

Computer graphics.

Book: computer Graphics

Author: Donald Heam M Pauline baker.

*Date: 2065/4/12*

### **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
- Compiler
- 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)

*Date:2065/4/13*

**1.1 HISTROY 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.

Date: 2065/4/14

- Another major hardware advance of late sixties was attaching a display to 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 device 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 store 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 user 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 famous technology in the computer 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 almost 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 indispensable 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 DESIGN (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.

*Date: 2065/4/15*

### **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 employed 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 from 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 computer. 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 often 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 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 cursor 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 two types of mouse are there. They are:

- Mechanical mouse and
- Optical Mouse.

Mechanical mouse contains two rollers. One roller counts 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 magnitude 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 reflect it alternatively. The alternate reflections are used to calculate the relative position change.

**Advantages:**

- None of the part of computer screen is obscured (hidden from head).
- Movement becomes easier as hand 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.

*Date:2065/4/16*

**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 from each side proportionally. The sensor calculates the change 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 on 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.

*Date: 2065/4/19*

## 2.2 Raster and vector display architecture:

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

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

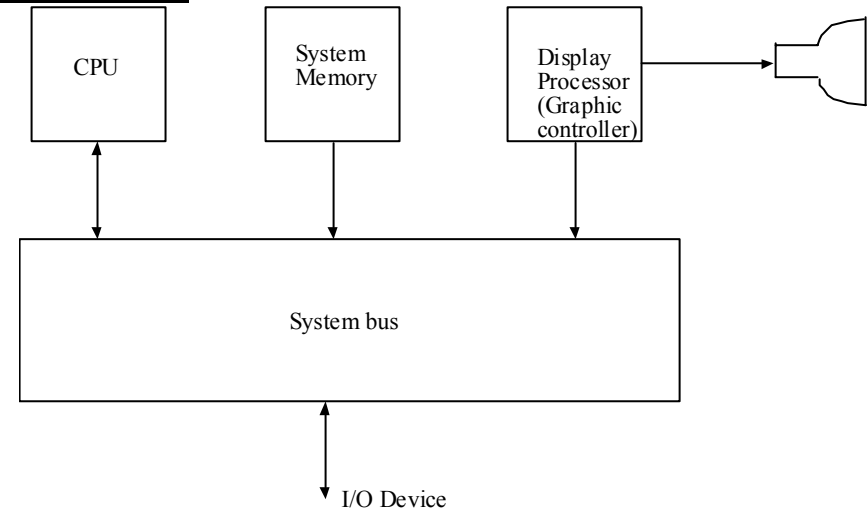
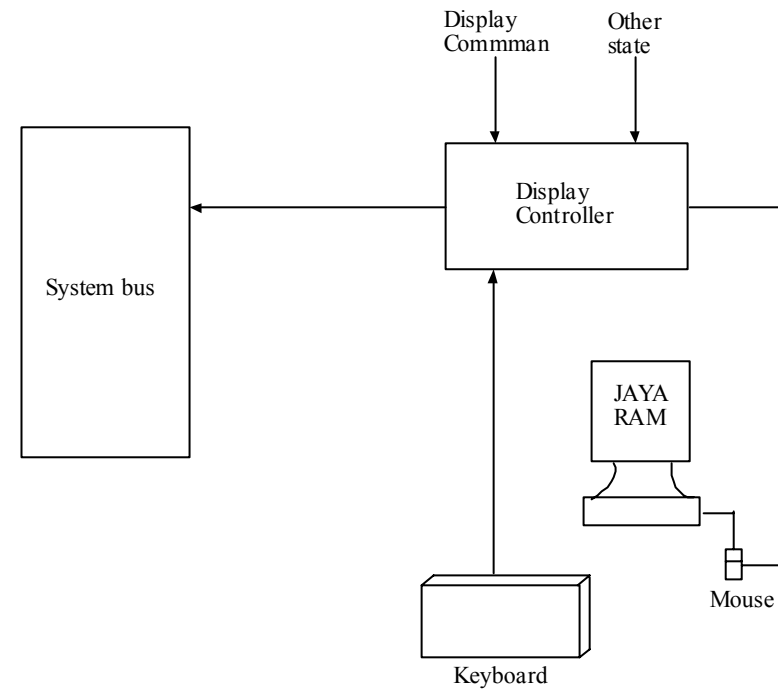


Fig. Architecture of a simple random scan system





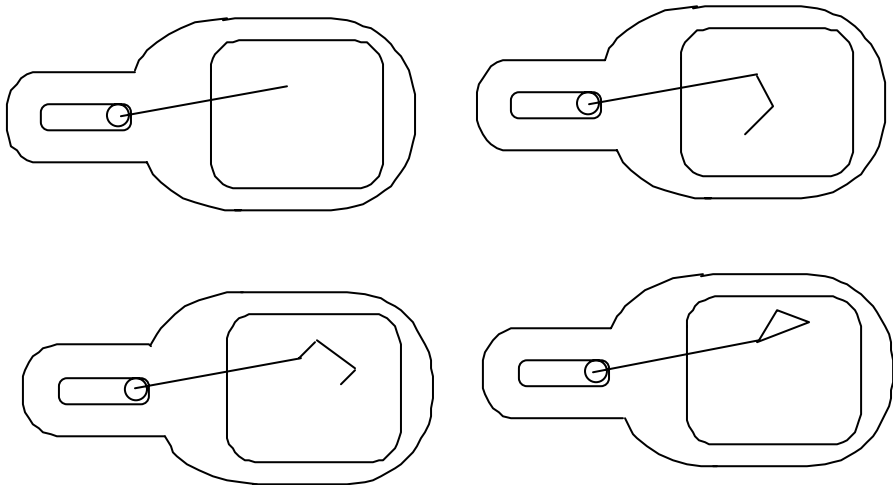


Fig. Random scan system drawing through the component lines of an object in any order specified.

- 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 from 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.

Date: 2065/4/20

## Raster Display:

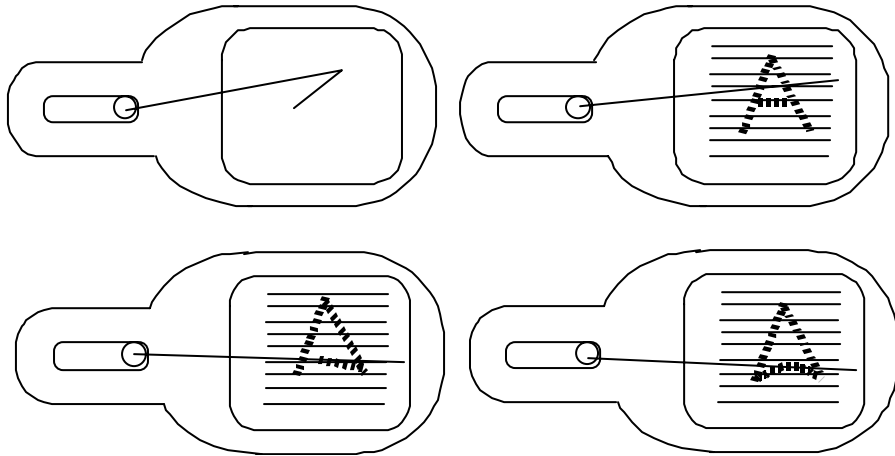


Fig. A raster scan system displays an object as a set of discrete points across each scan line.

- The raster graphics developed in early 70's.
- It is common CRT monitor.
- It is based on TV technology.
- In this system, electron beam sways across the screen one row at a time from top to bottom.
- Beam intensity vary (on or off) by the movement of electron beam across each row.
- 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 from 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:

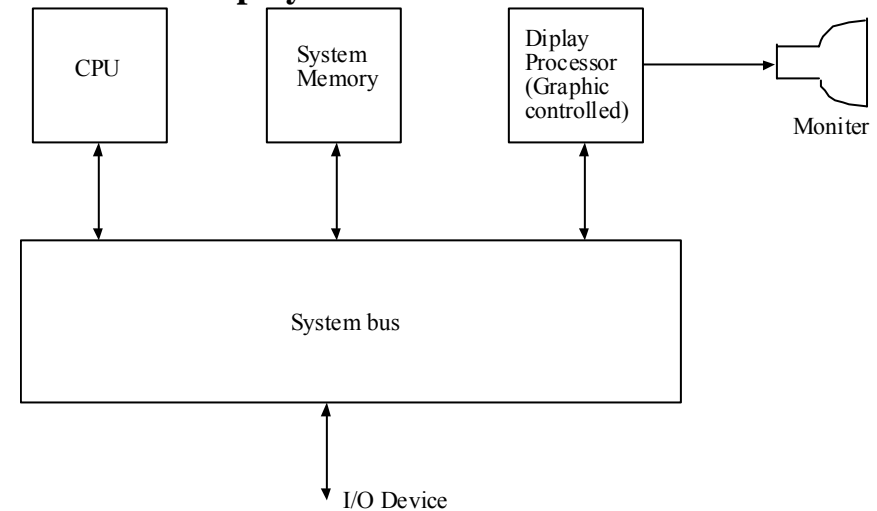


Fig: Architecture of simple random-scan system

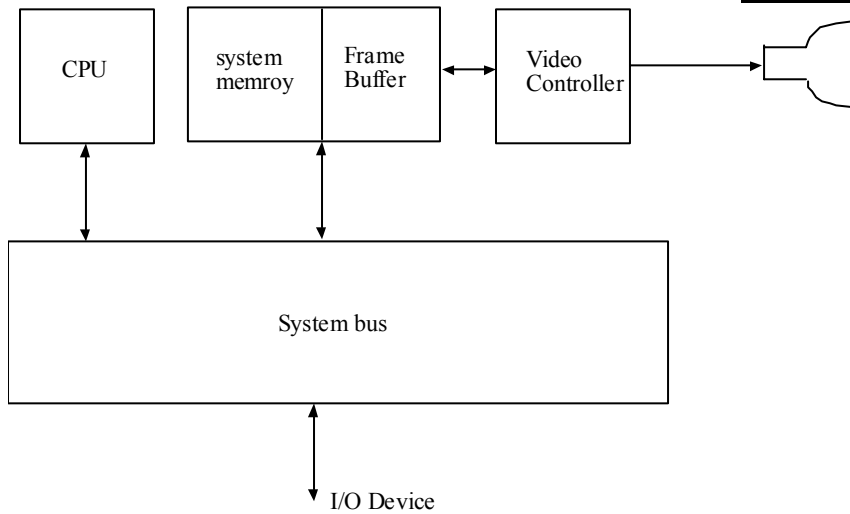


Fig. Architecture of a raster system with a fixed position of the system memory required for the frame buffer

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

Date: 2065/4/21

### Video Controller:

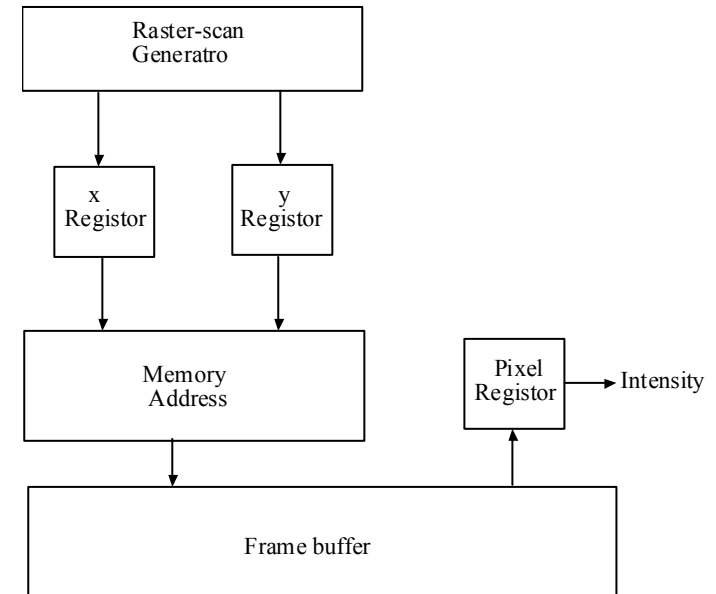


Fig. Basic video controller Refresh operation.

- 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.

- 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.

- 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.

Date: 2065/4/22

### 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.

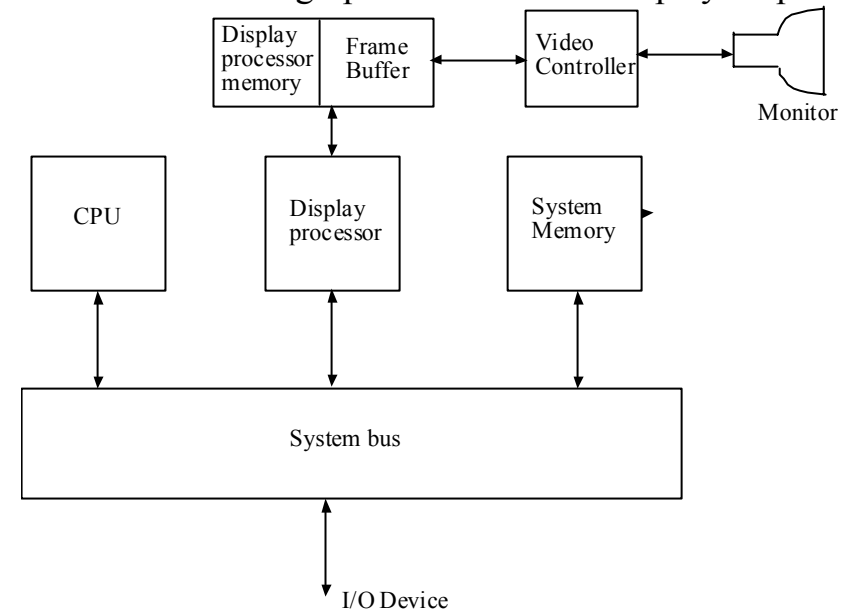


Fig. Architecture of raster graphics system with a display processor

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.

