COMPUTER GRAPHICS AND MULTIMEDIA

Module 01

Introduction

Computer Graphics is the use of computers to display and manipulate information in graphical or pictorial form, either on a visual-display unit or via a printer or plotter.

Or

Computer graphics are graphics created by computers and, more generally, the representation and manipulation of pictorial data by a computer.

The term computer graphics includes almost everything on computers that is not text or sound. Today nearly all computers use some graphics and users expect to control their computer through icons and pictures rather than just by typing. The term Computer Graphics has several meanings:

- the representation and manipulation of pictorial data by a computer
- the various technologies used to create and manipulate such pictorial data
- the images also produced

2D Computer Graphics

2D computer graphics are the computer-based generation of digital images mostly from two-dimensional models, such as 2D geometric models, text, and digital images, and by techniques specific to them. The word may stand for the branch of computer science that comprises such techniques, or for the models themselves. 2D computer graphics started in the 1950s.

2D computer graphics are mainly used in applications that were originally developed upon traditional printing and drawing technologies, such as typography, cartography, technical drawing, advertising, etc..

3D Computer Graphics

3D computer graphics in contrast to 2D computer graphics are graphics that use a three-dimensional representation of geometric data that is stored in the computer for the purposes of performing calculations and rendering 2D images. Such images may be for later display or for real-time viewing.3D computer graphics are often referred to as 3D models.

Computer Animation

Computer animation is the art of creating moving images via the use of computers. It is a subfield of computer graphics and animation. Increasingly it is created by means of 3D computer graphics, though 2D computer graphics are still widely used for stylistic, low bandwidth, and faster real-time rendering needs. Sometimes the target of the animation is the computer itself, but sometimes the target is another medium, such as film. It is also referred to as CGI (Computer-generated imagery or computer-generated imaging), especially when used in films.

To create the illusion of movement, an image is displayed on the computer screen then quickly replaced by a new image that is similar to the previous image, but shifted slightly. This technique is identical to the illusion of movement in television and motion pictures.

Applications of Computer Graphics

Computer graphics become a power field for the production of pictures. There is no area in which graphical displays can't be used to some advantages, so it is not surprising to find the use of computer graphics so widespread. The applications of computer graphics are:

- Computational biology
- Computational physics
- Computer-aided design
- Computer simulation
- Digital art
- Education
- Graphic design
- Infographics
- Information visualization
- Scientific visualization
- Video Games
- Virtual reality
- Web design

Computational Biology

Computational biology is an interdisciplinary field that applies the techniques of computer science, applied mathematics and statistics to address biological problems. The main focus lays on developing mathematical modeling and computational simulation techniques. By these means it addresses scientific reaserch topics with their theoretical and experimental questions without a laboratory. It encompasses the fields of:

- Bioinformatics: which applies algorithms and statistical techniques to the interpretation, classification and understanding of biological datasets.
- Computational biomodeling: a field within biocybernetics concerned with building computational models of biological systems.
- Computational genomics: a field within genomics which studies the genomes of cells and organisms.
- Molecular modeling: which consists of modelling the behaviour of molecules of biological importance.

Computational physics

Computational physics is the study and implementation of numerical algorithm to solve problems in physics for which a quantitative theory already exists. It is often regarded as a subdiscipline of theoretical physics but some consider it an intermediate branch between theoretical and experimental physics. It includes

- Solving differential equations.
- Evaluating integrals.
- Stochastic methods, especially Monte Carlo methods.
- The pseudo-spectral method.

Computer-Aided Design

Computer-aided design (CAD) is the use of computer technology for the design of objects, real or virtual. The design of geometric models for object shapes, in particular, is often called computer-aided geometric design (CAGD).

CAD may be used to design curves and figures in two-dimensional ("2D") space; or curves, surfaces, or solids in three-dimensional ("3D") objects. CAD is also widely used to produce computer animation for special effects in movies, advertising, technical manuals.

CAD is used in the design of tools and machinery and in the drafting and design of all types of buildings, from small residential types (houses) to the largest commercial and industrial structures (hospitals and factories). CAD is mainly used for detailed engineering of 3D models and/or 2D drawings of physical components, but it is also used throughout the engineering process from conceptual design and layout of products, through strength and dynamic analysis of assemblies to definition of manufacturing methods of components. CAD has become an especially important technology within the scope of computer-aided technologies, with benefits such as lower product development costs and a greatly shortened design cycle. CAD enables designers to lay out and develop work on screen, print it out and save it for future editing, saving time on their drawings.

Computer Simulation

A computer simulation, a computer model or a computational model is a computer program, or network of computers, that attempts to simulate an abstract model of a particular system. Computer simulations have become a useful part of mathematical modelling of many natural systems in physics (computational physics), chemistry and biology, human systems in economics, psychology, and social science and in the process of engineering new technology, to gain insight into the operation of those systems, or to observe their behaviour.

Digital Art

Digital art most commonly refers to art created on a computer in digital form. In advanced, is a term applied to contemporary art that uses the methods of mass production or digital media. The impact of digital technology has transformed traditional activities such as painting, drawing and sculpture, while new forms, such as net art, digital installation art, and virtual reality, have been recognized artistic practices. More generally the term digital artist is used to describe an artist who makes use of digital technologies in the production of art. Digital artists are artists who make digital art using computer graphics software, digital photography technology and computer assisted painting to create art.

Education

A computer simulation, a computer model or a computational model is a computer program, or network of computers, that attempts to simulate an abstract model of a particular system. Computer simulations have become a useful part of mathematical modelling of many natural systems in physics (computational physics), chemistry and biology, human systems in economics, psychology, and social science and in the process of engineering new technology, to gain insight into the operation of those systems, or to observe their behaviour.

Graphic Design

The term graphic design can refer to a number of artistic and professional disciplines which focus on visual communication and presentation. Various methods are used to create and combine symbols, images and/or words to create a visual representation of ideas and messages. A graphic designer may use typography, visual arts and page layout techniques to produce the final result. Graphic design often refers to both the process (designing) by which the communication is created and the products (designs) which are generated. Common uses of graphic design include magazines, advertisements, product packaging and web design.

Information Graphics

Information graphics or infographics are visual representations of information, data or knowledge. These graphics are used where complex information needs to be explained quickly and clearly, such as in signs, maps, journalism, technical writing, and education. They are also used extensively as tools by computer scientists, mathematicians, and statisticians to ease the process of developing and communicating conceptual information. Today information graphics surround us in the media, in published works both pedestrian and scientific, in road signs and manuals. They illustrate information that would be unwieldy in text form, and act as a visual shorthand for everyday concepts such as stop and go.

Information Visualization

Information visualization is the study of the visual representation of large-scale collections of non-numerical information, such as files and lines of code in software systems, and the use of graphical techniques to help people understand and analyze data. In contrast with scientific visualization, information visualization focuses on abstract data sets, such as unstructured text or points in high-dimensional space, that do not have an inherent 2D or 3D geometrical structure.

Scientific Visualization

Scientific visualization is a branch of science, concerned with the visualization of three dimensional phenomena, such as architectural, meteorological, medical, biological systems. The emphasis is on realistic rendering of volumes, surfaces, illumination sources, and so forth, perhaps with a dynamic (time) component. Scientific visualization focuses on the use of computer graphics to create visual images which aid in understanding of complex, often massive numerical representation of scientific concepts or results.

Video Game

A video game is an electronic game that involves interaction with a user interface to generate visual feedback on a video device (a raster display device). The electronic systems used to play video games are known as platforms; examples of these are personal computers and video game consoles. These platforms range from large computers to small handheld devices. The input device used to manipulate video games is called a game controller. Early personal computer games often needed a keyboard for gameplay, or more commonly, required the user to buy a separate joystick with at least one button. Many modern computer games allow, or even require, the player to use a keyboard and mouse simultaneously.

Virtual Reality

Virtual reality (VR) is a technology which allows a user to interact with a computer-simulated environment. Most current virtual reality environments are primarily visual experiences, displayed either on a computer screen or through special or stereoscopic displays, but some simulations include additional sensory information, such as sound through speakers or headphones. Virtual Reality is often used to describe a wide variety of applications, commonly associated with its immersive, highly visual, 3D environments. The development of CAD software, graphics hardware acceleration, head mounted displays, database gloves and miniaturization have helped popularize the notion.

Web Design

Web design is the skill of designing presentations of content (usually hypertext or hypermedia) that is delivered to an end-user through the World Wide Web, by way of a Web browser. The process of designing Web pages, Web sites, Web applications or multimedia for the Web may utilize multiple disciplines, such as animation, authoring, communication design, corporate identity, graphic design, human-computer interaction, information architecture, interaction design, marketing, photography, search engine optimization and typography.

Video Display Devices

A cathode ray tube (CRT) is a specialized vacuum tube in which images are produced when an electron beam strikes a phosphorescent surface. Most desktop computer displays make use of CRTs. The CRT in a computer display is similar to the "picture tube" in a television receiver.

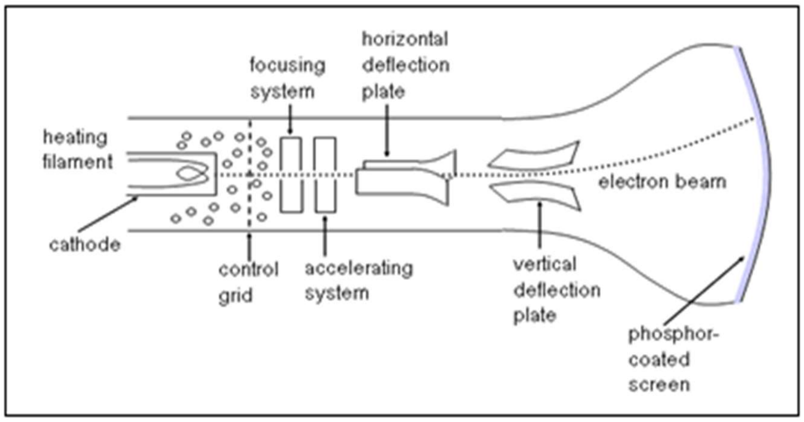
All CRT's have three main elements: an electron gun, a deflection system, and a screen. The electron gun provides an electron beam, which is a highly concentrated stream of electrons. The deflection system positions the electron beam on the screen, and the screen displays a small spot of light at the point where the electron beam strikes it.

Refresh CRT

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

Basic Operation of a CRT

The basic operation of CRT is shown in figure below:



Electron Gun

The primary components of an electron gun in a CRT are the heated metal cathode and a control grid. The cathode is heated by an electric current passed through a coil of wire called the filament. This causes electrons to be boiled off the hot cathode surface. In the vacuum inside the CRT envelope, negatively charged electrons are then accelerated toward the phosphor coating by a high positive voltage. The accelerating voltage can be generated with a positively charged metal coating on the inside of the CRT envelope near the phosphor screen, or an accelerating anode can be used. Sometimes the electron gun is built to contain the accelerating anode and focusing system within the same unit.

Focusing System

The focusing system is used to create a clear picture by focusing the electrons into a narrow beam. Otherwise, electrons would repel each other and beam would spread out as it reaches the screen. Focusing is accomplished with either electric or magnetic fields.

Deflection System

Deflection of the electron beam can be controlled by either electric fields or magnetic fields. In case of magnetic field, two pairs of coils are used, one for horizontal deflection and other for vertical deflection. In case of electric field, two pairs of parallel plates are used, one for horizontal deflection and second for vertical deflection as shown in figure above.

CRT Screen

The inside of the large end of a CRT is coated with a fluorescent material that gives off light when struck by electrons. When the electrons in the beam is collides with phosphor coating screen, they stopped and their kinetic energy is absorbed by the phosphor. Then a part of beam energy is converted into heat energy and the remainder part causes the electrons in the phosphor atom to move up to higher energy levels. After a short time the excited electrons come back to their ground state. During this period, we see a glowing spot that quickly fades after all excited electrons are returned to their ground state.

Persistence

It is defined as the time they continue to emit light after the CRT beam is removed. Persistence is defined as the time it takes the emitted light from the screen to decay to one-tenth of its original intensity. Lower-persistence phosphors require higher refresh rates to maintain a picture on the screen without flicker. A phosphor with low persistence is useful for animation; a high-persistence phosphor is useful for displaying highly complex, static pictures.

Resolution

The number of points per centimetre that can be used be plotted horizontally and vertically. Or Total number of points in each direction.

The resolution of a CRT is depending on

- type of phosphor
- intensity to be displayed
- focusing and deflection system

Aspect Ratio

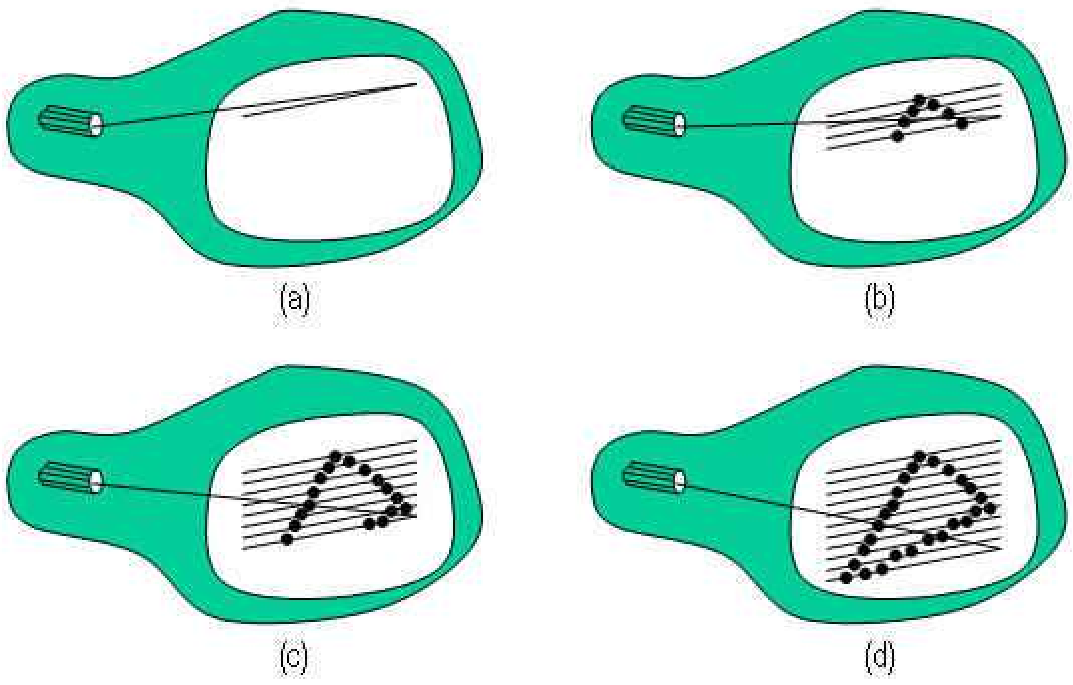
It is ratio of horizontal to vertical points.

Example: An aspect ratio of 3/4 means that a vertical line plotted with three points has same length as horizontal line plotted with four points.

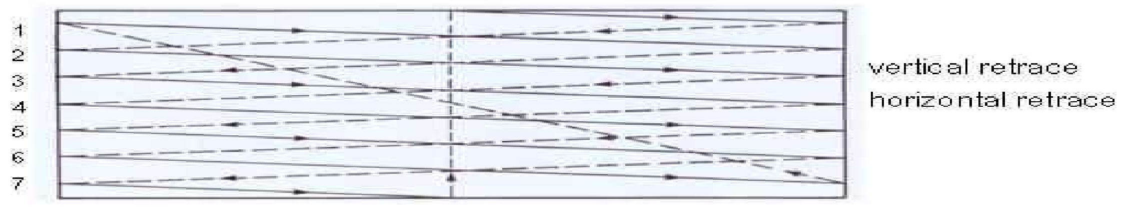
Display Technology

A. Raster Scan Systems

It is the most common type of graphics monitor based on television technology. In a raster scan system, the electron beam is swept across the screen, one row at a time from top to bottom. When electron beam moves across each row the beam intensity is turned ON and OFF to create a pattern of illuminated spots. Picture definition is stored in a memory called frame buffer which holds the set of intensity values, which are then retrieved from the frame buffer and pointed on the screen one row at a time as shown in figure below:



At the end of each line the beam must be turned off and redirect to the left hand side of the CRT, this is called Horizontal Retrace. At the end of each frame, the electron beam return to top left corner of the screen to begin the next frame called Vertical Retrace as shown in figure below:



Advantages

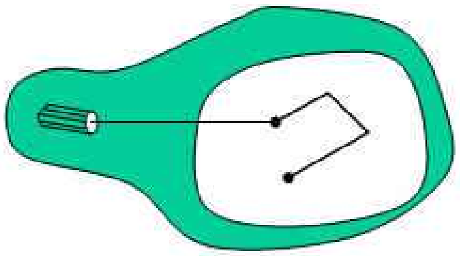
- produce realistic images
- also produced different colors
- and shadows scenes.

Disadvantages

- low resolution
- expensive
- electron beam directed to whole screen

B. Random Scan Systems

In Random Scan System, an electron beam is directed to only those parts of the screen where a picture is to be drawn. The picture is drawn one line at a time, so also called vector displays or stroke writing displays. After drawing the picture the system cycles back to the first line and design all the lines of the picture 30 to 60 time each second.



Advantages

- Produced smooth line drawings
- High resolution

Disadvantages

- Designed only for line drawing applications.
- Can't display realistic images.

C. Flat panel displays

Flat-panel display refers to a class of video devices that have reduced volume (thinner), weight and power consumption compared to CRT. These emerging display technologies tend to replace CRT monitors. Current uses of flat-panel displays include TV monitors, calculators, pocket video games, laptops, displays in airlines and ads etc.

Two categories of flat-panel displays:

a) **Emissive displays:** act as a light source themselves and convert electrical energy into light. Example: Plasma panels, electroluminescent displays and light-emitting diodes.
b) **Non-emissive displays:** need an external light source to function i.e. use optical effects to convert sunlight or light from other sources into graphics patterns. Example: liquid-crystal displays.

D. Direct View Storage Tube (DVST)

This is alternative method for method maintaining a screen image to store picture information inside the CRT **instead of refreshing** the system.

