the conduct of the project.
- The size and constituency of the team may fluctuate during the life of the project.

3. PROJECT MANAGEMENT SYSTEM

- The project management system consists of an organizational structures and information system.
- The organizational structure is specified by the top management and defines the relationship of the project team members to the project manager.
- Assistance from personnel in other functional areas must be negotiated by the project manager.

- Under this structure, the PM has minimal control over the timing of the project, but resource duplications across functional areas are minimized.
- Project management system also provides for the integrative planning and control of the performance, costs expressed in dollars or resource uses, the inter-related effects of schedule changes and the projected times and cost of project completion.
- Network planning methods provide the information needed to manager projects.

NETWORK PLANNING METHOD

Network planning method can help project manager to monitor and control project. These methods and treat a project at a set of inter-related activities that can be visually displaced in a network diagram which consists of nodes (circle) and arc (arrow) that shows the relationship between the activities.

Two network planning methods were developed in 1950's:

A. PERT (Program Evaluation and Review technique)
B. CPM (Critical Path Method)

Network planning method manages the projects which involves different four steps:

1. Describing the project.
2. Diagramming the network
3. Establishing time of completion
4. Monitoring project progress

1. Describing The Project:

- The project manager must first describe the project into that every one involve will understand.
- With the input of the team project manager must carefully define all project activities and precedence.
- An activity is the smallest unit of work effort consuming both time and resources that the project manager can schedule and control.
- If the precedence relationships determine a sequence of undertaking activities, it specifies that one activity can not start until the preceding activity has been completed.

2. Diagramming the Network:

- Diagramming the project as a network requires establishing the precedence relationship between the activities.
- The two different approaches are required for diagramming the project as a network.
    i. Activity on Arc(AOA)
    ii. Activity on Node(AON)

(i) Activity on Arc (AOA)

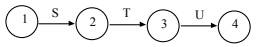
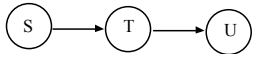
- It uses arcs to represent activity and nodes to represent events.

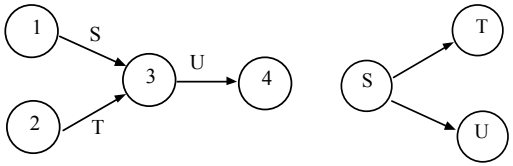
- An event is a point where one or more activities are to be completed and one or more other activities are to be begun.
- An event consumes neither time nor resources because AOA approach emphasizes activity connection points i.e. it is event oriented.
- Here the precedence relationship is required, an event cannot occur until all the preceding activities have been completed.
- Here connection used in AOA network is to number events sequentially from left to right.

(ii) Activity on Arc (AON)

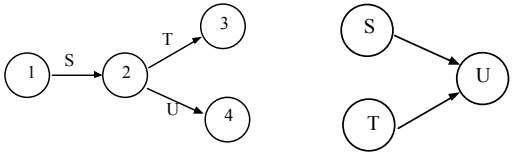
- Here node represents activity and arc indicates the relationship between them.
- This approach is activity oriented.
- Here the precedence relationship requires that an activity not begin until all preceding activities have been completed.

Fig. given below shows the AOA and AON approaches for several activities relationships commonly encountered;

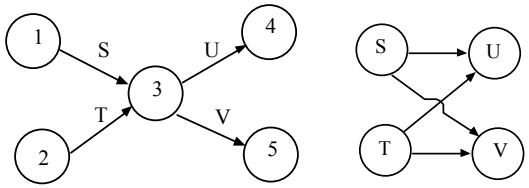
AOA	AON	Activity Relationship
		S precedes T which precedes U.



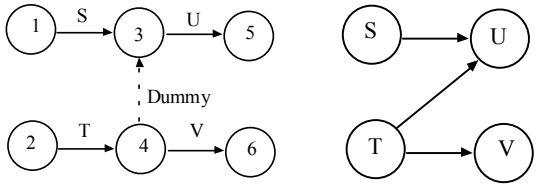
S and T must be completed before U can be completed.



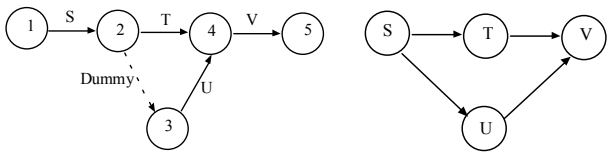
U and V can't begin until both S and T have been completed.



U and V can't begin until both S and T have been completed.



U can't begin until both S and T have been completed V can't begin until T has been completed.