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:

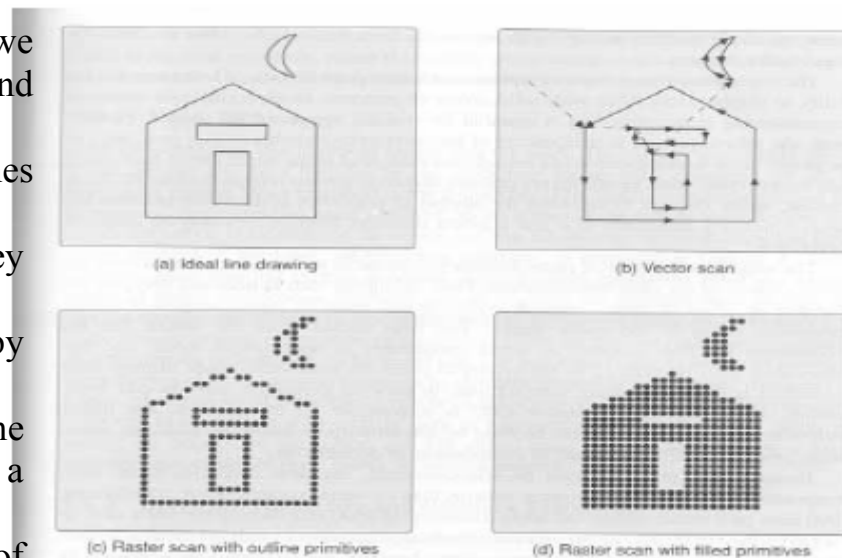


Fig.: Various Scan Methods

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

### Comparison between raster and vector graphics:

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:

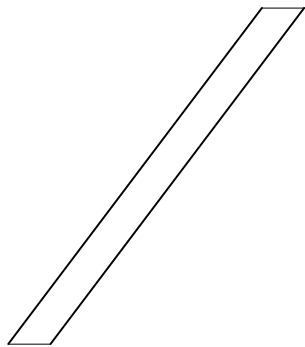


Fig (a): Actual thick

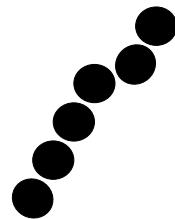


Fig (b): Scan - Converted

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 “jagging” or “staircasing”.

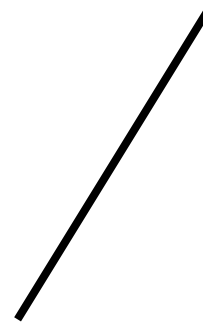


Fig. (a): Actual Line

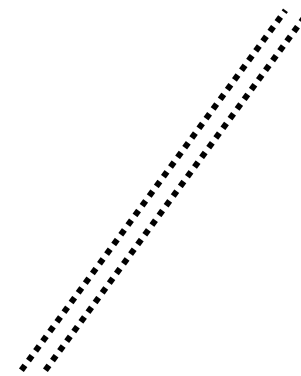


Fig. (b): Jagged Raster Line

Date:2065/4/23

### 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.

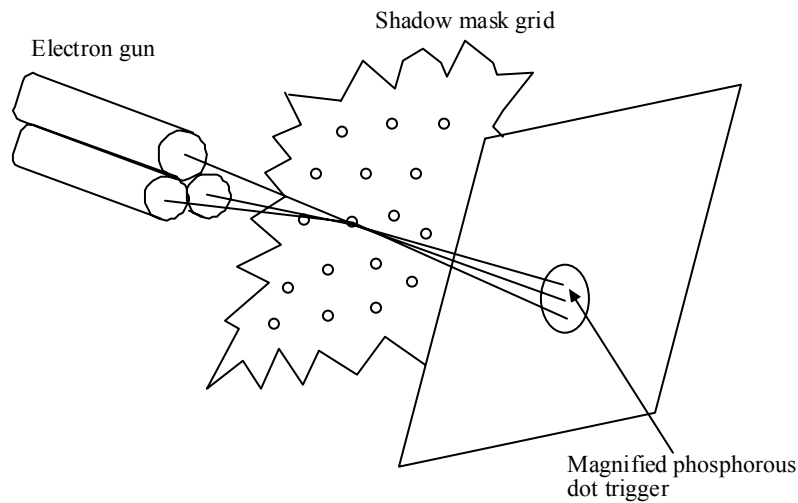


Figure: Operation of delta shadow mask method

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

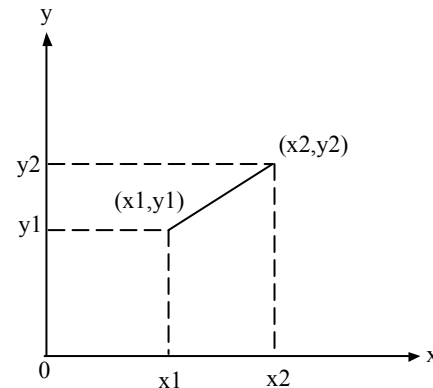
Date:2065/5/2

### 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.

$$m = \frac{y_2 - y_1}{x_2 - x_1} \dots\dots\dots(ii)$$

$$b = y - mx \dots\dots\dots(iii)$$

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

$$\Delta y = m \Delta x \dots\dots\dots(iv)$$

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

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

**Case: I**

For  $|m| < 1$ ,

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

**Case: II**

For  $|m| > 1$

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

**Case III**

When  $|m| = 1$ , then  $\Delta x = \Delta y$  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,

$$Y = mx+b\dots\dots\dots(i)$$

$$m = y_2-y_1/x_2-x_1 \dots\dots\dots(ii)$$

For any interval  $\Delta x$  , corresponding interval is given by  $\Delta y = m \Delta x$ .

**Case I :**

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

$$X_{k+1} = x_k+1 \dots\dots\dots(iii)$$

Then we compute each successive y-values,

$$\Delta y = m$$

$$Y_{k+1} = y_k + m \dots\dots\dots(iv)$$

**Case: II**

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

Therefore,  $1 = m \Delta x$

$$\Delta x = 1/m$$

$$X_{k+1} - x_k = 1/m\dots\dots\dots(v)$$

$$Y_{k+1} = y_k + 1\dots\dots\dots(vi)$$

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  $|m| < 1$ ,  $\Delta x = -1$

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

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

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

$$Y_{k+1} = y_k - 1 \dots\dots\dots(ix)$$

$$X_{k+1} = x_k + 1/m \dots\dots\dots(x)$$

Therefore, in general,

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

$$X_{k+1} = x_k \pm 1 \quad \text{for } |m| < 0$$

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

$$X_{k+1} = x_k \pm 1/m \quad \text{for } |m| > 0$$

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

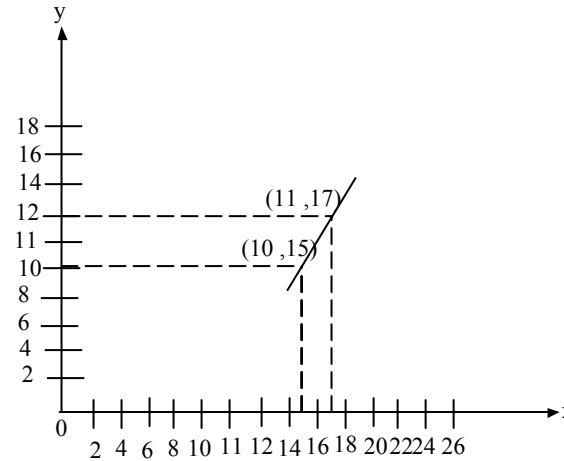
Solution:

The slope of line is

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{30 - 15}{15 - 10} = \frac{15}{5} = 3$$

$$|m| > 1$$

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  
 $dx = x_2 - x_1$   
 $dy = y_2 - y_1$
3. Test if  $|dy| < |dx|$  then  
 Steps =  $|dx|$

Else steps = /dy/

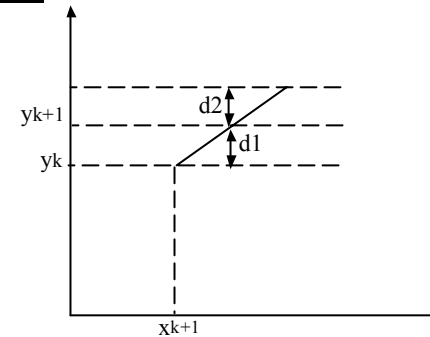
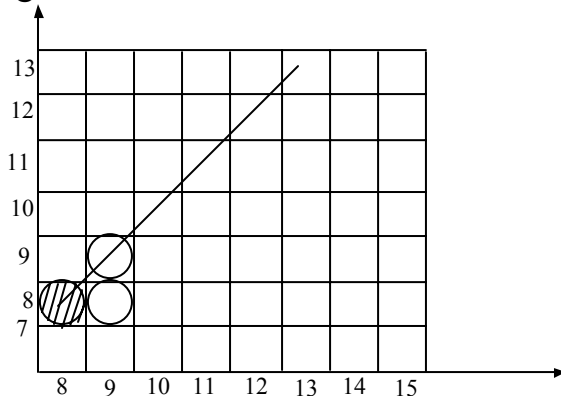
4. set del x = dx/steps/  
del y = dy/steps  
x = x1  
y = y1
5. Plot (x, y)
6. Do for k = 1 to steps  
x = x + delx  
y = y + del y

Plot (x,y)  
End do.

Date: 2065/5/3

### 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:

$$|m| < 1 \quad m > 0$$

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 separation 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 sampling 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  $k^{\text{th}}$  step by

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

$$\Delta x > 0,$$

Therefore,  $P_k < 0$  if  $d_1 < d_2$

$$P_k \geq 0, \text{ if } d_1 > d_2$$

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

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

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

$$P_x = \Delta x \{ 2mx_k + 2m + 2b - 2y_k - 1 \} \text{ since, } x_{k+1} = x_k + 1$$

$$P_x = 2mx_k \Delta x - 2y_k \Delta x + (2m + 2b - 1) \Delta x$$

$$P_x = 2 \cdot (\Delta y / \Delta x) \cdot x_k \Delta x - 2y_k \Delta x + (2m + 2b - 1) \Delta x$$

$$P_x = 2 \Delta y x_k - 2 \Delta x y_k + c$$

Where,  $C = (2m + 2b - 1) \Delta x$  is a constant.

Now, for next step,

$$P_{k+1} = 2 \Delta x x_{k+1} - 2 \Delta x y_{k+1} + c \dots \dots \dots \text{(ii)}$$

From (i) and (ii)

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

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

Where,

$$Y_{k+1} - y_k = '0' \text{ or } '1'$$

If  $p_k < 0$ , then we plot lower pixel

$$Y_{k+1} = y_k \dots \dots \dots \text{(iii)}$$

$$P_{k+1} = p_k + 2 \Delta y \dots \dots \dots \text{(iv)}$$

If  $p_k \geq 0$  then we plot upper pixel.

Therefore,  $y_{k+1} = y_k + 1 \dots \dots \dots \text{(v)}$

$$P_{k+1} = p_k + 2 \Delta y - 2 \Delta x \dots \dots \dots \text{(vi)}$$

Therefore, Initial decision parameter.

$$P_0 = 2 \Delta y x_0 - 2 \Delta x y_0 + c \quad [\text{from (i)}]$$

$$= 2 \Delta y x_0 - 2 \Delta x y_0 + (2m + 2b - 1) \Delta x$$

$$= 2 \Delta y x_0 - 2 \Delta x y_0 + 2 m \Delta x + 2 b \Delta x - \Delta x$$

$$= 2 \Delta y x_0 - 2 \Delta x y_0 + 2 \cdot (\Delta y / \Delta x) \cdot \Delta x + 2 (y_0 - mx_0) \Delta x - \Delta x$$

$$= 2 \Delta y x_0 - 2 \Delta x y_0 + 2 \Delta y + 2 \Delta x y_0 + 2 \Delta y / \Delta x \cdot x_0 \Delta x - \Delta x$$

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

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

Solution:

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

$$\Delta x = |10 - 15| = 5$$

$|m| < 1$ , 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  $p_k < 0$

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

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

if  $p \geq 0$

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

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

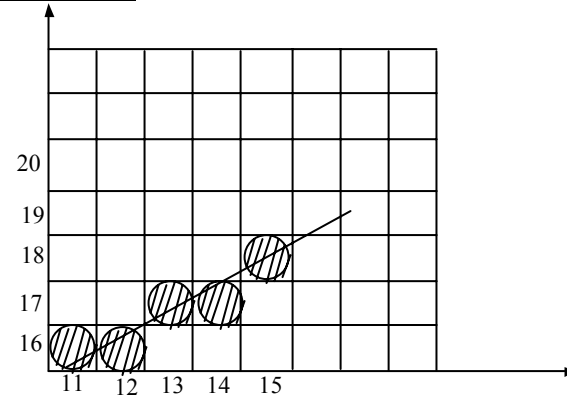
We have  $p_k \geq 0$ ,

$$\begin{aligned} \text{So, } p_1 &= p_0 + 2 \Delta y - 2 \Delta x \\ &= 1 + 2 \times 3 - 2 \times 5 \\ &= -3 \end{aligned}$$

$$\begin{aligned} P_2 &= p_1 + 2 \Delta y \\ &= -3 + 2 \times 3 \\ &= 3 \end{aligned}$$

$$\begin{aligned} P_3 &= p_2 + 2 \Delta y - 2 \Delta x \\ &= 3 + 2 \times 3 - 2 \times 5 \\ &= 9 - 10 \\ &= -1 \end{aligned}$$

K	P <sub>k</sub>	X <sub>k+1</sub> , Y <sub>k+1</sub>
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:

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

$$\Delta x = |10-15| = 5$$

$$m = 3/5 = 0.6$$

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

First pixel, ( 10, 15)

$$\begin{aligned} P_0 &= 2 \Delta y - \Delta x \\ &= 2 \times 3 - 5 \\ &= 1 \end{aligned}$$

We know that,

If  $p_k \geq 0$

$$\begin{aligned} P_1 &= p_0 + 2 \Delta y - 2 \Delta x \\ &= 1 + 2 \times 3 - 2 \times 5 \\ &= -3 \end{aligned}$$

$P_k < 0$

$$\begin{aligned} P_2 &= p_k + 2 \Delta y \\ &= -3 + 2 \times 3 \\ &= 3 \end{aligned}$$