- ➢ DVST stores the picture information as a charge distribution just behind the phosphor-coated screen.
- ➢ Two electron guns used: primary gun – to store picture pattern and flood gun – maintains the picture display.
- ➢ Pros: Since no refreshing is needed complex pictures can be displayed in high-resolution without flicker.
- ➢ Cons: Ordinarily do not display color and that selected parts of picture can not be erased. To eliminate a picture section, entire screen must be erased and modified picture redrawn, which may take several seconds for complex picture.

Color CRT Monitors

A color CRT monitor displays color picture by using a combination of phosphors that emit different colored light. By combining the emitted light a range of colors can be generated. Two basic methods for producing color displays are:

- Beam Penetration Method
- Shadow-Mask Method

Beam Penetration Method

Random scan monitors use the beam penetration method for displaying color picture. In this, the inside of CRT screen is coated two layers of phorphor namely red and green. A beam of slow electrons excites only the outer red layer, while a beam of fast electrons penetrates red layer and excites the inner green layer. At intermediate beam speeds, combination of red and green light are emitted to show two additional colors- orange and yellow.

Advantages

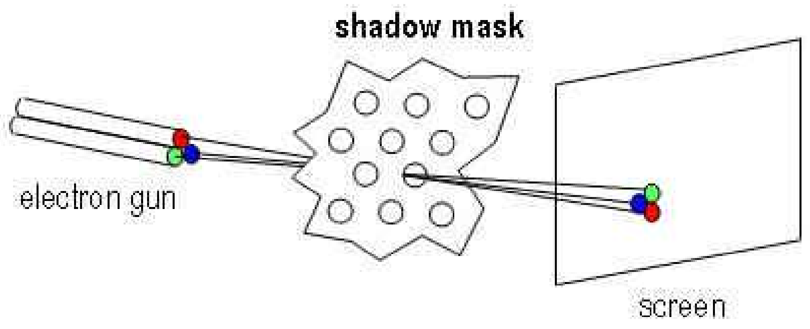
- Less expensive

Disadvantages

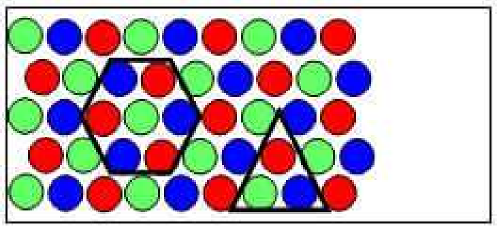
- Quality of images are not good as comparable with other methods
- Four colors are allowed only

Shadow Mask Method

Raster scan system are use shadow mask methods to produced a much more range of colors than beam penetration method. In this, CRT has three phosphor color dots. One phosphor dot emits a red light, second emits a green light and third emits a blue light. This type of CRT has three electrons guns and a shadow mask grid as shown in figure below:



In this figure, three electrons beams are deflected and focused as a group onto the shadow mask which contains a series of holes. When three beams pass through a hole in shadow mask they activate dot triangle as shown in figure below:



phosphor dot pattern

The colors we can see depends on the amount of excitation of red, green and blue phosphor. A white area is a reasult of all three dots with equal intensity while yellow is produced with green and red dots and so on.

Advantages

- produce realistic images
- also produced different colors
- and shadows scenes.

Disadvantages

- low resolution
- expensive
- electron beam directed to whole screen

Architecture of Raster-Scan System

The raster graphics systems typically consist of several processing units. Besides CPU, graphics system consists of a special purpose processor called video controller or display processor (DP). The organization of raster system is as shown below:

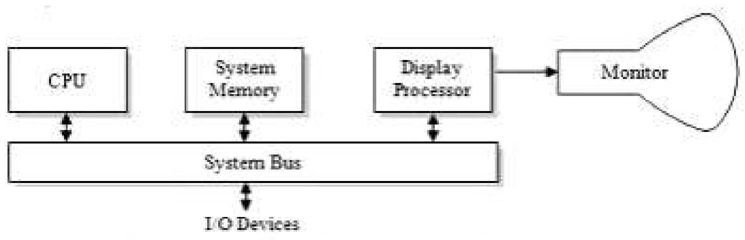
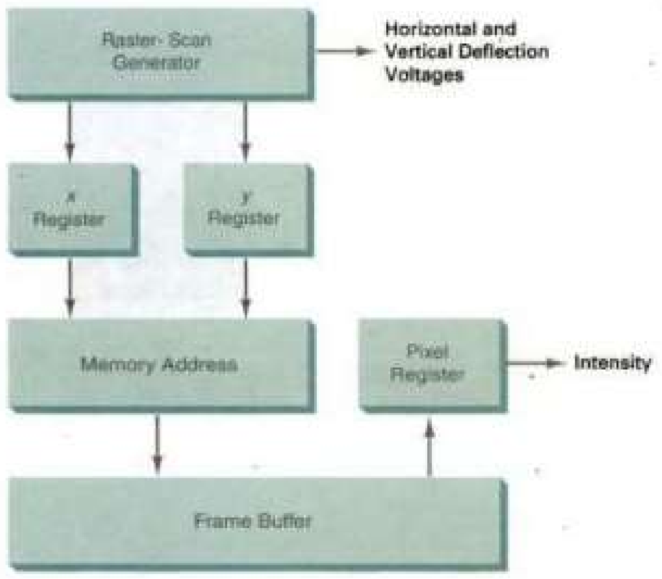


Fig: Architecture of simple raster-graphics system

- A fixed area of system memory is reserved for the frame buffer. The video controller has the direct access to the frame buffer for refreshing the screen.
- The video controller cycles through the frame buffer, one scan line at a time, typically at 60 times per second or higher. The contents of frame buffer are used to control the CRT beam's intensity or color.

The Video Controller

It is a special-purpose processor used to control the operation of the display device. The basic video-controller refresh operation is diagrammed below.



Raster Scan Display Processor

The raster scan with a peripheral display processor is a common architecture that avoids the disadvantage of simple raster scan system. It includes a separate graphics processor to perform graphics functions such as scan conversion and raster operation and a separate frame buffer for image refresh. The display processor has its own separate memory called display processor memory.

- System memory holds data and those programs that execute on the CPU, and the application program, graphics packages and OS.
- The display processor memory holds data plus the program that perform scan conversion and raster operations.
- The frame buffer stores displayable image created by scan conversion and raster operations.

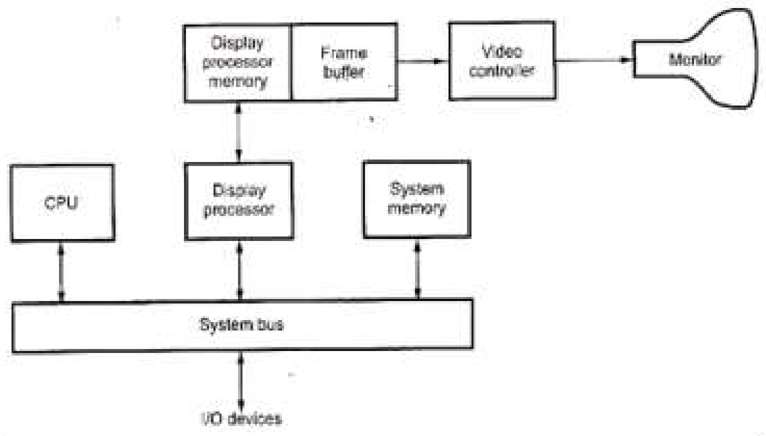


Fig: Architecture of a raster-graphics system with a display processor

Architecture of Random-Scan(Vector)System

The organization of simple vector system shown in the figure below:

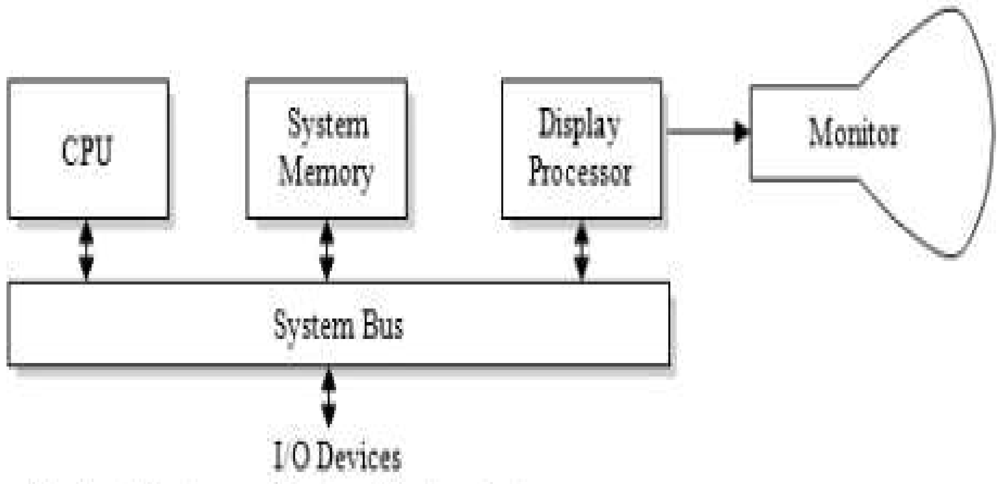


Fig: Architecture of Vector Display System

- Vector display system consists of several units along with peripheral devices. The display processor is also called as graphics controller.
- Graphics package creates a display list and stores in systems memory (consists of points and line drawing commands) called display list or display file.
- Vector display technology is used in monochromatic or beam penetration color CRT.
- Graphics are drawn on a vector display system by directing the electron beam along component line.

Advantages:

- Can produce output with high resolutions.
- Better for animation than raster system since only end point information is needed.

Disadvantages:

- Cannot fill area with pattern and manipulate bits.
- Refreshing image depends upon its complexity.

Input Devices

A piece of computer hardware that is used to enter and manipulate information on a computer.

Basic input devices include the

- Keyboard
- Mouse
- Digitizer
- Trackball
- Touch Screens
- Light Pens
- Microphones
- Bar code readers
- Joysticks
- Scanners
- Voice Systems

Keyboard

The keyboard is the most common input device for entering numeric and alphabetic data in to a computer system by pressing a set of keys which are mounted on the keyboard, which is connected to computer system.

The keys on computer keyboards are often classified as follows:

Alphanumeric Keys - letters and numbers.

Punctuation Keys - comma, period, semicolon, and so on.

Special Keys - function keys, control keys, arrow keys, Caps Lock key, and so on.

Mouse

A mouse is a small device that a computer user pushes across a desk surface in order to point to a place on a display screen and to select one or more actions to take from that position.

A mouse consists of a metal or plastic housing or casing, a ball that sticks out of the bottom of the casing and is rolled on a flat surface, one or more buttons on the top of the casing, and a cable that connects the mouse to the computer. As the ball is moved over the surface in any direction, a sensor sends impulses to the computer that causes a mouse-responsive program to reposition a visible indicator (called a cursor) on the display screen. The positioning is relative to some variable starting place. Viewing the cursor's present position, the user readjusts the position by moving the mouse.

Digitizer

A graphics tablet (or digitizing tablet, graphics pad, drawing tablet) is a computer input device that allows one to hand-draw images and graphics, similar to the way one draws images with a pencil and paper. These tablets may also be used to capture data or handwritten signatures.

A graphics tablet (also called pen pad or digitizer) consists of a flat surface upon which the user may "draw" an image using an attached stylus, a pen-like drawing apparatus. The image generally does not appear on the tablet itself but, rather, is displayed on the computer monitor.

Light Pen

A Light Pen is a pointing device shaped like a pen and is connected to a VDU. The tip of the light pen contains a light-sensitive element which, when placed against the screen, detects the light from the screen enabling the computer to identify the location of the pen on the screen. Light pens have the advantage of 'drawing' directly onto the screen, but this can become uncomfortable, and they are not as accurate as digitizing tablets.

Touch Screen

A touchscreen is a display which can detect the presence and location of a touch within the display area. The term generally refers to touch or contact to the display of the device by a finger or hand. Touchscreens can also sense other passive objects, such as a stylus. However, if the object sensed is active, as with a light pen, the term touchscreen is generally not applicable. The ability to interact directly with a display typically indicates the presence of a touchscreen.

The touchscreen has two main attributes. First, it enables one to interact with what is displayed directly on the screen, where it is displayed, rather than indirectly with a mouse or touchpad. Secondly, it lets one do so without requiring any intermediate device, again, such as a stylus that needs to be held in the hand. Such displays can be attached to computers or, as terminals, to networks. They also play a prominent role in the design of digital appliances such as the personal digital assistant (PDA), satellite navigation devices, mobile phones, and video games.

Image Scanners

A scanner is a device that optically scans images, printed text, handwriting, or an object, and converts it to a digital image. Common examples found in offices are variations of the desktop (or flatbed) scanner where the document is placed on a glass window for scanning. Hand-held scanners, where the device is moved by hand, have evolved from text scanning "wands" to 3D scanners used for industrial design, reverse engineering, test and measurement, orthotics, gaming and other applications.

Voice Systems

voice input device A device in which speech is used to input data or system commands directly into a system. Such equipment involves the use of speech recognition processes, and can replace or supplement other input devices. Some voice input devices can recognize spoken words from a predefined vocabulary, some have to be trained for a particular speaker.

Speech recognition (also known as automatic speech recognition or computer speech recognition) converts spoken words to machine-readable input (for example, to key presses, using the binary code for a string of character codes).

Joystick

A joystick is an input device consisting of a stick that pivots on a base and reports its angle or direction to the device it is controlling. Joysticks are often used to control video games, and usually have one or more push-buttons whose state can also be read by the computer. A popular variation of the joystick used on modern video game consoles is the analog stick.

Trackball

A trackball is a pointing device consisting of a ball held by a socket containing sensors to detect a rotation of the ball about two axes—like an upside-down mouse with an exposed protruding ball. The user rolls the ball with the thumb, fingers, or the palm of the hand to move a cursor. Large tracker balls are common on CAD workstations for easy precision.

Hard Copy Devices

Hard copy device or output devices accept data from a computer and converted them into a form which is suitable for use by the user.

Basic output devices include the

- Monitors
- Printers
- Plotters

Printer

Printers are the most commonly used output devices for producing hard copy output.

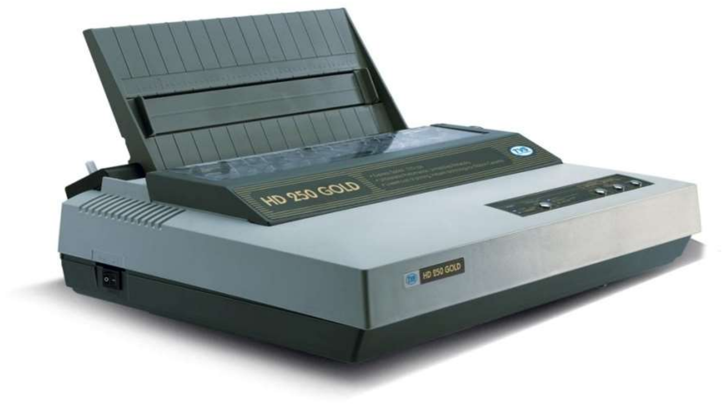
The various types of printers in used today are

- Dot-Matrix Printers
- Inkjet Printers
- Drum Printers
- Laser Printers

Dot-Matrix Printers

A dot matrix printer or impact matrix printer is a type of computer printer with a print head that runs back and forth, or in an up and down motion, on the page and prints by impact, striking an ink-soaked cloth ribbon against the paper, much like a typewriter. Unlike a typewriter or daisy wheel printer, letters are drawn out of a dot matrix, and thus, varied fonts and arbitrary graphics can be produced. Because the printing involves mechanical pressure, these printers can create carbon copies and carbonless copies.

Most dot matrix printers have a single vertical line of dot-making equipment on their print heads; others have a few interleaved rows in order to improve dot density.



Inkjet Printers

Inkjet printers form characters and images by spraying small drops of ink on to the paper. They are the most common type of computer printer for the general user due to their low cost, high quality of output, capability of printing in different colors, and ease of use.

If you ever look at a piece of paper that has come out of an inkjet printer, you know that:

- The dots are extremely small, so small that they are tinier than the diameter of a human hair (70 microns).
- The dots are positioned very precisely, with resolutions of up to 1440x720 dots per inch.
- The dots can have different colors combined together to create photo-quality images.

Laser Printers

A type of printer that utilizes a laser beam to produce an image on a drum. The light of the laser alters the electrical charge on the drum wherever it hits. The drum is then rolled through a reservoir of toner, which is picked up by the charged portions of the drum. Finally, the toner is transferred to the paper through a combination of heat and pressure. This is also the way copy machines work.

Because an entire page is transmitted to a drum before the toner is applied, laser printers are sometimes called page printers. There are two other types of page printers that fall under the category of laser printers even though they do not use lasers at all. One uses an array of LEDs to expose the drum, and the other uses LCDs. Once the drum is charged, however, they both operate like a real laser printer.

One of the chief characteristics of laser printers is their resolution -- how many dots per inch (dpi) they lay down. The available resolutions range from 300 dpi at the low end to 1,200 dpi at the high end. By comparison, offset printing usually prints at 1,200 or 2,400 dpi. Some laser printers achieve higher resolutions with special techniques known generally as resolution enhancement.

Laser printers produce very high-quality print and are capable of printing an almost unlimited variety of fonts.

Plotters

A plotter is a printer that interprets commands from a computer to make line drawings on paper with one or more automated pens. Unlike a regular printer, the plotter can draw continuous point-to-point lines directly from vector graphics files or commands. There are a number of different types of plotters: a drum plotter draws on paper wrapped around a drum which turns to produce one direction of the plot, while the pens move to provide the other direction; a flatbed plotter draws on paper placed on a flat surface; and an electrostatic plotter draws on negatively charged paper with positively charged toner. Plotters were the first type of printer that could print with color and render graphics and full-size engineering drawings. As a rule, plotters are much more expensive than printers. They are most frequently used for CAE (computer-aided engineering) applications, such as CAD (computer-aided design) and CAM (computer-aided manufacturing).

A plotter consists of an arm that moves across the paper on which the diagram or graph needs to be drawn. A pen moves along the arm. and the arm itself moves relative to the paper. A combination of the two thus provides movement along the horizontal and vertical axes.

