DMI – St. Eugene University

(Run by sisters of Daughters of Mary Immaculate and Collaborators)

Module Code : 351 CS 63

Module Name : Computer Graphics

Stream : B. Sc. (CS)

Unit I

Overview of Computer Graphics System: Over View of Computer Graphics System – Video display devices – Raster Scan and random scan system – Input devices – Hard copy devices

Unit II

Output Primitives and Attributes: Drawing line, circle and ellipse generating algorithms – Scan line algorithm – Character generation – attributes of lines, curves and characters – Antialiasing

Unit III

Two-dimensional Geometric Transformations – Windowing and Clipping – Clipping of lines and clipping of polygons

Unit IV

Three-dimensional concepts – Object representations- Polygon table, quadric surfaces, Splines, Bezier curves and surfaces – Geometric and Modelling transformations – Viewing - Parallel and perspective projections.

Unit V

Visible Surface Detection Methods – Computer Animation

UNIT – I

Overview of Computer Graphics System: Over View of Computer Graphics System – Video display devices – Raster Scan and random scan system – Input devices – Hard copy devices.

Definition

Pictures and **movies** created using **computers** - usually referring to **image** data created by a computer specifically with help from specialized graphic hardware and software**.** The development of computer graphics, has made computers easier to interact with, and better for understanding and interpreting many types of data. Developments in computer graphics have had a profound impact on many types of media and have revolutionized animation, movies and the video game industry.

Overview

The term computer graphics has been used in a broad sense to describe "almost everything on computers that is not text or sound. Typically, the term *computer graphics* refers to several different things

- the representation and manipulation of image data by a computer
- the various technologies used to create and manipulate images
- the images so produced, and
- the sub-field of computer science which studies methods for digitally synthesizing and manipulating visual content, see study of computer graphics

Today, computers and computer-generated images touch many aspects of daily life. Computer imagery is found on television, in newspapers, for example in weather reports, or for example in all kinds of medical investigation and surgical procedures. A well-constructed graph can present complex statistics in a form that is easier to understand and interpret. In the media "such graphs are used to illustrate papers, reports, theses", and other presentation material

Many powerful tools have been developed to visualize data. Computer generated imagery can be categorized into several different types: 2D, 3D, 5D, and animated graphics. As technology has improved, 3D computer graphics have become more common, but 2D computer graphics are still

widely used. Computer graphics has emerged as a sub-field of computer science which studies methods for digitally synthesizing and manipulating visual content. Over the past decade, other specialized fields have been developed like information visualization, and scientific visualization more concerned with "the visualization of three dimensional phenomena (architectural, meteorological, medical, biological, etc.), where the emphasis is on realistic renderings of volumes, surfaces, illumination sources, and so forth, perhaps with a dynamic (time) component".

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 are mainly used in applications that were originally developed upon traditional printing and drawing technologies, such as typography, cartography, technical drawing, advertising, etc.. In those applications, the two-dimensional image is not just a representation of a real-world object, but an independent artefact with added semantic value; two-dimensional models are therefore preferred, because they give more direct control of the image than 3D computer graphics, whose approach is more akin to photography than to typography.

Pixel art

Pixel art is a form of digital art, created through the use of raster graphics software, where images are edited on the pixel level. Graphics in most old (or relatively limited) computer and video games, graphing calculator games, and many mobile phone games are mostly pixel art.

Vector graphics

Vector graphics formats are complementary to raster graphics, which is the representation of images as an array of pixels, as it is typically used for the representation of photographic images. There are instances when working with vector tools and formats is best practice, and instances when working with raster tools and formats is best practice. There are times when both formats come together. An understanding of the advantages

and limitations of each technology and the relationship between them is most likely to result in efficient and effective use of tools.

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.

Despite these differences, 3D computer graphics rely on many of the same algorithms as 2D computer vector graphics in the wire frame model and 2D computer raster graphics in the final rendered display. In computer graphics software, the distinction between 2D and 3D is occasionally blurred; 2D applications may use 3D techniques to achieve effects such as lighting, and primarily 3D may use 2D rendering techniques.

3D computer graphics are often referred to as 3D models. Apart from the rendered graphic, the model is contained within the graphical data file. However, there are differences. A 3D model is the mathematical representation of any three-dimensional object. A model is not technically a graphic until it is visually displayed. Due to 3D printing, 3D models are not confined to virtual space. A model can be displayed visually as a two-dimensional image through a process called *3D rendering*, or used in non-graphical computer simulations and calculations.

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.

Concepts and principles

Image

An image or picture is an artefact that resembles a physical object or person. The term includes two-dimensional objects like photographs and sometimes includes three-dimensional representations. Images are captured by optical devices—such as cameras, mirrors, lenses, telescopes, microscopes, etc. and natural objects and phenomena, such as the human eye or water surfaces.

A digital image is a representation of a two-dimensional image in binary format as a sequence of ones and zeros. Digital images include both vector images and raster images, but raster images are more commonly used.

Pixel

In the enlarged portion of the image individual pixels are rendered as squares and can be easily seen.

In digital imaging, a pixel (or picture element) is a single point in a raster image. Pixels are normally arranged in a regular 2-dimensional grid, and are often represented using dots or squares. Each pixel is a sample of an original image, where more samples typically provide a more accurate representation of the original. The intensity of each pixel is variable; in color systems, each pixel has typically three components such as red, green, and blue.

Graphics

Graphics are visual presentations on some surface, such as a wall, canvas, computer screen, paper, or stone to brand, inform, illustrate, or entertain. Examples are photographs, drawings, line art, graphs, diagrams, typography, numbers, symbols, geometric designs, maps, engineering drawings, or other images. Graphics often combine text, illustration, and color. Graphic design may consist of the deliberate selection, creation, or arrangement of typography alone, as in a brochure, flier, poster, web site, or book without any other element. Clarity or effective communication may be the objective, association with other cultural elements may be sought, or merely, the creation of a distinctive style.

Rendering

Rendering is the process of generating an image from a model, by means of computer programs. The model is a description of three dimensional objects in a strictly defined language or data structure. It would contain geometry, viewpoint, texture, lighting, and shading information. The image is a digital image or raster graphics image. The term may be by analogy with an "artist's rendering" of a scene. 'Rendering' is also used to describe the process of calculating effects in a video editing file to produce final video output.

3D projection

> 3D projection is a method of mapping three dimensional points to a two dimensional plane. As most current methods for displaying graphical data are based on planar two dimensional media, the use of this type of projection is widespread, especially in computer graphics, engineering and drafting.

Ray tracing

> Ray tracing is a technique for generating an image by tracing the path of light through pixels in an image plane. The technique is capable of producing a very high degree of photorealism; usually higher than that of typical scan line rendering methods, but at a greater computational cost.

Shading

> Shading refers to depicting depth in 3D models or illustrations by varying levels of darkness. It is a process used in drawing for depicting levels of darkness on paper by applying media more densely or with a darker shade for darker areas, and less densely or with a lighter shade for lighter areas. There are various techniques of shading including cross hatching where perpendicular lines of varying closeness are drawn in a grid pattern to shade an area. The closer the lines are together, the darker the area appears. Likewise, the farther apart the lines are, the lighter the area appears. The term has been recently generalized to mean that shaders are applied.

Texture mapping

Texture mapping is a method for adding detail, surface texture, or colour to a computer-generated graphic or 3D model. Its application to 3D graphics was pioneered by Dr Edwin Catmull in 1974. A texture map is applied (mapped) to the surface of a shape, or polygon. This process is akin to applying patterned paper to a plain white box. Multitexturing is the use of more than one texture at a time on a polygon.[6] Procedural textures (created from adjusting parameters of an underlying algorithm that produces an output texture), and bitmap textures (created in an image editing application) are, generally speaking, common methods of implementing texture definition from a 3D animation program, while intended placement of textures onto a model's surface often requires a technique known as UV mapping.

Volume rendering

Volume rendered CT scan of a forearm with different colour schemes for muscle, fat, bone, and blood.

Volume rendering is a technique used to display a 2D projection of a 3D discretely sampled data set. A typical 3D data set is a group of 2D slice images acquired by a CT or MRI scanner.

Usually these are acquired in a regular pattern (e.g., one slice every millimeter) and usually have a regular number of image pixels in a regular pattern. This is an example of a regular volumetric grid, with each volume element, or voxel represented by a single value that is obtained by sampling the immediate area surrounding the voxel.

3D modeling

3D modeling is the process of developing a mathematical, wireframe representation of any three-dimensional object, called a "3D model", via specialized software. Models may be created automatically or manually; the manual modeling process of preparing geometric data for 3D computer graphics is similar to plastic arts such as sculpting. 3D models

may be created using multiple approaches: use of NURBS curves to generate accurate and smooth surface patches, polygonal mesh modeling (manipulation of faceted geometry), or polygonal mesh subdivision (advanced tessellation of polygons, resulting in smooth surfaces similar to NURBS models). A 3D model can be displayed as a two-dimensional image through a process called *3D rendering*, used in a computer simulation of physical phenomena, or animated directly for other purposes. The model can also be physically created using 3D Printing devices.

The study of computer graphics

The study of computer graphics is a sub-field of computer science which studies methods for digitally synthesizing and manipulating visual content. Although the term often refers to three-dimensional computer graphics, it also encompasses two-dimensional graphics and image processing.

As an academic discipline, computer graphics studies the manipulation of visual and geometric information using computational techniques. It focuses on the *mathematical* and *computational* foundations of image generation and processing rather than purely aesthetic issues. Computer graphics is often differentiated from the field of visualization, although the two fields have many similarities.

Applications

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

Many software applications include graphics components. Such programs are said to *support* graphics. For example, certain word processors support graphics because they let you draw or import pictures. All CAD/CAM systems support graphics. Some database management systems and spreadsheet programs support graphics because they let you display data in the form of graphs and charts. Such applications are often referred to as *business graphics.*

- The following are also considered *graphics applications :*

  paint programs : Allow you to create rough freehand drawings. The images are stored as bit maps and can easily be edited.

  illustration/design programs: Supports more advanced features than paint programs, particularly for drawing curved lines. The images are usually stored in vector-based formats. Illustration/design programs are often called *draw programs*.

  presentation graphics software : Lets you create bar charts, pie charts, graphics, and other types of images for slide shows and reports. The charts can be based on data imported from spreadsheet applications.

  animation software: Enables you to chain and sequence a series of images to simulate movement. Each image is like a frame in a movie.

  CAD software: Enables architects and engineers to draft designs.

  desktop publishing : Provides a full set of word-processing features as well as fine control over placement of text and graphics, so that you can create newsletters, advertisements, books, and other types of documents.

Video display devices
Definition:

A **display device** is an output device for presentation of information for visual, tactile or auditive reception, acquired, stored, or transmitted in various forms. When the input information is supplied as an electrical signal,

the display is called *electronic display*. *Electronic displays* are available for presentation of visual, tactile and auditive information.

Tactile electronic displays (aka *refreshable Braille display*) are usually intended for the blind or visually impaired, they use electro-mechanical parts to dynamically update a tactile image (usually of text) so that the image may be felt by the fingers.

Common applications for *electronic visual displays* are television sets or computer monitors.

Some of Video display Devices:

- Cathode ray tube (CRT)
  - Storage tube
- Bistable display
- Electronic paper
- Nixie tube displays
- Vector display
- Flat panel display
- Vacuum fluorescent display (VF)
- Light-emitting diode (LED) displays
- Electroluminescent display (ELD)
- Plasma display panels (PDP)
- Liquid crystal display (LCD)
  - HPA display
  - Thin-film transistor displays (TFT)
- Organic light-emitting diode displays (OLED)
- Surface-conduction electron-emitter display (SED) (experimental)
- Laser TV (forthcoming)
- Carbon nanotubes (experimental)
- Nanocrystal displays (experimental), using quantum dots to make vibrant, flexible screens.
- Head-mounted display

Projectors

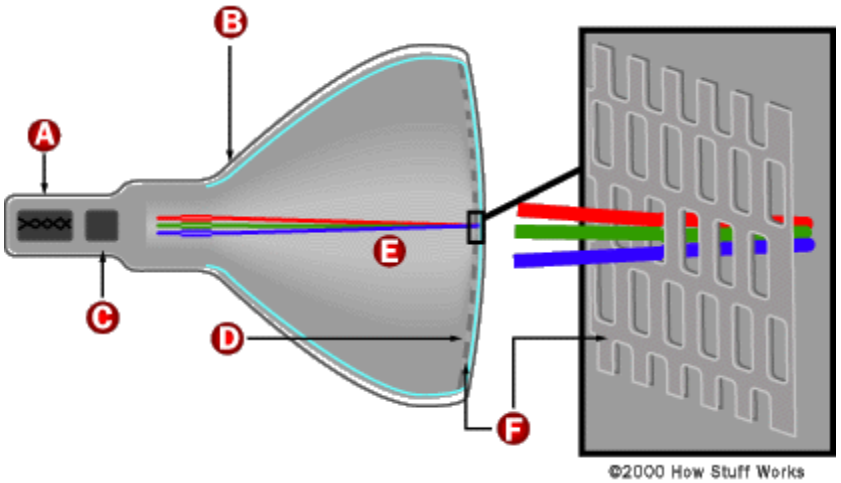
- Film projectors
  - Movie projector
  - Slide projector
- Digital projector

    - Video projector
    - LCD projector
- Laser projector
- Head-up display

The Cathode Ray Tube

Cathode rays (also called an **electron beam** or **e-beam**) are streams of electrons observed in vacuum tubes, i.e. evacuated glass tubes that are equipped with at least two metal electrodes to which a voltage is applied, a cathode or negative electrode and an anode or positive electrode. Electrons were first discovered as the constituents of cathode rays.

Almost all TVs in use today rely on a device known as the **cathode ray tube**, or **CRT**, to display their images. LCDs and plasma displays are sometimes seen, but they are still rare when compared to CRTs. It is even possible to make a television screen out of thousands of ordinary 60-watt light bulbs! You may have seen something like this at an outdoor event like a football game. Let's start with the CRT, however, because CRTs are the most common way of displaying images today.



The terms **anode** and **cathode** are used in electronics as synonyms for positive and negative terminals. For example, you could refer to the positive terminal of a battery as the anode and the negative terminal as the cathode.

Principal CRT Advantages

1.Resolution and Aspect Ratio

They operate at any resolution, geometry and aspect ratio without the need for rescaling the image.

2. Highest Resolutions

CRTs run at the highest pixel resolutions generally available.

3. Black-Level and Contrast

Produce a very dark black and the highest contrast levels normally available. Suitable for use even in dimly lit or dark environments.

4. Color and Gray-Scale Accuracy

CRTs produce the very best color and gray-scale and are the reference standard for all professional calibrations. They have a perfectly smooth gray-scale with an infinite number of intensity levels. Other display technologies are expected to reproduce the natural power-law Gamma curve of a CRT, but can only do so approximately.

5. Motion Artifacts

CRTs have fast response times and no motion artifacts. Best for rapidly moving or changing images.

6. Cost

CRTs are less expensive than comparable displays using other display technologies.

Principal CRT Disadvantages

1. Sharpness

The CRT's Gaussian beam profile produces images with softer edges that are not as sharp as an LCD at its native resolution. Imperfect focus and color registration also reduce sharpness. Generally sharper than LCDs at other than native resolutions.

2. Interference

All color CRTs produce annoying Moré patterns. Many monitors include Moré reduction, which normally doesn't eliminate the Moré interference patterns entirely.

3. Geometric Distortion

Subject to geometric distortion and screen regulation problems. Also affected by magnetic fields from other equipment including other CRTs.

4. Brightness

Relatively bright but not as bright as LCDs. Not suitable for very brightly lit environments.

5. Screen Shape

Some CRTs have a rounded spherical or cylindrical shape screen. Newer CRTs are flat.