**Note:**

For,  $P_k \geq 0$ ,

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

$$y_{k+1} = y_k - 1 \text{ (when the slope is -ve)}$$

$P_k < 0$

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

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



$$P_3 = p_2 + 2 \Delta y - 2 \Delta x$$

$$= 3 + 2 \times 3 - 2 \times 5$$

$$= -1$$

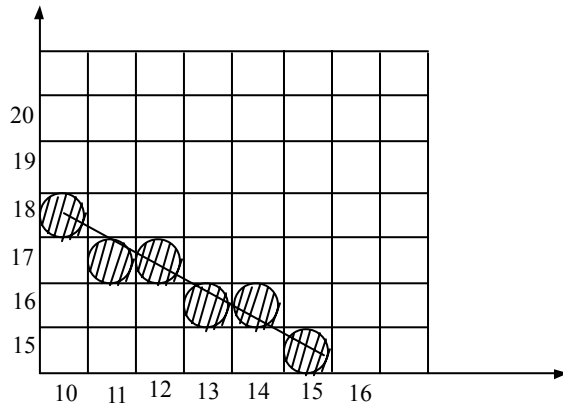
$$P_k < 0$$

$$P_4 = p_3 + 2 \Delta y$$

$$= -1 + 2 \times 3$$

$$= 5$$

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



Date: 2065/5/4

**Bresenham's line algorithm:**

1. Input the two 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  $\Delta x, \Delta y, 2 \Delta y$  and  $2 \Delta y - 2 \Delta x$  and obtained the starting value for the decision parameter as ;  
 $p_0 = 2 \Delta y - \Delta x$

4. At each  $x_k$  along the line, starting at  $k= 0$  , perform the following test:

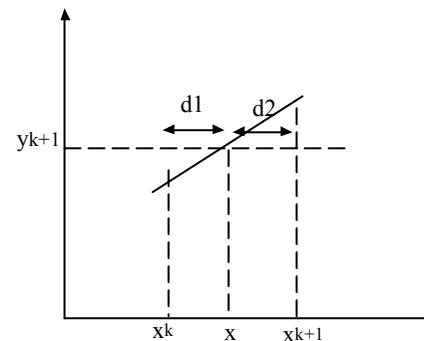
If  $p_k < 0$  the next point to plot is  $(x_{k+1}, y_k)$  and  $p_{k+1} = p_k + 2 \Delta y$ .

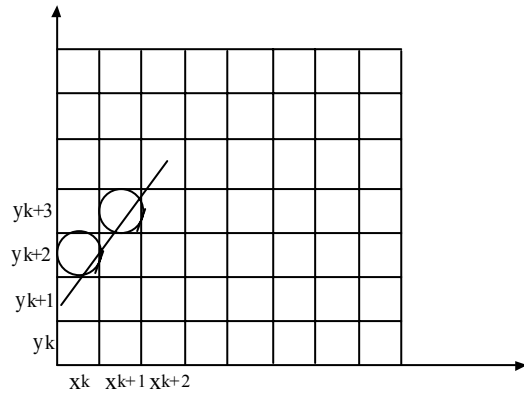
Otherwise , the next point to plot is  $(x_{k+1}, y_{k+1})$  and  $p_{k+1} = p_k + 2 \Delta y - 2 \Delta x$

5. Repeat step 4  $\Delta x$  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 positions  $(x_k, y_{k+1})$  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  $y_{k+1}$

From figure,

$$d1 = x - x_k$$

$$d2 = x_{k+1} - x$$

Let us define a decision parameter  $p_k$  and  $P_k$  is defined by,

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

$$\Delta y > 0$$

$$P_k < 0 \text{ if } d1 < d2$$

$$P_k \geq 0, \text{ if } d1 \geq d2$$

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

$$\begin{aligned} &= \Delta y (x - x_{k+1} + x) \\ &= \Delta y (2x - x_{k+1} + x) \\ &= \Delta y (2x - 2x_k + 1) \quad [\text{since, } x_{k+1} = x_k + 1] \\ &= \Delta y \left\{ 2 \left( \frac{y_{k+1} - b}{m} \right) - 2x_k - 1 \right\} \\ &= \Delta y / m \{ 2(y_{k+1} - b) - 2x_k m - m \} \\ &= \Delta x \{ 2 (y_{k+1} - b) - 2 x_k (\Delta y / \Delta x) - \Delta y / \Delta x \} \\ &= 2 \Delta x (y_{k+1} - b) - 2 \Delta y \cdot x_k - \Delta y \\ &= 2 \Delta x y_k - 2 \Delta y x_k + 2(1-b) \Delta x - \Delta y \end{aligned}$$

$$P_x = 2 \Delta x y_k - 2 \Delta y x_k + c \dots \dots \dots (ii)$$

$$\text{Let } C = 2 (1-b) \Delta x - \Delta y$$

Now for next step,

$$P_{k+1} = 2 \Delta x y_{k+1} - 2 \Delta y x_{k+1} + c \dots \dots \dots (iii)$$

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,

$$\begin{aligned}
 P_o &= 2 \Delta x y_o - 2 \Delta y x_o + c \\
 &= 2 \Delta x y_o - 2 \Delta y x_o + 2 (1-b) \Delta x - \Delta y \\
 &= 2 \Delta x y_o - 2 \Delta y x_o + 2 \Delta x - 2b \Delta x - \Delta y \\
 &= 2 \Delta x y_o - 2 \Delta y x_o + 2 \Delta x - 2 (y_o - mx_o) \Delta x - \Delta y \\
 &= 2 \Delta x y_o - 2 \Delta y x_o + 2 \Delta x - 2 \Delta x y_o + 2. (\Delta y / \Delta x). x_o
 \end{aligned}$$

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

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

Date: 2005/5/10

**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  $dy = |y_2 - y_1|$
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,  
 $x = x_1$ ;  $x_1 = x_2$ ;  $x_2 = t$   
 $y = y_1$ ,  $y_1 = y_2$ ;  $y_2 = t$ .
- 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 circle is defined as a set of points that are all at a given distance 'r' from the centre position  $(x_c, y_c)$ . This distance relationship is expressed by the pythagorean 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:**

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

**Bresenham's mid point algorithm :**

This is also a scan converting 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,

$$2x + 2y \frac{dy}{dx} = 0$$

Or  $\frac{dy}{dx} = -x/y$ , where,  $\frac{dy}{dx}$  = slope

Now, if  $\frac{dy}{dx} = 0$ , then  $x = 0$ .

If,  $\frac{dy}{dx} = -1$ , then  $x = y$

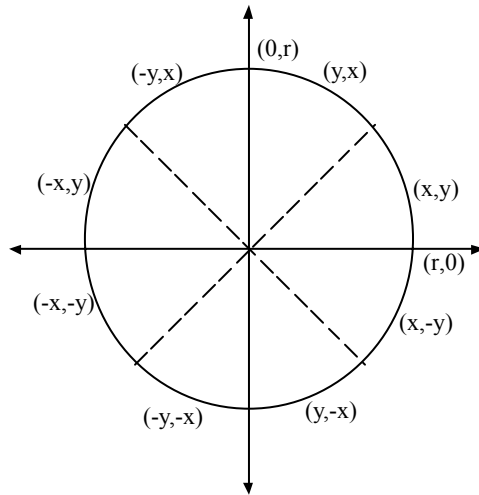


Fig. Symmetry of circle

Consider circle section for  $x = 0, x = y_0$  where slope of the curve varies from 0 to 1. 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,

$$f_{\text{circle}}(x,y) = x^2 + y^2 - r^2$$
 .....(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 circle function tests are performed for the mid position between pixels near the circle path at each sampling step.

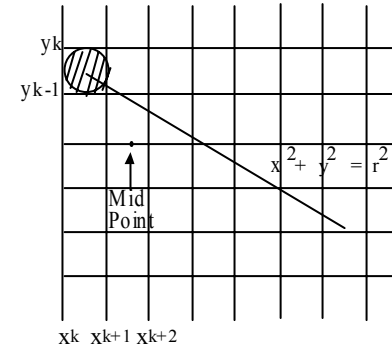


Fig. Midpoint between candidate pixels at sampling position  $x_{k+1}$  along a circular path.

Here assume that we have just plotted the pixel  $(x_k, y_k)$ . We next need to determine whether the pixel at position  $(x_{k+1}, y_{k+1})$  or 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.

$$\begin{aligned} \text{I.e } P_k &= f_{\text{circle}}(x_k+1, y_k-1/2) \\ &= (x_k+1)^2 + (y_k-1/2)^2 - r^2 \end{aligned}$$
 .....(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 evaluation the circle function at sampling position.

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

$$P_{k+1} = f_{\text{circle}}(x_{k+1} + 1, y_{k+1} - 1/2) \\ = (x_{k+1} + 1)^2 + (y_{k+1} - 1/2)^2 - r^2 \quad \dots\dots\dots(v)$$

Substituting equation (iv) from (v).

$$P_{k+1} - p_k = (x_{k+1} + 1)^2 + (y_{k+1} - 1/2)^2 - (x_k + 1)^2 - (y_k - 1/2)^2 \quad \dots(VI)$$

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)

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

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

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)

$$P_{k+1} = p_k + (x_{k+1} + 1)^2 + (y_k - 1/2)^2 - (x_k + 1)^2 - (y_k - 1/2)^2 \\ = P_k + (x_k + 1)^2 + 2(x_k + 1) \cdot 1 + 1 - (x_k + 1)^2 \\ = P_k + 2(x_k + 1) + 1 \\ = P_k + 2(x_{k+1}) + 1$$

$$\text{Therefore, } P_{k+1} = p_k + 2x_{k+1} + 1 \quad \dots\dots\dots(viii)$$

For initial decision parameter.

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

$$\text{Hence, } P_0 = f(x_0 + 1, y_0 - 1/2) \\ = f(1, r - 1/2) \\ = 1 + (r - 1/2)^2 - r^2 \\ = 1 + r^2 - r + 1/4 - r^2$$

$$P_0 = 5/4 - r$$

$$P_0 = 1 - r$$

$P = 1 - r \quad \dots\dots\dots(ix)$  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,

$$\text{Centre} = (0, 0)$$

$$\text{Radius } (r) = 10$$

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

$$P_0 = 1 - r \\ = 1 - 10 \\ = -9$$

Successive decision parameter values are positions along the circle path are calculated using mid-point method as,

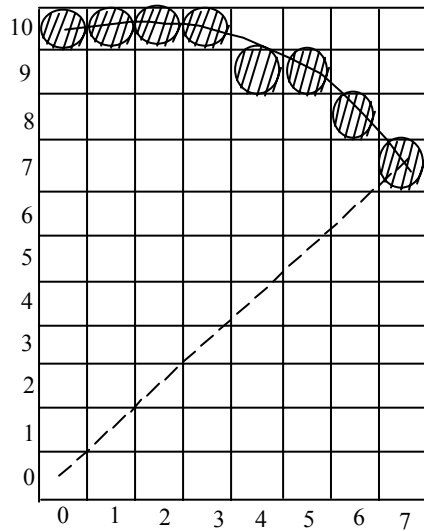
$$P_{k+1} = p_k + 2x_{k+1} + 1 \quad (P_k < 0)$$

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

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

$$\text{Set } x_{k+1} = x_k + 1, y_{k+1} = y_k - 1$$

K	P <sub>k</sub>	x <sub>k+1</sub>	y <sub>k+1</sub>	2x <sub>k+1</sub>	2y <sub>k+1</sub>
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

1<sup>st</sup> pixel (0,5)

P<sub>0</sub> = 1-1

= 1-5

= -4

P<sub>k+1</sub> = p<sub>k</sub> + 2x<sub>k+1</sub> + 1 (p<sub>k</sub> < 0)

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

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

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

K	p <sub>k</sub>	x <sub>k+1</sub>	y <sub>k+1</sub>	2x <sub>k+1</sub>	2y <sub>k+1</sub>
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

(1,5)

(1,5)

(-1,5)

(-1,-5)

(1,-5)

(5,1)

(5,-1)

(-5,-1)

(-5,1)

Actual pixel

(3,8) (x,y)

(1,8) (-x,y)

(1,-2) (-x,-y)

(3,-2) (x,-y)

(7,4) (y,x)

(7,2) (y,-x)

(-3,2) (-y,-x)

(-3,4) (-y,x)

(2,5)

(2,5)

(-2,5)

(-2,-5)

(2,-5)

(5,2)

(5,-2)

(-5,-2)

(-5,2)

(4,8)

(0,8)

(0,-2)

(4,-2)

(7,5)

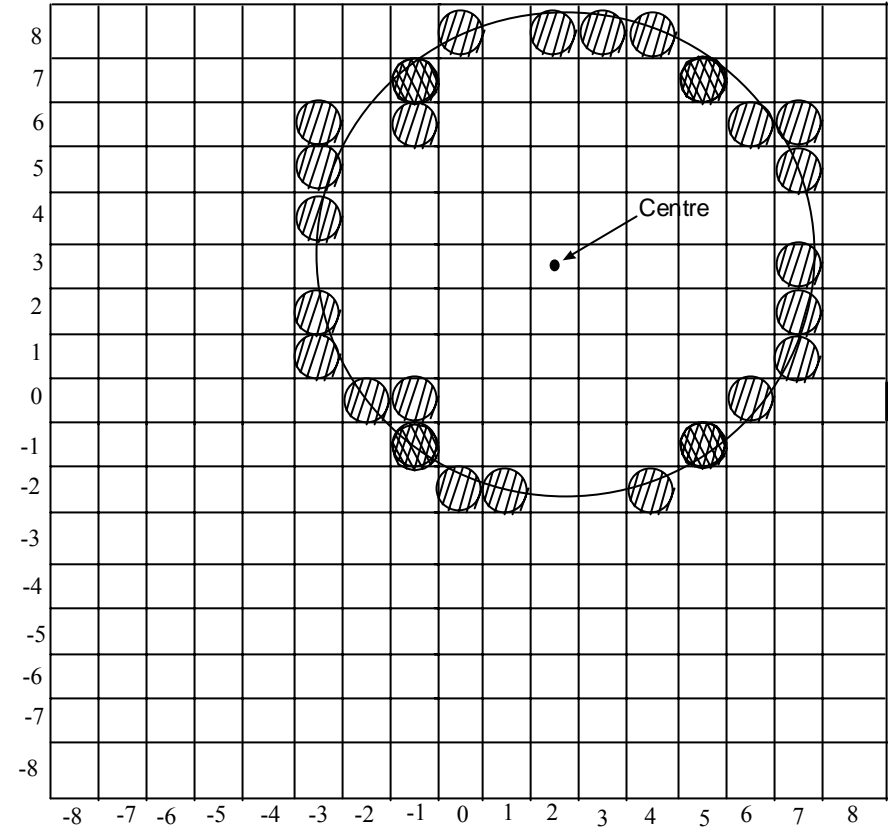
(7,1)

(-3,1)

(-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.