Plotter are used in applications like CAD, which require high-quality graphics on paper. Many of the plotters now available in the market are desktop models that can be used with PCs. Businesses typically use plotters to present an analysis in terms (bar charts, graphs, diagrams, etc.) as well as for engineering drawings.

Two commonly type plotters are:

- Drum Plotters
- Flatbed Plotters

Drum Plotters

In this plotter the pen moves in the horizontally and the drum rolls on the other axis. Generally it is a graphical output device, and it is generally used to plot drawings. the width of the plot is limited by the length of the drum. It is the first graphical output device produced to print large scaled engineering drawings. Colored prints can be made if you use colored ink. This electronic equipment used to plot large sized drawings on a tracing sheet. Plans for building (architectural drawing), engineering drawings, dress models ETC are drawn using suitable packages. These drawings are plotted on a tracing sheet, which can be used to produce a large no of blue prints.

It is very good for line drawings but it is very slow. Here the pen is held in gantry and it moves in x-axis and the paper moves on y-axis.

Flatbed Plotters

This is a plotter where the paper is fixed on a flat surface and pens are moved to draw the image. This plotter can use several different colour pens to draw with. The size of the plot is limited only by the size of the plotter's bed.

Output primitives

Output primitives are the geometric structures such as straight line segments (pixel array) and polygon color areas, used to describe the shapes and colors of the objects. Points and straight line segments are the simplest geometric components of pictures. Additional output primitive includes: circles and other conic sections, quadric surfaces, spline curves and surfaces, polygon color areas and character strings. Here, we discuss picture generation algorithm by examining device-level algorithms for displaying two dimensional output primitives, with emphasis on scan-conversion methods for raster graphics system.

Points and Lines

Point plotting is accomplished by converting a single coordinate position furnished by an application program into appropriate operations for the output device. With a CRT monitor, for example, the electron beam is turned on to illuminate the screen phosphor at the selected location

Line drawing is accomplished by calculating intermediate positions along the line path between two specified end points positions. An output device is then directed to fill in these positions between the end points

Pixel positions are referenced according to scan-line number and column number (pixel position across a scan line). Scan lines are numbered consecutively from 0, starting at the bottom of the screen; and pixel columns are numbered from 0, left to right across each scan line

To load an intensity value into the frame buffer at a position corresponding to column x along scan line y, set pixel (x, y)

To retrieve the current frame buffer intensity setting for a specified location we use a low level function get pixel (x, y)

Line Drawing Algorithms

The Cartesian slope-intercept equation of a straight line is

$$y = mx + b \quad \text{........... (1), where m = slope of line and b = y-intercept.}$$

For any two given points (x_1, y_1) and (x_2, y_2) of a line segment, we can determine *m* and *b* with following calculations:

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

$\therefore$ (1) becomes,

$$b = y - \frac{y_2 - y_1}{x_2 - x_1}x$$

At any point (x_k, y_k),

$$y_k = mx_k + b \quad \text{.......................... (2)}$$

At (x_{k+1}, y_{k+1}),

$$y_{k+1} = mx_{k+1} + b \text{ .... (3)}$$

Subtracting (2) from (3) we get,

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

Here ($y_{k+1} - y_k$) is increment in y as corresponding increment in x.

$$\therefore \Delta y = m.\Delta x$$

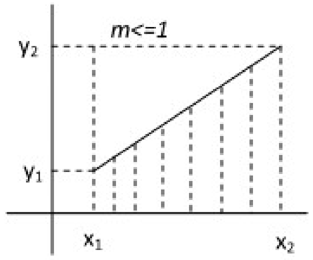
$$\text{or } m = \frac{\Delta y}{\Delta x}$$

DDA Line Algorithm (Incremental Algorithm)

The digital differential analyzer (DDA) is a scan conversion line drawing algorithm based on calculating either Δx or Δy form the equation,

$$\Delta y = m.\Delta x$$

We sample the line at unit intervals in one co-ordinate and determine the corresponding integer values nearest to the line path for the other co-ordinates.

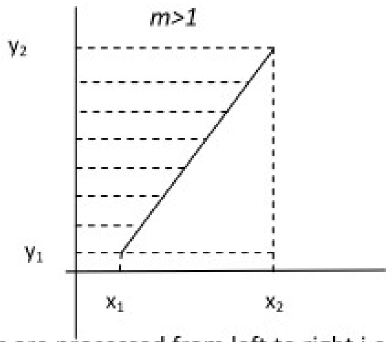


Consider first the line with **positive slope**.

For m<=1, we sample x co-ordinate. So $\Delta x = 1$ and compute each successive *y* value as:

$$y_{k+1} = y_k + m \quad .... (4) \because m = \frac{\Delta y}{\Delta x}, \Delta x = 1$$

Here *k* takes value from starting point and increase by 1 until final end point. *m* can be any real value between 0 and 1.



For m > 1, we sample y coordinate. So $\Delta y = 1$ and calculate corresponding x values as:

$$x_{k+1} = x_k + \frac{1}{m} \quad .... (5) \because m = \frac{\Delta y}{\Delta x}, \Delta y = 1$$

The above equations are under the assumption that the lines are processed from left to right i.e. left end point is starting. If the processing is from right to left, then either we have $\Delta x = -1$ and

$$x_{k+1} = x_k - \frac{1}{m} .... (6)$$

or (m > 1) we have $\Delta y = -1$ with

$$y_{k+1} = y_k - m .... (7)$$

Equations (4) through (7) can also be used to calculate pixel positions along a line with **negative slope**.

- If |m|<1
  - Start endpoint is at left: we set $\Delta x = 1$ and calculate y values with (4).
  - Start endpoint is at right: we set $\Delta x = -1$ and calculate y values with (6).
- If |m|>1
  - Start endpoint is at left: we set $\Delta y = 1$ and calculate x values with (5).
  - Start endpoint is at right: we set $\Delta y = -1$ and calculate x values with (7).

The DDA algorithm is faster method for calculating pixel position than direct use of line equation but it has problems:

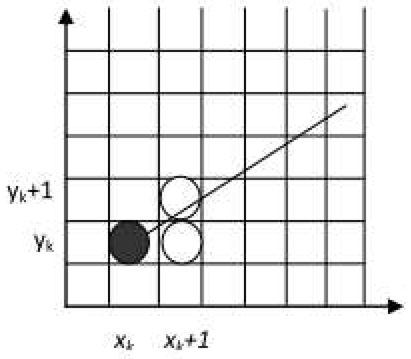
- Accumulation of roundoff error in successive additions can cause calculated pixel positions to drift away from the actual line path for long line segments.
- Rounding operations and floating-point-arithmetic are time consuming.

Brenham's Line Algorithm

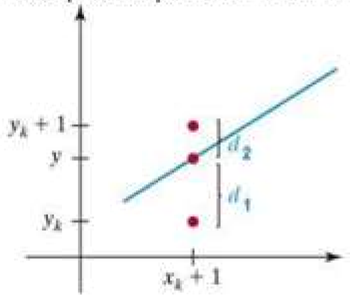
An accurate and efficient line generating algorithm, developed by Bresenham that scan converts lines only using integer calculation to find the next *(x, y)* position to plot. It avoids incremental error accumulation.

Line with positive slope less than 1 (0<m<1)
Pixel position along the line path is determined by sampling at unit x intervals. Starting from left end point, we step to each successive column and plot the pixel closest to line path.

Assume that (x_k, y_k) is pixel at kth step then next point to plot may be either (x_k+1, y_k) or (x_k+1, y_k+1).



At sampling position x_k+1, we label vertical pixel separation from line path as d_1 & d_2 as in figure.



The y-coordinate on the mathematical line path at pixel column x_k+1 is $y = m(x_k+1)+b$.

Then $d_1 = y - y_k = m(x_k + 1) + b - y_k$

$d_2 = (y_k + 1) - y = (y_k + 1) - m(x_k + 1) - b$

Now $d_1 - d_2 = 2m(x_k + 1) - (y_k + 1) - y_k + 2b = 2m(x_k + 1) - 2y_k + 2b - 1$

A decision parameter p_k for the kth step in the line algorithm can be obtained by rearranging above equation so that it involves only integer calculations. We accomplish this by substituting $m = \frac{\Delta y}{\Delta x}$ in above eqn and defining

$$p_k = \Delta x(d_1 - d_2) = \Delta x[2\frac{\Delta y}{\Delta x}(x_k + 1) - 2y_k + 2b - 1] = 2\Delta y.x_k - 2\Delta x.y_k + c$$

Where the constant $c = 2\Delta y - \Delta x(2b - 1)$ which is independent of the pixel position. Also, sign of p_k is same as the sign of $d_1 - d_2$.

If decision parameter p_k is negative i.e. $d_1 < d_2$, pixel at y_k is closer to the line path than pixel at y_k+1. In this case we plot lower pixel *(x_k+1 , y_k), otherwise plot upper pixel (x_k+1, y_k+1).*

Co-ordinate change along the line occur in unit steps in either x, or y direction. Therefore we can obtain the values of successive decision parameters using incremental integer calculations.
At step *k+1*, decision parameter p_{k+1} *is* evaluated as.

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

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

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

The term $y_{k+1} - y_k$ is either 0 or 1 depending upon the sign of p_k.

The first decision parameter p_0 is evaluated as $p_o = 2\Delta y - \Delta x$

and successively we can calculate decision parameter as $p_{k+1} = p_k + 2\Delta y - 2\Delta x(y_{k+1} - y_k)$

So if p_k is negative, $y_{k+1} = y_k$ so $p_{k+1} = p_k + 2\Delta y$

Otherwise $y_{k+1} = y_k + 1$, *then* $p_{k+1} = p_k + 2\Delta y - 2\Delta x$

Algorithm:

1. Input the two line endpoint and store the left endpoint at (x_o, y_o)
2. Load (x_o, y_o) in to frame buffer, i.e. Plot the first point.
3. Calculate constants $2\Delta x, 2\Delta y$ calculating $\Delta x, \Delta y$ and obtain first decision parameter value as

   $p_o = 2\Delta y - \Delta x$
4. At each x_k along the line, starting at k=0, perform the following test,

   if $p_k < 0$, next point is $(x_k + 1, y_k)$

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

   otherwise

   next point to plot is $(x_k + 1, y_k + 1)$

   $p_{k+1} = p_k + 2\Delta y - 2\Delta x$
5. Repeat step 4 Δx times.

Midpoint Circle Algorithm

In mid point circle algorithm, we sample at unit intervals and determine the closest pixel position to the specified circle path at each step.

For a given radius r, and screen center position (xc, yc), we can first set up our algorithm to calculate pixel positions around a circle path centered at (0, 0) and then each calculated pixel position (x, y) is moved to its proper position by adding xc to x and yc to y

i.e. x = x + xc, y = y + yc.

To apply the midpoint method, we define a circle function as:

$$f_{circle} = x^2 + y^2 - r^2$$

To summarize the relative position of point (x, y) by checking sign of f_{circle} function,

$$f_{circle}(x, y) \begin{cases} <0, \text{ if } (x, y) \text{ lies inside the circle boundary} \\ =0, \text{ if } (x, y) \text{ lies on the circle boundary} \\ >0, \text{ if } (x, y) \text{ lies outside the circle boundary.} \end{cases}$$

Steps of Mid-Point Circle drawing algorithm

1. Input radius r and circle centre (x_c, y_c) and obtain the first point on circle centered at origin as.

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

2. Calculate initial decision parameter

   $p_0 = 5/4 - r$

3. At each x_k position, starting at $k = 0$, perform the tests:

   If $p_k < 0$ next point along the circle centre at $(0,0)$ is $(x_k + 1, y_k)$

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

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

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

   Where $2x_{k+1} = 2x_k + 2.$ and $2y_{k+1} = 2y_k - 2.$

4. Determine symmetry point on the other seven octants.
5. Move each calculated pixels positions (x, y) in to circle path centered at (x_c, y_c) as

   $x = x + x_c, y = y + y_c$

6. Repeat 3 through 5 until $x \geq y$.

Ellipse drawing algorithm

1. Input center (xc, yc) and r_x and r_y for the ellipse and obtain the first point as $(x_0, y_0) = (0, r_y)$
2. Calculate initial decision parameter value in Region 1 as

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

3. At each x_k position, in Region 1, starting at k = 0 , compute $x_{k+1} = x_k + 1$

   If $p_{1k} < 0$, then the next point to plot is

   $p_{1k+1} = p_{1k} + 2r_y^2 x_{k+1} + r_y^2$

   $y_{k+1} = y_k$

   Otherwise next point to plot is

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

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

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

   $p_{20} = r_y^2 (x_0 + \frac{1}{2})^2 + r_x^2 (y_0 - 1)^2 - r_x^2 r_y^2$

5. At each y_k position in Region 2 starting at k = 0, perform computation y_{k+1} = y-1;

   if $p_{2k} > 0$, then

   $x_{k+1} = x_k$

   $p_{2k+1} = p_{2k} - 2r_x^2 (y_k - 1) + r_x^2$

   Otherwise

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

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

6. Determine the symmetry points in other 3 quadrants.
7. Move each calculated point (x_k, y_k) on to the centered (xc,yc) ellipse path as

   $x_k = x_k + xc$;

   $y_k = y_k + yc$
8. Repeat the process for region 1 until $2r_y^2 x_k \geq 2r_x^2 y_k$ and region until (x_k, y_k) =(r_x,0)

Filled Area Primitives

A standard output primitive in general graphics package is solid color or patterned polygon area. Other kinds of area primitives are sometimes available, but polygons are easier to process since they have linear boundaries. There are two basic approaches to area filling in raster systems. One way to fill an area is to determine the overlap intervals for scan lines that cross the area. Another method for area filling is to start from a given interior position and point outward from this until a specified boundary is met.

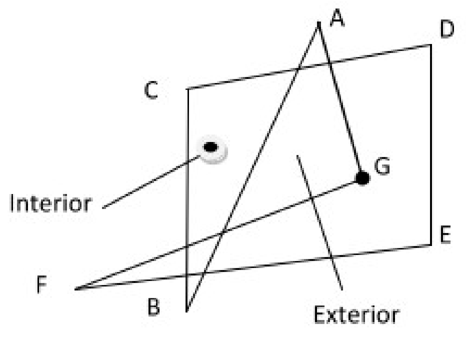
Scan-line Polygon Fill Algorithm

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

Inside-Outside Test

Area filling algorithms and other graphics package often need to identify interior and exterior region for a complex polygon in a plane. Viz. in figure below, it needs to identify interior and exterior region.

We apply add-even rule, also called odd-parity rule. To identify the interior or exterior point, we conceptually draw a line from a point p to a distant point outside the co-ordinate extents of the object and count the number of intersecting edge crossed by this line. If the intersecting edge crossed by this line is odd, P is **interior** otherwise P is **exterior**.

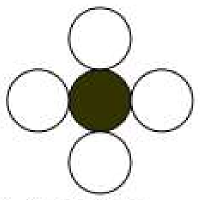


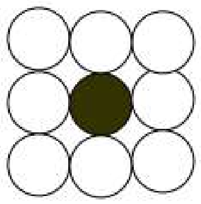
Boundary-Fill Algorithm

In Boundary filling algorithm starts at a point inside a region and paint the interior outward the boundary. If the boundary is specified in a single color, the fill algorithm proceeds outward pixel by until the boundary color is reached.

A boundary-fill procedure accepts as input the co-ordinates of an interior point (x, y), a fill color, and a boundary color. Starting from (x, y), the procedure tests neighboring positions to determine whether they are of boundary color. If not, they are painted with the fill color, and their neighbours are tested. This process continues until all pixels up to the boundary color area have tested. The neighbouring pixels from current pixel are proceeded by two methods:

- 4- Connected if they are adjacent horizontally and vertically.
- 8- Connected if they adjacent horizontally, vertically and diagonally.


4- Connected


8- Connected

- Fill method that applies and tests its 4 neighbouring pixel is called **4-connected**.
- Fill method that applies and tests its 8 neighbouring pixel is called **8-connected**.