6. Emissions

CRTs give off electric, magnetic and electromagnetic fields. There is considerable controversy as to whether any of these pose a health hazard, particularly magnetic fields. The most authoritative scientific studies conclude that they are not harmful but some people remain unconvinced.

7. Physical

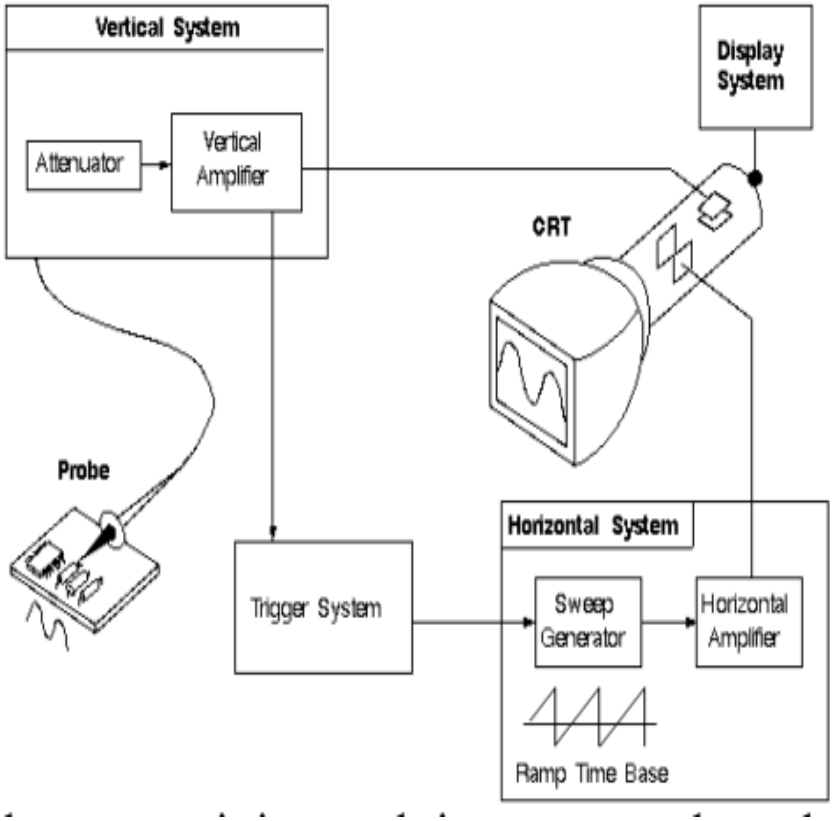
They are large, heavy, and bulky. They consume a lot of electricity and produce a lot of heat.

Vector Scan Display Devices

Vector scan CRTs are used extensively in the Data Display Group ANK/UYA-4(V) plan position indicators (PPIs). The circular display screens provide control and display of conventional radar sweep and video data and computer-generated symbology.The CRTs used in the PPIs use electrostatic deflection The methods used todevelop the deflection and unblinking signals forradar sweep and video are similar because the sameCRT beam is used to develop both presentations. However, the methods used to develop the radar sweep and video are different from the two methods used to develop symbology

In a Electrostatic CRT the best way to "write" on the screen is using vectors. Each vector has 4 associated voltage levels: Xs, Ys, Xe, Ye, which define the position on the screen. A wave form can be divided into a finite number of vectors. there are 2 more parameters to complete the vector image information: intensity and time (from start to end).The persistence (luminance decay) of the phosphors will determine the minimum speed at which I can draw the vector tobe "seen" stable on the screen Prevalent monochromatic display (Black and White or Green)

Vectors application: Analog Scopes



Analog scope: it is a real time scope, where the sweep time is N x Period of the represented wfm. For high frequency signal an high speed (BW) CRT is required.

Strength

- » Well applies to monitors and vector scopes.
- » Real time application
- » Analog memory by Phosphor persistance
- » fast response (350 psstep response in 1GHz scope)

Weaknesses

- » With time the cost of memory went down, the manufacturing cost of electrostatic display increased
- » substantially.
- » Required colors on screen.
- » More information required, for a more complex display has made vector scan circuits more expensive and complex.

Raster Scan Displays

A **raster scan**, or **raster scanning**, is the rectangular pattern of image capture and reconstruction in television. By analogy, the term is used for raster graphics, the pattern of image storage and transmission used in most computer bitmap image systems. The word *raster* comes from the Latin word *rastrum* (a rake), which is derived from *radere* (to scrape); see also rastrum, an instrument for drawing musical staff lines. The pattern left by the tines of a rake, when drawn straight, resembles the parallel lines of a raster: this line-by-line scanning is what creates a raster. It's a systematic process of covering the area progressively, one line at a time. Although often a great deal faster, it's similar in the most-general sense to how one's gaze travels when one reads text.

Description

Scan lines

In a raster scan, an image is subdivided into a sequence of (usually horizontal) strips known as "scan lines". Each scan line can be transmitted in the form of an analog signal as it is read from the video source, as in television systems, or can be further divided into discrete pixels for processing in a computer system. This ordering of pixels by rows is known as raster order, or raster scan order. Analog television has discrete scan lines (discrete vertical resolution), but does *not* have discrete pixels (horizontal resolution) – it instead varies the signal continuously over the scan line. Thus, while the number of scan lines (vertical resolution) is unambiguously defined, the horizontal resolution is more approximate, according to how quickly the signal can change over the course of the scan line.

Strength

- To each pixel is possible to add chrominanceinformation
- The frame and horizontal SCAN is set by the screen resolution
- (1024x728). We talk about pixels and no more about vectors.
- the information resides in the luminance intensity of each pixel.
- Position on screen is predefined by the scan process.
- Less expensive than vector display
- The DSP and graphic processors are cheap and extremely powerful
- Very efficient to represent full images

Weaknesses

- » Slow screen update rate, normally25-120 screen/sec
- » Extensive DSP required to achieve analog persistance simulation
- » At low resolutions pixels are quite “big”
- » NON real time display
- » Improper Interpolation of digital samples can produce visual artefacts

Differences between Raster-Scan –System and Random Scan System

Raster-scan system:-

1)raster displays have less resolution.

2)the lines produced are zig-zag as the plotted values are discrete.

3)high degree realism is achived in picture with the aid of advanced shading and hidden surface technique.

4)decreasing memory costs have made raster systems popular.

Random scan system:-

1)random displays have high resolutions since the picture defination is stored as a set of line drawing commands and not as a set of intensity values.

2)smooth lines are produced as the electron beam directly follows the line path.

3)realism is difficult to achieve.

4)random-scan system's are generally costlier.

Color monitor

A monitor or display (sometimes called a visual display unit) is an electronic visual display for computers. The monitor comprises the display device, circuitry, and an enclosure. The display device in modern monitors is typically a thin film transistor liquid crystal display (TFT-LCD), while older monitors use a cathode ray tube (CRT).

Performance measurements

The performance of a monitor is measured by the following parameters:

- Luminance is measured in candelas per square meter.
- Viewable image size is measured diagonally. For CRTs, the viewable size is typically 1 in (25 mm) smaller than the tube itself.
- Aspect ratios is the ratio of the horizontal length to the vertical length. 4:3 is the standard aspect ratio, for example, so that a screen with a width of 1024 pixels will have a height of 768 pixels. If a

widescreen display has an aspect ratio of 16:9, a display that is 1024 pixels wide will have a height of 576 pixels.

- Display resolution is the number of distinct pixels in each dimension that can be displayed. Maximum resolution is limited by dot pitch.
- Dot pitch is the distance between subpixels of the same color in millimeters. In general, the smaller the dot pitch, the sharper the picture will appear.
- Refresh rate is the number of times in a second that a display is illuminated. Maximum refresh rate is limited by response time.
- Response time is the time a pixel in a monitor takes to go from active (black) to inactive (white) and back to active (black) again, measured in milliseconds. Lower numbers mean faster transitions and therefore fewer visible image artifacts.
- Contrast ratio is the ratio of the luminosity of the brightest color (white) to that of the darkest color (black) that the monitor is capable of producing.
- Power consumption is measured in watts.
- Viewing angle is the maximum angle at which images on the monitor can be viewed, without excessive degradation to the image. It is measured in degrees horizontally and vertically.

Flat panel displays

Flat panel displays encompass a growing number of technologies enabling video displays that are much lighter and thinner than traditional television and video displays that use cathode ray tubes, and are usually less than 100 mm (4 inches) thick. They can be divided into two general categories; *volatile* and *static*.

In many applications, specifically modern portable devices such as laptops, cellular phones, and digital cameras, whatever disadvantages exist are overcome by the portability requirements.

Volatile

Volatile displays require pixels be periodically refreshed to retain their state, even for a static image. This refresh typically occurs many times a second. If this is not done, the pixels will gradually lose their coherent state, and the image will "fade" from the screen.

Examples of volatile flat panel displays

- Plasma displays
- Liquid crystal displays (LCDs)
- Organic light-emitting diode displays (OLEDs)
- Light-emitting diode displays (LED)
- Electroluminescent displays (ELDs)
- Surface-conduction electron-emitter displays (SEDs)
- Field emission displays (FEDs) (also called **Nano-emissive displays** (NEDs))

Only the first five of these displays are commercially available today, though OLED displays are beginning deployment only in small sizes (mainly in cellular telephones). SEDs were promised for release in 2006, while the FEDs (also NEDs) are (as of November 2005) in the prototype stage.

Static

Static flat panel displays rely on materials whose color states are bistable. This means that the image they hold requires no energy to maintain, but instead requires energy to change. This results in a much more energy-efficient display, but with a tendency towards slow refresh rates which are undesirable in an interactive display.

Bistable flat panel displays are beginning deployment in limited applications (Cholesteric displays, manufactured by Magink, in outdoor advertising; electrophoretic displays in e-book products from Sony and iRex; anlabels).

Advantages:

» They consume less desk space
» Flat panels also run alot cooler than normal CRT
» The images and text are sharper
» The visual image does not fade at the edges
» Less reflection from other lights or outdoor sunshine
» Consumes less electricity.

Plasma display panel (PDP)

A **plasma display panel (PDP**) is a type of flat panel display common to large TV displays (80 cm or larger). Many tiny cells between just two panels of glass hold a mixture of noble gases. The gas in the cells is electrically turned into a plasma which emits ultraviolet light which then excites phosphors to emit visible light. Plasma displays should not be confused with LCDs, another lightweight flatscreen display using very different technology, which is not based on phosphorescence.

General characteristics

A typical modern plasma screen television, made by Gradiente of Brazil.

Plasma displays are bright (1,000 lux or higher for the module), have a wide color gamut, and can be produced in fairly large sizes—up to 150 inches (3.8 m) diagonally. They have a very low-luminance "dark-room" black level compared to the lighter grey of the unilluminated parts of an LCD screen. The display panel itself is only about 6 cm (2.5 inches) thick, generally allowing the device's total thickness (including electronics) to be less than 10 cm (4 inches). Plasma displays use as much power per square meter as a CRT or an AMLCD television. Power consumption varies greatly with picture content, with bright scenes drawing significantly more power than darker ones - this is also true of CRTs. Typical power consumption is 400 watts for a 50-inch (127 cm) screen. 200 to 310 watts for a 50-inch (127 cm) display when set to cinema mode. Most screens are set to 'shop' mode by default, which draws at least twice the power (around 500-700 watts) of a 'home' setting of less extreme brightness.Panasonic has greatly reduced power consumption by using Neo-PDP screens in their 2009 series of Viera plasma HDTVs. Panasonic claims that PDPs will consume only half the power of their previous series of plasma sets to achieve the same overall

brightness for a given display size. The lifetime of the latest generation of plasma displays is estimated at 100,000 hours of actual display time, or 27 years at 10 hours per day. This is the estimated time over which maximum picture brightness degrades to half the original value.

Plasma display screens are made from glass, which reflects more light than the material used to make an LCD screen. This causes glare from reflected objects in the viewing area. Companies such as Panasonic coat their newer plasma screens with an anti-glare filter material.Currently, plasma panels cannot be economically manufactured in screen sizes smaller than 32 inches. Although a few companies have been able to make plasma EDTVs this small, even fewer have made 32in plasma HDTVs. With the trend toward larger and larger displays, the 32in screen size is rapidly disappearing. Though considered bulky and thick compared to their LCD counterparts, some sets such as Panasonic's Z1 and Samsung's B860 series are as slim as one inch thick making them comparable to LCDs in this respect.

Competing display technologies include CRT, OLED, LCD, DLP, SED, LED, and FED.

Plasma display advantages and disadvantages

Advantages

- Slim profile
- Can be wall mounted
- Less bulky than rear-projection televisions
- Achieves better and more accurate color reproduction than LCDs (68 billion/2^{36} versus 16.7 million/2^{24})
- Produces deep, true blacks allowing for superior contrast ratios (up to 1:2,000,000
- Far wider viewing angles than those of LCD (up to 178°); images do not suffer from degradation at high angles unlike LCDs
- Virtually no motion blur, thanks in large part to very high refresh rates and a faster response time, contributing to superior performance when displaying content with significant amounts of rapid motion.

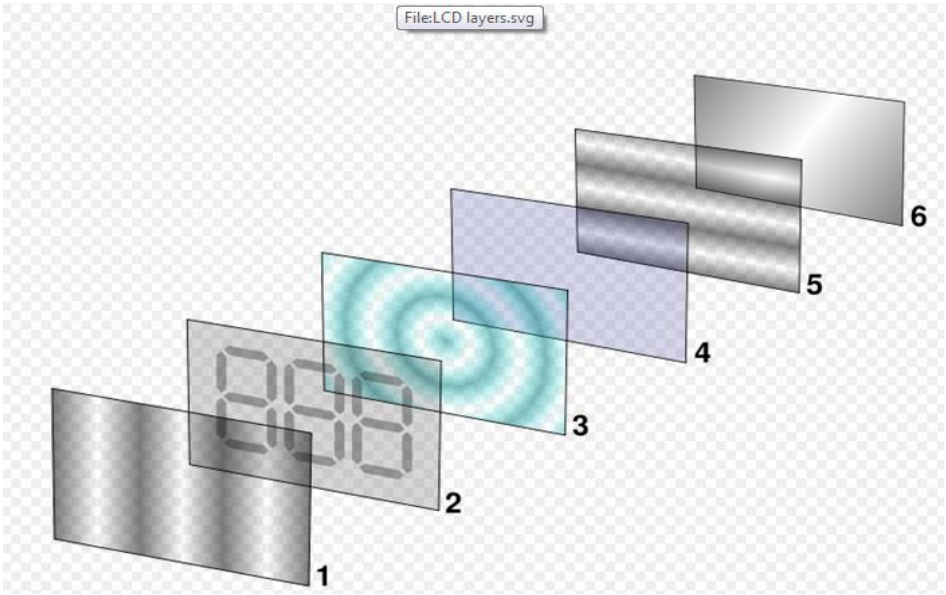
Disadvantages

- Earlier models are susceptible to screen burn-in and image retention (however, newer models have green phosphors and built-in technologies to eliminate this, such as pixel shifting
- Phosphors in older models lose luminosity over time, resulting in gradual decline of absolute image brightness (newer models are less susceptible to this, having lifespans exceeding 60,000 hours, far longer than older CRT technology
- Susceptible to "large area flicker"
- Generally do not come in smaller sizes than 37 inches
- Susceptible to reflection glare in bright rooms
- Heavier than LCD due to the requirement of a glass screen to hold the gases
- Use more electricity, on average, than an LCD TV
- Do not work as well at high altitudes due to pressure differential between the gases inside the screen and the air pressure at altitude. It may cause a buzzing noise. Manufacturers rate their screens to indicate the altitude parameters.
- For those who wish to listen to AM radio, or are Amateur Radio operators (Hams) or Shortwave Listeners (SWL) , the Radio Frequency Interference (RFI) from these devices can be irritating or disabling

Liquid crystal display

A **liquid crystal display** (**LCD**) is a thin, flat electronic visual display that uses the light modulating properties of liquid crystals (LCs). LCs do not emit light directly. This digital technology was first invented in 1971.Initially used in watches and calculators, its functionality was quickly adopted and began redefining computer, medical, and industrial electronics.