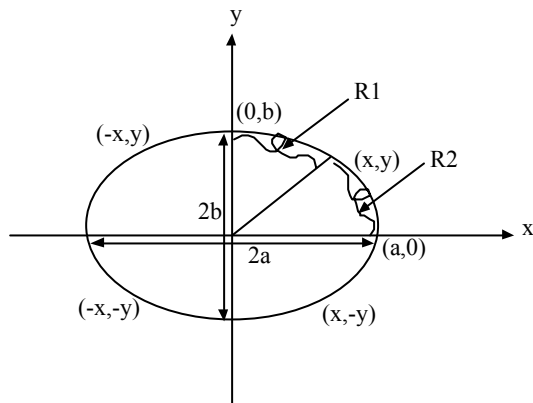
iv. sdf

**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,  $y = \pm b \sqrt{1 - x^2/a^2}$



This algorithm is used to display the ellipse in standard form. The algorithm is applied throughout the 1<sup>st</sup> quadrant according to the slope of the ellipse. For the region (R<sub>1</sub>) where the slope of the curve is less than one. We process by taking unit steps in x-direction and for the region (R<sub>2</sub>) 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<sub>1</sub>, R<sub>2</sub>). 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

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

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

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

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

Therefore, we move out of region 1 (R<sub>1</sub>) when  $2b^2x \geq 2a^2y$ . **For region R<sub>1</sub>.**

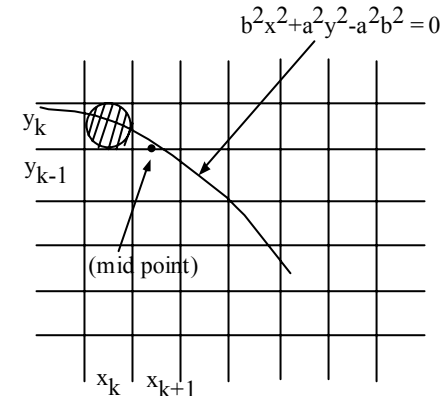


Fig. Mid point between candidate pixel at sampling position  $x_k+1$  along in elliptical path.

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

$$P_k = f(x_k+1, y_k-1/2) = b^2(x_k+1)^2 + a^2(y_k-1/2)^2 - a^2b^2 \dots\dots(i)$$

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

Substituting (i) from (ii)

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

If  $P_k < 0$ , mid point lies inside the ellipse, So we plot,

$$(x_{k+1}, y_k) \text{ i.e } x_{k+1} = x_k + 1, y_{k+1} = y_k$$

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

$$p_{k+1} = p_k + 2b^2x_{k+1} + b^2 \dots\dots(iii) \text{ Where,}$$

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

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

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

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

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

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

Where,

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

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

Now initial decision parameter is given by is given by ,

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

$$= f(0+1, b-1/2) = b^2 \cdot 1^2 + a^2(b-1/2)^2 - a^2b^2$$

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

$$P_0 = b^2 - a^2b + a^2/4 \dots\dots(v)$$

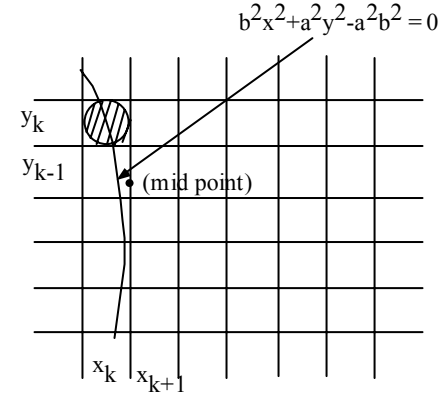


Fig. Mid point between candidate pixel at sampling position  $y_k-1$  along an elliptical path.

**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

$$P_k = f(x_k+1/2, y_k-1) = b^2(x_k+1/2)^2 + a^2(y_k-1)^2 - a^2b^2 \dots\dots(i)$$

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

Substituting (i) from (ii)

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

If  $p_k > 0$ , the mid point lies outside the ellipse, so we plot

$$(x_k, y_{k-1})$$

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

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

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

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

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

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

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

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

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

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

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

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

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

The initial value of initial decision parameter for region 2 is

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

$$P_0 = b^2 (x_0 + 1/2)^2 + a^2 (y_0 - 1)^2 - a^2 b^2 \dots\dots\dots (vi)$$

[**Note:**  $(x_0, y_0)$  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  $R_1, P_1 = b^2 - a^2 b^2 + a^2 / 4$

5. Repeat the following till  $2b^2 x < 2a^2 y$

- i.  $x = x + 1$
- ii. Test if  $P_1 < 0$  then

$$P_1 = p_1 + 2b^2 x + b^2$$

Else,

$$y = y - 1$$

$$P_1 = p_1 + 2b^2 x - 2a^2 y = b^2$$

iii. Plot the points  $(x, y)$  and its symmetry points at approximate position by

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

6. Compute initial decision parameter for  $R_2$

$$P_2 = b^2 (x + 0.5)^2 + a^2 (y - 1)^2 - a^2 b^2$$

7. Repeat the following till  $y > 0$

- i.  $y = y - 1$
- ii. Test if  $P_2 > 0$  then

$$P_2 = p_2 - 2a^2 y + a^2$$

Else,

$$x = x + 1$$

$$P_2 = p_2 + 2b^2 x - 2a^2 y + a^2$$

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:

$$\text{Center} = (2, -5)$$

$$a = 8$$

$$b = 6$$

For region  $R_1$

First pixel is (0,6)

$$P_0 = b^2 - a^2b + a^2/4$$

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

$$= -332$$

Successive decision parameter values and positions along 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  $p_k \geq 0$ ,

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

$$P_0 = b^2(x_0+1/2) + a^2(y_0-1)^2 - a^2b^2$$

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

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

$$= -151$$

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.

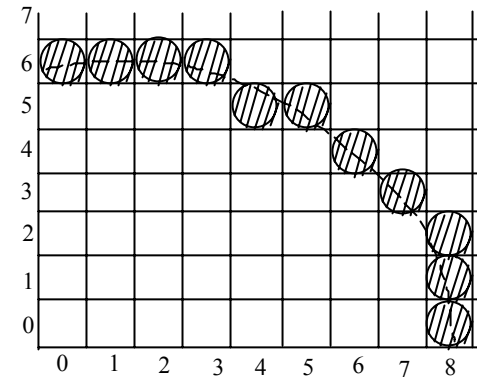


Fig. Positions along an elliptical path centered on the origin with  $a=8$  and  $b=6$  using the midpoint algorithm to calculate pixel addresses in first quadrant.

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 deformation in shape. )
- Non rigid body transformation (transformation with change in shape)

**Basic Transformations:**

1. Translation
2. Reflection
3. Scaling

**Other Transformation**

1. reflection
2. Shearing

**Two dimensional transformation:**

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

If P(x,y) is translated to a position p'(x',y') by t<sub>x</sub> units parallel to x-axis and t<sub>y</sub> units parallel to y axis then,

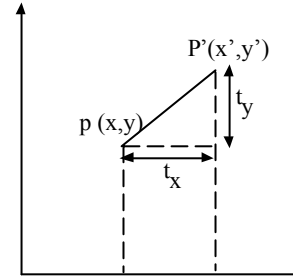
$$X' = x + t_x$$

$$Y' = y + t_y$$

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:

$$(x,y) = (2,5)$$

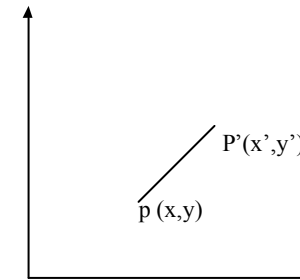
$$T_x = 3$$

$$T_y = 3$$

$$X' = x + t_x$$

$$Y' = y + t_y$$

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



**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,

$$x' = x \cdot s_x$$

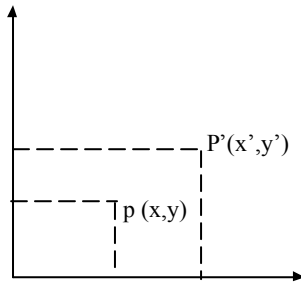
$$y' = y \cdot s_y$$

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

$$p' = S (s_x, s_y)$$



Where

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

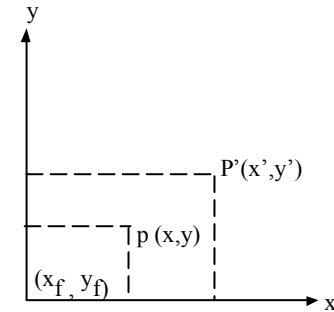
If  $s_x = s_y$  scaling type is uniform

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

c. **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

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

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



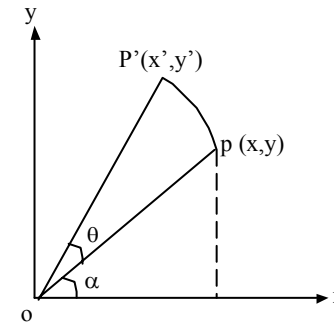
In matrix form,

.....

**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:**

- About origin
- About any point
- About origin.

Figure:

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

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

$$x = r \cos\alpha$$

$$y = r \sin\alpha$$

Also,

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

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

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

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

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

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

In matrix form:

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

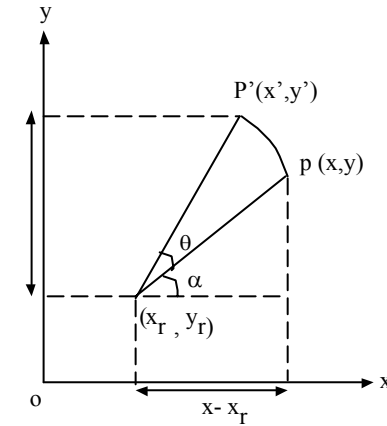
Note:  $\theta$  +ve ( anticlock wise)

$\theta$  -ve ( clock wise )

**b. About any point:** if  $(x,y)$  is rotated to a new position  $p'(x',y')$  by an angle  $\theta$ , 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  $60^\circ$  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 rerepresentation. 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  $(2 \times 2)$  matrix is changed into  $3 \times 3$  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,  $p' = T(t_x, t_y) \cdot p$

For Rotation (about origin)

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

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

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.

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

Solution:

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

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

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

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

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

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

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

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

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

Which demonstrates that two successive translations are additive.

# Prove that two successive rotations are additive. i.e  $R(\theta_2) \cdot$

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

Solution:

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

$\cdot p$

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

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

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

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

# Prove that two successive scalings are multiplicative.

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

Solution:

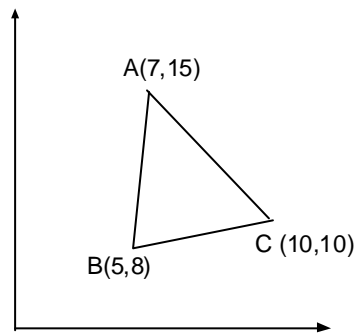
Concatenating transformation matrices for two successive scaling operations produces the following composite scaling matrix.

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

Or  $s(s_{x2}, s_{y2}) S(s_{x1}, s_{y1}) = S(s_{x1}, s_{x2}, s_{y1}, s_{y2})$

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

Solution:



Step 1: Rotation by  $45^\circ$  (clock wise)

Step-II: Scaling by (2,3)

$$S(2,3)$$

Net transformation:  $= S(2,3) \cdot R(-45)$

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

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

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

The transformation points are ,

$$A' = TA$$

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

$$B' = TB$$

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

$$C' = TC$$

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



**Note from miss:**

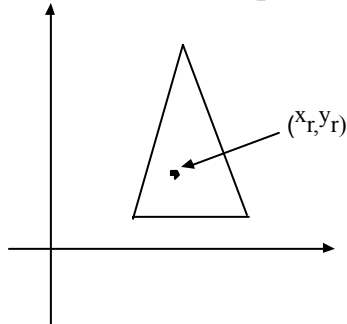
**General pivot point rotation:** We can rotate any selected point  $(x_r, y_r)$  by following the sequence of translate – rotate –translate operations.

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

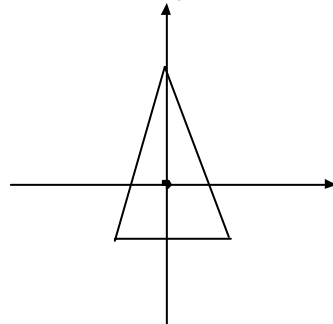
**Step 2:** Rotate the object coordinate origin i.e  $R(\theta)$ .

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

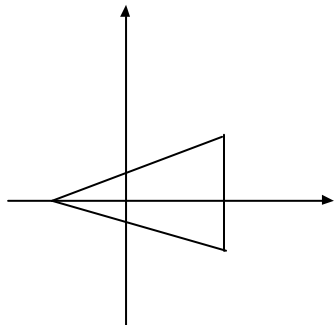
This transformation sequence is illustrated in fig below.