<u>Algorithm Outline</u>: Recursive procedures

Boundary-Fill 4-Connected

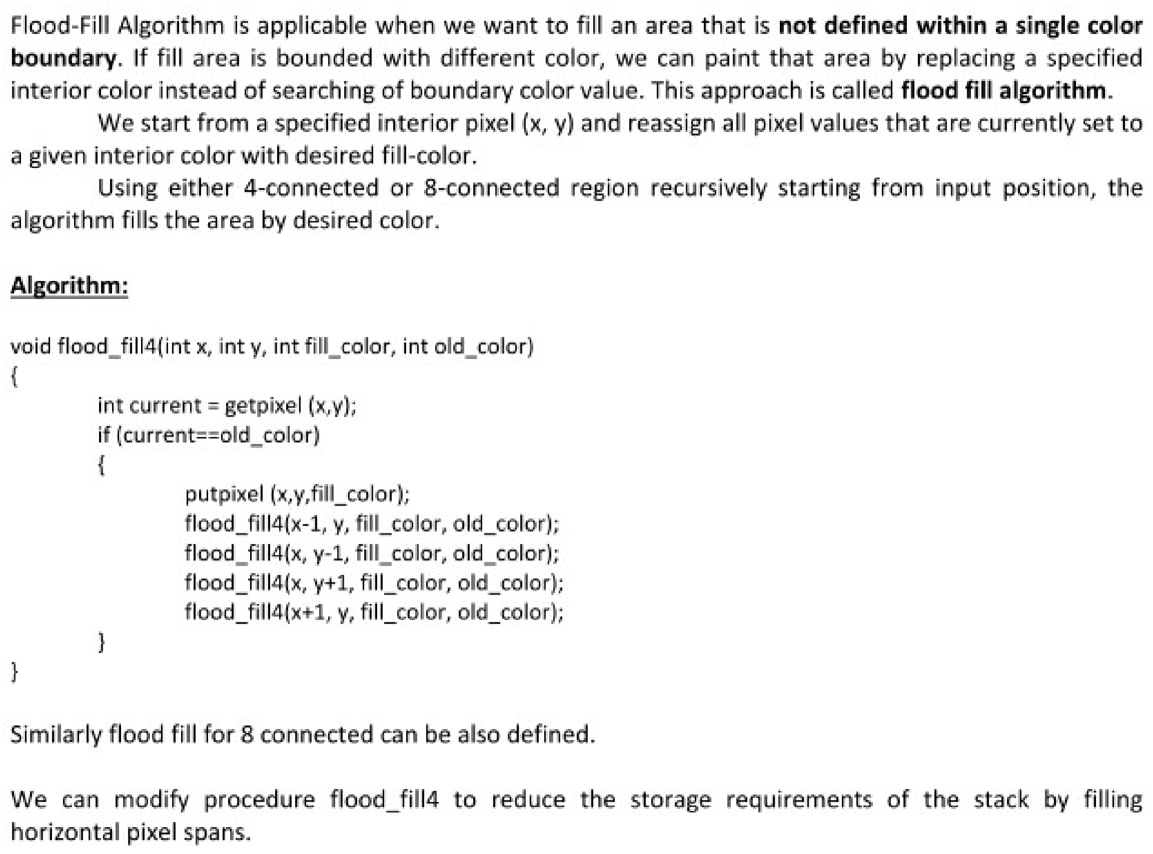
```
void Boundary_fill4(int x, int y, int b_color, int fill_color)
{
        int value = getpixel (x, y);
        if (value! =b_color && value!=fill_color)
        {
                putpixel (x, y, fill_color);
                Boundary_fill 4 (x-1, y, b_color, fill_color);
                Boundary_fill 4 (x+1, y, b_color, fill_color);
                Boundary_fill 4 (x,y-1, b_color, fill_color);
                Boundary_fill 4 (x,y+1, b_color, fill_color);
        }
 }
```

Boundary fill 8- connected:

```
void Boundary-fill8(int x,int y,int b_color, int fill_color)
{
        Int current  = getpixel (x, y);
        if (current !=b_color && current!=fill_color)
        {       putpixel (x,y,fill_color);
                Boundary_fill8(x-1, y, b_color,fill_color);
                Boundary_fill8(x+1, y, b_color, fill_color);
                Boundary_fill8(x, y-1, b_color, fill_color);
                Boundary_fill8(x, y+1, b_color, fill_color);
                Boundary_fill8(x-1, y-1, b_color,fill_color);
                Boundary_fill8(x-1,y+1,b_color,fill_color);
                Boundary_fill8(x+1,y-1,b_color,fill_color);
                Boundary_fill8(x+1,y+1,b_color,fill_color);
        }
}
```

Recursive boundary-fill algorithm does not fill regions correctly if some interior pixels are already displayed in the fill color. Encountering a pixel with the fill color can cause a recursive branch to terminate, leaving other interior pixel unfilled. To avoid this we can first change the color of any interior pixels that are initially set to the fill color before applying the boundary fill procedure.

Flood-Fill Algorithm



Flood-Fill Algorithm is applicable when we want to fill an area that is **not defined within a single color boundary**. If fill area is bounded with different color, we can paint that area by replacing a specified interior color instead of searching of boundary color value. This approach is called **flood fill algorithm.**

We start from a specified interior pixel (x, y) and reassign all pixel values that are currently set to a given interior color with desired fill-color.

Using either 4-connected or 8-connected region recursively starting from input position, the algorithm fills the area by desired color.

Algorithm:

```
void flood_fill4(int x, int y, int fill_color, int old_color)
{
        int current = getpixel (x,y);
        if (current==old_color)
        {
                putpixel (x,y,fill_color);
                flood_fill4(x-1, y, fill_color, old_color);
                flood_fill4(x, y-1, fill_color, old_color);
                flood_fill4(x, y+1, fill_color, old_color);
                flood_fill4(x+1, y, fill_color, old_color);
        }
}
```

Similarly flood fill for 8 connected can be also defined.

We can modify procedure flood_fill4 to reduce the storage requirements of the stack by filling horizontal pixel spans.

Cell Array

The cell array is a primitive that allows users to display an arbitrary shape defined as a two-dimensional grid pattern. A predefined matrix of color values is mapped by this function onto a specified rectangular coordinate region. The PHIGS version of this function is

Cellarray(wcpoints,n,m,colorArray)

Where colorarray is the n by m matrix of integer color values and wcpoints lists the limits of the rectangular coordinate region (xmin, ymin) and (xmax,ymax).

Character Generation

Computer graphics involves display of picture, lines and other graphics like designs. These picture and graph will belong to some data. Some information and instruction should be given to the user about this data. This is possible with the help of text display.

Since text consists of string of characters. so a character is a basc unit of text
There are three methods for character generation. These are:
1) Stroke Method
2) Bitmap Method
3) Starbust Method

1)STROKE METHOD

Stroke method is based on natural method of text written by human being. In this method graph is drawing in the form of line by line.
Line drowing algorithm DDA follow this method for line drawing.

2)BITMAP METHOD

Bitmap method is a called dot-matrix method. as the name suggest this method use array of bits fot generating a character. This dots are the points for array whose size is fixed.
In bitmatrix method when the dots is stored in the form of array the value 1 in array represent the characters i.e where the dots appear we represent that position with numerical value 1 and the value where dots are not present is reprented by 0 in array.

3)STARBUST METHOD

Starbust method is user in a particular pattern where only 24 strokes are defined for character generation.

Attributes of output primitives

Any parameter that affects the way a primitive is to be displayed is referred to as an attribute parameter. Example attribute parameters are
color, size etc. A line drawing function for example could contain parameter to set color, width and other properties.

1.Line Attributes

2.Curve Attributes

3.Color and Grayscale Levels

4.Area Fill Attributes

5.Character Attributes

6.Bundled Attributes

1. Line Attributes

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

- Line Type
- Line Width
- Pen and Brush Options
- Line Color

Line type

Possible selection of line type attribute includes solid lines, dashed lines and dotted lines. To set line type attributes in a **PHIGS** application program, a user invokes the function

setLinetype (lt)

Where parameter lt is assigned a positive integer value of 1, 2, 3 or 4 to generate lines that are solid, dashed, dash dotted respectively. Other values for line type parameter it could be used to display variations in dot-dash patterns.

Line width

Implementation of line width option depends on the capabilities of the output device to set the line width attributes.

setLinewidthScaleFactor(lw)

Line width parameter lw is assigned a positive number to indicate the relative width of line to be displayed. A value of 1 specifies a standard width line. A user could set lw to a value of 0.5 to plot a line whose width is half that of the standard line. Values greater than 1 produce lines thicker than the standard.

Line Cap

We can adjust the shape of the **line** ends to give them a better appearance by adding line caps.

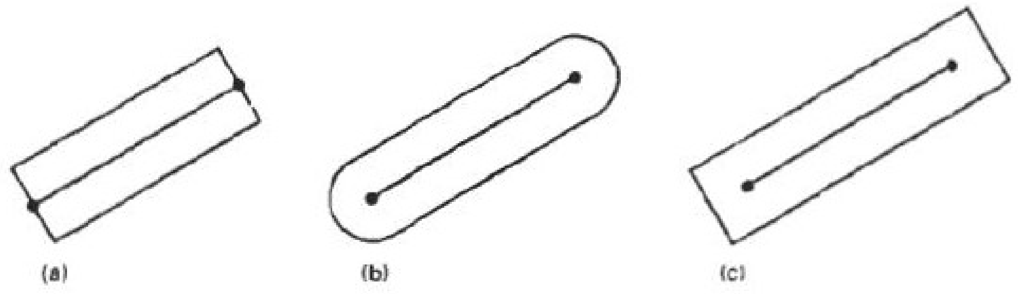
There are three types of line cap. They are

- Butt cap
- Round cap
- Projecting square cap

Butt cap obtained by adjusting the end positions of the component parallel **lines** so that the thick line is displayed with square ends that are perpendicular to the line path.

Round cap obtained by adding a filled semicircle to each butt cap. The circular arcs are centered on the line endpoints and have a diameter equal to the line thickness

Projecting square cap extend the line and add butt caps that are positioned one-half of the line width beyond the specified endpoints.

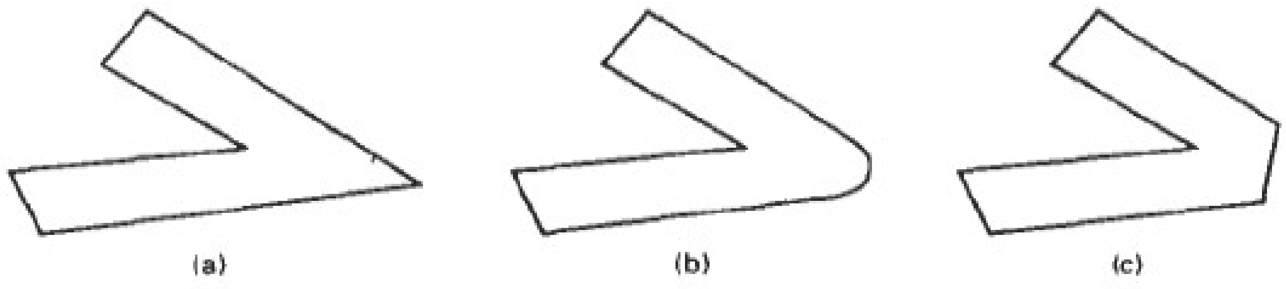


Thick lines drawn with (a) butt caps, (b) round caps, and (c) projecting square caps.

Three possible methods for smoothly joining two line segments

- Mitter Join
- Round Join
- Bevel Join

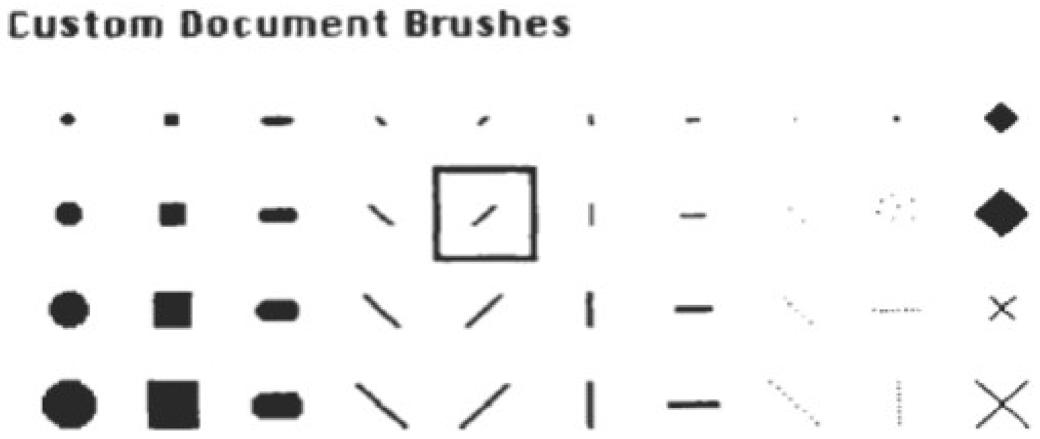
1. A **miter join** accomplished by extending the outer boundaries of each of the two lines until they meet.
2. A **round join** is produced by capping the connection between the two segments with a circular boundary whose diameter is equal to the width.
3. A **bevel join** is generated by displaying the line segment with but caps and filling in tri angular gap where the segments meet



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

Pen and Brush Options

With some packages, lines can be displayed with pen or brush selections. Options in this category include shape, size, and pattern. Some possible pen or brush shapes are given in Figure



Line color

A poly line routine displays a line in the current color by setting this color value in the frame buffer at pixel locations along the line path using the set pixel procedure.
We set the line color value in **PHIGS** with the function

setPolylineColourIndex (lc)

Nonnegative integer values, corresponding to allowed color choices, are assigned to the line color parameter lc

2. Curve Attributes

Parameters for curve attribute are same as those for line segments. Curves displayed with varying colors, widths, dot –dash patterns and available pen or brush options

3. Color and Grayscale level

Various color and intensity-level options can be made available to a user, depending on the capabilities and design objectives of a particular system

In a color raster system, the number of color choices available depends on the amount of storage provided per pixel in the frame buffer

Color-information can be stored in the frame buffer in two ways:

- We can store color codes directly in the frame buffer
- We can put the color codes in a separate table and use pixel values as an index into this table

A user can set color-table entries in a PHIGS applications program with the function

setColourRepresentation (ws, ci, colorptr)

Grayscale

With monitors that have no color capability, color functions can be used in an application program to set the shades of gray, or grayscale, for displayed primitives. Numeric values over the range from 0 to 1 can be used to specify grayscale levels, which are then converted to appropriate binary codes for storage in the raster.

4. Area fill Attributes

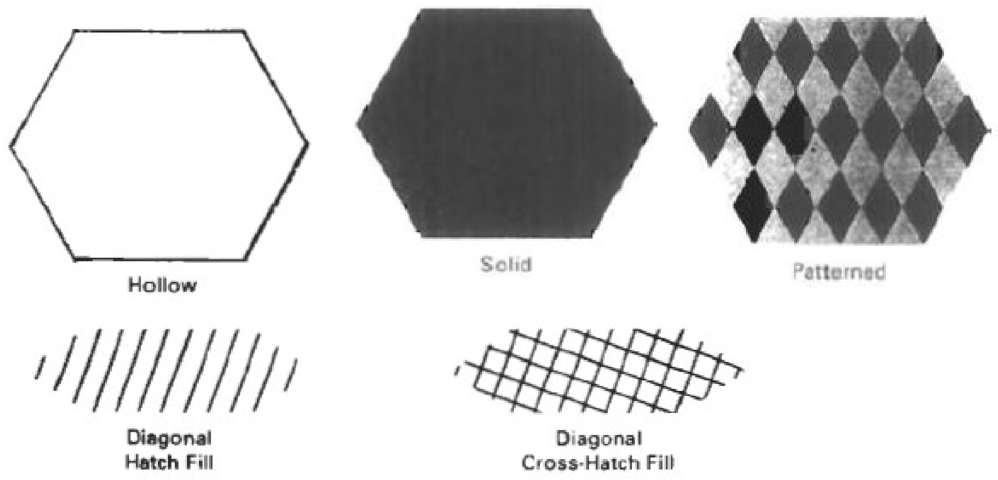
Options for filling a defined region include a choice between a solid color or a pattern fill and choices for particular colors and patterns

Fill Styles

Areas are displayed with three basic fill styles: hollow with a color border, filled with a solid color, or filled with a specified pattern or design. A basic fill style is selected in a **PHIGS** program with the function

setInteriorStyle(fs)

Values for the fill-style parameter fs include hollow, solid, and pattern. Another value for fill style is hatch, which is used to fill an area with selected hatching patterns-parallel lines or crossed lines



The color for a solid interior or for a hollow area outline is chosen with where fill color parameter fc is set to the desired color code

setInteriorColourIndex(fc)

Pattern Fill

We select fill patterns with setInteriorStyleIndex (pi) where pattern index parameter pi specifies a table position

5. Character Attributes

The appearance of displayed character is controlled by attributes such as font, size, color and orientation. Attributes can be set both for entire character strings (text) and for individual characters defined as marker symbols

Text Attributes

The choice of font or type face is set of characters with a particular design style as courier, Helvetica, times roman, and various symbol groups.

The characters in a selected font also be displayed with styles. (solid, dotted, double) in **bold face** in *italics*, and in outline or shadow styles.

A particular font and associated stvle is selected in a PHIGS program by setting an integer code for the text font parameter tf in the function

setTextFont(tf)

Control of text color (or intensity) is managed from an application program with

setTextColourIndex(tc)

where text color parameter tc specifies an allowable color code.

Text size can be adjusted without changing the width to height ratio of characters with

SetCharacterHeight (ch)

6. Bundled Attributes

The procedures considered so far each function reference a single attribute that specifies exactly how a primitive is to be displayed these specifications are called individual attributes.

A particular set of attributes values for a primitive on each output device is chosen by specifying appropriate table index. Attributes specified in this manner are called bundled attributes. The choice between a bundled or an unbundled specification is made by setting a switch called the aspect source flag for each of these attributes

setIndividualASF(attributeptr, flagptr)

where parameter attributer ptr points to a list of attributes and parameter flagptr points to the corresponding list of aspect source flags. Each aspect source flag can be assigned a value of individual or bundled.

Bundled line attributes

Entries in the bundle table for line attributes on a specified workstation are set with the function

setPolylineRepresentation (ws, li, lt, lw, lc)

Parameter ws is the workstation identifier and line index parameter li defines the bundle table position. Parameter lt, lw, tc are then bundled and assigned values to set the line type, line width, and line color specifications for designated table index.

Bundle area fill Attributes

Table entries for bundled area-fill attributes are set with

setInteriorRepresentation (ws, fi, fs, pi, fc)

Which defines the attributes list corresponding to fill index fi on workstation ws. Parameter fs, pi and fc are assigned values for the fill style pattern index and fill color.

Bundled Text Attributes

setTextRepresentation (ws, ti, tf, tp, te, ts, tc)

bundles values for text font, precision expansion factor size an color in a table position for work station ws that is specified by value assigned to text index parameter ti.

Bundled marker Attributes

setPolymarkerRepresentation (ws, mi, mt, ms, mc)

That defines marker type marker scale factor marker color for index mi on workstation ws.

Antialiasing

In computer graphics, the process by which smooth curves and other lines become jagged because the resolution of the graphics device or file is not high enough to represent a smooth curve.

In the line drawing algorithms, we have seen that all rasterized locations do not match with the true line and we have to select the optimum raster locations to represent a straight line. This problem is severe in low resolution screens. In such screens line appears like a stair-step, as shown in the figure below. This effect is known as **aliasing**. It is dominant for lines having gentle and sharp slopes.

The **aliasing effect** can be reduced by adjusting intensities of the pixels along the line. The process of adjusting intensities of the pixels along the line to minimize the effect of aliasing is called **antialiasing**.

The aliasing effect can be minimized by increasing resolution of the raster display. By increasing resolution and making it twice the original one, the line passes through twice as many column of pixels and therefore has twice as many **jags**, but each jag is half as large in x and in y direction.