They are used in a wide range of applications including: computer monitors, television, instrument panels, aircraft cockpit displays, signage, etc. They are common in consumer devices such as video players, gaming devices, clocks, watches, calculators, and telephones. LCDs have displaced cathode ray tube(CRT) displays in most applications. They are usually more compact, lightweight, portable, and lower cost. They are available in a wider range of screen sizes than CRT and other flat panel displays.



1. Vertical filter film to polarize the light as it enters.
2. Glass substrate with ITO electrodes. The shapes of these electrodes will determine the dark shapes that will appear when the LCD is turned on. Vertical ridges are etched on the surface so the liquid crystals are in line with the polarized light.
3. Twisted nematic liquid crystals.
4. Glass substrate with common electrode film (ITO) with horizontal ridges to line up with the horizontal filter.
5. Horizontal filter film to block/allow through light.
6. Reflective surface to send light back to viewer.

An example of a modern LCD display

Advantages of LCDs

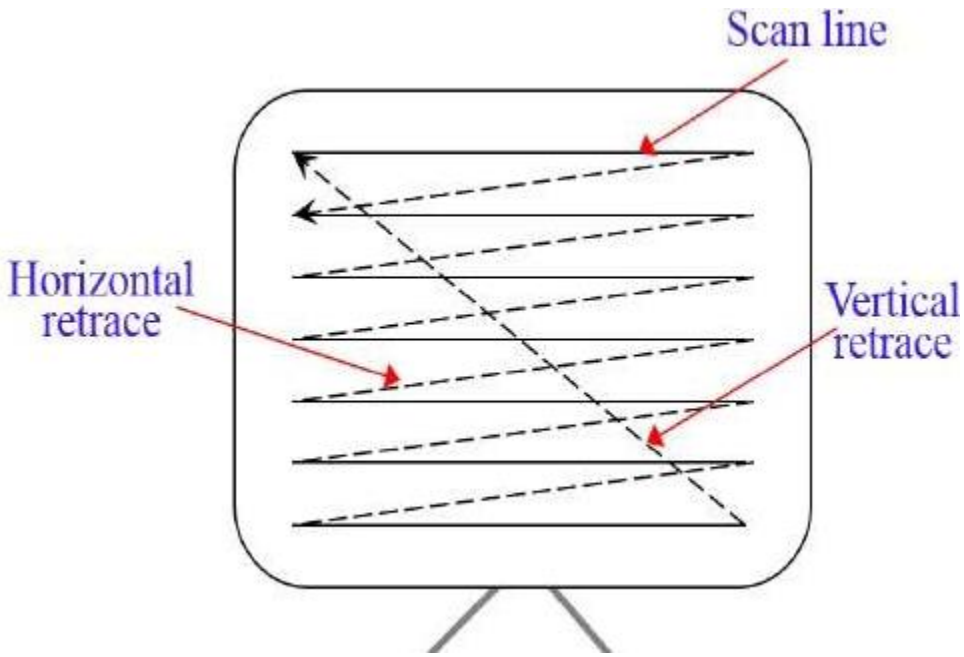
- » Light weight; can be about 15 lbs for a thin LCD.
- » Smaller footprint on desk leaving, freeing up work area on the user's desk.
- » Flicker free screen.
- » More usable display area than on comparably sized CRTs.
- » Low frequency radiation is practically eliminated.
- » Easy adjustment, storage, and movement
- » Energy efficient, using only 1/3 to 1/2 the electricity of CRTs, and they don't ge
- » Potentially less eyestrain due to reduced screen glare.
- » Good for basic web surfing.

Disadvantages:

- » Fragile screen; may result in both screen and backlight lamp damage if touched or handled. Thus, not recommended in environments where it may be handled roughly.
- » Contrast ratio causes darkness to not be displayed true. Darkest areas may be viewed as dark gray rather than black
- » Designed only for one optimum resolution; can't adjust images
- » Best view of screen is straight-on, limiting clarity and colors of information for those viewing from an angle.
- » The backlight is the potential weak link and its failure can be costly. Many warranties only cover it for 1 year

Random Scan

Random Scan (Vector **Scan**) In this technique, the electron beam is directed only to the part of the screen where the picture is to be drawn rather than **scanning** from left to right and top to bottom as in raster **scan**. It is also called vector display, stroke-writing display, or calligraphic display.



Input Devices

An **input device** is any peripheral (piece of computer hardware equipment) used to provide data and control signals to an information processing system (such as a computer). Input and output devices make up the hardware interface between a computer as a scanner or 6DOF controller.

A hardware device that sends information to the computer. Without any input devices a computer would simply be a display device and not allow users to interact with it, much like a TV. To the right is a Logitech trackball mouse and an example of an input device

The following are the commonly used input devices.

a.Keyboard
b.Mouse
c.Image scanners
d.Touch panels
e.Light pen
f.Voice systems
g.Track ball and space ball
h.Joysticks
i.Data Glove
j.Digitizers

Keyboard

An alphanumeric keyboard is used as a device for entering text strings. Function keys allow users to enter frequently used operations in a single keystroke and cursor control keys can be used to select displayed objects or coordinates portions by positioning the screen cursor.

A numeric keypad is included on the keyboard for fast entry of numeric data.

Mouse

A mouse is small handled box used to position the screen cursor.Wheels or rollers on the bottom of the mouse can be used to record the amount and direction of movement.Optical sensor is also used to detect mouse movement.It detects movement across the lines in the grid.

Image scanners

Drawing graphs,color and black-and-white photos, or text can be stored for computer processing with an image scanner by passing an optical scanning mechanism over the information to be stored.

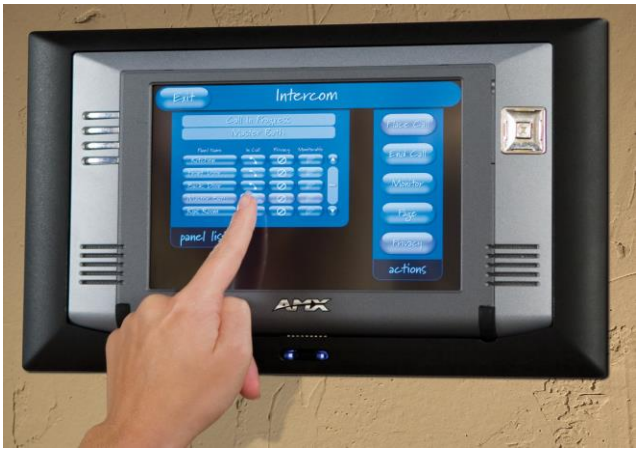
The gradations of gray scale or color are recorded and stored in an array. After getting the internal representation of the picture, we can apply transformation to rotate ,scale or crop the picture to a particular screen area.

Various image processing methods can also be applied to modify the array representation of the picture. For scanned text input,various editing operations can be performed nhe stored documents.

Touch panels

Touch panels allow displayed objects or screen positions to be selected with the touch of the finger. The touch panel is used to select the processing options that are represented with graphical icons.

The touch panels system can be adapted for touch input by fitting a transparent device with a touch sensitive mechanism over the video monitor screen. Touch input can be recorded using optical, electrical, or acoustical methods.



Optical touch panels

Optical touch panels employ a line of infrared light emitting diodes along one vertical edge and along horizontal edge of the frame. The opposite vertical and horizontal edges contain light detectors.These detectors are used to record which beams are interrupted when the panel is touched. The crossing beams that are interrupted identify the horizontal and vertical coordinates of the screen position selected.Positions can be selected.Positions can be selected with an accuracy of about 1/4 inch.

With closely spaced LED's it is possible to break into horizontal or two vertical beams simultaneously. In this case an average positions between the two interrupted beams is recorded. The LEDs operate at infrared frequencies,So that the light is not visible to the user.

Electrical touch panels

An electrical touch panel is constructed with two transparent plates separated by a small distance.One of the plate is coated with conducting material,and the other plate is coated with the resistive material.

When the outer plate is touched, it is forced into contact with the inner plate.This contact creates a voltage drop across the resistive plate that is converted to the coordinate values of the selected screen positions.

Acoustical touch panels

In acoustical touch panels,high frequency sound waves are generated in the horizontal and vertical directions across the glass plate.

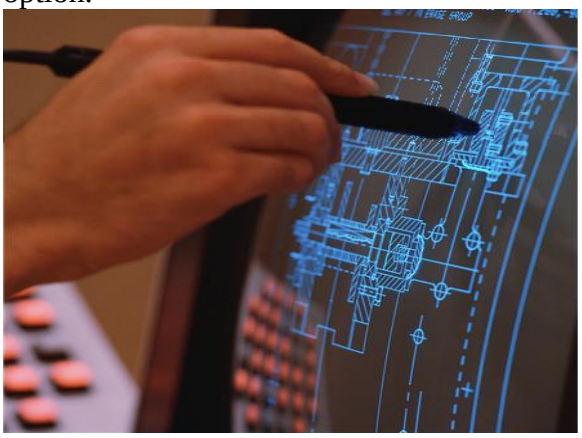
Touching the screen causes part of each wave to be reflected from the finger to the emitter. The screen position at the point of contact is calculated from a measurement of the time interval between the transmission of each wave and its reflection to the emitter.

Light Pens

Pencil shaped device are used to select screen positions by detecting the light coming from points on the CRT screen.
They are sensitive to the short burst of light emitted from the phosphor coating at the instant the electron beam strikes a particular point. Other light sources as background light in the room are not detected by a light pen.

An activated light pen generates an electrical pulse that causes the coordinate position of the electron beam to be recorded. The recorded light pen coordinates can be used to position an object or to select a processing

option.



Voice systems

Speech recognizers are used to accept voice commands as inputs. The voice systems input can be used to initiate graphics operations or to enter data. These systems operate by matching an input against a predefined dictionary of words and phrases.

When a voice command is given,the systems searches the dictionary for a frequency pattern match.Voice input is typically spoken into a microphone mounted on a headset.The microphone is designed to minimize input of other background sounds.

Track ball and Space ball

A track ball is a ball that can be rotated with the fingers or palm to produce screen cursor movement.potentiometers attached to the ball, measure the amount and direction of rotation.Trackballs can be mounted on keyboards or other devices such as the z mouse.

Space balls are used for three dimensional positioning and selection operations in virtual reality systems,modelling ,animation , CAD and other applications.

Joysticks

A joystick consists of a small, vertical lever mounted on a base that is used to steer the screen cursor around.Some joysticks mounted on a keyboard,others function as stand alone units.

Most joysticks select screen positions with actual stick movement, others respond to pressure on the stick.The distance that the stick is moved in any direction from its center position corresponds to screen cursor movement in that direction.

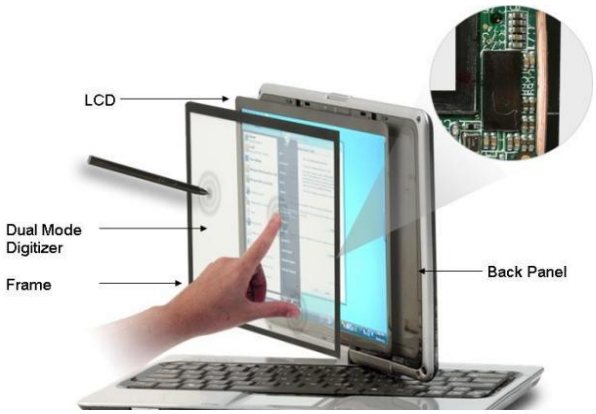
Data Glove

The glove is constructed with a series of sensors that detect hand and finger motions.Electro magnetic coupling between transmitting antennas and receiving antennas is used to provide information about the position and orientation of the hand.

Digitizers

A common device for drawing,painting or interactively selecting coordinate positions on an object is digitizer.These devices can be used to input coordinate values in either a two-dimensional or a three-dimensional space.

One type digitizer is the graphic tablet which is used to input two-dimensional coordinates by activating a hand cursor or styles positions on a flat surface.



Hard Copy Devices

Output that is readable by the user can be categorized into two categories:

- Hard copy - Hard copy is a relatively permanent form of output that can be read immediately or stored for later use, such as paper. *Printers* are the most common hard copy output devices.
- Soft copy - Soft copy is a transient form of output, for example, text on a screen display. It is lost when the computer is turned off unless it is saved in the main memory or on a disk.

The quality of the pictures obtained from a device depends on dot size and the number of dots per inch, or lines per inch, that can be displayed. To produce smooth characters in printed text strings, higher-quality printers shift dot positions so that adjacent dots overlap.

Important Characteristics of Hardcopy Devices

Dot Size: It is the diameter of a so=ingle dot on the device's output. It is also referred to as spot size

Addressability: It is the number of individual dots per inch that can be created. If the address of current dot is (x,y),then address of next dot in the horizontal direction is given as (x+1,y).Similarly, the address of next dot in vertical direction is (x,y+1)

Interdot Distance: It is the reciprocal of addressability.

Resolution: It is the number of distinguishable lines per inch that a device can create. It depends on a dot size and the cross-sectional intensity distribution of a spot.

Printers are also classified by the following characteristics

 Quality of type: The output produced by printers is said to be either *letter quality* (as good as a typewriter), *near letter quality*, or *draft quality*. Only daisy-wheel, ink-jet, and laser printers produce letter-quality type. Some dot-matrix printers claim letter-quality print, but if you look closely, you can see the difference.

 Speed: Measured in characters per second (cps) or pages per minute (ppm), the speed of printers varies widely. Daisy-wheel printers tend to be the slowest, printing about 30 cps. Line printers are fastest (up to 3,000 lines per minute). Dot-matrix printers can print up to 500 cps, and laser printers range from about 4 to 20 text pages per minute.

 Impact or non-impact*:* *Impact printers* include all printers that work by striking an ink ribbon. Daisy-wheel, dot-matrix, and line printers are impact printers. *Non-impact printers* include laser printers and ink-jet printers. The important difference between impact and non-impact printers is that impact printers are much noisier.

A *plotter* is another hard copy output device that reproduces graphic images on paper using a pen that is attached to a movable arm. Pen plotters can draw complex line art, including text, but do so very slowly because of the mechanical movement of the pens. When computer memory was very expensive, and processor power was very limited, this was often the fastest way to produce color high-resolution vector-based artwork, or very large drawings efficiently.

 Graphics: Some printers (daisy-wheel and line printers) can print only text. Other printers can print both text and graphics.

 Fonts : Some printers, notably dot-matrix printers, are limited to one or a few fonts. In contrast, laser and ink-jet printers are capable of printing an almost unlimited variety of fonts. Daisy-wheel

printers can also print different fonts, but you need to change the daisy wheel, making it difficult to mix fonts in the same document.

Impact & Non-impact printers

Refers to a class of printers that work by banging a head or needle against an ink ribbon to make a mark on the paper. Impact printers include dot-matrix printers, daisy-wheel printers, and line printers. In contrast, laser and ink-jet printers are *non-impact printers.* The distinction is important because impact printers tend to be considerably noisier than non-impact printers but are useful for multipart forms such as invoices.

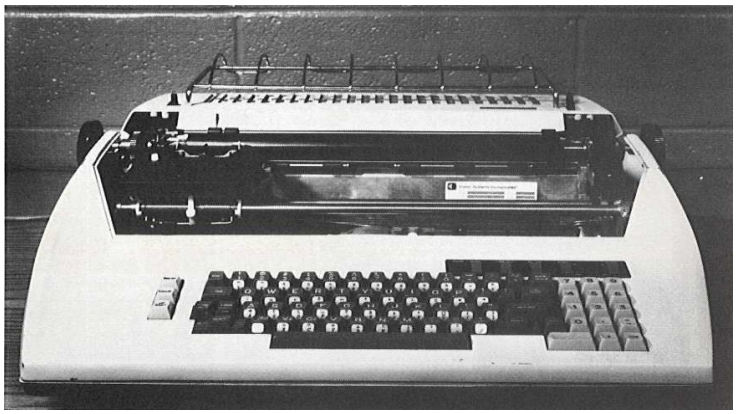
Impact printers

Of all the printer types, however, impact printers have relatively low consumable costs. Ink ribbons and paper are the primary recurring costs for impact printers. It press formed character faces against an inked ribbon onto the paper.