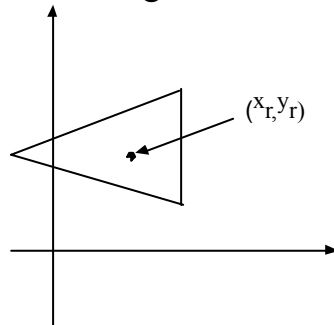
(a) Original position of object  
And pivot point



(b) Translation of object so  
that pivot 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) =  $T(x_r, y_r) \cdot R(\theta) \cdot T(-x_r, -y_r)$

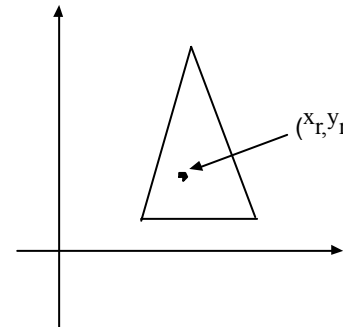
$$= \begin{bmatrix} 1 & 0 & x_r \\ 0 & 1 & y_r \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_r \\ 0 & 1 & -y_r \\ 0 & 0 & 1 \end{bmatrix}$$

### General Fixed point Scaling:

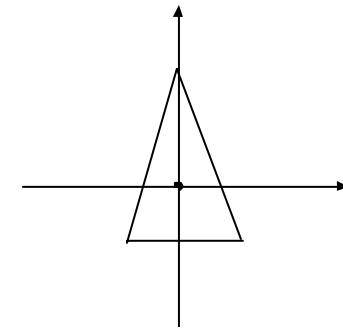
A transformation sequence to produce scaling with respect to a selected fixed position  $(x_f, y_f)$  using a scaling function is illustrated below:

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.

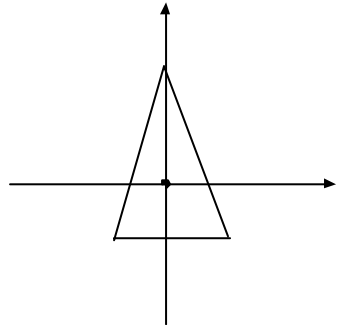
The transformation sequence is illustrated in fig below:



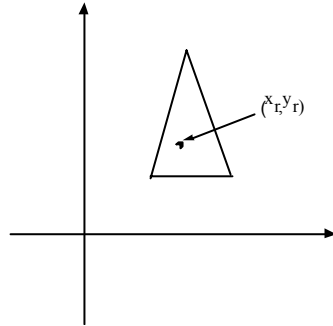
(a) Original position of object  
And fixed point



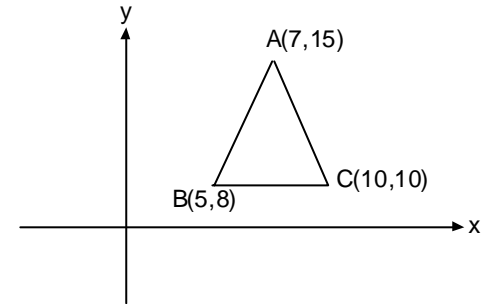
(b) Translate object  
so that fixed point  
 $(x_r, y_r)$  is at origin.



(c) Scale object with respect To origin



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



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

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

Date: 2065/6/5

# Rotate the  $\triangle ABC$  by  $90^\circ$  anticlock wise about  $(5, 8)$  and scale it by  $(2, 2)$  about  $(10, 10)$

Solution:

Step:1

$T(-5, -8)$

Step: 2

$R(90^\circ)$

Step: 3

$T(5, 8)$

Step: 4

$T(-10, -10)$

Step:5

$S(2, 2)$

Step: 6

$T(10, 10)$

Net transformation:

$T(10, 10) \cdot S(2, 2) \cdot T(-10, -10) \cdot T(5, 8) \cdot R(90^\circ) \cdot T(-5, -8)$

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

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

**Other transformation:**

1. Reflection

## 2. Shearing.

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

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

$$x' = x$$

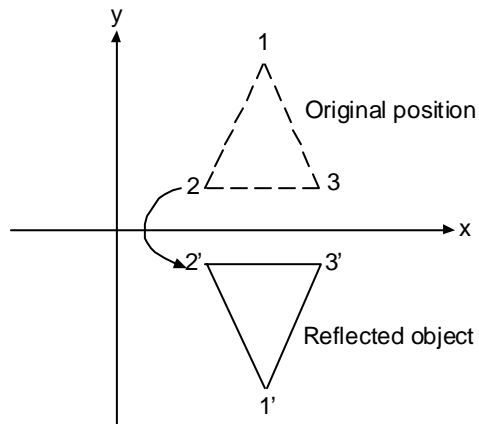
$$y' = -y$$

In matrix from,

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

$$P' = R_{fx} \cdot P$$

$R_{fx}$  = Reflection matrix about x-axis



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

$$x' = -x$$

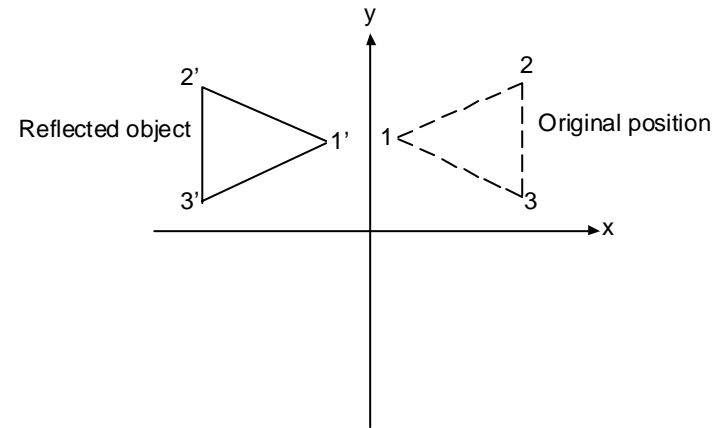
$$y' = y$$

In matrix from,

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

$$P' = R_{fy} \cdot P$$

$R_{fy}$  = Reflection matrix about y-axis



Date: 2065/6/6

### Alternative way for reflection:

Step: 1

Rotate the object so that y axis coincide 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:

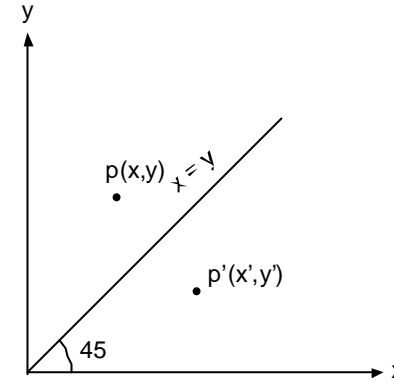
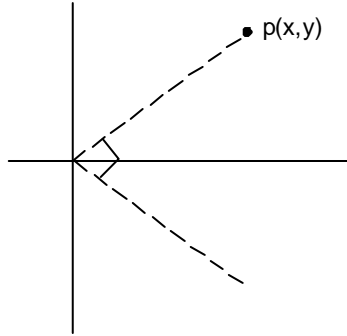
$$\text{i.e } R_{fy} = R(90) \cdot R_{fx} \cdot R(-90)$$

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

$$y' = x$$

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

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



# Determine transformation matrix responsible for reflection of object about the line  $y = x$

- (1)  $R(-45)$
- (2)  $R_{fx}$
- (3)  $R(45^\circ)$

Net transformation ,

$$R(45) \cdot R_{fx} \cdot R(-45)$$

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

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

i.e  $x' = y$

# 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,

$$R(-45) \cdot R_{fx} \cdot R(+45)$$

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

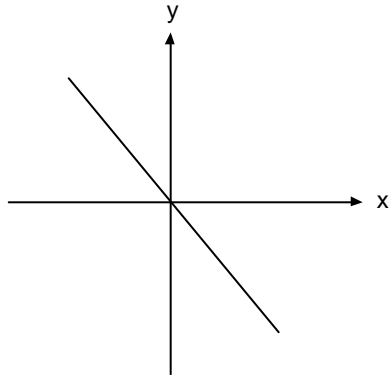
$$= \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}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



(5) R(0,2)

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

i. e  $x' = -y$   
 $y' = -x$

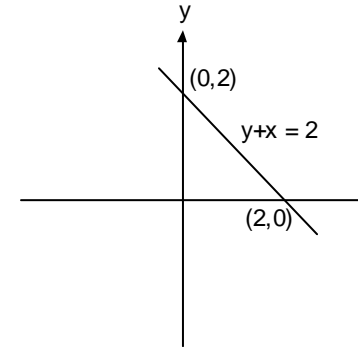


# Determine transformation responsible for reflection of object about the line  $y + x = 2$ .

- (i) T(0,-2)
- (ii) R(45°)
- (iii) R<sub>fx</sub>
- (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<sub>fx</sub>
- (4) R(-45)



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

- (i) T(0,-b)
- (ii) R(-θ)
- (iii) R<sub>fx</sub>
- (iv) R(θ)
- (v) T(0,b)

Net transformation = T(0,b). R(θ) . R<sub>fx</sub>.R(-θ) . T(0,-b)

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

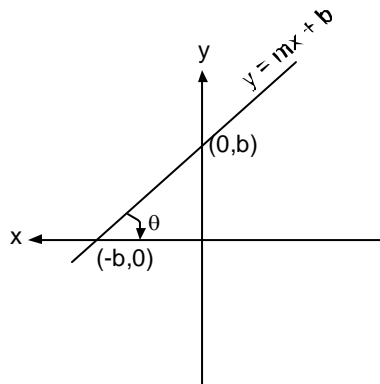
$$\begin{bmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & -b \\ 0 & 0 & 1 \end{bmatrix} \\ = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & b \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin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

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

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

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

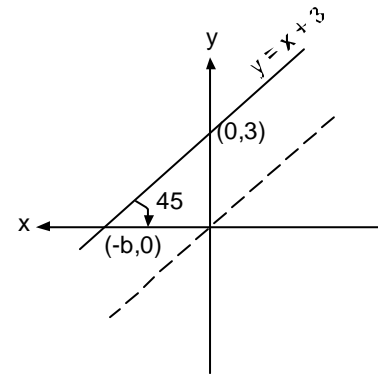
$$= \begin{bmatrix} \frac{1-m^2}{1+m^2} & \frac{2m}{1+m^2} & \frac{-2bm}{1+m^2} \\ \frac{2m}{1+m^2} & \frac{m^2-1}{1+m^2} & \frac{2b}{1+m^2} \\ 0 & 0 & 1 \end{bmatrix}$$



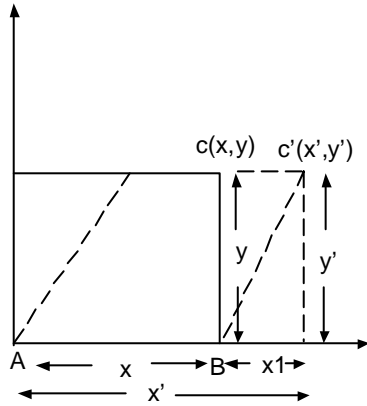
### Reflection:

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

- (1) T(0,-3)
- (2) R(-45)
- (3) R<sub>fx</sub>
- (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

$$y' = y$$

$$x' = x + x_1 \text{ where,}$$

$$x_1 \propto y$$

$$x_1 = Sh_x \cdot y$$

where,  $Sh_x$  is shearing constant.

$$y' = y$$

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

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

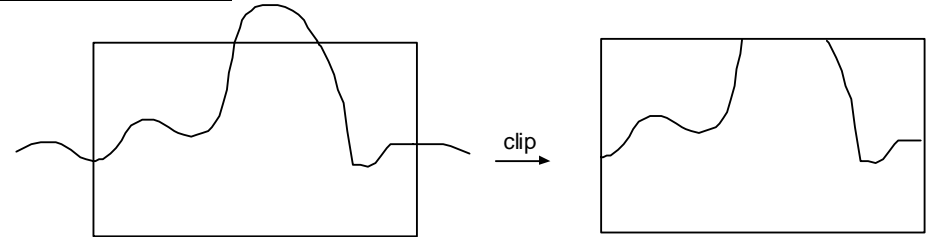
$$x' = x$$

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

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

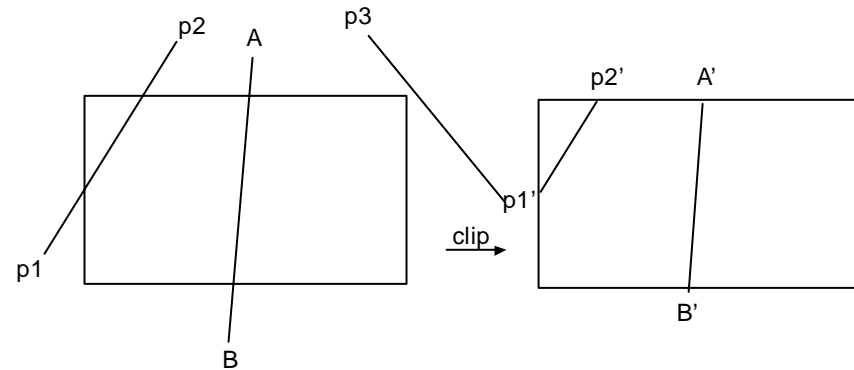
$$P' = SH_y \cdot p$$

## 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.

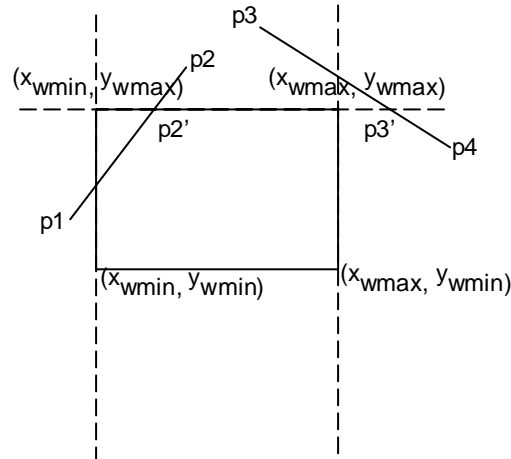
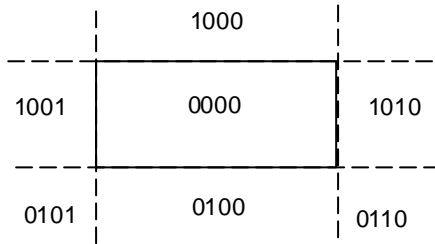
For,

$b_1$  : 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  $R_A \text{ AND } R_B \neq 0$

Line is completely outside .

The lines which can 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

$$P_1 = 0001$$

$$P_2 = 1000$$

$$P_1 \text{ AND } P_2 = 0$$

So we need future calculation.