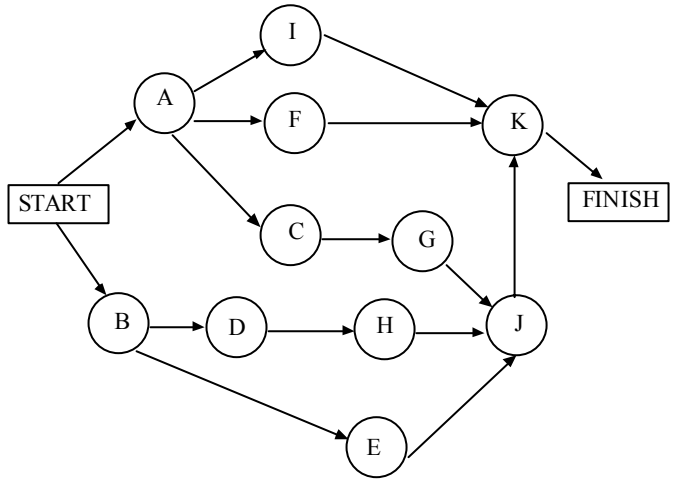
T and U can't begin until S has been completed. V can't begin until both T and U have been completed.

In a Hospital project ,11 major activities are identified. It also specified the immediate predecessors (those activities that must be completed before a particular activity can begin),for each activity, as shown in the following table.

Activity	Description	Immediate Predecessors
A	Select administrative and medical staff	-
B	Select site and do site survey	-
C	Select equipment	A
D	Prepare final construction plants and layout.	B
E	Bringing utilities to the site	B
F	Interview applicants and and fill positions in nursing, support staff, maintenance and security.	A
G	Purchase and take delivery Of equipment	C
H	Construct the hospital	D
I	Develop an information system.	A
J	Install the equipment	E,G,H
K	Train nurses and support staffs.	F,I,J

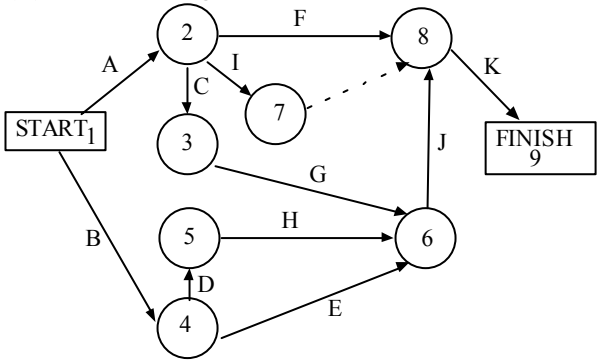
(a) Draw the AON network diagram.
(b) Draw the AOA network diagram.

(a) AON Diagram.



AOA Network for a given Hospital Project.

(b) AOA Diagram .



AOA Network for a given Hospital Project.

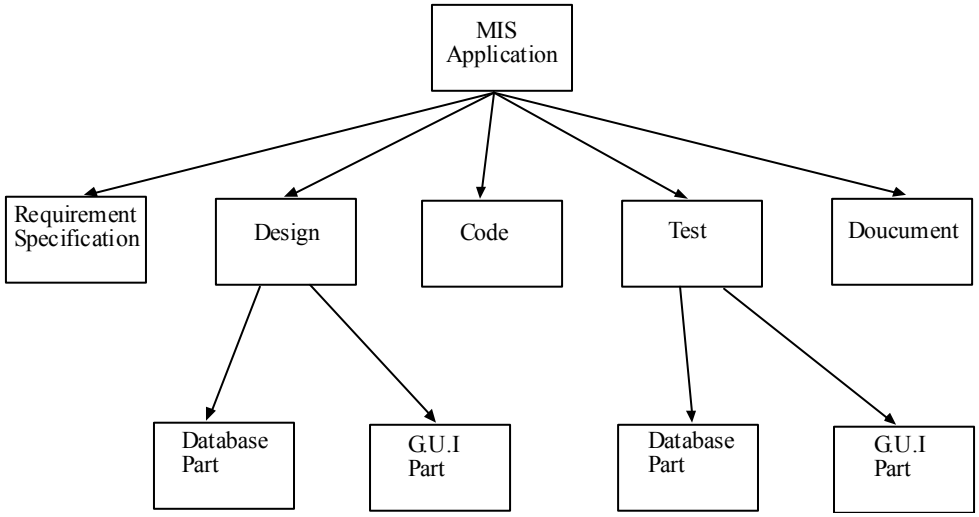
4.Monitoring Project progress:

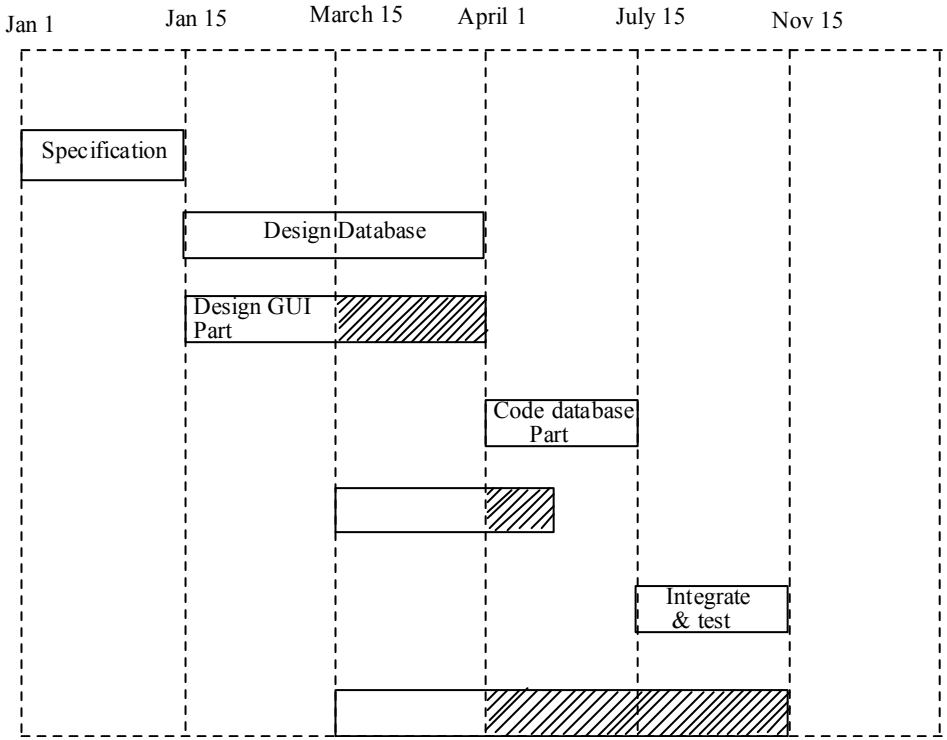
Gantt Chart:

Gantt Chart (named after its developer Henry Gantt) are useful for scheduling, budgeting and resource planning. A Gantt chart is a special type of bar chart where each bar represents an activity. The bars are drawn along a time line. The length of each

bar is proportional to the duration of time planned for the corresponding activity.

Gantt Charts are very useful in software project management. If the Gantt charts used for software project management; each bar consist of a white part and a shaded part. The white part of the bar shows the length of time each task is estimated to take. The shaded part shows the slack time, i.e , the latest time by which a task must be finished.





REVIEW OF PROGRAM DEBUGGING TECHNIQUES

LOGICAL DEBUGGING:

We have seen that syntactical errors and certain type of logical errors will cause diagnostic messages to be generated when compiling or executing a program. Errors of this type are easy to find and correct. Some types of logical errors can be much more difficult to detect, however, since the output

resulting from a logically incorrect program may appear to be error-free. Moreover, logical errors are often hard to find even when they are known to exist (as, for example, when the computed output is obviously incorrect). Thus a good bit of detective work may be required in order to find and correct errors of this type. Such detective work is known as *logical debugging.*

Detecting Errors:

The first step is attacking logical errors are to find out if they are present. This can sometimes be accomplished by testing a new program with data that will yield a known answer. If the correct results are not obtained, then the program obviously contains errors. Even if the correct results are obtained, however, one cannot be absolutely certain that the program is error-free, since some errors cause incorrect results only under certain circumstances (as, for example, with certain values of the input data or with certain program options). Therefore a new program should receive thorough testing before it is considered to be debugged. This is especially true of complicated programs or programs that will be used extensively by others.

As a rule, a calculation will have to be carried out by hand, with the aid of a calculator, in order to obtain a known answer. For some problems, however, the amount of work involved in carrying out a hand calculation is prohibitive. (Remember, a calculation that requires a few minutes of computer time may require several weeks to solve by hand!). Therefore a sample calculation cannot always be developed to test a new program. The logical debugging of such programs can be particularly difficult, though an observant programmer can often detect the presence of logical errors by studying the computed results carefully to see if they are reasonable.

Correcting Errors:

Once it has been established that a program contains a logical error, some resourcefulness and ingenuity may be required to find the error. Error detection should always begin with a thorough review of each logical group of statements within program. Armed with the knowledge that an error exists somewhere, the programmer can often spot the error by such careful study. If the error cannot found, it sometimes helps to get the program aside for a while. (This is especially true if the programmer is experiencing some fatigue or frustration). It is not unusual for an overly intent programmer to miss an obvious error the first time around.

If an error cannot be located simply by inspection, the program should be modified to print out certain intermediate results and the return. (This technique is sometimes referred to as *tracing*). The source of error will often become evident once these intermediate calculations have been carefully examined. In particular, the programmer can usually identify the particular area within the program where things begin to go wrong. The greater the amount of intermediate output, the more likely the chances of pinpointing the source of error.

Sometimes an error simply cannot be located, despite the most elaborate debugging techniques. On such occasions beginning programmers are often inclined to suspect a problem that is beyond their control, such as a hardware error or an error in the complier. In almost all cases, however, the problem turns out to be some subtle error in the program logic. Thus the beginning programmer should resist the temptation to simply blame the computer and not look further for that elusive programming error. (Hardware errors do occur on rare occasions, though they usually produce very bizarre results, such as the computer "dying" or the terminal spewing out random, unintelligible characters. Also, compiler errors occasionally crop with a new compiler but are usually corrected after a compiler has been in use for a while).

Finally, the reader should recognize the fact that some logical errors are inescapable in computer programming, though a conscientious programmer will make every attempt to minimize their occurrence. The programmer should therefore anticipate the need for some logical debugging when writing realistic, meaningful Pascal programs or any programs.