Types of impact printers

- **Daisy wheel printer :** A type of printer that produces letter-quality type. A daisy-wheel printer works on the same principle as a ball-head typewriter. The daisy wheel is a disk made of plastic or metal on which characters stand out in relief along the outer edge. To print a character, the printer rotates the disk until the desired letter is facing the paper. Then a hammer strikes the disk, forcing the character to hit an ink ribbon, leaving an impression of the character on the paper. You can change the daisy wheel to print different fonts.

Daisy-wheel printers cannot print graphics, and in general they are noisy and slow, printing from 10 to about 75 characters per second. As the price of laser and ink-jet printers has declined, and the quality of dot-matrix printers has improved, daisy-wheel printers have become obsolete.



- **Dot matrix printer:** A type of printer that produces characters and illustrations by striking pins against an ink ribbon to print closely spaced dots in the appropriate shape. Dot-matrix printers are relatively expensive and do not produce high-quality output. However, they can print to multi-page forms (that is, carbon copies), something laser and ink-jet printers cannot do.

Compared to laser and ink-jet printers, dot-matrix printers are notorious for making a racket.

- **Line printer:** A high-speed printer capable of printing an entire line at one time. A fast line printer can print as many as 3,000 lines per minute. Because of the nature of the print mechanism, line printers are much faster than dot-matrix or daisy-wheel printers; The disadvantages of line printers are that they cannot print graphics, the print quality is low, and they are very noisy.

	Line printer	Dot matrix printer	Daisy wheel printer
1	Printers one line at a time	Prints a character at a time	Prints a character at a time
2	Characters are embossed on the drum or chain	Characters are formed by combination of dots.	Characters are formed disk until the desired letter is facing the paper
3	Characters cannot be printed with different fonts	Characters can be printed with various character fonts	Characters can be printed with various character fonts
4	Better printing quality	Poor printing quality as characters are formed by combination of dots.	Poor printing quality
5	Better Printing speed	Poor printing speed	Noise and Slow
6	Heavy duty printers	Light duty printers	Light duty printers

Non-Impact printers

Non-impact printers are much quieter than impact printers as their printing heads do not strike the paper. A type of printer that does not operate by striking a head against a ribbon. The term *non-impact* is important primarily in that it distinguishes quiet printers from noisy (impact) printers

Types of non-impact printers

- **Laser printer:** 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. The

standard of print is very good and laser printers can also produce very good quality printed graphic images too.

- **Ink-jet printers:** A type of printer that works by spraying ionized ink at a sheet of paper. Magnetized plates in the ink's path direct the ink onto the paper in the desired shapes. Ink-jet printers are capable of producing high quality print approaching that produced by laser printers.

A typical ink-jet printer provides a resolution of 360 dots per inch, although some newer models offer higher resolutions. In general, the price of ink-jet printers is lower than that of laser printers. However, they are also considerably slower. Another drawback of ink-jet printers is that they require a special type of ink that is apt to smudge on inexpensive copier paper.

Features of ink-Jet Printers

1. They can print from two to four pages per minute.
2. Resolution is about 360 dots per inch,therefore better printing quality is achived.
3. The operating cost is quite low ,the only part that needs replacement is the ink cartridage

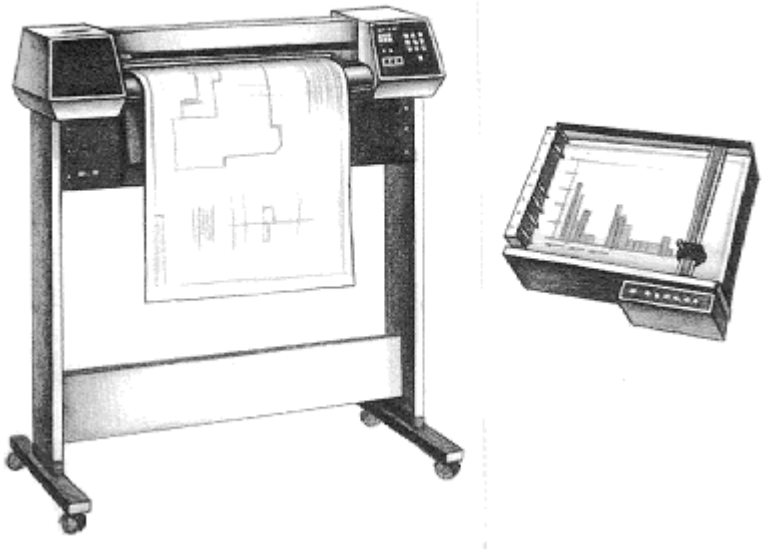
4. Colour ink jet printers have four ink nozzles with colours cyan,magenta,yellow and black,because it is possible to combine these colours to create any colour in the visible spectrum.

Plotter

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.

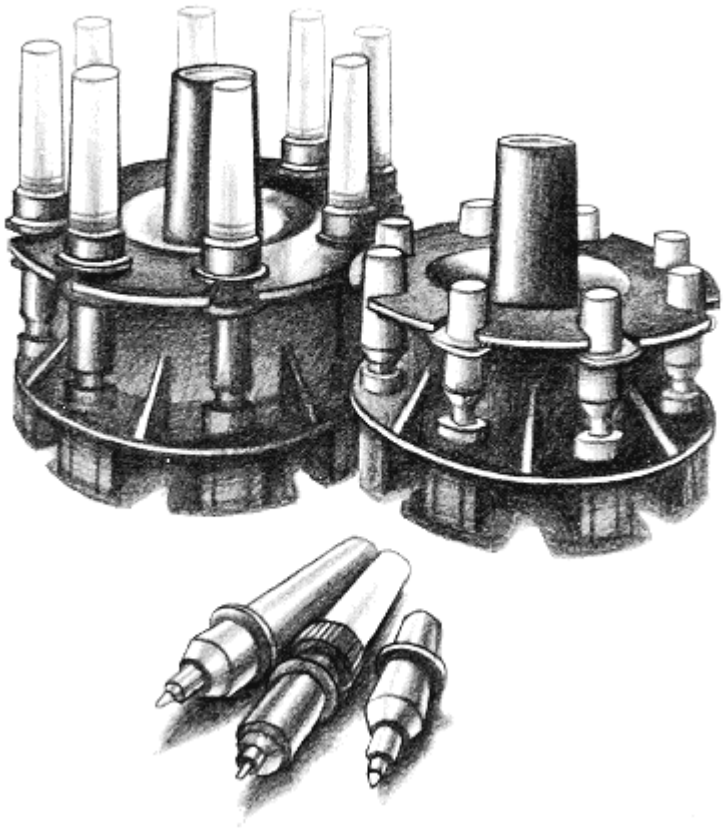
Plotters were the first type of printer that could print with color and render graphics and full-size engineering drawings. As a rule, plotters are much more expensive than printers. They are most frequently used for CAE (computer-aided engineering) applications, such as CAD (computer-aided design) and CAM (computer-aided manufacturing). Hewlett-Packard is the leading vendor of plotters worldwide.

Pen plotters are still the most affordable printing device for CAD use and offer resolution unlike any other printer. The lines are not made up of dots. They are actually drawn, providing infinite resolution. See drum plotter, flatbed plotter, electrostatic plotter and inkjetprinter.



Drum and Flatbed Plotters

Both types of plotters actually "draw" the images. The drum plotter (left) wraps the paper around a drum with pin feeds. It moves the paper back and forth for one direction of the plot. The pens move across the paper, creating the other axis. The bed of the flatbed unit (right) determines the maximum size of the total drawing.



Drawing Pens

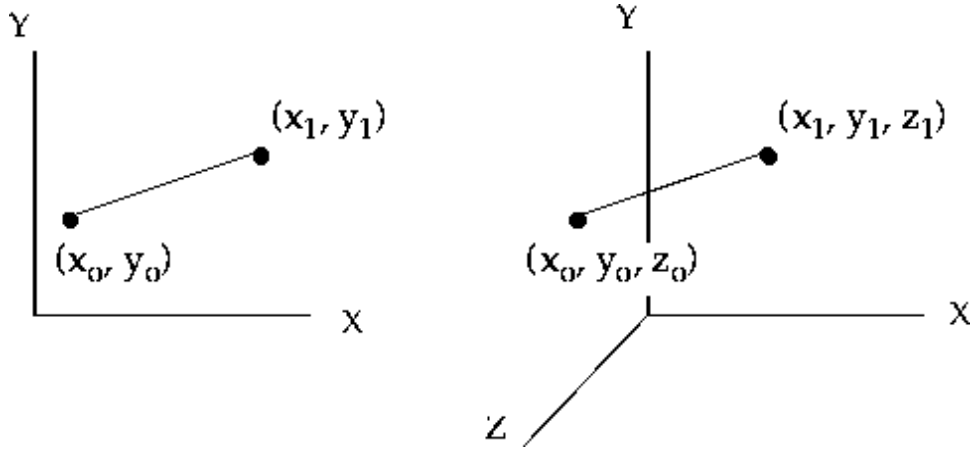
Pen plotters use drawing pens that provide infinite resolution, because the lines are actually drawn. All other printing devices print dots.

Advantages

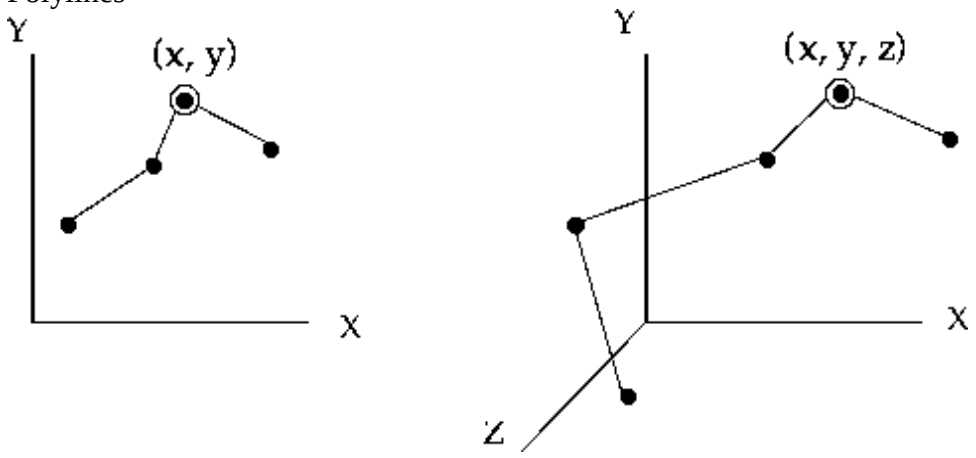
The advantages are that because they are a vetor graphics printer, meaning they can only print lines and minimum curves, they are great for printing blueprints and maps. They are very expensive so big businesses are usually the only ones that use them. They aren't designed for printing ordinary photographs so you would be disappointed with the quality if you tried.

Primitives in 2D and 3D: generalized elements

Lines

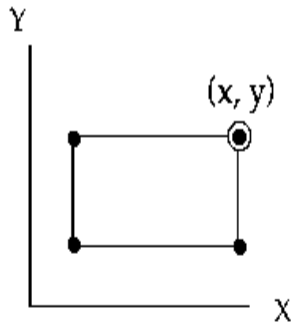


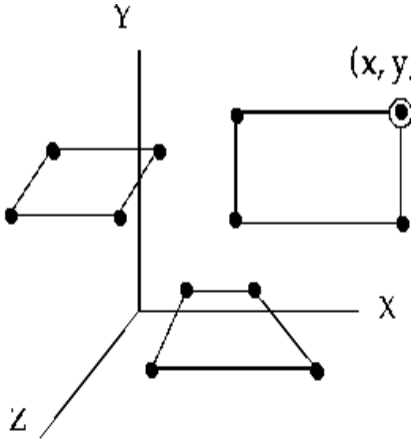
Polylines



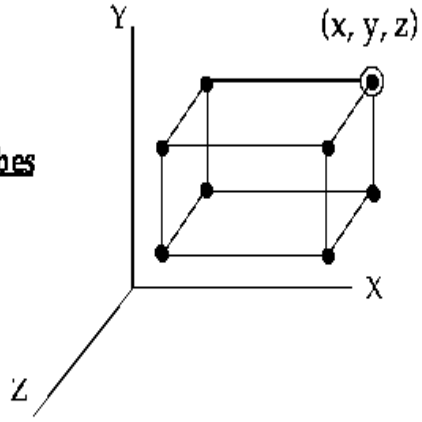
Primitives in 2D and 3D (continued)

Rectangles

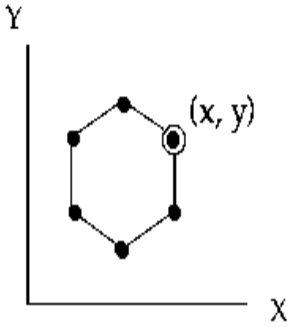




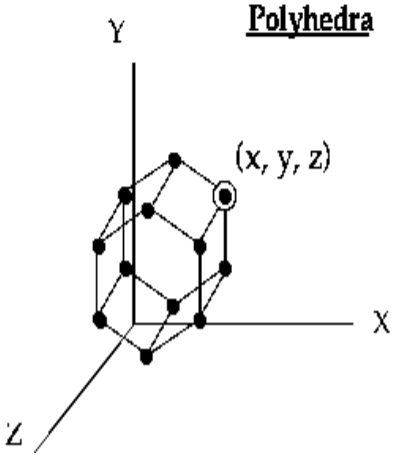
Cubes



Polygons



Polyhedra



UNIT - II

Output Primitives and Attributes: Drawing line, circle and ellipse generating algorithms – Scan line algorithm – Character generation – attributes of lines, curves and characters – Antialiasing

OUTPUT PRIMITIVES

The basic elements constituting a graphic are called output primitives. The following output primitives.

- Polyline
- Polymarker
- Text
- Tone

There are other secondary primitives besides these.

- Line
- Arrow

Five of these six primitives (excluding the tone primitive) all consist of line segments. Line segments have two attributes: the line index and line type. A tone has the attribute of tone pattern index. These attributes are defined as follows.

Text Output Attributes

Text and objects have line and fill attributes. Fill attributes are the color, trapping and fill style inside any shape, path or character. Line attributes are the color, trapping and stroke, or outline, of any shape, path or character. Color can be applied to lines and fills independently.

The appearance of text on a workstation is controlled by a number of attributes including:

- Text path

- Text alignment
- Text font and precision
- Text color index
- Character height
- Character spacing
- Character up vector
- Character expansion factor

Each character is surrounded by an imaginary box called the character body. The ratio of height to width in a character body is part of the font definition. Most fonts allow the width of the character body to vary from character to character.. The character body is delimited by the lines labeled left, right, bottom, and top. The character itself is delimited in the vertical by a base and a cap. The distance between the base and the cap is the character height. The character body usually will contain some white space to the left and right of the character in order to provide pleasing character separations when character bodies are concatenated to produce an output text string.

Drawing line

Basic Concepts in line Drawing

Before discussing specific line drawing algorithms it is useful to note the generation requirements for such algorithms. These requirements specify the descried characteristics of line.

- The line should appear as a straight line and it should start and end accurately
- The line should be displayed with constant brightness along its length independent of its length and orientation
- The line should be drawn rapidly

Line Drawing Algorithms.