Starting from  $p_1$ ,

Intersection of  $P_1$  with left boundary is calculated.

Region code of  $P_1' = 0000$

$$P_1' \text{ AND } P_2 = 0$$

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  $P_3$  ,  $P_4$ .

$$P_3 = 1000$$

$$P_4 = 0010$$

$$P_3 \text{ AND } P_4 = 0$$

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 0010.

Again  $p_3' \text{ AND } P_4 \neq 0$

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  $m = \frac{y_2 - y_1}{x_2 - x_1}$ .

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 part 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 viewport.

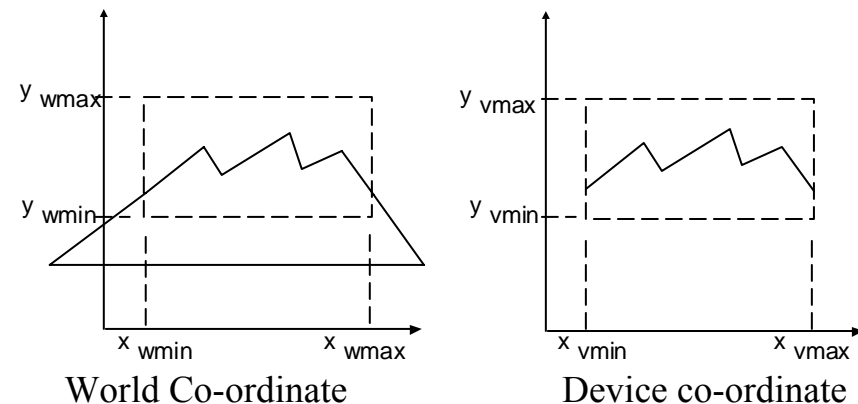


Fig. A viewing transformation using standard rectangles for the window and viewpoint.

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.

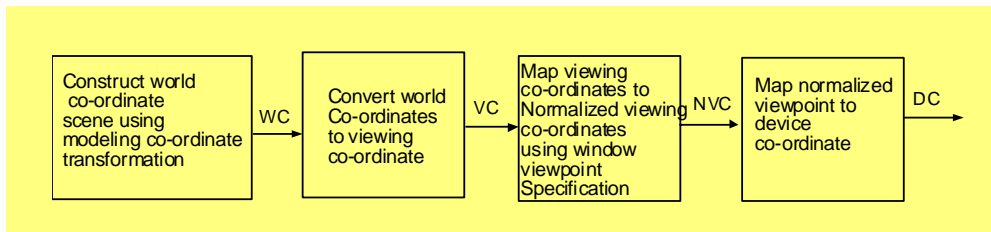
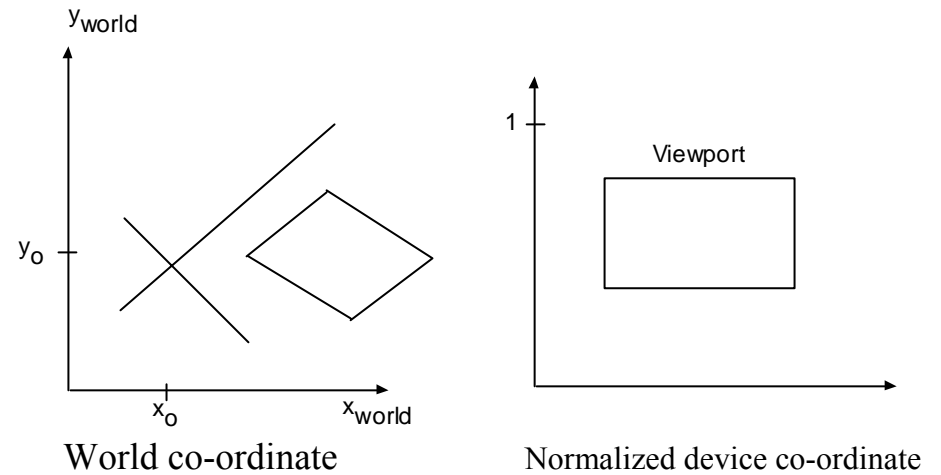


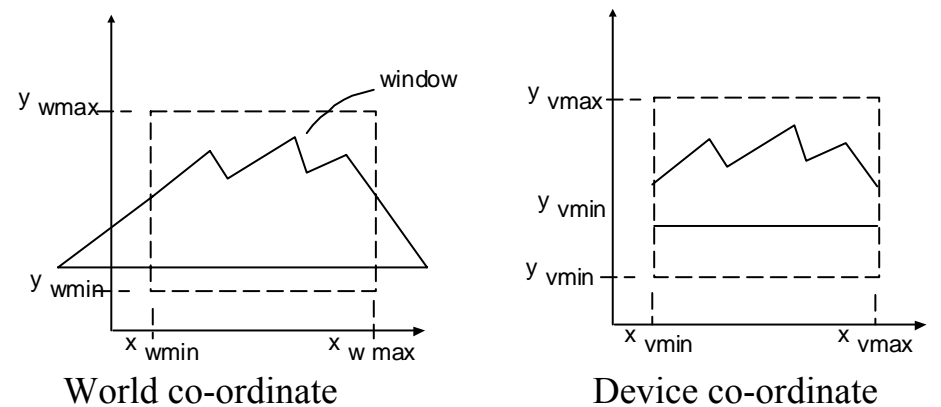
Fig. The two dimensional viewing transformation pipeline

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



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:**



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 scene to device co-ordinate is called viewing transformation or window-to-viewport transformation.

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 viewport.

To maintain the same relative position, so

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

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

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

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

$$\therefore T_{wv} = \begin{bmatrix} 1 & 0 & x_{v \min} \\ 0 & 1 & y_{v \min} \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{v \max} - x_{w \min} & 0 & 0 \\ x_{w \max} - x_{w \min} & 0 & 0 \\ 0 & y_{v \max} - y_{w \min} & 0 \\ 0 & y_{w \max} - y_{w \min} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -x_{w \min} \\ 0 & 1 & -y_{w \min} \\ 0 & 0 & 1 \end{bmatrix}$$

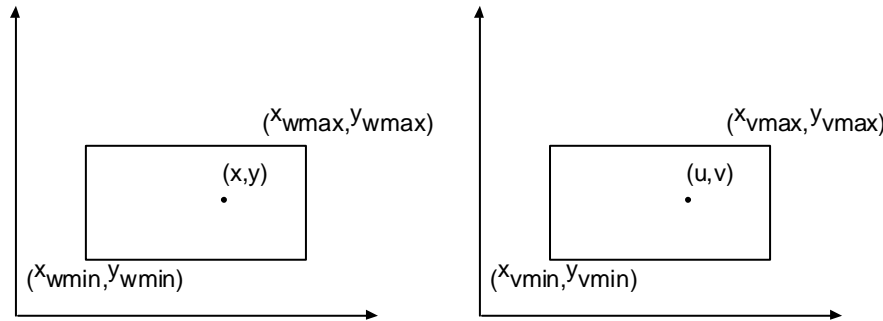
Is the required window- to – viewpoint transformation.

# 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)



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:  $T(-x_{w \min}, -y_{w \min})$

Step 2:  $S(s_x, s_y)$

Step 3:  $T(x_{v \min}, y_{v \min})$

$\therefore$  Net transformation,

$$T_{wv} = T(x_{v \min}, y_{v \min}) \cdot S(s_x, s_y) \cdot T(-x_{w \min}, -y_{w \min})$$

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.

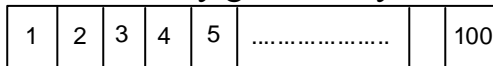


Fig. Array of 100 students

(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:

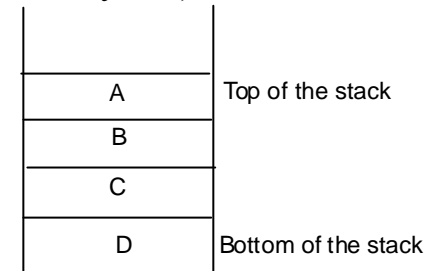
- push - insertion
- pop - deletion
- peep – extract information

- 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  $TOS \geq size$

Output: “stack overflow and exit”.

Step 2: Increment the pointer value by one

$TOS = TOS + 1$

Step 3: Perform insertion



$S[\text{TOS}] = \text{value}$

Step 4: Exit

**POP operation:**

Step 1: Check for the underflow

If  $\text{TOS} = 0$

Output: “Stack underflow and exit”

Step 2: Decrement the pointer value by one

$\text{Value} = S[\text{TOS}]$

$\text{TOS} = \text{TOS} - 1$

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  $\text{TOS} - i + 1 < 0$

Output : “ Stack underflow” and exit.

Step 2: [Return the  $i^{\text{th}}$  element from the top of the stack]

Return[ $S(\text{TOS} - i + 1)$ ]

**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  $i^{\text{th}}$  location in the stack ‘s’. We move TOS pointer to the  $i^{\text{th}}$  location from the top of the stack and input the new value of that location.

Step 1: [Check for underflow]

If  $\text{TOS} - i + 1 < 0$

Output “stack underflow” and exit.

Step 2: [Change the element]

$S(\text{TOS} - i + 1) = \text{value}$

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:

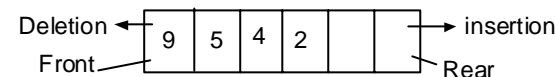
- Infix
- Postfix
- Prefix

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  $+ab$ .

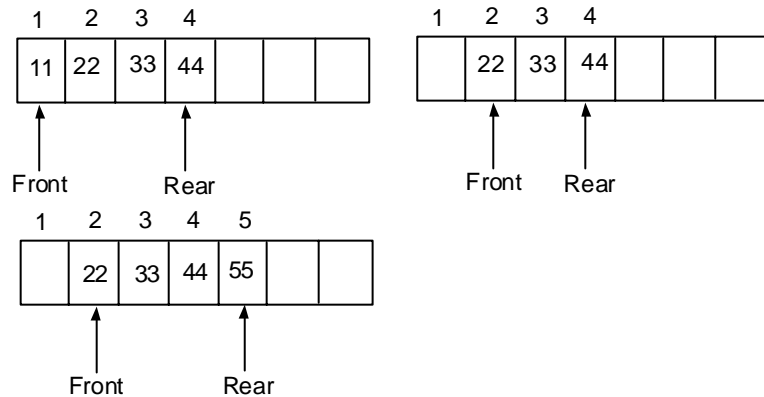
*Date: 2065/6/8*

**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 in 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 pointers; 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:**

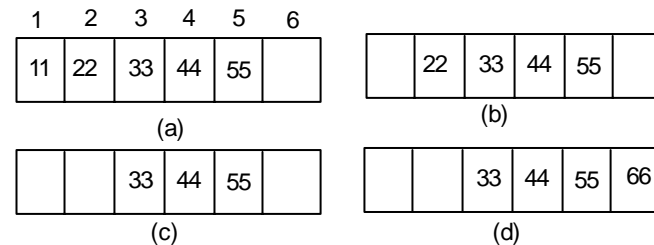
- Step 1: [check overflow condition]
  - If rear  $\geq$  size
  - o/p : “overflow” and return
- Step 2: [ Increment rear pointer]
  - Rear = rear+1
- Step 3: [Insert an item ]

- 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

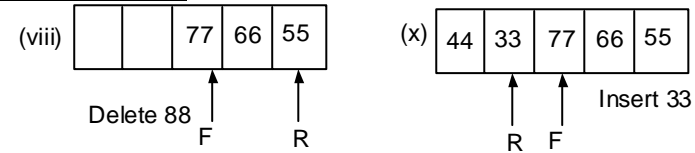
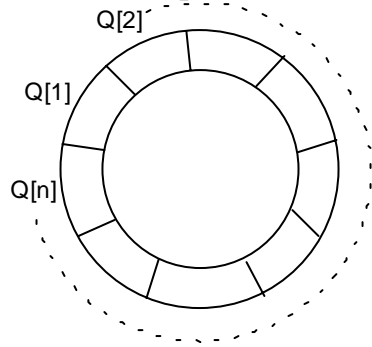
Step 4: Return (Value )



**Circular queue:**

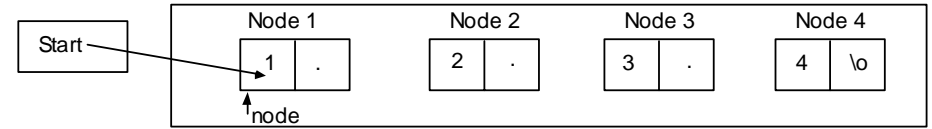
Suppose we have an array Q that contains n element in 50 which Q<sub>1</sub> comes after Q<sub>n</sub> in the array. When this technique is

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.

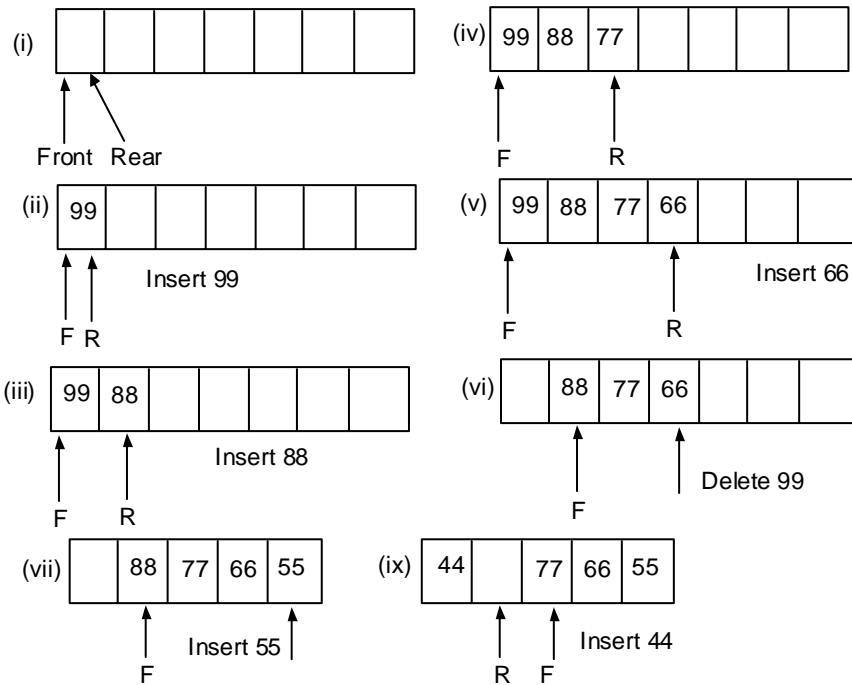
Information part may consist of one or more than one fields. In other words a linked list consist of a series of a structure contains one or more then one contiguous information fields and a pointer to a structure containing its successor. The last node of the list contains NULL ('\0').