Antialiasing methods are basically classified as :-

Supersampling or Postfiltering:-

Supersampling or Postfiltering is the process by which aliasing effects in graphics are reduced by increasing the frequency of the sampling grid and then averaging the results down. This process means calculating a virtual image at a higher spatial resolution than the frame store resolution and then averaging down to the final resolution. It is called Postfiltering as the filtering is carried out after sampling.

Supersampling is basically a three stage process:

1. A continuous image I(x , y) is sampled at n times the frame resolution. This is a virtual image.
2. The virtual image is then lowpass filtered.

3. The filtered image is then resampled at the final frame resolution.

Area sampling or Prefiltering:-

In this antialiasing method pixel intensity is determined by calculating the areas of overlap of each pixel with the objects to be displayed. Antialiasing by computing area is referred to as **Area sampling or Prefiltering**. A modification to Bresenham's algorithm was developed by Pitteway and Watkinson. In this algorithm, each pixel is given intensity depending on the area of overlap of the pixel and the line. So, due to the blurring effect along the line edges, the effect of anti-aliasing is not very prominent, although it still exists. For sampling shapes other than polygons, this can be very computationally intensive.

Module 03

Two Dimesional Geometric Transformations

Changes in orientations, size and shape are accomplished with geometric transformations that alter the coordinate description of objects.

Basic transformation

- Translation
  - $T(t_x, t_y)$
  - Translation distances
- Scale
  - $S(s_x, s_y)$
  - Scale factors
- Rotation
  - $R(\theta)$
  - Rotation angle

Translation

A translation is applied to an object by representing it along a straight line path from one coordinate location to another adding translation distances, tx, ty to original coordinate position (x,y) to move the point to a new position (x’,y’) to

$$x' = x + tx, \quad y' = y + ty$$

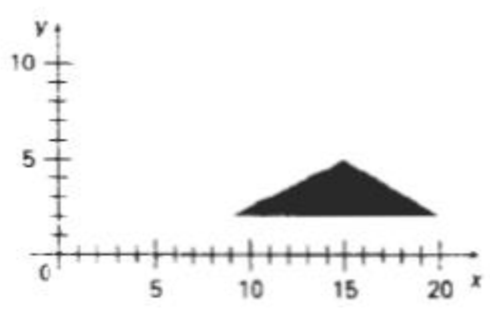
The translation distance point (tx,ty) is called translation vector or shift vector.

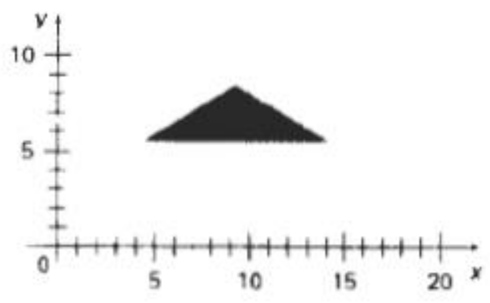
Translation equation can be expressed as single matrix equation by using column vectors to represent the coordinate position and the translation vector as

$$P = (x, y)$$
$$T = (t_x, t_y)$$

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

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$
$$P' = P + T$$

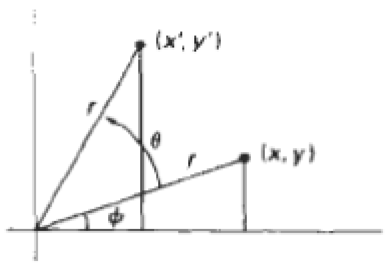




Rotations:

A two-dimensional rotation is applied to an object by repositioning it along a circular path on xy plane. To generate a rotation, specify a rotation angle θ and the position (x_r,y_r) of the rotation point (pivot point) about which the object is to be rotated.

Positive values for the rotation angle define counter clock wise rotation about pivot point. Negative value of angle rotate objects in clock wise direction. The transformation can also be described as a rotation about a rotation axis perpendicular to xy plane and passes through pivot point



Rotation of a point from position (x,y) to position (x',y') through angle θ relative to coordinate origin

The transformation equations for rotation of a point position P when the pivot point is at coordinate origin. In figure r is constant distance of the point positions Φ is the original angular of the point from horizontal and θ is the rotation angle.

The transformed coordinates in terms of angle θ and Φ

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

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

The original coordinates of the point in polar coordinates

$$x = r\cos\Phi, \quad y = r\sin\Phi$$

the transformation equation for rotating a point at position (x,y) through an angle θ about origin

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

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

Rotation equation

$P' = R \cdot P$

Rotation Matrix

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

Scaling

A scaling transformation alters the size of an object. This operation can be carried out for polygons by multiplying the coordinate values (x,y) to each vertex by scaling factor Sx & Sy to produce the transformed coordinates (x',y')

$x' = x.Sx$ **$y' = y.Sy$**

scaling factor Sx scales object in x direction while Sy scales in y direction.

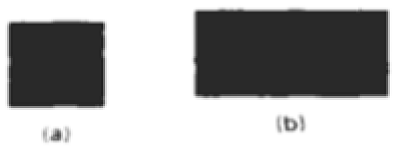
The transformation equation in matrix form

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

or

$P' = S. P$

Where S is 2 by 2 scaling matrix



Turning a square (a) Into a rectangle (b) with scaling factors sx = 2 and sy= 1.

Any positive numeric values are valid for scaling factors sx and sy. Values less than 1 reduce the size of the objects and values greater than 1 produce an enlarged object.

Matrix Representation and homogeneous Coordination

Many graphics applications involve sequences of geometric transformations. An animation, for example, might require an object to be translated and rotated at each increment of the motion. In order to combine sequence of transformations we have to eliminate the matrix addition. To achieve this we have represent matrix as 3 X 3 instead of 2 X 2 introducing an additional dummy coordinate h. Here points are specified by three numbers instead of two. This coordinate system is called as Homogeneous coordinate system and it allows to express transformation equation as matrix multiplication

Cartesian coordinate position (x,y) is represented as homogeneous coordinate triple(x,y,h)

- Represent coordinates as (x,y,h)
- Actual coordinates drawn will be (x/h,y/h)

For Translation

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

$$P' = T(t_x, t_y) \bullet P$$

For Scaling

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

$$P' = S(s_x, s_y) \bullet P$$

For rotation

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

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

COMPOSITE TRANSFORMATIONS

A composite transformation is a sequence of transformations; one followed by the other. we can set up a matrix for any sequence of transformations as a **composite transformation matrix** by calculating the matrix product of the individual transformations

Translation

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

P'=T(tx2,ty2).{T(tx1,ty1).P}

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

Where P and P' are represented as homogeneous-coordinate column vectors.

$$\begin{bmatrix} 1 & 0 & tx2 \\ 0 & 1 & ty2 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} 1 & 0 & tx1 \\ 0 & 1 & ty1 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & tx1+tx2 \\ 0 & 1 & ty1+ty2 \\ 0 & 0 & 1 \end{bmatrix}$$

Or

T(tx2,ty2).T(tx1,ty1) = T(tx1+tx2,ty1+ty2)

Which demonstrated the two successive translations are additive.

Rotations

Two successive rotations applied to point P produce the transformed position

P'=R(θ2).{R(θ1).P}={R(θ2).R(θ1)}.P

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

R(θ2).R(θ1) = R(θ1+ θ2)

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

P' = R(θ1+ θ2).P

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

Scaling

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

$$\begin{bmatrix} sx2 & 0 & 0 \\ 0 & sy2 & 0 \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} sx1 & 0 & 0 \\ 0 & sy1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} sx2.sx1 & 0 & 0 \\ 0 & sy2.sy1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

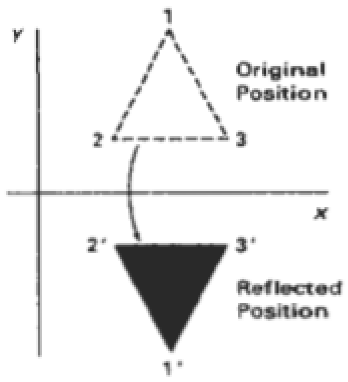
Other Transformations

1. Reflection
2. Shear

Reflection

A reflection is a transformation that produces a mirror image of an object. The mirror image for a two-dimensional reflection is generated relative to an axis of reflection by rotating the object 180^o about the reflection axis. We can choose an axis of reflection in the xy plane or perpendicular to the xy plane or coordinate origin

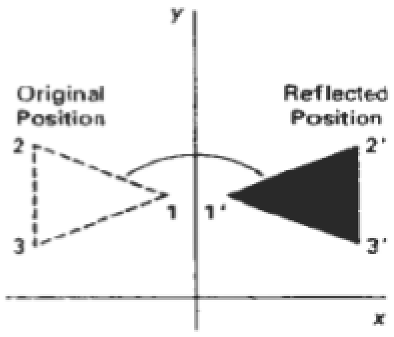
Reflection of an object about the x axis



Reflection the x axis is accomplished with the transformation matrix

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

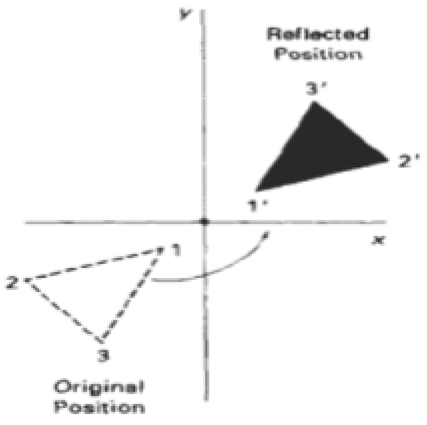
Reflection of an object about the y axis



Reflection the y axis is accomplished with the transformation matrix

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

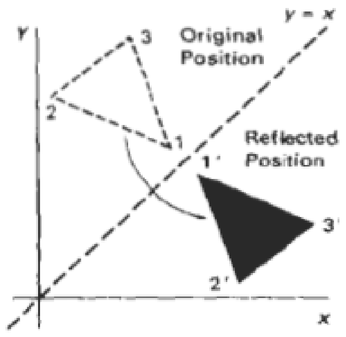
Reflection of an object about the coordinate origin



Reflection about origin is accomplished with the transformation matrix

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

Reflection axis as the diagonal line y = x



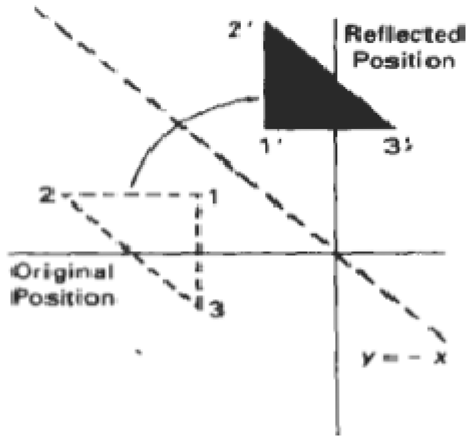
To obtain transformation matrix for reflection about diagonal y=x the transformation sequence is

1. Clock wise rotation by 45^0
2. Reflection about x axis
3. counter clock wise by 45^0

Reflection about the diagonal line y=x is accomplished with the transformation matrix

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

Reflection axis as the diagonal line y = -x



To obtain transformation matrix for reflection about diagonal y=-x the transformation sequence is

1. Clock wise rotation by 45^0
2. Reflection about y axis
3. counter clock wise by 45^0

Reflection about the diagonal line y=-x is accomplished with the transformation matrix

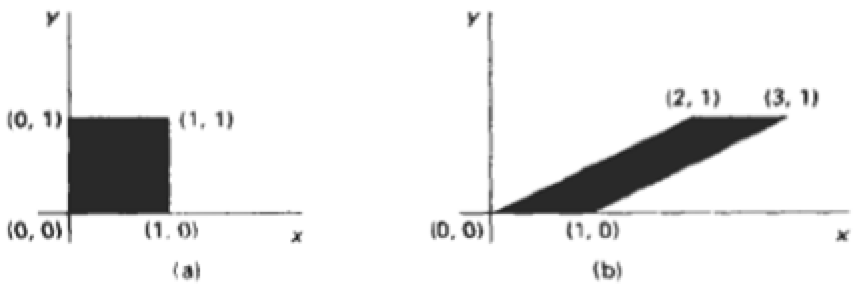
$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Shear

A Transformation that slants the shape of an object is called the shear transformation. Two common shearing transformations are used. One shifts x coordinate values and other shift y coordinate values. However in both the cases only one coordinate (x or y) changes its coordinates and other preserves its values.

X- Shear

The x shear preserves the y coordinates, but changes the x values which cause vertical lines to tilt right or left as shown in figure



The Transformations matrix for x-shear is

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

which transforms the coordinates as

x' =x+ sh_x.y

y' = y

Y Shear

The y shear preserves the x coordinates, but changes the y values which cause horizontal lines which slope up or down

The Transformations matrix for y-shear is

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

which transforms the coordinates as

x' =x

y' = y+ sh_y.x

XY-Shear

The transformation matrix for xy-shear

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

which transforms the coordinates as

x' =x+ sh_x .y

y' = y+ sh_y .x

Shearing Relative to other reference line

We can apply x shear and y shear transformations relative to other reference lines. In x shear transformations we can use y reference line and in y shear we can use x reference line.

X shear with y reference line

We can generate x-direction shears relative to other reference lines with the transformation matrix

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

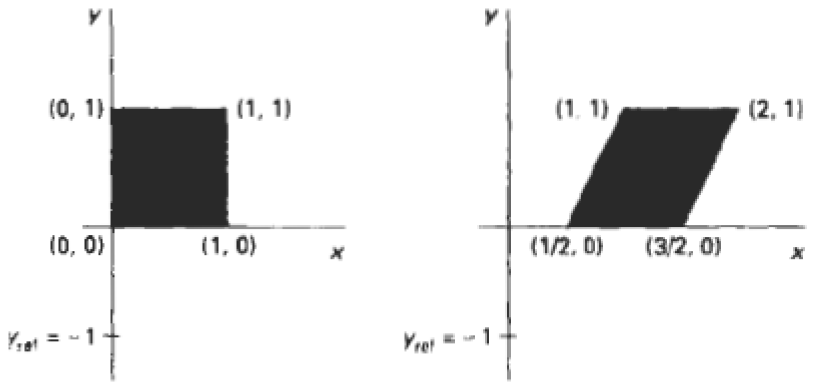
which transforms the coordinates as

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

y' = y

Example

Sh_x = ½ and y_{ref}=-1



Y shear with x reference line

We can generate y-direction shears relative to other reference lines with the transformation matrix

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

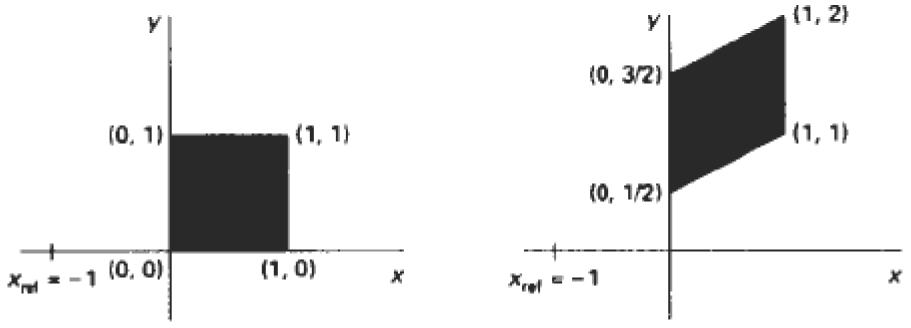
which transforms the coordinates as

x' =x

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

Example

Sh_y = ½ and x_{ref}=-1



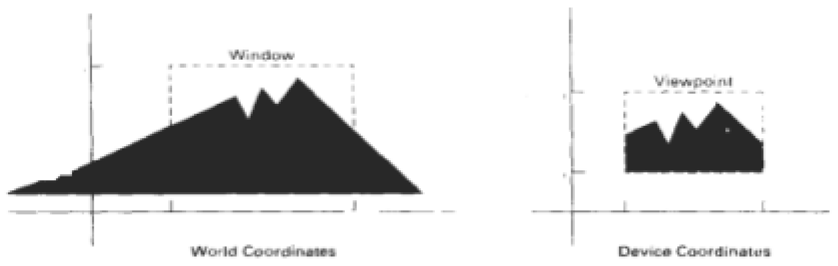
Two Dimension Viewing

The Viewing Pipeline

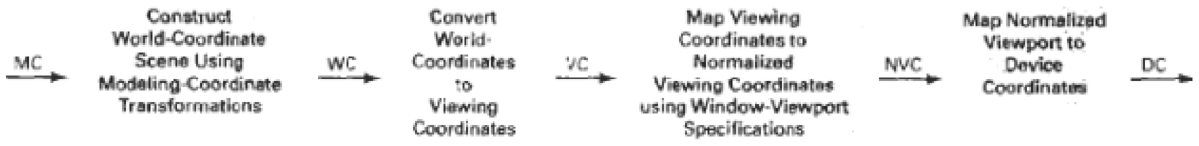
A world coordinate area selected for display is called a window. An area on a display device to which a window is mapped is called a view port. The window defines what is to be viewed the view port defines where it is to be displayed.

The mapping of a part of a world coordinate scene to device coordinate is referred to as viewing transformation. The two dimensional viewing transformation is referred to as window to view port transformation of windowing transformation.

A viewing transformation using standard rectangles for the window and viewport



The two dimensional viewing transformation pipeline

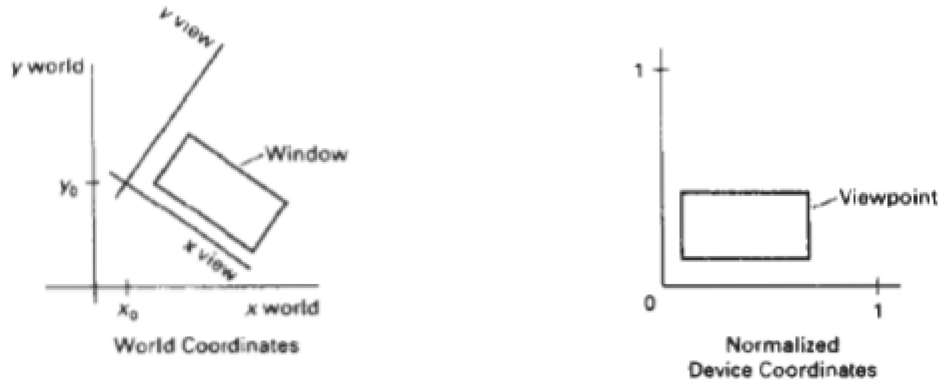

MC
Construct World-Coordinate Scene Using Modeling-Coordinate Transformations
WC
Convert World-Coordinates to Viewing Coordinates
VC
Map Viewing Coordinates to Normalized Viewing Coordinates using Window-Viewport Specifications
NVC
Map Normalized Viewport to Device Coordinates
DC

The viewing transformation in several steps, as indicated in Fig. First, we construct the scene in world coordinates using the output primitives. Next to obtain a particular orientation for the window, we can set up a two-dimensional viewing-coordinate system in the world coordinate plane, and define a window in the viewing-coordinate system.