Line Equation

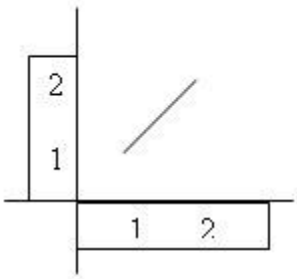
The Cartesian slop-intercept equation for a straight line is

y=mx+b

------ (1)

with
m->slope
b->y intercept

The 2 end points of a line segment are specified at a position(x1,y1)



Line path between the end point(x1,y1) and(x2,y2)

Line DDA Algorithm:

The Digital Differential Analyzer is a scan conversion line algorithm based on calculating either Δx *or* Δy. Sample the line at unit intervals in one co-ordinate and determine corresponding values nearest to line path for the other co-ordinate.

- The digital differential analyzer(DDA) is a scan conversion line algorithm based on calculation either Dy or Dx.
- The line at unit intervals is one coordinate and determine corresponding integer values nearest line for the other coordinate.
- Consider first a line with positive slope.

DDA Algorithm

Step1: Read the line end points (x_1, *y1*) and (*x2*, *y2*), such that they are not equal .[if equal then plot that point and exit]

Step2: $\Delta x = |x2-x1|$

$\Delta y = |y2 - y1|$

Step3: if($\Delta x >= \Delta y$) *then*

Length= Δx

else

Length = Δy

endif

Step4: $\Delta x = (x_2 - x_1) / length$

$\Delta y = (y_2-y_1)/ length$

Step5: $x= x_1 + 0.5 * sign(\Delta x)$

$Y= y_1+ 0.5 * sign(\Delta y)$

Step6: i=1

While(i<=length)

{

Plot (Integer(x), Integer(y))

$x =x + \Delta x$

$y = y + \Delta y$

i=i+1

}

Step7: Stop

Example:

Consider the line from (0,0) to (4,6)

Solution:

$x_1 = 0 \quad x_2 = 4$

$y_1 = 0 \quad y_2 = 6$

length = $| y_2 - y_1| = 6$

$\Delta x = | x_2 - x_1| / \text{length} = 4/6$

$\Delta y = | y_2 - y_1| / \text{length} = 6/6 = 1$

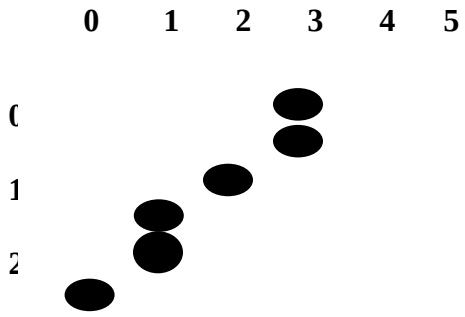
$x = 0 + 0.5 * \sin(4/6) = 0.5$

$y = 0 + 0.5 * \sin(1) = 0.5$

Tabulate the step6:

I	Plot	X	y
		0.5	0.5
1	(0,0)	1.17	1.5
2	(1,1)	1.85	2.5
3	(1,2)	2.5	3.5
4	(2,3)	3.167	4.5
5	(3,4)	3.85	5.5
6	(3,5)	4.5	6.5

OUTPUT:



Example:

xa,ya=>(2,2)
xb,yb=>(8,10)
dx=6
dy=8

xincrement=6/8=0.75
yincrement=8/8=1

1) for(k=0;k<8;k++)

xincrement=0.75+0.75=1.50
yincrement=1+1=2
1=>(2,2)

2) for(k=1;k<8;k++)

xincrement=1.50+0.75=2.25
yincrement=2+1=3
2=>(3,3)

it will be incremented upto the final end point is reached.

Advantages & disadvantages of DDA:-

- Faster than the direct use of the line equation and it does not do any floating point multiplication.
- Floating point Addition is still needed.
- Precision loss because of rounding off.
- Pixels drift farther apart if line is relatively larger.

Bresenham line algorithm:

Introduction:

It is commonly used to draw lines on a computer screen, as it uses only integer addition, subtraction and bit shifting, all of which are very cheap operations in standard computer architectures. It is one of the earliest algorithms developed in the field of computer graphics.

The Bresenham line algorithm is an algorithm which determines which points in an n-dimensional raster should be plotted in order to form a close approximation to a straight line between two given points.

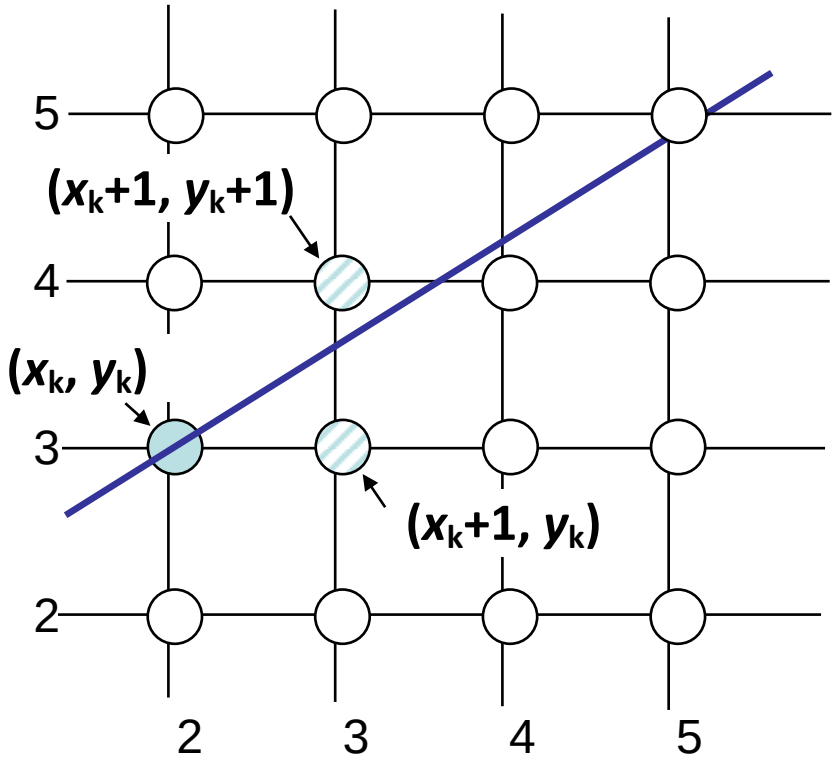
The speed and simplicity of Bresenham's line algorithm mean that it is still important. The algorithm is used in hardware such as plotters and in the graphics chips of modern graphics cards. It can also be found in many software graphics libraries, because the algorithm is very simple.

Deriving The Bresenham Line Algorithm:

Move across the *x* axis in unit intervals and at each step choose between two different *y* coordinates. The endpoints of the line are the pixels at (x_0, y_0) and (x_1, y_1), where the first coordinate of the pair is the column and the second is the row. The algorithm will be initially presented only for the octant in which the segment goes down and to the right ($x_0 \leq x_1$ and $y_0 \leq y_1$).

For example, from position (2, 3) we have to choose between (3, 3) and (3, 4)

We would like the point that is closer to the original line.



pk is a decision parameter. If *pk* is negative, then we choose the lower pixel, otherwise we choose the upper pixel.

The first decision parameter p0 is evaluated at (x0, y0) is given as:

$$p_0 = 2\Delta y - \Delta x$$

BRESENHAM'S LINE DRAWING ALGORITHM

1. Input the two line end-points, storing the left end-point in (*x0*, *y0*)
2. Plot the point (*x0*, *y0*)
3. Calculate the constants Δx, Δy, $2\Delta y$, and ($2\Delta y$ - $2\Delta x$) and get the first value for the decision parameter as:

$$p_0 = 2\Delta y - \Delta x$$

4. At each *xk* along the line, starting at *k = 0*, perform the following test. If *pk < 0*, the next point to plot is *(xk+1, yk)* and:

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

Otherwise, the next point to plot is (*xk+1*, *yk+1*) and:

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

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

Bresenham Example:

Let's plot the line from (20, 10) to (30, 18)

Step1: Input the two line end-points, storing the left end-point in (*x0*, *y0*) as (20,10).

Step2: Plot the point (*x0, y0*) ie (20,10).

Step3: First off calculate all of the constants:

- Δx: (x1 – x0) = (30-20) = 10
- Δy: (y1 – y0) = (18-10) = 8
- $2\Delta y$: 2(8)= 16
- $2\ \Delta x$: *2(10) =20*
- $2\Delta y - 2\Delta x$: ((16-20) = -4

Calculate the initial decision parameter *p0*:

- $p0 = 2\Delta y - \Delta x = 6$

Step4.1: p0 = 6<0

next point to plot is (x_{k+1}, y_{k+1}) ie (21, 11)

$p_{k+1}=p_k+2\Delta y - 2\Delta x = 0+16-20 = -4$

Step4.2: p1 = -4<0

next point to plot is (x_{k+1}, y) ie (22, 11)

$p_{k+1}=p_k+2\Delta y = 1+16 = 17$

Step4.3: p2 = 17<0

next point to plot is (x_{k+1}, y_{k+1}) ie (23, 12)

$p_{k+1}=p_k+2\Delta y - 2\Delta x = 2+16-20 = -2$

Step4.4: p3 = -2<0

next point to plot is (x_{k+1}, y) ie (24, 12)

$p_{k+1}=p_k+2\Delta y = 3+16 = 19$

Step4.5: p4 = 19<0

next point to plot is (x_{k+1}, y_{k+1}) ie (25, 13)

$p_{k+1}=p_k+2\Delta y - 2\Delta x = 4+16-20 = 0$

Step4.6: p5 = 0=0

next point to plot is (x_{k+1}, y_{k+1}) ie (26, 14)

$p_{k+1}=p_k+2\Delta y - 2\Delta x = 5+16-20 = 1$

Step4.7: p6 = 1<0

next point to plot is (x_{k+1}, y_{k+1}) ie (27, 15)

$p_{k+1}=p_k+2\Delta y - 2\Delta x = 6+16-20 = 2$

Step4.8: p7 = 2<0

next point to plot is (x_{k+1}, y_{k+1}) ie (28, 16)

$p_{k+1}=p_k+2\Delta y - 2\Delta x = 7+16-20 = 3$

Step4.9: p8 = 3<0

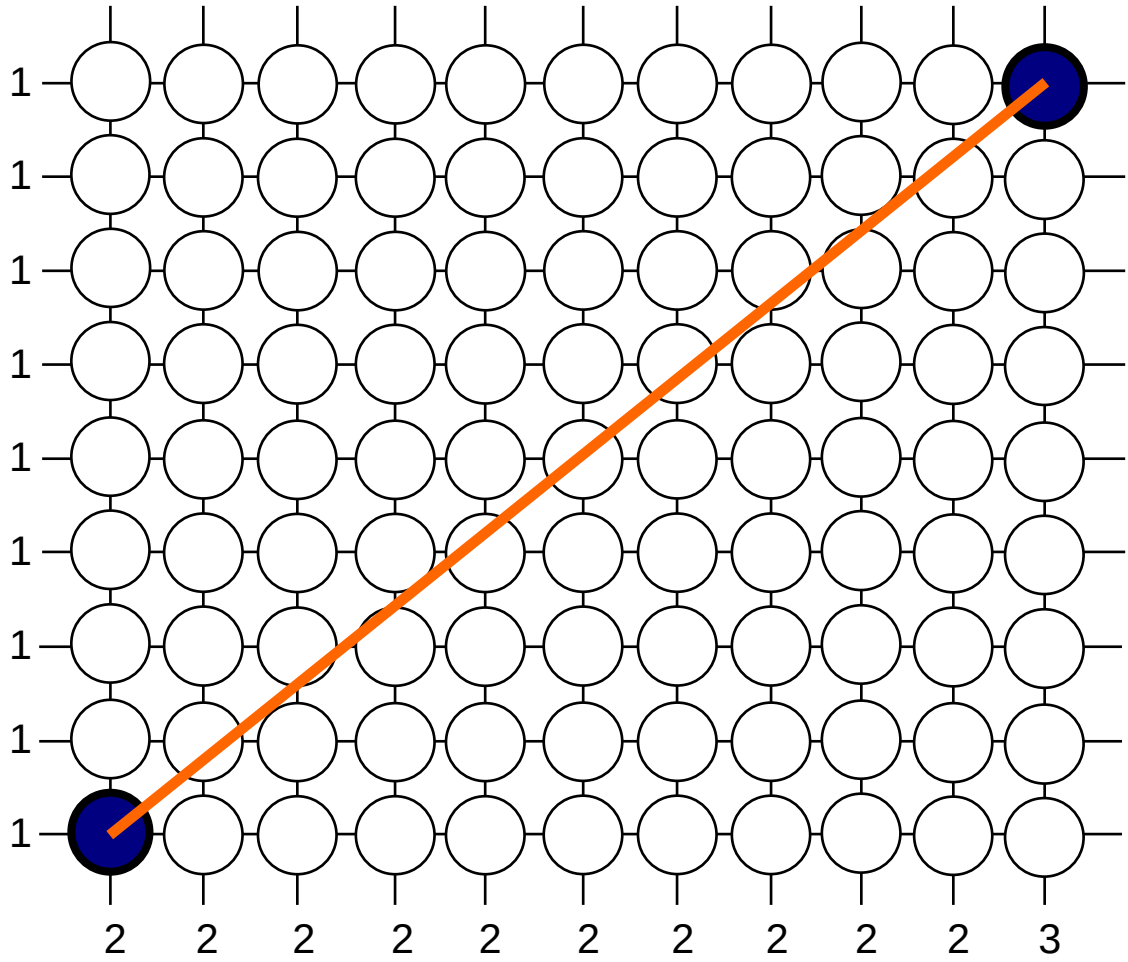
next point to plot is (x_{k+1}, y_{k+1}) ie (29, 17)

$p_{k+1}=p_k+2\Delta y - 2\Delta x = 8+16-20 = 4$

Step4.10: p9 = 4<0

next point to plot is (x_{k+1}, y_{k+1}) ie (30, 18)

Step:5 Repeat step 4 (Δx – 1) times (10-1=9), hence 9 times step 4 will be repeated.



CIRCLE

Definition:

Circle is defined by its center xc, yc and its radius in user coordinate units. The

equation of the circle is (x-xc) + (y-yc) = r2.

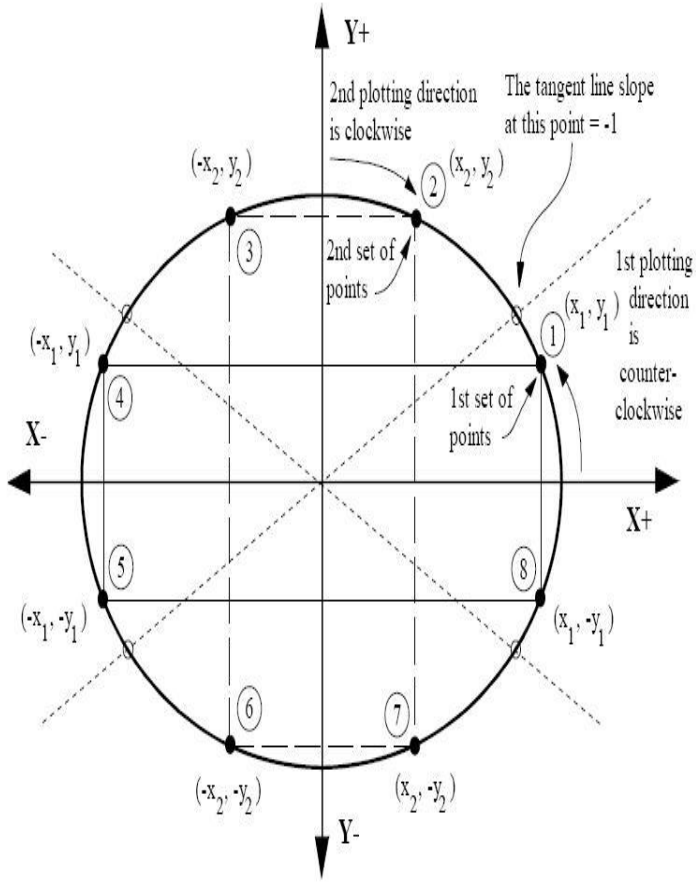
Mid-Point Circle Algorithm

In computer graphics, the **midpoint circle algorithm** is an algorithm used to determine the points needed for drawing a circle. The algorithm is a variant of Bresenham's line algorithm, and is thus sometimes known as **Bresenham's circle algorithm.**

In the mid-point circle algorithm we use eight-way symmetry so only ever calculate the points for the top right eighth of a circle, and then use symmetry to get the rest of the points.