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:

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



c. Inset a node between the nodes.

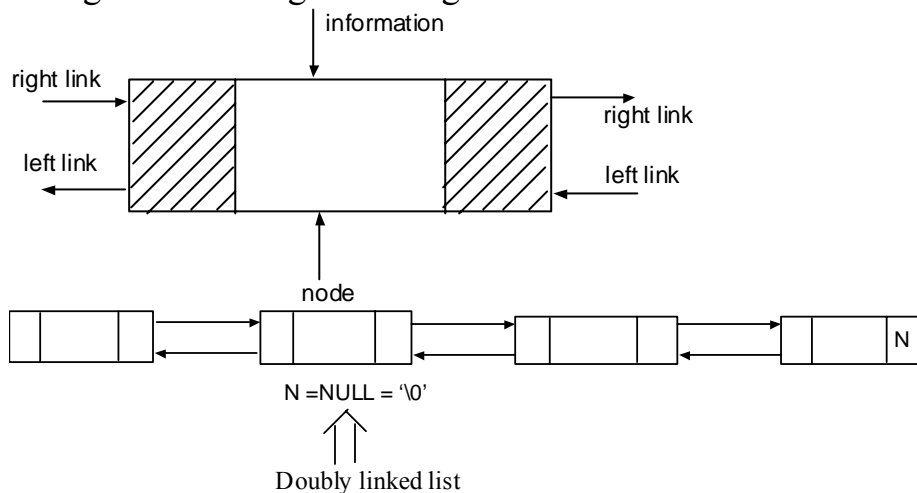
(2) Deletion

- 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

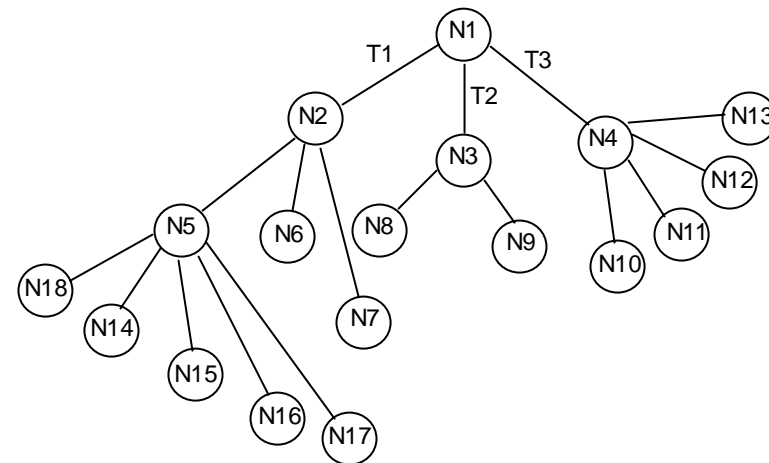


Fig. Tree T having 18 nodes

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

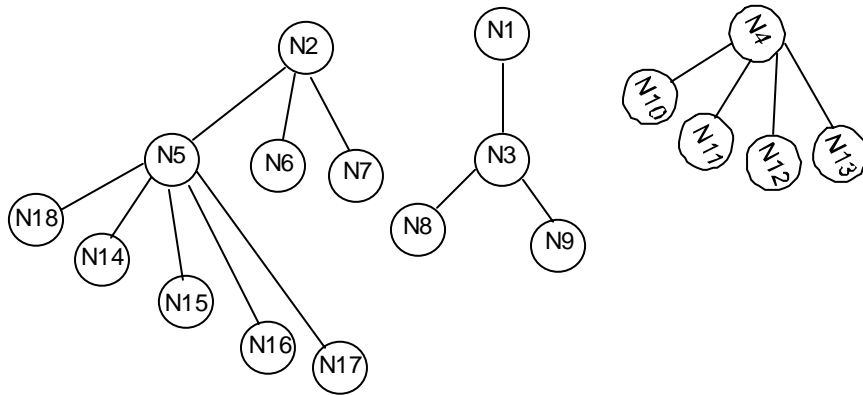
- There is a special node called the root R.
- The remaining nodes are divided into  $n \geq 0$

Disjoint sets  $T_1, T_2, T_3, \dots, T_n$  , where each of these set is a tree .  $T_1, T_2, T_3, \dots, T_n$  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:**

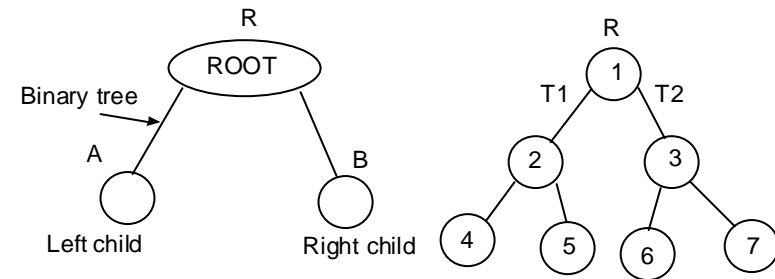


A forest is a set of  $n \geq 0$  disjoint trees, where n represents no of nodes in the tree. The meaning of forest is very close to a tree because if we remove the root of a tree, we get a forest. For e.g in the fig. above we remove N1, we get the forest of three trees. It is shown in the figure above.

**Binary tree:**

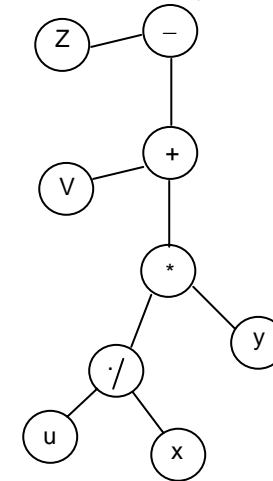
A binary tree T is a finite set of nodes, which is either empty or consists of special nodes called R and two disjoint binary tree T1 and T2 (Which are called the left subtree and right subtree respectively). If T1 is not empty then the root of T1 is called the

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.

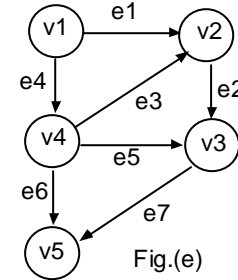
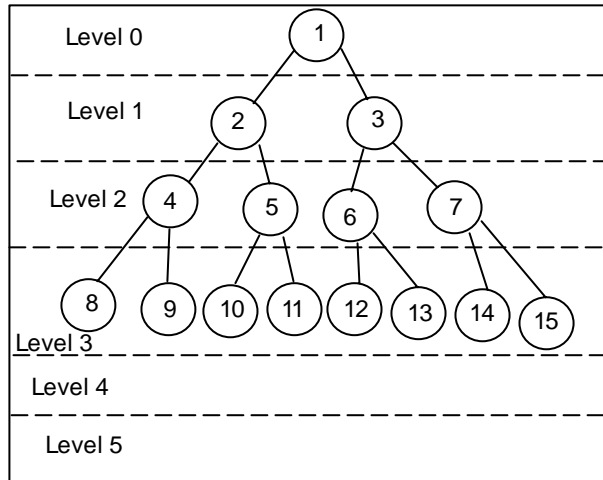


Fig.(e)

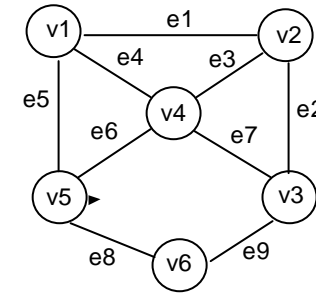


Fig.(d)

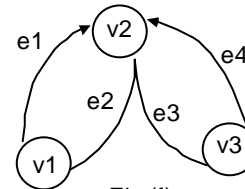


Fig.(f)

### 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  $V$ . For eg. given below shows the different types of graphs.



Fig.(a)

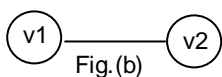


Fig.(b)

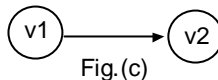
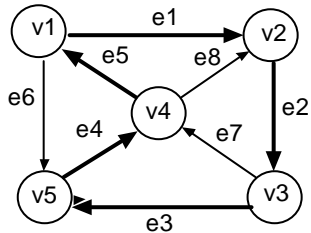


Fig.(c)

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 edges 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 exists 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.

$e_1 = (v1,v2)$

$e_2 = (v2,v3)$

$e_3 = (v3,v4)$

$e_4 = (v4,v5)$

$e_5 = (v5,v1)$

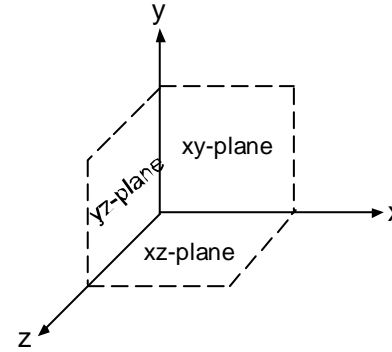
From  $e_1$  and  $e_2$  we find that  $v_2$  is common in both. In the  $e_1$ ,  $v_2$  is the terminal vertex and it is initial vertex of  $e_2$ . Similarly in  $e_2$  and  $e_3$ ,  $v_3$  is corner and so on. Thus we can say that there is cycle and there exist a circular path that is shown by the dark line in the above fig.

Date: 2065/7/24

### Three dimensional Graphics:

#### 6.1 Three Dimensional object to screen perspective viewing

##### Transformation:



The 3D viewing process is inherently more complex than is the 2D viewing process. In 2D we simply 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 complexity 3D-viewing is caused in part by the added dimensions and in part by the fact that display devices are only 2D.

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

In 3D-viewing, we specify a new volume in the world, a projection onto a projection plane, and a viewport on the view surface. Conceptually objects in the 3D-new volume 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 viewport for display). Figure below shows the conceptual model of the 3D-viewing process.

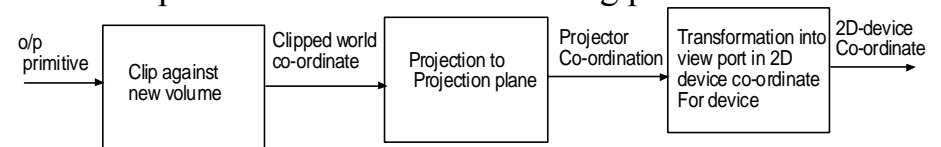


Fig. Conceptual model of 3D-viewing process.

**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 defined by straight projection rays (Called projectors) emanating 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 projectors (rather than curved) then we call it planar-geometric projection.

**Types of Geometric Projection:**

1. Parallel projection
2. Perspective projection.

**Parallel projection:**

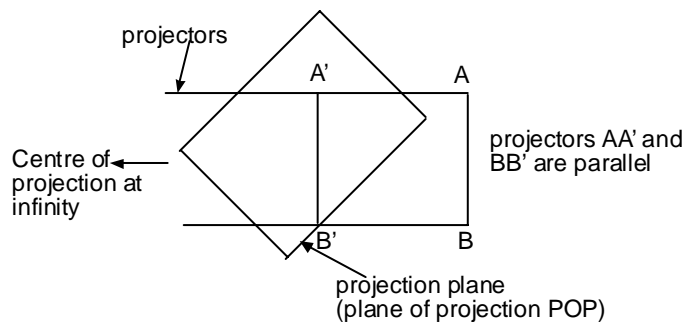


Fig. line AB and its parallel projection A'B'

If the distance between COP and POP is infinite then projection is called parallel projection, i.e the rays from COP are parallel. 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,

$$x = x_1 + (x_2 - x_1)u$$

$$y = y_1 + (y_2 - y_1)u$$

$$z = z_1 + (z_2 - z_1)u$$

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 ,

$$x = x_1 + x_p u$$

$$y = y_1 + y_p u$$

$$z = z_1 + z_p u$$

But on the xy plane

$$0 = z_1 + z_p u$$

$$\therefore u = -z_1/z_p$$

$$x_2 = x_1 - z_1(x_p/z_p)$$

$$y_2 = y_1 - z_1(y_p/z_p)$$



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

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 doesnot 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. Persepctive 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

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

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

$$Z = z_c + (z_1 - z_c)u$$

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

$$\text{i.e } z_c = 0, z_c + (z_1 - z_c)u = 0$$

$$\text{i.e } u = -z_c / (z_1 - z_c)$$

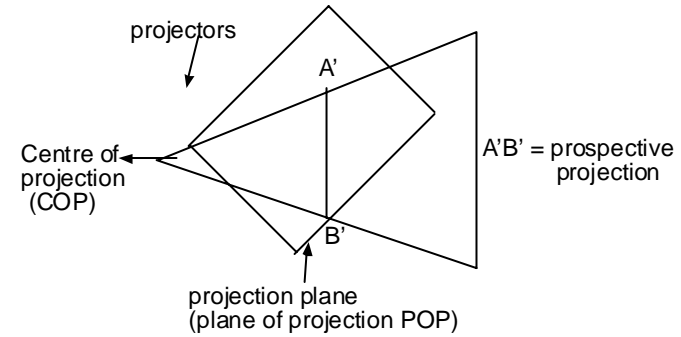
$$x_2 = (x_c z_1 - x_1 z_c) / (z_1 - z_c)$$

$$y_2 = (y_c z_1 - y_1 z_c) / (z_1 - z_c)$$

$$[x_2, y_2, z_2] = [(x_c z_1 - x_1 z_c) / (z_1 - z_c), (y_c z_1 - y_1 z_c) / (z_1 - z_c), 0]$$

In homogenous form (Homogenous co-ordinate system)

$$[x_2, y_2, z_2, 1] = [x_c z_1 - x_1 z_c, y_c z_1 - y_1 z_c, 0, z_1 - z_c]$$



A'B' = perspective projection of AB

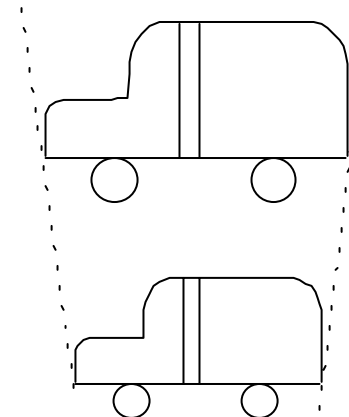


Fig. Same vehicle

In Matrix form ,

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

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

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

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

However while drawing we ignore the z value.