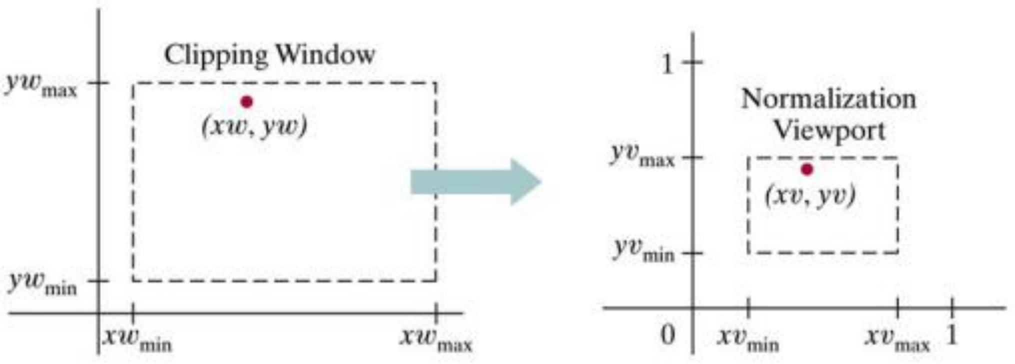
The viewing- coordinate reference frame is used to provide a method for setting up arbitrary orientations for rectangular windows. Once the viewing reference frame is established, we can transform descriptions in world coordinates to viewing coordinates.

We then define a viewport in normalized coordinates (in the range from 0 to 1) and map the viewing-coordinate description of the scene to normalized coordinates.

At the final step all parts of the picture that lie outside the viewport are clipped, and the contents of the viewport are transferred to device coordinates. By changing the position of the viewport, we can view objects at different positions on the display area of an output device.



Window to View port coordinate transformation:



A point (xw, yw) in a world-coordinate clipping window is mapped to viewport coordinates (xv, yv), within a unit square, so that the relative positions of the two points in their respective rectangles are the same.

A point at position (x_w,y_w) in a designated window is mapped to viewport coordinates (x_v,y_v) so that relative positions in the two areas are the same. The figure illustrates the window to view port mapping.

A point at position (x_w,y_w) in the window is mapped into position (x_v,y_v) in the associated view port. To maintain the same relative placement in view port as in window

$$\frac{xv - xv_{min}}{xv_{max} - xv_{min}} = \frac{xw - xw_{min}}{xw_{max} - xw_{min}}$$

$$\frac{yv - yv_{min}}{yv_{max} - yv_{min}} = \frac{yw - yw_{min}}{yw_{max} - yw_{min}}$$

solving these expressions for view port position (x_v,y_v)

$$xv = xv_{min} + (xw - xw_{min})\frac{(xv_{max} - xv_{min})}{xw_{max} - xw_{min}}$$

$$yv = yv_{min} + (yw - yw_{min})\frac{(yv_{max} - yv_{min})}{yw_{max} - yw_{min}}$$

where scaling factors are

$$sx = \frac{xv_{max} - xv_{min}}{xw_{max} - xw_{min}} \qquad sy = \frac{yv_{max} - yv_{min}}{yw_{max} - yw_{min}}$$

The conversion is performed with the following sequence of transformations.

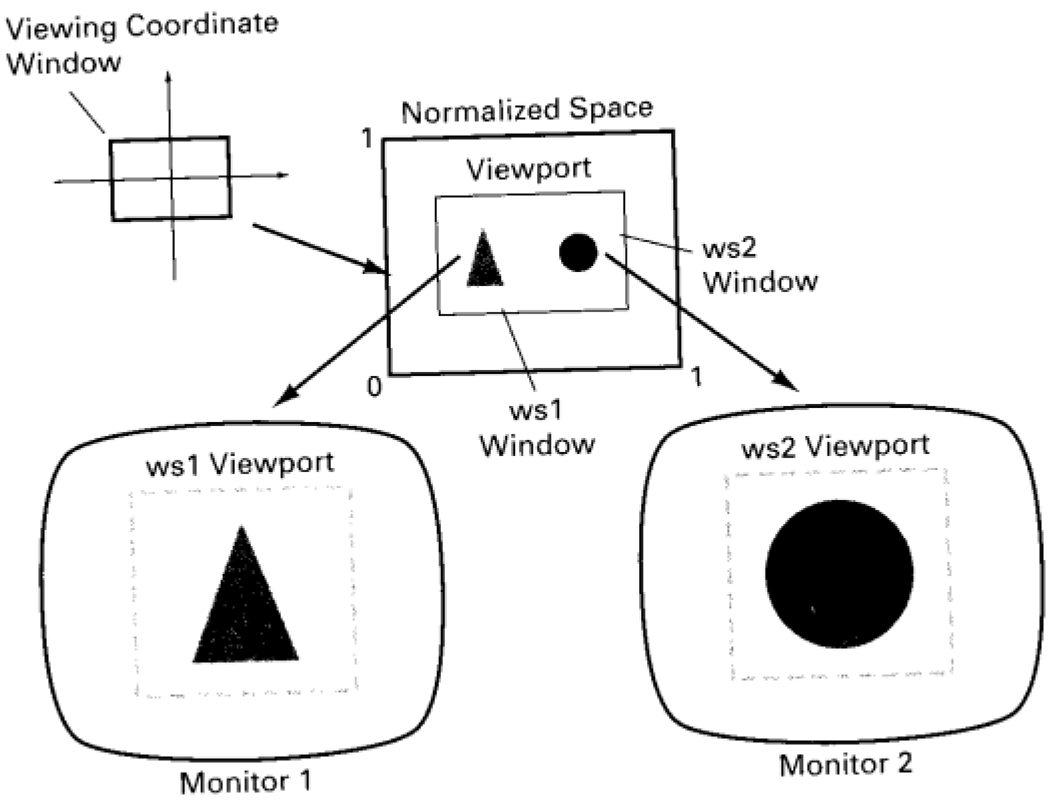
1. Perform a scaling transformation using point position of (x_w min, y_w min) that scales the window area to the size of view port.

2. Translate the scaled window area to the position of view port. Relative proportions of objects are maintained if scaling factor are the same(Sx=Sy).

Otherwise world objects will be stretched or contracted in either the x or y direction when displayed on output device. For normalized coordinates, object descriptions are mapped to various display devices.

Any number of output devices can be open in particular application and another window view port transformation can be performed for each open output device. This mapping called the work station transformation is accomplished by selecting a window area in normalized apace and a view port are in coordinates of display device.

Mapping selected parts of a scene in normalized coordinate to different video monitors with work station transformation.



Clipping Operations

Any procedure that identifies those portions of a picture that are inside or outside of a specified region of space is referred to as **clipping algorithm or clipping**. The region against which an object is to be clipped is called **clip window**.

Algorithm for clipping primitive types:

Point clipping
Line clipping (Straight-line segment)
Area clipping
Curve clipping
Text clipping

Point Clipping

Clip window is a rectangle in standard position. A point P=(x,y) for display, if following inequalities are satisfied:

$$xw_{min} <= x <= xw_{max}$$

$$yw_{min} <= y <= yw_{max}$$

where the edges of the clip window **(xw_{min},xw_{max},yw_{min},yw_{max})** can be either the world-coordinate window boundaries or viewport boundaries. If any one of these four inequalities is not satisfied, the point is clipped (not saved for display).

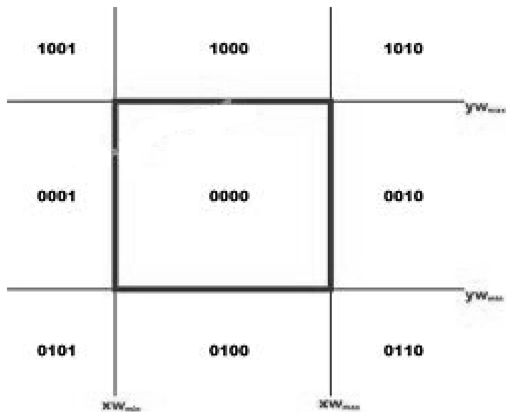
Line Clipping

A line clipping procedure involves several parts. First we test a given line segment whether it lies completely inside the clipping window. If it does not we try to determine whether it lies completely outside the window . Finally if we can not identify a line as completely inside or completely outside, we perform intersection calculations with one or more clipping boundaries.

Cohen-Sutherland Line Clipping

This is one of the oldest and most popular line-clipping procedures. The method speeds up the processing of line segments by performing initial tests that reduce the number of intersections that must be calculated.

Every line endpoint in a picture is assigned a four digit binary code called a **region code** that identifies the location of the point relative to the boundaries of the clipping rectangle.



Binary region codes assigned to line end points according to relative position with respect to the clipping rectangle.

Regions are set up in reference to the boundaries. Each bit position in region code is used to indicate one of four relative coordinate positions of points with respect to clip window: to the left, right, top or bottom. By numbering the bit positions in the region code as 1 through 4 from right to left, the coordinate regions are corrected with bit positions as

bit 1:	left
bit 2:	right
bit 3:	below
bit4:	above

A value of 1 in any bit position indicates that the point is in that relative position. Otherwise the bit position is set to 0. If a point is within the clipping rectangle the region code is 0000. A point that is below and to the left of the rectangle has a region code of 0101.

Bit values in the region code are determined by comparing endpoint coordinate values (x,y) to clip boundaries. Bit1 is set to 1 if $x < xw_{min}$.

For programming language in which bit manipulation is possible region-code bit values can be determined with following two steps.

(1) Calculate differences between endpoint coordinates and clipping boundaries.

(2) Use the resultant sign bit of each difference calculation to set the corresponding value in the region code.

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

bit 2 is the sign bit of $xw_{max} - x$

bit 3 is the sign bit of $y - yw_{min}$

bit 4 is the sign bit of $yw_{max} - y$.

Once we have established region codes for all line endpoints, we can quickly determine which lines are completely inside the clip window and which are clearly outside.

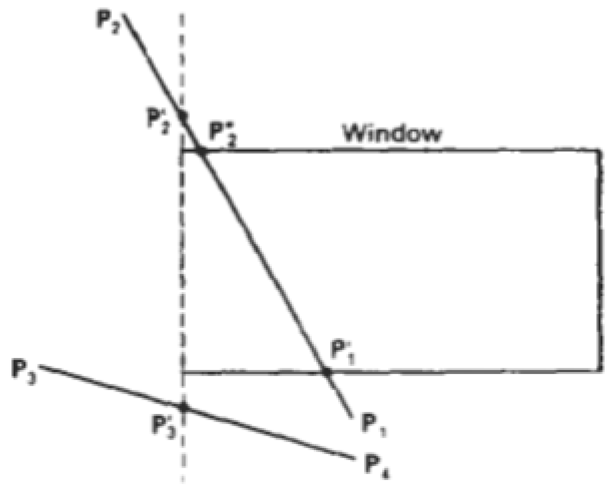
Any lines that are completely contained within the window boundaries have a region code of 0000 for both endpoints, and we accept

these lines. Any lines that have a 1 in the same bit position in the region codes for each endpoint are completely outside the clipping rectangle, and we reject these lines.

We would discard the line that has a region code of 1001 for one endpoint and a code of 0101 for the other endpoint. Both endpoints of this line are left of the clipping rectangle, as indicated by the 1 in the first bit position of each region code.

A method that can be used to test lines for total clipping is to perform the logical and operation with both region codes. If the result is not 0000,the line is completely outside the clipping region.

Lines that cannot be identified as completely inside or completely outside a clip window by these tests are checked for intersection with window boundaries.



Line extending from one coordinates region to another may pass through the clip window, or they may intersect clipping boundaries without entering window.

Cohen-Sutherland line clipping starting with bottom endpoint left, right , bottom and top boundaries in turn and find that this point is below the clipping rectangle.

Starting with the bottom endpoint of the line from **P_1** to **P_2,** we check P_1 against the left, right, and bottom boundaries in turn and find that this point is below the clipping rectangle. We then find the intersection point P_1' with the bottom boundary and discard the line section from P_1 to P_1'.

The line now has been reduced to the section from P_1' to P_2,Since P_2**,** is outside the clip window, we check this endpoint against the boundaries and find that it is to the left

of the window. Intersection point P_2' is calculated, but this point is above the window. So the final intersection calculation yields P_2''**,** and the line from P_1' to P_2''is saved. This completes processing for this line, so we save this part and go on to the next line.

Point P_3 in the next line is to the left of the clipping rectangle, so we determine the intersection P_3'**,** and eliminate the line section from P_3 to P_3'. By checking region codes for the line section from P_3'to P_4 we find that the remainder of the line is below the clip window and can be discarded also.

Intersection points with a clipping boundary can be calculated using the slope-intercept form of the line equation. For a line with endpoint coordinates (x_1,y_1) and (x_2,y_2) and the y coordinate of the intersection point with a vertical boundary can be obtained with the calculation.

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

where x value is set either to xw_{min} or to xw_{max} and slope of line is calculated as

$$m = (y_2 - y_1) / (x_2 - x_1)$$

the intersection with a horizontal boundary the x coordinate can be calculated as

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

with y set to either to yw_{min} or to yw_{max}.

Liang – Barsky line Clipping:

Based on analysis of parametric equation of a line segment, faster line clippers have been developed, which can be written in the form :

$$x = x_1 + u \Delta x$$
$$y = y_1 + u \Delta y \qquad 0<=u<=1$$

where $\Delta x = (x_2 - x_1)$ and $\Delta y = (y_2 - y_1)$

In the Liang-Barsky approach we first the point clipping condition in parametric form :

$$xw_{min} <= x_1 + u\ \Delta x <=.\ xw_{max}$$

$$yw_{min} <= y_1 + u\ \Delta y <= yw_{max}$$

Each of these four inequalities can be expressed as

$\mu p_k <= q_k$.   k=1,2,3,4

the parameters p & q are defined as

$p_1 = -\Delta x$	$q_1 = x_1 - xw_{min}$
$p_2 = \Delta x$	$q_2 = xw_{max} - x_1$
$P_3 = -\Delta y$	$q_3 = y_1 - yw_{min}$
$P_4 = \Delta y$	$q_4 = yw_{max} - y_1$

Any line that is parallel to one of the clipping boundaries have $p_k=0$ for values of k corresponding to boundary k=1,2,3,4 correspond to left, right, bottom and top boundaries. For values of k, find $q_k<0$, the line is completely out side the boundary.

If $q_k >=0$, the line is inside the parallel clipping boundary.

When $p_k<0$ the infinite extension of line proceeds from outside to inside of the infinite extension of this clipping boundary.

If $p_k>0$, the line proceeds from inside to outside, for non zero value of p_k calculate the value of u, that corresponds to the point where the infinitely extended line intersect the extension of boundary k as

$$u = qk / pk$$

For each line, calculate values for parameters $u_1$and u_2 that define the part of line that lies within the clip rectangle. The value of u1 is determined by looking at the rectangle edges for which the line proceeds from outside to the inside (p<0).

For these edges we calculate

$$r_k = qk / pk$$

The value of u1 is taken as largest of set consisting of 0 and various values of r. The value of u2 is determined by examining the boundaries for which lines proceeds from inside to outside (P>0).

A value of $r_k$is calculated for each of these boundaries and value of u2 is the minimum of the set consisting of 1 and the calculated r values.

If u1>u2, the line is completely outside the clip window and it can be rejected.

Line intersection parameters are initialized to values u1=0 and u2=1. for each clipping boundary, the appropriate values for P and q are calculated and used by function

Cliptest to determine whether the line can be rejected or whether the intersection parameter can be adjusted.

When p<0, the parameter r is used to update u1.

When p>0, the parameter r is used to update u2.

If updating u1 or u2 results in u1>u2 reject the line, when p=0 and q<0, discard the line, it is parallel to and outside the boundary.If the line has not been rejected after all four value of p and q have been tested , the end points of clipped lines are determined from values of u1 and u2.

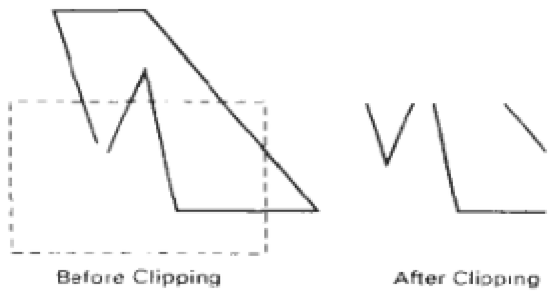
The Liang-Barsky algorithm is more efficient than the Cohen-Sutherland algorithm since intersections calculations are reduced. Each update of parameters u1 and u2 require only one division and window intersections of these lines are computed only once.

Cohen-Sutherland algorithm, can repeatedly calculate intersections along a line path, even through line may be completely outside the clip window. Each intersection calculations require both a division and a multiplication.

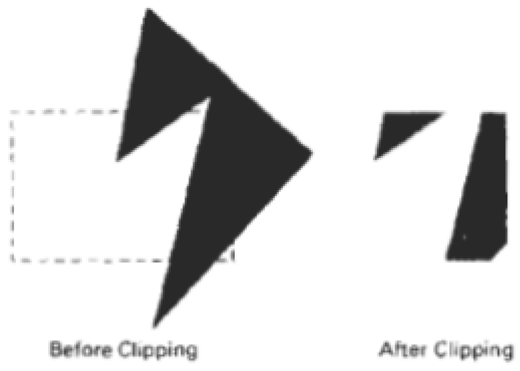
Polygon Clipping

To clip polygons, we need to modify the line-clipping procedures. A polygon boundary processed with a line clipper may be displayed as a series of unconnected line segments (Fig.), depending on the orientation of the polygon to the clipping window.