Assume that we have just plotted point (x_k, y_k).The next point is a choice between (x_k+1, y_k) and (x_k+1, y_k-1).We would like to choose the point that is nearest to the actual circle.

Eight way symmetry of the circle



Let's the equation of the circle to give us:

$$f_{circ}(x, y) = x^2 + y^2 - r^2$$

The equation evaluates as follows:

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

By evaluating this function at the midpoint between the candidate pixels we can make our decision.

THE MID-POINT CIRCLE ALGORITHM

Step1: Input radius *r* and circle centre (x_c, y_c), then set the coordinates for the first point on the circumference of a circle centred on the origin as:

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

Step2: Calculate the initial value of the decision parameter as:

$$p_0 = 5/4 - r$$

Step3: Starting with $k = 0$ at each position x_k, perform the following test. If $p_k < 0$, the next point along the circle centred on $(0, 0)$ is (x_k+1, y_k) and:

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

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

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

Step4: Determine symmetry points in the other seven octants

Step5: Move each calculated pixel position (x, y) onto the circular path centred at (x_c, y_c) to plot the coordinate values:

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

Step6: Repeat steps 3 to 5 until $x >= y$.

What is difference between mid-point and bresenhams circle algorithm ?

bresenhams circle algorithm results in a much more smoother circle,comparred to midpoint circle algorithm..In mid point,decision parameter depends on previous decision parameter and corresponding pixels whereas in bresenham decision parameter only depends on previous decision parameter...

ELLIPSE

Definition:

An ellipse can use the same parameters xc, yc ,r as a circle, in addition to the eccentricity.

Let's the equation of the circle to give us:

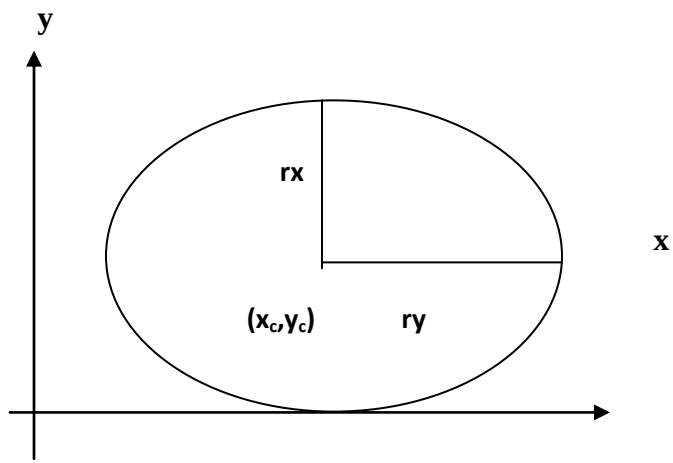
$$(x-x_c)^2/a^2 + (y-y_c)^2/b^2 = 1$$

The equation evaluates as follows:

$$f_{circ}(x,y)\begin{cases} <0, & \text{if } (x,y) \text{ is inside the ellipse boundary} \\ =0, & \text{if } (x,y) \text{ is on the ellipse boundary} \\ >0, & \text{if } (x,y) \text{ is outside the ellipse boundary} \end{cases}$$

Ellipse

Ellipse centered at (x_c,y_c) with semi major axis for r_x and semnior axis for r_y . $(x-x_c/r_x)^2 + (y-y_c/r_y)^2 = 1$



By evaluating this function at the midpoint between the candidate pixels we can make our decision.

Midpoint Ellipse Algorithm:

Step1: Input (r_x,r_y) and ellipse centre(x_c,y_c) and obtain the Ist point. On ellipse centered on origin as $(x_0,y_0)=(0,r_y)$.

Step2: Calculate the initial value of decision parameter in region 1 as,

$$\mathbf{p1_0=r_y^2 - r_x^2 r_y + 1/4\ r_x^2}$$

Step3: At each xk position in region1, Starting at k=0, perform the following test:

If $\mathbf{p1_k<0}$, the next point along the ellipse centered on (0,0) is $\mathbf{(x_{k+1}, y_k)}$ and

$$\mathbf{P1_{k+1} = p1_k + 2\ r_y^2\ (x_{k+1}) + r_y^2}$$

Otherwise,

the next point along the ellipse centered on (0,0) is $\mathbf{(x_{k+1}, y_k - 1)}$ and

$$\mathbf{P1_{k+1} = p1_k + 2\ r_y^2\ (x_{k+1}) - + r_y^2(y_{k-1})}$$

Step4: calculate intial value of decision parameter in region2 using last point. (x_0,y_0) which is calculated in region1,

$$\mathbf{P2_0=r_y^2\ (x_0+1/2)^2 + r^2(y_0 - 1) - r_x^2\ r_y^2}$$

Step5: At each yk position in region2, starting at k=0, perform the following test:

1. if $p2_k>0$, the next point along the ellipse centered on (0,0) is $\mathbf{(x_k,y_{k-1})}$

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

otherwise,

2. for(x_k+1,y_k-1),

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

using the same incremental calculation for x+y as in region-1.

Step6: Determine symmetry points in other 3 quadrants.

Step7: Move each calculated pixel position (x,y) on to the ellipitical path centered on (x_c,y_c) and plot the co-ordinate values,

$$\mathbf{X=x+x_c \quad and \quad y=y+y_c}$$

Step8: Repeat the steps for Region 1 until $2r_y^2 x >= 2\, r_x^2 y$

Example:

$r_x=8$ and $r_y=6$

Solution:

(x_0,y_0) => x_0 =0 and y_0 + 6

$P1_0$ = -332

Region 1:

K	$P1_k$	X_{k+1}, y_{k+1}	$2r_y^2x_{k+1}$	$2r_x^2y_{k+1}$
0	-332	(1,6)	72	768
1	-224	(2,6)	144	768
2	-44	(3,6)	216	768
3	208	(4,5)	288	640
4	-108	(5,5)	360	640
5	288	(6,4)	432	512
6	244	(7,3)	504	384

Region 2:

K	$P2_k$	X_{k+1}, y_{k+1}	$2r_y^2x_{k+1}$	$2r_x^2y_{k+1}$
0	-151	(8,2)	576	256
1	233	(8,1)	576	128
2	745	(8,0)		

SCAN LINE ALGORITHM

Definition:

One way to fill the polygon is to apply the inside test. i.e to check whether the pixel is inside the polygon or outside the polygon and then highlight the pixel which lie inside the polygon. This approach is known as scan-line algorithm.

For each scan line:

1. Find the intersections of the scan line with all edges of the polygon.

2. Sort the intersections by increasing x coordinate.

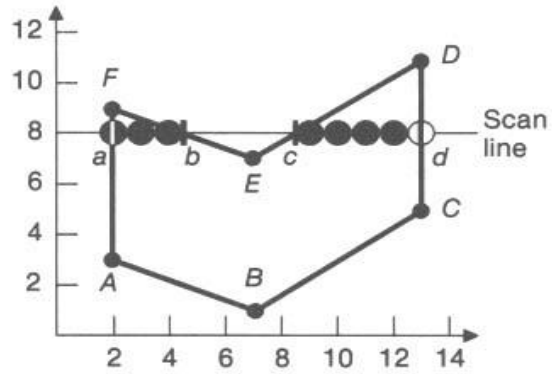
3. Fill in all pixels between pairs of intersections.

Problem:

Calculating intersections is slow.

Solution:

Incremental computation/coherence. For scan line number 8 the sorted list of x-coordinates is (2, 4, 9, 13) (b and c are initially no integers) Therefore fill pixels with x coordinates 2-4 and 9-13.



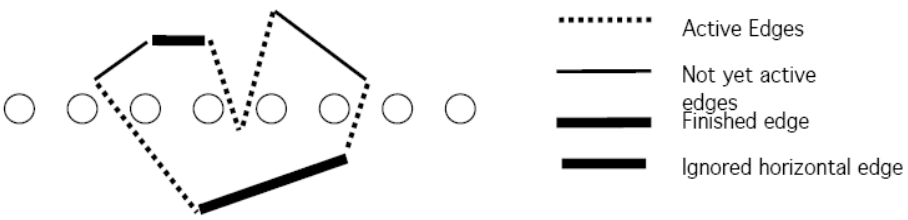
Processing Polygons

• Polygon edges are sorted according to their minimum / maximum Y.

• Scan lines are processed in increasing (upward) / decreasing (downward) Y order.

• When the current scan line reaches the lower / upper endpoint of an edge it becomes active.

• When the current scan line moves above the upper / below the lower endpoint, the edge becomes inactive.



• Active edges are sorted according to increasing X. Filling the scan line starts at the leftmost edge intersection and stops at the second. It restarts at the third intersection and stops at the fourth. . . (spans)

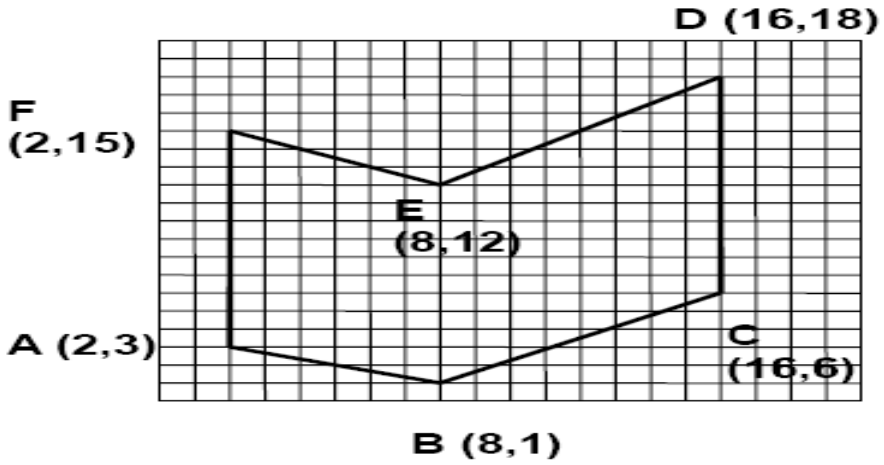
Polygon fill example:

The edge table (ET) with edges entries sorted in increasing y and x of the lower end.

• ymax: max y-coordinate of edge

• xmin: x-coordinate of lowest edge point

• 1/m: x-increment used for stepping from one scan line to the next



Character Generator

Character generator, often abbreviated as **CG**, is a device or software that produces static or animated text. Modern character generators are computer-based, and can generate graphics as well as text.

For example: The integrated circuit, usually in the form of a PROM, that decodes a keystroke in a keyboard, and outputs a corresponding character, is also referred to as a "Hardware character generator."

Character generators are primarily used in the broadcast areas of live sports or news presentations, given that the modern character generator can rapidly (i.e., "on the fly") generate high-resolution, animated graphics for use when an unforeseen situation in the game or newscast dictates an opportunity for broadcast coverage.

For example, when, in a football game, a previously unknown player begins to have what looks to become an outstanding day, the character generator operator can rapidly, using the "shell" of a similarly-designed graphic composed for another player, build a new graphic for the previously unanticipated performance of the lesser known player.

The character generator, then, is but one of many technologies used in the remarkably diverse and challenging work of live television, where events on the field or in the newsroom dictate the direction of the coverage. In such an environment, the quality of the broadcast is only as good as its weakest link, both in terms of personnel and technology. Hence, character generator development never ends, and the distinction between hardware and software CG's begins to blur as new platforms and operating systems evolve to meet the live television consumer's expectations.

Types of Character Generator:

- **Hardware CGs**
- **Software CGs**

Hardware CGs :

Hardware CGs are used in television studios and video editing suites. A DTP-like interface can be used to generate static and moving text or

graphics, which the device then encodes into some high-quality video signal.

Software CGs

Software CGs run on standard off-the-shelf hardware and is often integrated into video editing software such as nonlinear video editing applications. Some stand-alone products are available, however, for applications that do not even attempt to offer text generation on their own, as high-end video editing software often does, or whose internal CG effects are not flexible and powerful enough.

Some software CGs can be used in live production with special software:

- Computer video interface cards.
- Video editing software

LINE ATTRIBUTES

ATTRIBUTES

Attribute parameter:

Any parameter that affects the way a primitive is to be displayed is referred to as an attribute parameter.

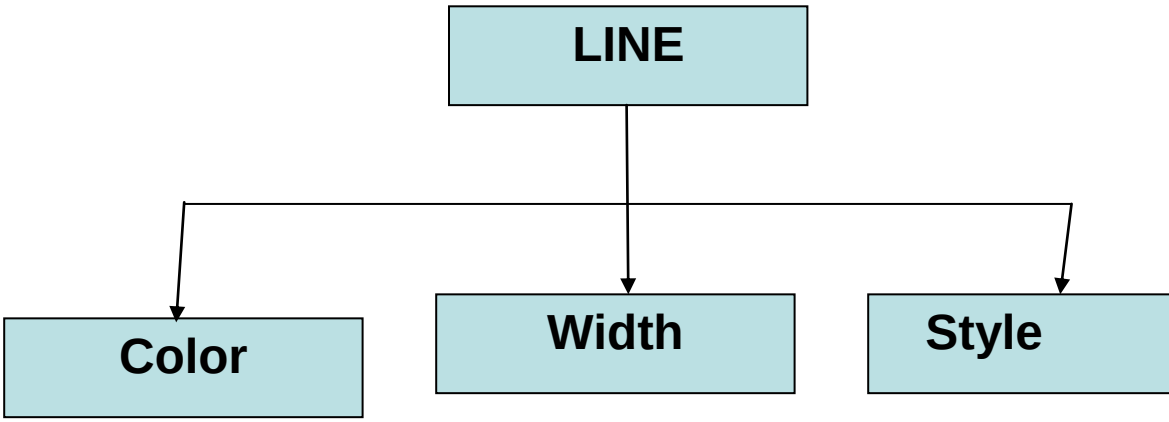
The various attributes are :

1. **Line attribute**
2. **Curve attribute**
3. **Color attribute**
4. **Character attribute**

1. Line Attribute

The line type, width and color are the attributes of the line. The line type include

solid line, dashed lines, and dotted lines.



1.1. **Line Style:**

- Solid lines :
- Dashed Lines
- Dotted Lines
- Dash-dotted Lines

The set line attribute in a application program: **setLinetype(lt)**

All lines, except for those used to draw markers or outline filled regions, are affected by the attributes described in this section. These include axis lines and lines drawn when a Data node is rendered with its DataType attribute having its DATA_TYPE_LINE bit set.

Attribute LineColor

LineColor is a Color-valued attribute that determines the color of the line. Its default value is Color.Black.

Attribute LineWidth

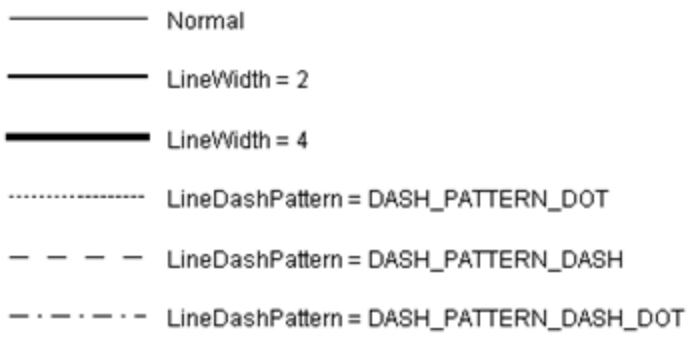
LineWidth is a double-valued attribute that determines the thickness of the lines. Its default value is 1.0.