*Date: 2065/7/25*

### Extension of two-dimensional to three dimensional transformation:

Just as 2D-transformation can be represented by 3x3 matrices using homogeneous co-ordinate can be represented by 4x4 matrices, provided we use homogeneous 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 co-ordinate 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,

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

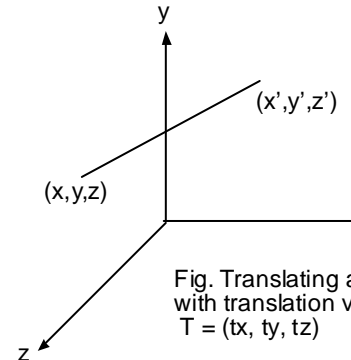


Fig. Translating a point with translation vector  $T = (t_x, t_y, t_z)$

#### 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.

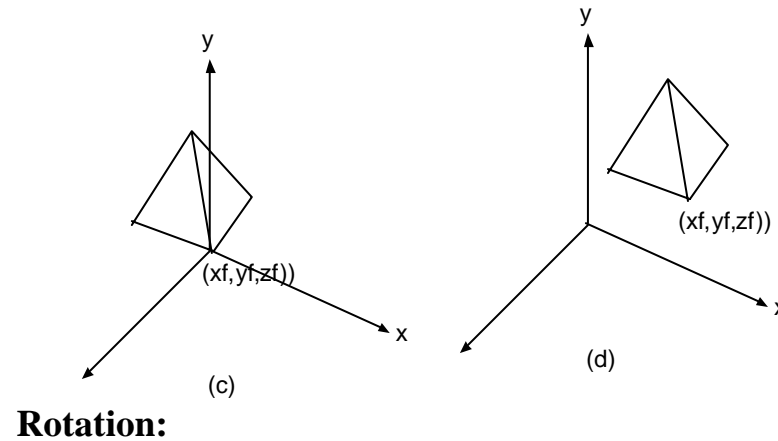
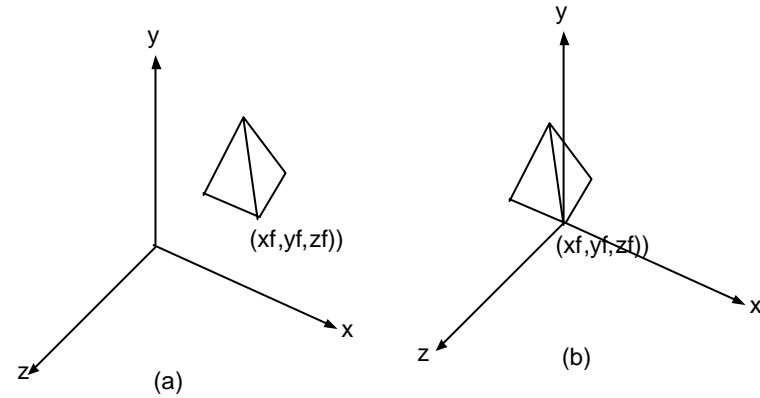
$S(s_x, s_y, s_z)$

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

$T(x_f, y_f, z_f)$

Net transformation

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



**Rotation:**

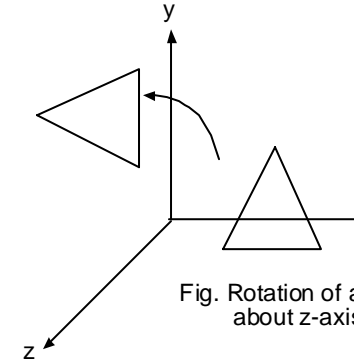
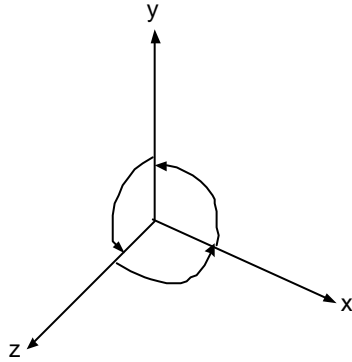


Fig. Rotation of an object about z-axis

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 z-axis 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

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

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

$$Z' = z$$

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

**Rotation about x-axis:**

$$x' = x$$

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

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

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

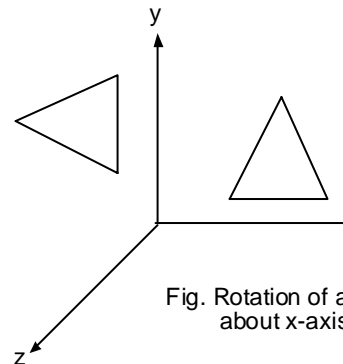


Fig. Rotation of an object about x-axis

**About y-axis:**

$$Y' = y$$

$$z = z \cos \theta - x \sin \theta$$

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

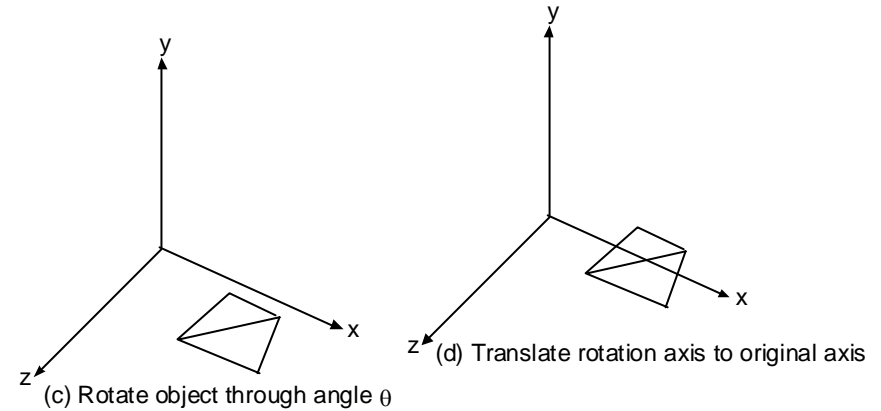
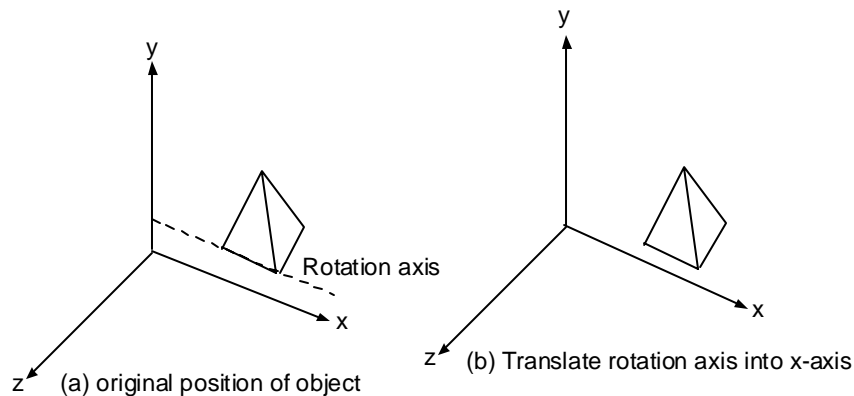
Date: 2065/7/27

### 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,

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



### (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 co-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.

$$\Delta = \vec{p}_2 - \vec{p}_1 = (x_2, y_2, z_2) - (x_1, y_1, z_1) \\ = (x_2 - x_1, y_2 - y_1, z_2 - z_1)$$

Unit vector along rotation axis is

$$\vec{u} = \frac{\vec{v}}{|\vec{v}|} = \frac{(x_2 - x_1, y_2 - y_1, z_2 - z_1)}{\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}} \\ = (a, b, c) \text{ [let]}$$

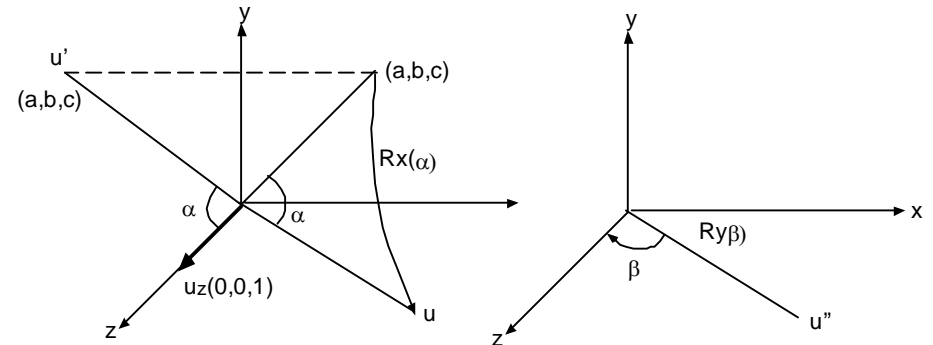
Where (a,b,c) are direction cosines of rotation axis i.e  $a^2 + b^2 + c^2 = 1$ .

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

(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.

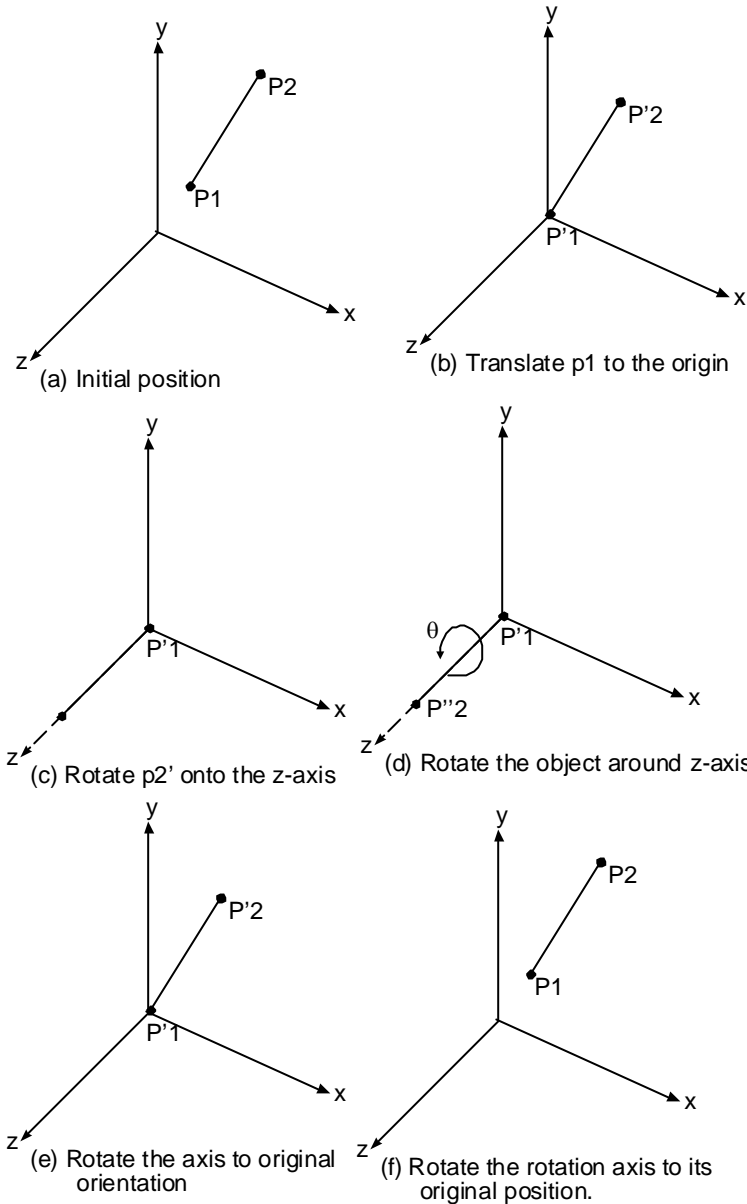
(a) Rotation by angle  $\alpha$  about x axis.

(b) Rotate by angle  $\beta$  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).



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

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

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

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

Steps (b)

U moves to the position u'' after being rotated by  $\alpha$  about x-axis so the position of  $\vec{u}''$  is given by

$$\begin{aligned} \vec{u}'' &= R_x(\alpha) \cdot \vec{u} \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c/d & -b/d & 0 \\ 0 & b/d & c/d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ 1 \end{bmatrix} = \begin{bmatrix} a \\ 0 \\ d \\ 1 \end{bmatrix} \end{aligned}$$

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

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

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

$$R_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\beta & 0 & \cos\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} d/\sqrt{a^2+d^2} & 0 & -a/\sqrt{a^2+d^2} & 0 \\ 0 & 1 & 0 & 0 \\ a/\sqrt{a^2+d^2} & 0 & d/\sqrt{a^2+d^2} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

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)  $R_y^{-1}(\beta)$

(b)  $R_x^{-1}(\alpha)$

Step: 5

$T(x_1, y_1, z_1)$

$\therefore$  Net transformation,

$$= T(x_1, y_1, z_1) \cdot R_x^{-1}(\alpha) \cdot R_y^{-1}(\beta) \cdot R_z(\theta) \cdot R_y(\beta) \cdot R_x(\alpha) \cdot T(-x, -y, -z)$$

Date: 2065/7/28

**Reflection:**

(i) **About Axis:** Reflection about axis is equivalent to  $180^\circ$  rotation about the axis.

**About z-axis:**

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

**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:

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

(iii) About plane:

$$X' = x,$$

$$Y' = y$$

$$Z' = z$$

(a) about xy plane (z-axis)

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

(c) 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.

$$\text{i.e } x' = x + S_{hx} \cdot z$$



$$y' = y + S_{hy} \cdot z$$

$$z' = z$$

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

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 object 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  $60^\circ$  about axis joining the points  $(1,2,3)$  and  $(6,5,7)$ .