Display of a polygon processed by a line clipping algorithm



For polygon clipping, we require an algorithm that will generate one or more closed areas that are then scan converted for the appropriate area fill. The output of a polygon clipper should be a sequence of vertices that defines the clipped polygon boundaries.



Sutherland – Hodgeman polygon clipping:

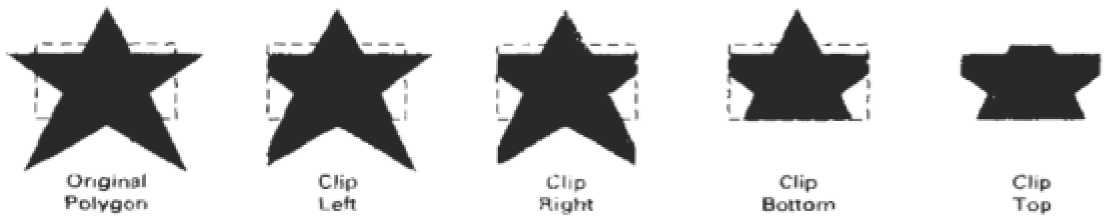
A polygon can be clipped by processing the polygon boundary as a whole against each window edge. This could be accomplished by processing all polygon vertices against each clip rectangle boundary.

There are four possible cases when processing vertices in sequence around the perimeter of a polygon. As each point of adjacent polygon vertices is passed to a window boundary clipper, make the following tests:

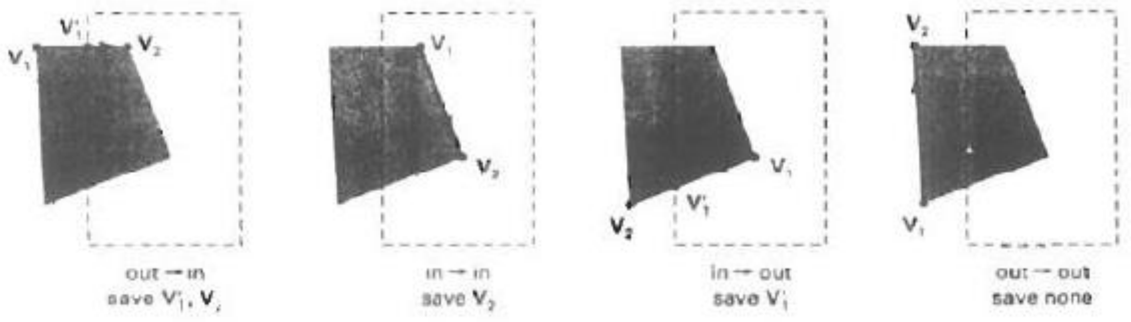
1. If the first vertex is outside the window boundary and second vertex is inside, both the intersection point of the polygon edge with window boundary and second vertex are added to output vertex list.
2. If both input vertices are inside the window boundary, only the second vertex is added to the output vertex list.

3. If first vertex is inside the window boundary and second vertex is outside only the edge intersection with window boundary is added to output vertex list.
4. If both input vertices are outside the window boundary nothing is added to the output list.

Clipping a polygon against successive window boundaries.



Successive processing of pairs of polygon vertices against the left window boundary



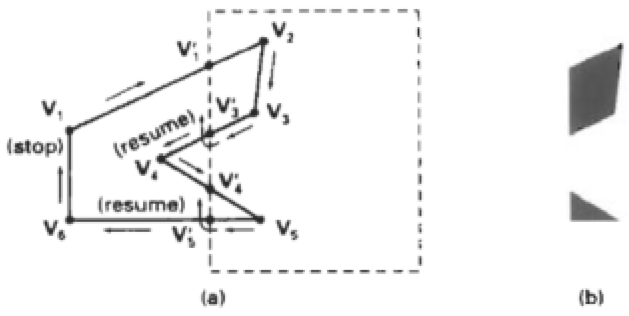
Weiler- Atherton Polygon Clipping

This clipping procedure was developed as a method for identifying visible surfaces, and so it can be applied with arbitrary polygon-clipping regions.

The basic idea in this algorithm is that instead of always proceeding around the polygon edges as vertices are processed, we sometimes want to follow the window boundaries. Which path we follow depends on the polygon-processing direction (clockwise or counterclockwise) and whether the pair of polygon vertices currently being processed represents an outside-to-inside pair or an inside- to-outside pair. For clockwise processing of polygon vertices, we use the following rules:

- For an outside-to-inside pair of vertices, follow the polygon boundary.
- For an inside-to-outside pair of vertices,. follow the window boundary in a clockwise direction.

In the below Fig. the processing direction in the Weiler-Atherton algorithm and the resulting clipped polygon is shown for a rectangular clipping window.



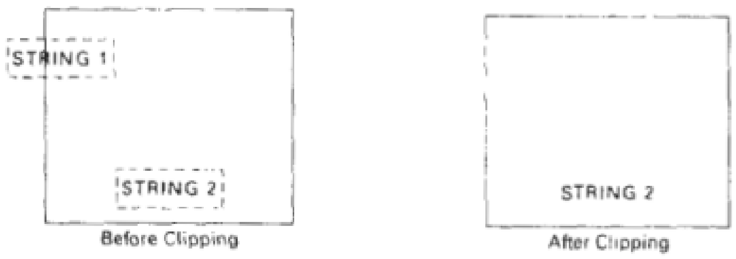
An improvement on the Weiler-Atherton algorithm is the Weiler algorithm, which applies constructive solid geometry ideas to clip an arbitrary polygon against any polygon clipping region.

Text Clipping

There are several techniques that can be used to provide text clipping in a graphics package. The clipping technique used will depend on the methods used to generate characters and the requirements of a particular application.

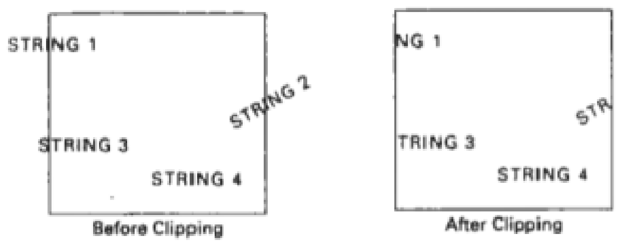
The simplest method for processing character strings relative to a window boundary is to use the **all-or-none string-clipping** strategy shown in Fig. . If all of the string is inside a clip window, we keep it. Otherwise, the string is discarded. This procedure is implemented by considering a bounding rectangle around the text pattern. The boundary positions of the rectangle are then compared to the window boundaries, and the string is rejected if there is any overlap. This method produces the fastest text clipping.

Text clipping using a bounding rectangle about the entire string



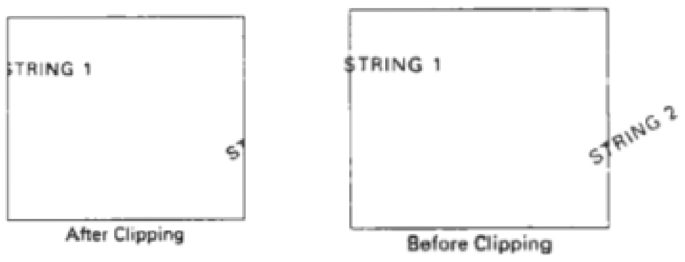
An alternative to rejecting an entire character string that overlaps a window boundary is to use the **all-or-none character-clipping** strategy. Here we discard only those characters that are not completely inside the window .In this case, the boundary limits of individual characters are compared to the window. Any character that either overlaps or is outside a window boundary is clipped.

Text clipping using a bounding rectangle about individual characters.



A final method for handling text clipping is to clip the components of individual characters. We now treat characters in much the same way that we treated lines. If an individual character overlaps a clip window boundary, we clip off the parts of the character that are outside the window.

Text Clipping performed on the components of individual characters



Exterior Clipping

Procedure for clipping a picture to the interior of a region by eliminating everything outside the clipping region. By these procedures the inside region of the picture is saved. To clip a picture to the exterior of a specified region. The picture parts to be saved are those that are outside the region. This is called as exterior clipping.

Objects within a window are clipped to interior of window when other higher priority window overlap these objects. The objects are also clipped to the exterior of overlapping windows.

Three-Dimensional Concept

Three-Dimensional Display Methods:

- To obtain a display of a three dimensional scene that has been modeled in world coordinates, we must setup a co-ordinate reference for the 'camera'.
- This coordinate reference defines the position and orientation for the plane of the camera film which is the plane we want to use to display a view of the objects in the scene.
- Object descriptions are then transferred to the camera reference coordinates and projected onto the selected display plane.
- The objects can be displayed in wire frame form, or we can apply lighting and surface rendering techniques to shade the visible surfaces.

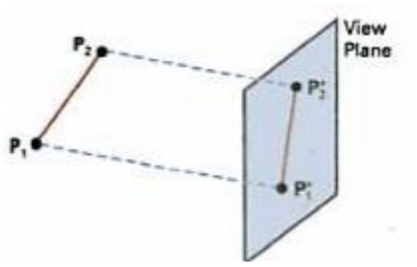
Projections

Once world coordinate description of the objects is converted to viewing coordinates, we can project the 3-dimensional objects onto the two dimensional view planes.

There are two basic types of projections:

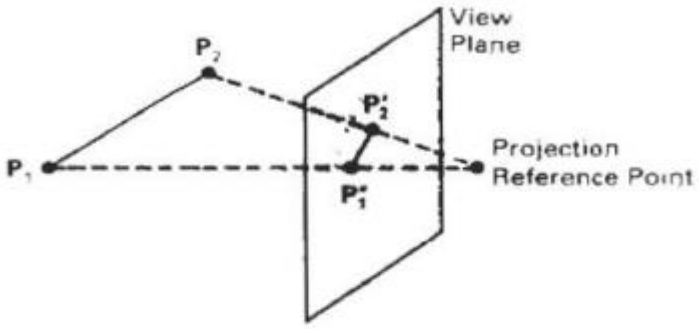
1. Parallel Projection: - Here the coordinate positions are transformed to the view plane along parallel lines.

   Parallel project of an object to the view plane



2. Perspective Projection – Here, object positions are transformed to the view plane along lines that converge to a point called the projection reference point.

Perspective projection of an object to the view plane



Parallel Projections

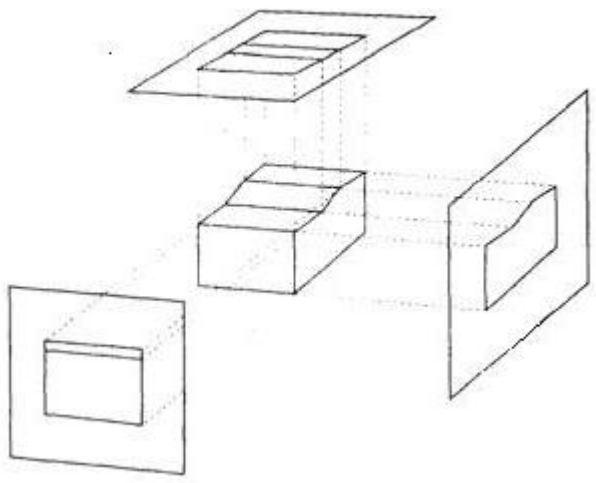
- Parallel projections are specified with a projection vector that defines the direction for the projection lines.
- When the projection in perpendicular to the view plane, it is said to be an orthographic parallel projection, otherwise it said to be an oblique parallel projection.

Orientation of the projection vector Vp to produce an orthographic parallel projection (a) and an oblique projection(b).

Orthographic Projection

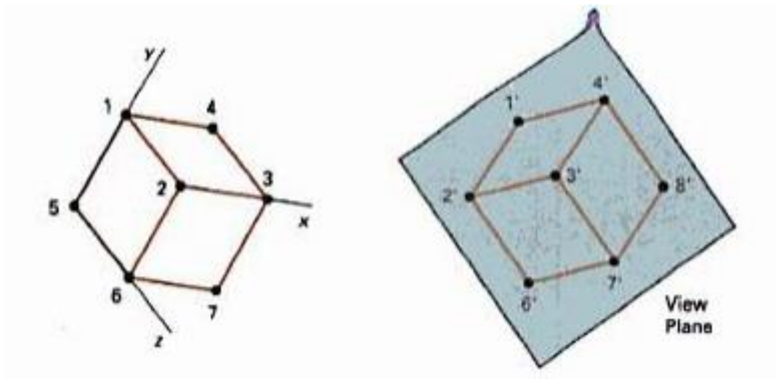
- Orthographic projections are used to produce the front, side and top views of an object.
- Front, side and rear orthographic projections of an object are called elevations.
- A top orthographic projection is called a plan view.
- This projection gives the measurement of lengths and angles accurately.

Orthographic projections of an object, displaying plan and elevation views



- The orthographic projection that displays more than one face of an object is called axonometric orthographic projections.
- The most commonly used axonometric projection is the isometric projection.
- It can be generated by aligning the projection plane so that it interests each coordinate axis in which the object is defined as the same distance from the origin.

Isometric Position of a Cube



- Transformation equations for an orthographic parallel projection are straight forward.

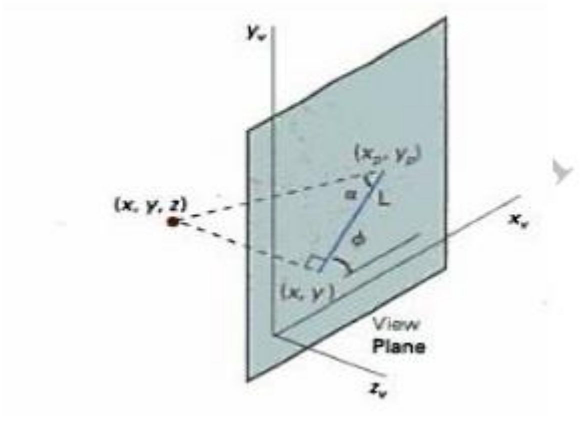
- If the view plane is placed at position z_{vp} along the z_v axis then any point (x,y,z) in viewing coordinates is transformed to projection coordinates as

$$x_p = x, \quad y_p = y$$

where the original z coordinates value is kept for the depth information needed in depth cueing and visible surface determination procedures.

Oblique Projection

- An oblique projection in obtained by projecting points along parallel lines that are not perpendicular to the projection plane.
- The below figure α and φ are two angles.



- Point (x,y,z) is projected to the position (Xp,Yp) on the view plane.
- The oblique projection line form(x,y,z) to (Xp,Yp) and (X,Y).
- This line of length L in at an angle ϕ with the horizontal direction in the projection plane.

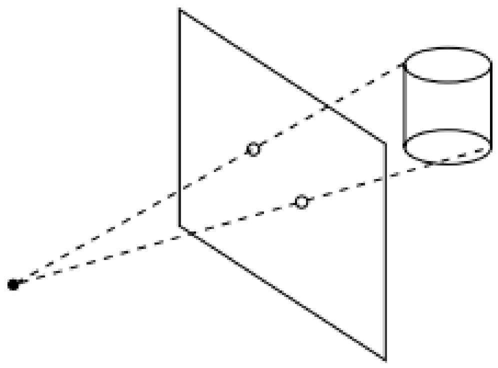
Perspective Projections

- To obtain perspective projection of a 3D object, we transform points along projection lines that meet at the projection reference point.
- Size Varies inversely with distance-looks realistic.

Projections

Projections transform points in n-space to m-space, where m < n.

In 3D, we map points from 3-space to the **projection plane (PP)** along **projectors** emanating from the **center of projection (COP)**.

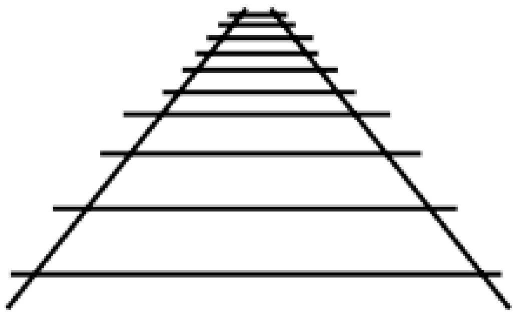


There are two basic types of projections:

- **Perspective** - distance from COP to PP finite
- **Parallel** - distance from COP to PP infinite

Vanishing points

Under perspective projections, any set of parallel lines that are not parallel to the PP will converge to a **vanishing point**.



Vanishing points of lines parallel to a principal axis x, y, or z are called **principal vanishing points**.

How many of these can there be?

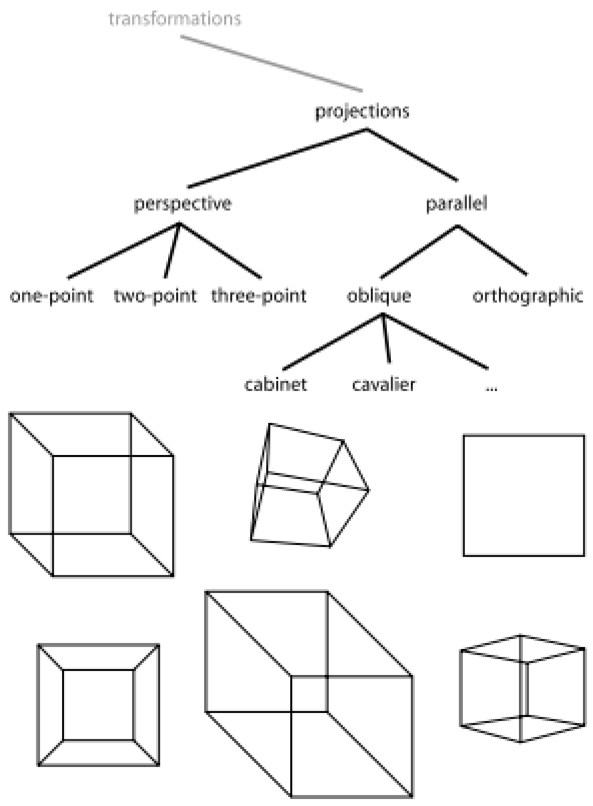
Types of perspective drawing

Perspective drawings are often classified by the number of principal vanishing points.