Attribute LineDashPattern

LineDashPattern is a double-array-valued attribute that determines the line pattern used to draw the line. It defaults to a solid line. Alternate entries in the array represent the lengths of the opaque and transparent segments of the dashes.

Some dash patterns are defined. They are DASH_PATTERN_DOT, DASH_PATTERN_DASH and DASH_PATTERN_DASH_DOT.

Samples



Line cap:

Line caps can be used to adjust the shape of the line ends to give a better appearance. There are three types of line caps. Butt cap which has a square end, round cap which has a semi circle end, projecting square cap which has one half of the line width beyond the specified end points.

Generation of thick lines

– Butt cap

– Round cap

– Projecting square cap

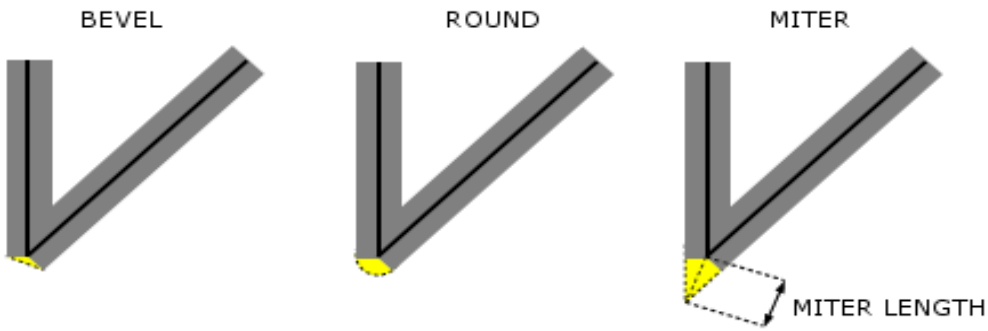
1.2. Line color:

The line color is the attributes of the line. The line color specifies colors for lines.

The set line attribute in a application program:

setPolylinecolorIndex(lc)

Generation of thick polylines



The methods used for smoothly joining two line segments:

- **Mitter join**- by extending the outer boundaries of each of the two lines until they meet.

- **Round join** – by capping the connection between the two segments with a circular boundary whose diameter is equal to the line width.
- **Bevel join** – by displaying the line segments with butt caps and filling in the triangular gap where the segment meet.

1.3. Line width:

The line width is depends upon the capabilities of the output device. Set the current line-width value in the attribute list: **setLinewidthScaleFactor(lw)**

where, lw=1 : a standard width line, lw>1 : thicker line

2. Color Attributes

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

Set the current color value in the attribute list:

glColor3i(1,1,1) -> Integer values

glColor3f(1.0,1.0,1.0) -> Floating point value

There are three types of color attributes, they are:

- RGB color attribute
- CMY color Attribute
- Gray scale attribute

2.1 RGB Color Attribute

– Coordinate system with R, G, B as axes.

- Color information can be stored in 2 ways:
  - Index
  - Color Lookup Tables (Color separation – R, G & B)

Color Look up table:

In color displays, 24 bits per pixel are commonly used, where 8 bits represent 256 levels for each color. It is necessary to read 24- bit for each pixel from frame buffer. This is very time consuming. To avoid this video controller uses look up table to store many entries to pixel values in RGB format. This look up table is commonly known as color table.

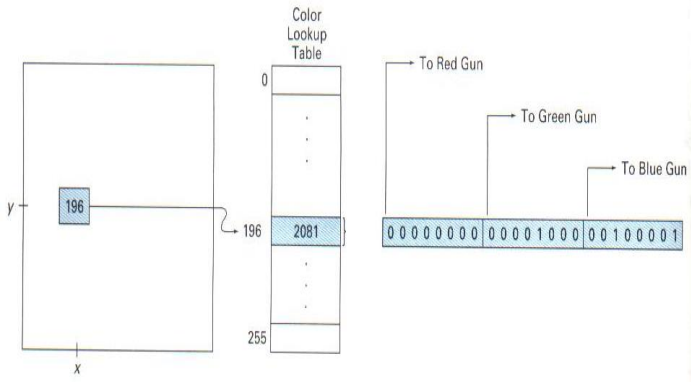
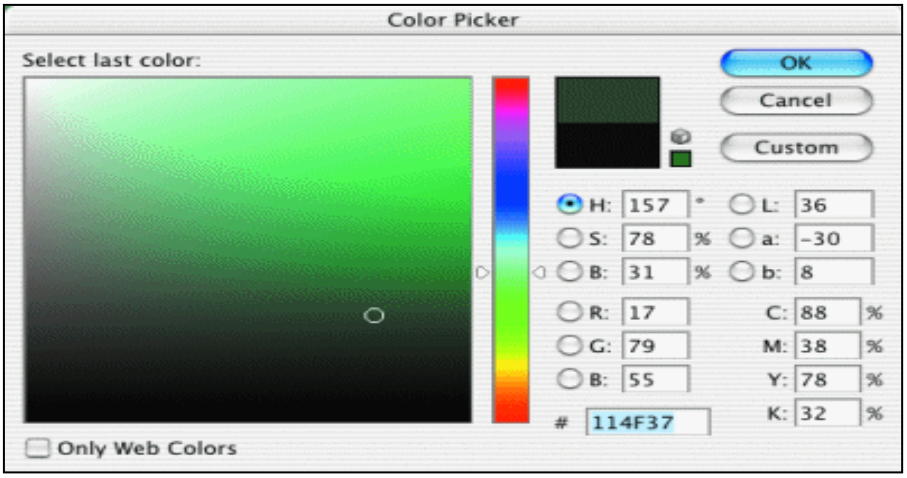


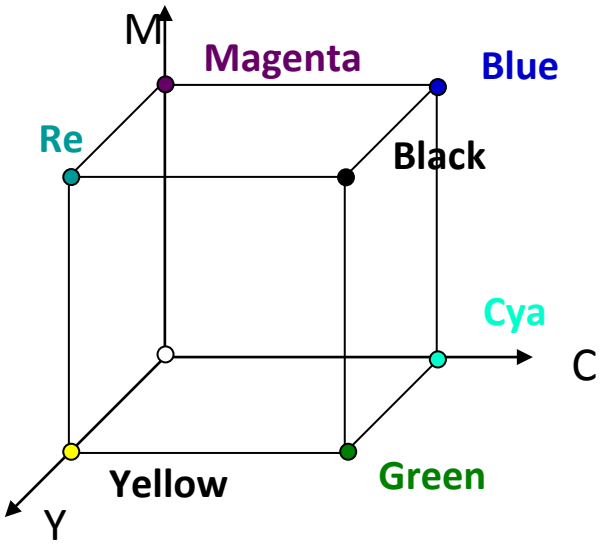
Figure 4-16
A color lookup table with 24 bits per entry accessed from a frame buffer with 8 bits per pixel. A value of 196 stored at pixel position (x, y) references the location in this table containing the value 2081. Each 8-bit segment of this entry controls the intensity level of one of the three electron guns in an RGB monitor.

INDEX:



2.2 CMY Color Attribute:

Co-ordinate system with C,M,Y as axis. Useful for describing color output to hard-copy devices.



Gray scale Attribute:

Composed of shades of gray, varying from black at the weakest intensity to white at the strongest.

3. Curve Attribute

- Display curves with varying colors, widths, dot-dash patterns, and available pen or brush options.
- Raster curves of various widths:
  - Where the magnitude of the curve slope is less than 1, we plot vertical spans, where the magnitude is greater than 1, we plot horizontal spans
    - Circles, ellipses, and other curves will appear thinnest where the slope has a magnitude of 1.

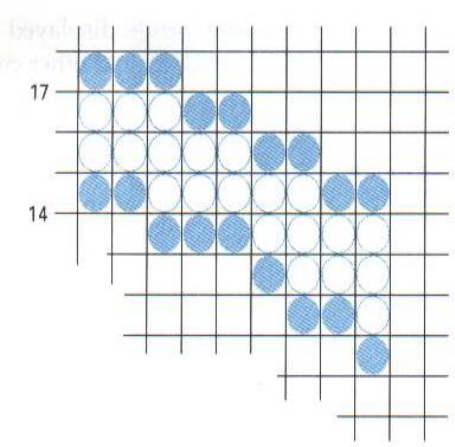


Figure 4-14
A circular arc of width 4 and radius 16 displayed by filling the region between two concentric arcs.

- Using the specified curve path as one boundary and setting up the second boundary either inside or outside the original curve path.

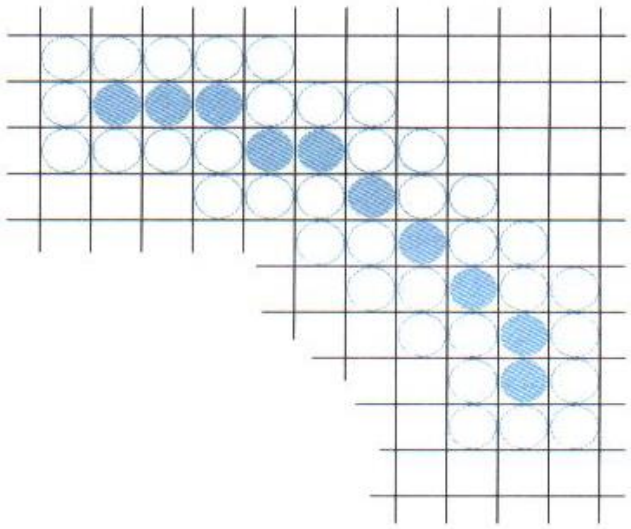


Figure 4-15
Circular arc displayed with rectangular pen.

4. Character Attributes (or) Text Attribute

- The characters in a selected font can also be displayed with assorted underlining styles (solid, dotted, double), in boldface, in italics, and in outline or shadow styles
  - setTextFont (tf)
  - setTextColourIndex (tc)
- Point measurements specify the size of the body of a character
- Text size can be adjusted without changing the width-to-height ratio of characters with,
  - setCharacterHeight (ch)

- The width only of text
  - setCharacterExpansionFactor (cw)

- Spacing between characters
  - setCharactersSpacing (cs),

    where cs<0 :overlap, cs>0:spread

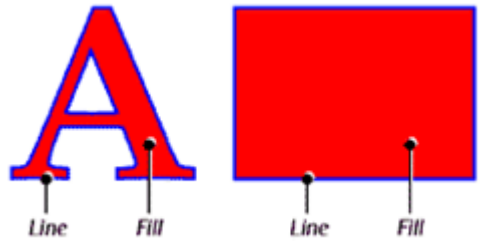
- The direction of the character up vector
  - setCharacterUpVector (upvect)
  - setTextPath (tp) :right, left, up, or down

- Alignment
  - setTextAlignment (h, v)
    - h:left, right, center v:top, cap, half, base, bottom

- Marker Attributes

- A marker symbol is a single character that can be displayed in different colors and in different sizes
    - setMarkerType (mt)
    - setMarkerSizeScaleFactor (ms)
        - ms>1 → enlargement , 0<ms<1→reduce
    - setPolymarkerColourIndex (mc)

Text and objects have line and fill attributes. Fill attributes are the color, trapping and fill style inside any shape, path or character. Line attributes are the color, trapping and stroke, or outline, of any shape, path or character. Color can be applied to lines and fills independently.



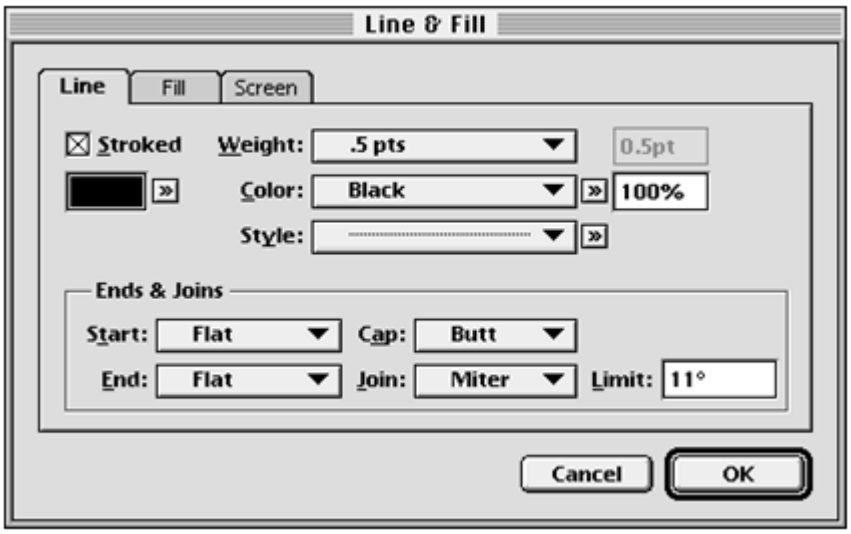
To apply line, fill, color and trapping attributes to selected objects: Select the object(s) with the Object tool. Choose Line & Fill from the Object menu to display the Line & Fill dialog box. Set the attributes and click OK.

To apply line, fill, color and trapping attributes to text: Select the text with the Text tool. Choose Line & Fill from the Type menu to display the Line & Fill dialog box. Set the attributes and click OK.

The Line & Fill dialog box is identical for text and objects. The four tabs allow you to view Line, Fill, Screen and Trapping attributes.

To fill an object, click on the Fill tab and check the Filled checkbox. To stroke an object, click the Line tab and check the Stroked checkbox.

When the Filled or Stroked checkboxes are unchecked, the object will not be filled or stroked, as applicable. When the Filled or Stroked checkboxes are set to mixed, the applicable attributes will not be changed when you click on OK.



Unlike most checkboxes in PageStream, the Filled and Stroked checkboxes can be manually set to mixed. This allows you to apply line attributes without applying fill attributes and vice versa. Applying many attributes at once to multiple objects can be slow, so set one of these to mixed when you are only changing one type of attribute.

The Color Palette

The color palette lists the colors defined for the current document. The Color palette can be used to easily apply colors to objects and text in the document. To display the Color palette if it's not shown, choose Show Color Palette from the Window menu.

If objects are selected with the Object tool, you can change the line and fill color of the selected objects.

If text is selected with the Text tool, you can change the line and fill color of the selected text.

Note: You cannot change the color of text by selecting a text object with the Object tool; this will change the text frame's line and fill color.

To apply color to the fill/stroke: Choose the Fill icon (the right icon) in the Color palette then click on a color name, and to apply color to the line stroke, choose the Line icon (the middle icon). Choose the Both icon (the left icon) to apply color to the line and fill simultaneously.

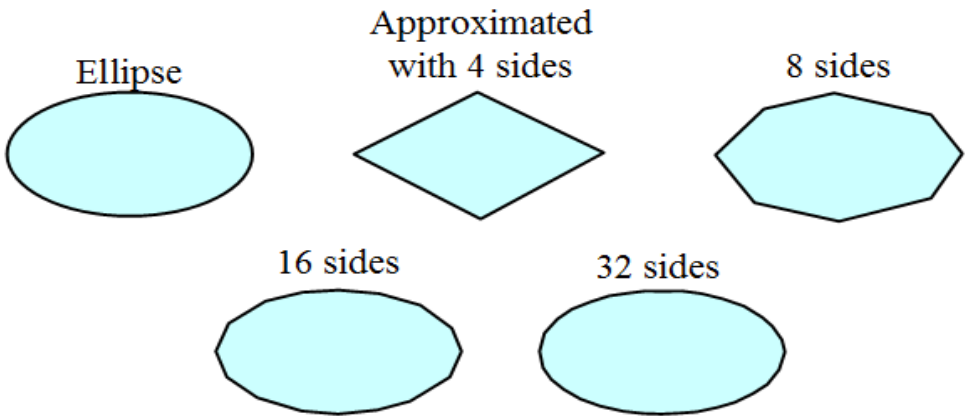
To apply From/To Gradient Colors: Select the number 1 icon for the From color, choose from one of the available color options, then select the number 2 icon for the To color and choose from one of the available color options.