- One-point perspective — simplest to draw
- Two-point perspective — gives better impression of depth
- Three-point perspective — most difficult to draw

All three types are equally simple with computer graphics.

Projection taxology



COMPUTER GHRAPHICS AND MULTIMEDIA

Module 04

Introduction to Multimedia

Multimedia

Multimedia is a process of putting many media together. Presenting information in various formats is nothing new, but multimedia generally implies presenting information in various digital formats. Multimedia finds its application in various areas including, but not limited to, art, education, entertainment, engineering, medicine, mathematics, business, and scientific research.

In education, multimedia is used to produce computer-based training courses (popularly called CBTs) and reference books like encyclopaedia and almanacs. A CBT lets the user go through a series of presentations, text about a particular topic, and associated illustrations in various information formats.

The Multimedia Messaging System, or MMS, is an application that allows one to send and receive messages containing Multimedia - related content. MMS is a common feature of most cell phones. The basic elements of multimedia on a computer are:

- Text
- Still images
- Sound
- Movies
- Animations
- Special Effects

Text, still images and the video portion of movies are functions of your monitor, your video card and the software driver that tells Windows how your video card works. Your monitor is essentially a grid of closely spaced little luminous points called pixels which can be turned on and off like tiny light bulbs. For the sake of simplicity we'll extend our above example to say that the little bulbs can be lighted with a number of colors. Just how close together those points of light are packed is a function of your monitor. The number of colors that the luminescent points can display is a function of the monitor in concert with the video card.

Uses of Multimedia

Multimedia finds its application in various areas including, but not limited to, advertisements, art, education, entertainment, engineering, medicine, mathematics, business, scientific research and spatial temporal applications.

Below are the several examples as follows:

Entertainment and fine arts:

Multimedia is heavily used in the entertainment industry, especially to develop special effects in movies and animations. Multimedia games are a popular pastime and are software programs available either as CD-ROMs or online. Some video games also use multimedia features.

In the Arts there are multimedia artists, whose minds are able to blend techniques using different media that in some way incorporates interaction with the viewer. One of the most relevant could be Peter Greenaway who is melding Cinema with Opera and all sorts of digital media.

Education:

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

Engineering:

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

Industry:

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

Mathematical and Scientific Research:

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

Medicine:

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

Applications of Multimedia

Examples of Multimedia Applications include:

- World Wide Web
- Hypermedia courseware
- Video conferencing
- Video-on-demand
- Interactive TV
- Groupware
- Home shopping
- Games
- Virtual reality
- Digital video editing and production systems

Multimedia authoring Tools

Multimedia authoring tools provide the important framework you need for organizing and editing the elements of multimedia like graphics, sounds, animations and video clips. Authoring tools are used for designing interactivity and the user interface, for presentation your project on screen and assembling multimedia elements into a single cohesive project. Authoring software provides an integrated environment for binding together the content and functions of your project. Authoring systems typically include the ability to create, edit and import specific types of data; assemble raw data into a playback sequence or cue sheet and provide structured method or language for responding to user input.

Types of Authoring Tools

The various authoring tools can be classified in three categories based on the metaphor used for sequencing or organizing multimedia elements and events.

•Card or page based tools
•Icon base, event driven tools
•Time base and presentation tools

Now let us discuss each of them in detail.

(1) Card or page based tools

In these authoring systems, elements are organized as pages of a book or a stack of cards. These tools are best used when the bulk of your content consists of elements that can be viewed individually, like the pages of a book or cards in a card file. The authoring system lets you link these pages or cards into organized sequences. You can jump, on command, to any page you wish in the structured navigation pattern. It allows you to play sound elements and launch animations and digital video.

(2) Icon based, event driven tools

In these authoring system, multimedia elements and interactions cues are organized as objects in a structural framework or process. Icon-base, event-driven tools simplify the organization of your project and typically display flow diagrams of activities along branching paths. In complicate structures, this charting is particularly useful during development.

(3) Time based tools

In these authoring systems, elements and events are organized along a timeline, with resolutions as high or higher than 1/30 second. Time based tools are best to use when you have a message with a beginning and an end. Sequentially organized graphic frames are played back at a speed that you can set. Other elements are triggered back at a given time or location in the sequence of events. The more powerful time based tools let you program jumps to any location in a sequence, thereby adding navigation and interactive control.

Features of Authoring Tools

Features of multimedia authoring tools are as mention below:

•Editing features
•Organizing features
•Programming features
•Interactive features
•Performance tuning features

•Playback features
•Delivery features
•Cross-Platform features
•Internet Playability
Now let us discuss each of them in detail.

(1) Editing features

The elements of multimedia – image, animation, text, digital audio and MIDI music and video clips – need to be created, edited and converted to standard file formats and the specialized applications, provide these capabilities. Editing tools for these elements, particularly text and still images are often included in your authoring system.

(2) Organizing features

The organization, design and production process for multimedia involves storyboarding and flowcharting. Some authoring tools provide a visual flowcharting system or overview facility for illustrating your project's structure at a macro level. Storyboards or navigation diagrams too can help organize a project. Because designing the interactivity and navigation flow of you project often requires a great deal of planning and programming effort, your story board should describe not just graphics of each screen but the interactive elements as well.

(3) Programming features

Authoring tools that offer a very high level language or interpreted scripting environment for navigation control and for enabling user inputs – such as Macromedia Director, Macromedia Flash, HyperCard, MetaCard and ToolBook are more powerful. The more commands and functions provided in the scripting language, the more powerful the authoring system.
As with traditional programming tools looks for an authoring package with good debugging facilities, robust text editing and online syntax reference.

(4) Interactivity features

Interactivity empowers the end users of your project by letting them control the content and flow of information. Authoring tools should provide one or more levels of interactivity: Simple branching, which offers the ability to go to another section of the multimedia production.Conditional branching, which supports a go-to based on the result of IF-THEN decision or events.

(5) Performance tuning features

Complex multimedia projects require extra synchronization of events. Accomplishing synchronization is difficult because performance varies widely among the different computers used for multimedia development and delivery. Some authoring tools allow you to lock a production's playback speed to specified computer platform, but other provides no ability what so ever to control performance on various systems.

(6) Playback features
When you are developing multimedia project, your will continually assembling elements and testing to see how the assembly looks and performs. Your authoring system should let you build a segment or part of your project and then quickly test it as if the user were actually using it.
(7) Delivery features
Delivering your project may require building a run-time version of the project using the multimedia authoring software. A run-time version allows your project to play back without requiring the full authoring software and all its tools and editors. Many times the run time version does not allow user to access or change the content, structure and programming of the project. If you are going to distribute your project widely, you should distribute it in the run-time version.
(8) Cross-Platform features
It is also increasingly important to use tools that make transfer across platforms easy. For many developers, the Macintosh remains the multimedia authoring platform of choice, but 80% of that developer's target market may be Windows platforms. If you develop on a Macintosh, look for tools that provide a compatible authoring system for Windows or offer a run-time player for the other platform.
(9) Internet Playability
Due to the Web has become a significant delivery medium for multimedia, authoring systems typically provide a means to convert their output so that it can be delivered within the context of HTML or DHTML, either with special plug-in or embedding Java, JavaScript or other code structures in the HTML document.

Multimedia Components

Text

The most basic type of additional information is that which tells us what text or texts we are looking at. A computer file name may give us a clue to what the file contains, but in many cases filenames can only provide us with a tiny amount of information. Information about the nature of the text can often consist of much more than a title and an author. These information fields provide the document with a whole document header which can be used by retrieval programs to search and sort on particular variables. For example, we might only be interested in looking at texts in a corpus that were written by women, so we could ask a computer program to retrieve texts where the author's gender variable is equal to "FEMALE".

When text is correctly structured and formatted, it can be the most flexible way to present content. To make distributed online learning accessible, developers of learning platforms must provide a means to render digital text in alternative formats.

Specifically, it should be possible to render text as:

1).Visual information. Text can be displayed on computer screens or other electronic devices (e.g. personal digital assistants, cell phones, e-book readers).

2).Audio information. Text can be translated into speech using recordings or via synthesized speech provided by a computer.

3).Tactile information. Text can be displayed on refreshable Braille displays or printed using a Braille embosser.

Audio

Audio elements can add to the general appeal of online learning materials while making them more accessible to those who are print-impaired learners, such as those with visual impairments or dyslexia. However, developers should provide alternatives to ensure that learners who are deaf or hard-of-hearing are not disadvantaged.

Images

Images can provide essential information. But without text support, images are not accessible for users who are blind or have low-vision. Developers must provide users with a way to access visual information. Providing text identification, or alternative text, will also benefit users of text-only browsers, such as mobile phones. In addition to providing, developers should ensure that images are scalable, so that users can enlarge them for better clarity.

Animations

Computer animation is the art of creating moving images via the use of computers. It is a subfield of computer graphics and animation. Increasingly it is created by means of 3D computer graphics, though 2D computer graphics are still widely used for stylistic, low bandwidth, and faster real-time rendering needs. Sometimes the target of the animation is the computer itself, but sometimes the target is another medium, such as film. It is also referred to as CGI (Computer-generated imagery or computer-generated imaging), especially when used in films.

To create the illusion of movement, an image is displayed on the computer screen then quickly replaced by a new image that is similar to the previous image, but shifted slightly. This technique is identical to the illusion of movement in television and motion pictures.

Hypertext/Hypermedia

Hypertext is text which is not constrained to be linear. Hypertext is text which contains links to other texts. The term was coined by Ted Nelson around 1965.

Hypertext is text, displayed on a computer, with references (hyperlinks) to other text that the reader can immediately access, usually by a mouse click or keypress sequence. Apart from running text, hypertext may contain tables, images and other presentational devices. Other means of interaction may also be present, such as a bubble with text appearing when the mouse hovers over a particular area, a video clip starting, or a form to complete and submit. The most extensive example of hypertext today is the World Wide Web.

Hypertext documents can either be static or dynamic. Static hypertext can be used to cross-reference collections of data in documents, software applications, or books on CDs. A well-constructed system can also incorporate other user-interface conventions, such as menus and command lines. Hypertext can develop very complex and dynamic systems of linking and cross-referencing. The most famous implementation of hypertext is the World Wide Web and later added to the Internet.

HyperMedia is a term used for hypertext which is not constrained to be text: it can include graphics, video and sound, for example. Apparently Ted Nelson was the first to use this term too.

Hypertext and HyperMedia are concepts, not products.

Hyper Graphics

A linkage between related information by means of a **graphic** image. A hypergraphic is the **graphics** counterpart of hypertext on the Web. Instead of clicking a word or phrase, an icon or image is clicked to jump to another section of the page or to another page.

Animation

Animation has been around for a long time. Mickey Mouse, Donald Duck, Tom & Jerry and many of our favorite cartoons are animated. To become an animator, you have to possess a basic understanding of how body parts work in a given space. For those who are entirely new to the field of animation, the process involves creating some sort of movement by showing a series of pictures in quick succession to give the illusion of movement. For the uninitiated, here are the most well-known styles of animation and what they involve.

Traditional Animation :Traditional animation is a long process. Each frame is painstakingly drawn. Whether the frame showcases the movement of a finger or a change in facial expression, each movement has a set of frames.This tedious process is used in old- school animated movies. Re-watch an old Disney favorite to see how accurately each frame was drawn for the end product to look this great.

2D Animation:Vector Based The vector-based 2D animation is similar to traditional animation. However, this style of animation is created using computer programs such as Flash. Animators using this process have the option of not drawing each frame individually. They can instead opt to move only a body part or object in the frame to show movement.
3D Animation: Computer Based Have you ever been wowed by the on-screen graphics in a movie? That is 3D animation.3D animation requires a unique set of technical skills. This technique is essentially like showing the movement of a puppet on screen, rather than showing movement through frames.
Motion Graphics Animation:This animation method is different from the entire lot since it involves moving around graphical elements. Animated logos, film titles, ad commercials and educational videos employ this method. More often than not, motion graphics involves using pieces of text to create an animation.

Stop Motion Animation :If you are someone who possesses an infinite amount of patience, stop motion animation is just the thing for you. Falling under this category is Claymation, Lego figure animation, Cut-outs, and Pixelation. Stop motion requires photographing an object in a sequence of pictures. Even the slightest movement gets its own shot. To ensure that there is fluidity in the animation and no hard breaks, the animator has to ensure that each movement is captured in a proper order. The history of animation is quite fascinating. What started as a hand-drawn process has slowly evolved into one that is done entirely on a computer. Technological advances have opened up new frontiers in the field of animation. However, each technique has remained unique in its own way.

What Is MATLAB?

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include:

- Math and computation
- Algorithm development
- Modeling, simulation, and prototyping
- Data analysis, exploration, and visualization
- Scientific and engineering graphics
- Application development, including Graphical User Interface building

MATLAB is an interactive system whose basic data element is an array that does not require dimensioning. This allows you to solve many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar noninteractive language such as C or Fortran.
The name MATLAB stands for matrix laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects, which together represent the state-of-the-art in software for matrix computation.

MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis.
MATLAB features a family of application-specific solutions called toolboxes. Very important to most users of MATLAB, toolboxes allow you to *learn* and *apply* specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

The MATLAB System

The MATLAB system consists of five main parts:

The MATLAB language.

This is a high-level matrix/array language with control flow statements, functions, data structures, input/output, and object-oriented programming features. It allows both "programming in the small" to rapidly create quick and dirty throw-away programs, and "programming in the large" to create complete large and complex application programs.

The MATLAB working environment.

This is the set of tools and facilities that you work with as the MATLAB user or programmer. It includes facilities for managing the variables in your workspace and importing and exporting data. It also includes tools for developing, managing, debugging, and profiling M-files, MATLAB's applications.

Handle Graphics.

This is the MATLAB graphics system. It includes high-level commands for two-dimensional and three-dimensional data visualization, image processing, animation, and presentation graphics. It also includes low-level commands that allow you to fully customize the appearance of graphics as well as to build complete Graphical User Interfaces on your MATLAB applications.

The MATLAB mathematical function library.

This is a vast collection of computational algorithms ranging from elementary functions like sum, sine, cosine, and complex arithmetic, to more sophisticated functions like matrix inverse, matrix eigenvalues, Bessel functions, and fast Fourier transforms.

The MATLAB Application Program Interface (API).

This is a library that allows you to write C and Fortran programs that interact with MATLAB. It include facilities for calling routines from MATLAB (dynamic linking), calling MATLAB as a computational engine, and for reading and writing MAT-files.

Uses of MATLAB

MATLAB is used in a lot of different ways by lots of people in occupations you might not necessarily think about when it comes to a math product. In fact, math is used in ways that many people don't consider.

Engineering new solutions

Many engineering disciplines rely on various kinds of math to ensure that the results of any design process or new theory of how the universe works actually make sense. A new building isn't much use if it can't hold up to the stresses placed on it.

Getting an education

Mathworks places a strong emphasis on education. In fact, you can find a special place for education-related materials at MathWorks.com. Even if the organization that employs you doesn't use MATLAB, the principles you learn by working through problems with MATLAB follow standards that apply equally well to other products. If you're a student and you need a copy of MATLAB, you can easily purchase it. Several versions of MATLAB are available for student use, so make sure that you pick the package that best suits your needs. Note that you may require some special add-ons for your classes.

Working with linear algebra

It may be hard to believe, but linear algebra really is part of the workplace. For example, to calculate Return on Investment (ROI), you must know algebra. The same holds true for the following:

- Predicting the amount of turnover a company will have
- Determining how many items to keep in inventory
- Making life and business decisions, such as whether it's cheaper to rent a car or to buy one outright
- Creating a financial plan, such a determining whether it makes more sense to pay down a credit card or build up savings

Performing numerical analysis

Numerical analysis relies on approximation rather than the precision you see in symbolic math. Performing certain building construction tasks is impossible without applying numerical analysis, and astronomy seems to require heavy use of it as well. You probably won't see a carpenter applying numerical analysis, but you will see architects who might need to do so.

Getting involved in science

MATLAB is likely to be used to explore new theories. When applied to science, MATLAB helps you perform “what if” analysis that helps you confirm the viability of a theory.

Of course, science is used in many different ways. For example, you might be involved in the health industry and using science to find a cure for cancer or the Ebola virus. A computer scientist might look for a new way to use computer technology to aid those with accessibility needs.

Engaging mathematics

Some people simply enjoy playing with math. It’s the reason that so many theorems are available today to solve problems. These people are engaged with math in a way that few others can readily understand. MATLAB makes it possible to play with math, to create new ways of using numbers to perform useful tasks.

Exploring research

After a question is asked and an answer is given, a researcher must convince colleagues that the answer is correct and then viable to put into practice. MATLAB lets you check the answer and verify that it does, in fact, work as the researcher suggests. After an answer is proven, the researcher can use MATLAB further to define precisely how the answer is used.

Walking through a simulation

Using a simulation rather than a real-world counterpart is a low-cost approach to testing that is an essential part of any sort of scientific or engineering endeavor today, for these reasons:

- Saves human lives
- Saves time
- Enhances the ability of the people involved to try various solutions
- Reduces costs
- Improves the chances of a new technology succeeding
- Increases the security surrounding a new technology

MATLAB makes simulations possible in several different ways. It may not always provide a complete solution, but you can use it to perform these kinds of tasks:

- Define the original math model used to define the technology and therefore the simulation
- Create individual snapshots showing how the technology will work based on the model

- Demonstrate the workflow for a technology using animation techniques so that even less-skilled stakeholders can see the technology at work

Employing image processing

Image processing is the act of managing the pixels in an image using math techniques to modify the matrix values. Techniques such as adding two matrices together are common when performing image processing.

Embracing programming using computer science

Computer scientists rely heavily on math to perform tasks. MATLAB, with its rich toolbox, can be used to rapidly prototype an algorithm before committing the development resources to implementing the algorithm in another language, such as C++ or Java. Programmers commonly depend on MATLAB to enhance their productivity.

When creating an application, you must ensure that the output is valid. However, verification is just one use. Computer scientists deal with a nearly inexhaustible supply of unknowns that could potentially benefit from the use of MATLAB.