To apply a tint color: Enter the percentage of the color to use into the text box beside the number 2 icon and press Return to apply the tint.

To change a color: Hold down a Shift key and click on the color name to open the Edit Color dialog box, or click on the pop-up menu button to create a new color, edit any existing color, delete colors, append colors to the list and save the color list to an external color file.

Curves and Surfaces

- Most real world objects are smooth continuous curves
- We can approximate curves using polygons



- Increasing the number of edges (or polygons) gives a better approximation to the original curve or surface.
- However, using polygons in modeling gives a faceted look.
- We can shade continuously across polygons and between polygons, but edges still cause problems

Hermite curves

Hermite curves are very easy to calculate but very powerfull to use. They are used to smoothly interpolate data between key-points (like object movement in keyframe animation or camera control). Understanding the mathematical background of hermite curves will help you to understand the entire family of splines.

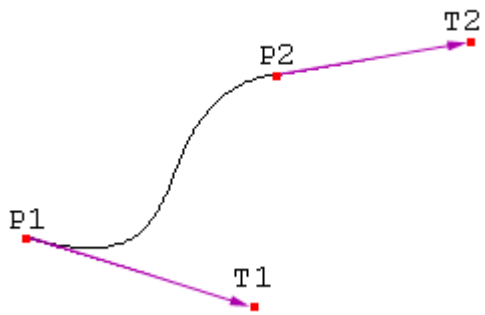
Maybe you have some experiences with 3D programming and already used them without knowing that.. (the so called kb-splines, curves with control over tension, continuity and bias are just a special form of the hermite curves).

The Math

To keep it simple we first start with some simple stuff. We also only talk about 2 dimensional curves here. If you need a 3d curve just do the same with the z-coordinate what you did with y or x. Hermite curves work in in any dimension.

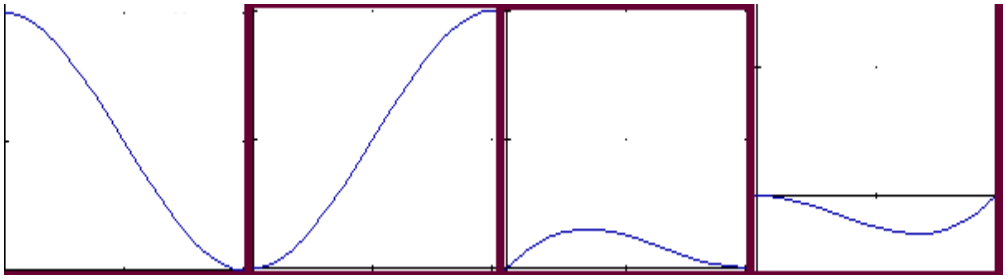
To calculate a hermite curve you need the following vectors:

P1: the startpoint of the curve
T1: the tangent (e.g. direction and speed) how the curve lefts the startpoint
P2: he endpoint of the curve
T2: the tangent (e.g. direction and speed) how the curves enters the endpoint



These 4 vectors are simply multiplied with 4 hermite basis functions and summed together.

h1(s) = 2s^3 - 3s^2 + 1
h2(s) = -2s^3 + 3s^2
h3(s) = s^3 - 2s^2 + s
h4(s) = s^3 - s^2
Below are the 4 graphs of the 4 functions (from left to right: h1, h2, h3, h4)



(all graphs except the 4th have been plotted from 0,0 to 1,1)

Take a closer look at functions h1 and h2:
h1 starts at 1 and goes slowly to 0.
h2 starts at 0 and goes slowly to 1.

If you now multiply the startpoint with **h1** and the endpoint with **h2**. Let **s** go from 0 to 1 to interpolate between start and endpoint.
h3 and **h4** are applied to the tangents in the same manner. They take care, that the curve bends into the desired direction at the start and endpoint.

spline - Cubic spline

Syntax

yy=spline(x,Y,xx)
pp = spline(x,Y)

Description

yy = spline(x,Y,xx) uses a cubic spline interpolation to find yy, the values of the underlying function Y at the values of the interpolant xx. For the

interpolation, the independent variable is assumed to be the final dimension of Y with the breakpoints defined by x.

The sizes of xx and yy are related as follows:

- If Y is a scalar or vector, yy has the same size as xx.
- If Y is an array that is not a vector,
    - If xx is a scalar or vector, size(yy) equals [d1, d2, ..., dk, length(xx)].
    - If xx is an array of size [m1,m2,...,mj], size(yy) equals [d1,d2,...,dk,m1,m2,...,mj].

pp = spline(x,Y) returns the piecewise polynomial form of the cubic spline interpolant for later use with ppval and the spline utility unmkpp. x must be a vector. Y can be a scalar, a vector, or an array of any dimension, subject to the following conditions:

- If x and Y are vectors of the same size, the not-a-knot end conditions are used.
- If x or Y is a scalar, it is expanded to have the same length as the other and the not-a-knot end conditions are used. (See Exceptions (1) below).
- If Y is a vector that contains two more values than x has entries, the first and last value in Y are used as the endslopes for the cubic spline.

Exceptions

1. If Y is a vector that contains two more values than x has entries, the first and last value in Y are used as the endslopes for the cubic spline. If Y is a vector, this means
    - f(x) = Y(2:end-1)
    - df(min(x)) = Y(1)
    - df(max(x)) = Y(end)
2. If Y is a matrix or an N-dimensional array with size(Y,N) equal to length(x)+2, the following hold:
    - f(x(j)) matches the value Y(:,...,:,j+1) for j=1:length(x)

- Df(min(x)) matches Y(:,:,...:,1)
- Df(max(x)) matches Y(:,:,...:,end)

Example 1

This generates a sine curve, then samples the spline over a finer mesh.

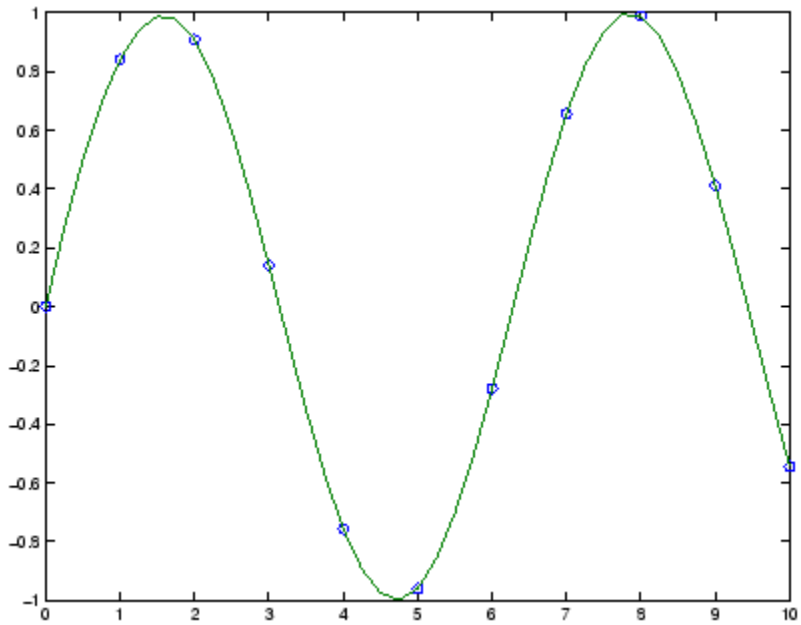
```
x = 0:10;

y = sin(x);

xx = 0:.25:10;

yy = spline(x,y,xx);

plot(x,y,'o',xx,yy)
```



Example 2

This illustrates the use of clamped or complete spline interpolation where end slopes are prescribed. Zero slopes at the ends of an interpolant to the values of a certain distribution are enforced.

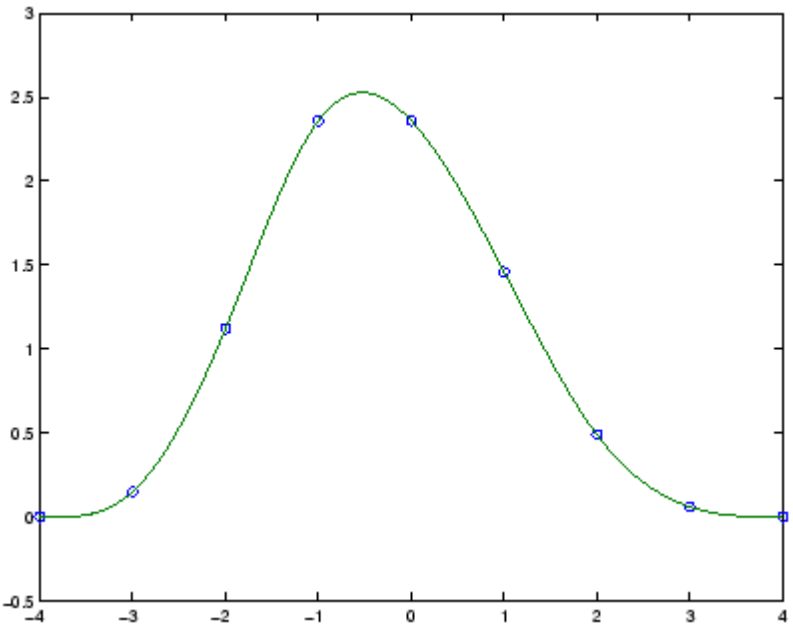
```
x = -4:4;

y = [0 .15 1.12 2.36 2.36 1.46 .49 .06 0];

cs = spline(x,[0 y 0]);

xx = linspace(-4,4,101);

plot(x,y,'o',xx,ppval(cs,xx),'-');
```



Bézier curve

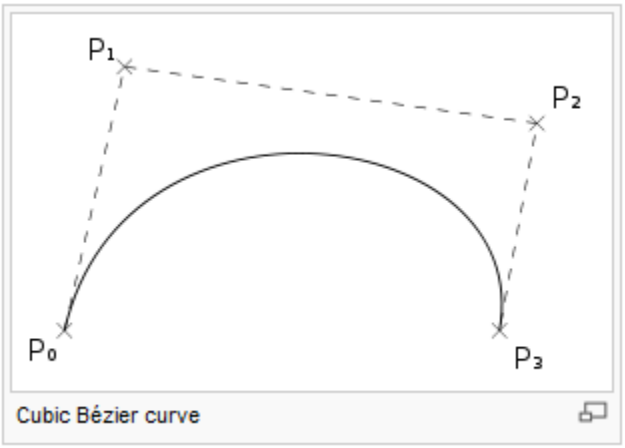
A **Bézier curve** is a parametric curve frequently used in computer graphics and related fields. Generalizations of Bézier curves to higher dimensions are called Bézier surfaces, of which the Bézier triangle is a special case.

In vector graphics, Bézier curves are used to model smooth curves that can be scaled indefinitely. "Paths," as they are commonly referred to in image manipulation programs[note 1] are combinations of linked Bézier

curves. Paths are not bound by the limits of rasterized images and are intuitive to modify. Bézier curves are also used in animation as a tool to control motion

Bézier curves are also used in the time domain, particularly in animation and interface design, e.g., a Bézier curve can be used to specify the velocity over time of an object such as an icon moving from A to B, rather than simply moving at a fixed number of pixels per step. When animators or interface designers talk about the "physics" or "feel" of an operation, they may be referring to the particular Bézier curve used to control the velocity over time of the move in question.

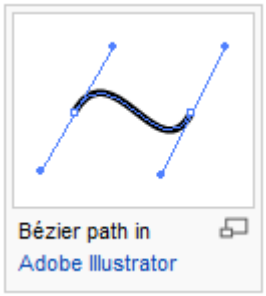
Bézier curves were widely publicized in 1962 by the French engineer Pierre Bézier, who used them to design automobile bodies. The curves were first developed in 1959 by Paul de Casteljau using de Casteljau's algorithm, a numerically stable method to evaluate Bézier curves.



Cubic Bézier curve

Applications

Computer graphics



Bézier path in Adobe Illustrator

Bézier curves are widely used in computer graphics to model smooth curves. As the curve is completely contained in the convex hull of its control points, the points can be graphically displayed and used to manipulate the curve intuitively. Affine transformations such as translation, and rotation can be applied on the curve by applying the respective transform on the control points of the curve.

Quadratic and cubic Bézier curves are most common; higher degree curves are more expensive to evaluate. When more complex shapes are needed, low order Bézier curves are patched together. This is commonly referred to as a "path" in programs like Adobe Illustrator or Inkscape. These poly-Bézier curves can also be seen in the SVG file format. To guarantee smoothness, the control point at which two curves meet must be on the line between the two control points on either side.

The simplest method for scan converting (rasterizing) a Bézier curve is to evaluate it at many closely spaced points and scan convert the approximating sequence of line segments. However, this does not guarantee that the rasterized output looks sufficiently smooth, because the points may be spaced too far apart. Conversely it may generate too many points in areas where the curve is close to linear. A common adaptive method is recursive subdivision, in which a curve's control points are checked to see if the curve approximates a line segment to within a small tolerance. If not, the curve is subdivided parametrically into two segments, $0 \le t \le 0.5$ and $0.5 \le t \le 1$, and the same procedure is applied recursively to each half. There are also forward differencing methods, but great care must be taken to analyse error propagation. Analytical methods where a spline is intersected with each scan line involve finding roots of

cubic polynomials (for cubic splines) and dealing with multiple roots, so they are not often used in practice.

Animation

In animation applications, such as Adobe Flash and Synfig, Bézier curves are used to outline, for example, movement. Users outline the wanted path in Bézier curves, and the application creates the needed frames for the object to move along the path. For 3D animation Bézier curves are often used to define 3D paths as well as 2D curves for keyframe interpolation.

Examination of cases

Linear Bézier curves

Given points $\mathbf{P}_0$ and $\mathbf{P}_1$, a linear Bézier curve is simply a straight line between those two points. The curve is given by

$$\mathbf{B}(t) = \mathbf{P}_0 + t(\mathbf{P}_1 - \mathbf{P}_0) = (1-t)\mathbf{P}_0 + t\mathbf{P}_1 \ , \ t \in [0,1]$$

and is equivalent to linear interpolation.

Quadratic Bézier curves

A quadratic Bézier curve is the path traced by the function $\mathbf{B}(t)$, given points $\mathbf{P}_0$, $\mathbf{P}_1$, and $\mathbf{P}_2$

$$\mathbf{B}(t) = (1-t)^2\mathbf{P}_0 + 2(1-t)t\mathbf{P}_1 + t^2\mathbf{P}_2 \ , \ t \in [0,1].$$

It departs from $\mathbf{P}_0$ in the direction of $\mathbf{P}_1$, then bends to arrive at $\mathbf{P}_2$ in the direction from $\mathbf{P}_1$. In other words, the tangents in $\mathbf{P}_0$ and $\mathbf{P}_2$ both pass through $\mathbf{P}_1$. This is directly seen from the derivate of the Bezier curve:

$$\mathbf{B}'(t) = 2(1-t)(\mathbf{P}_1 - \mathbf{P}_0) + 2t(\mathbf{P}_2 - \mathbf{P}_1)\,.$$

A quadratic Bézier curve is also a parabolic segment.

TrueType fonts use Bézier splines composed of quadratic Bézier curves.

Cubic Bézier curves

Four points $\mathbf{P}_0$, $\mathbf{P}_1$, $\mathbf{P}_2$ and $\mathbf{P}_3$ in the plane or in three-dimensional space define a cubic Bézier curve. The curve starts at $\mathbf{P}_0$ going toward $\mathbf{P}_1$ and arrives at $\mathbf{P}_3$ coming from the direction of $\mathbf{P}_2$. Usually, it will not pass through $\mathbf{P}_1$ or $\mathbf{P}_2$; these points are only there to provide directional information. The distance between $\mathbf{P}_0$ and $\mathbf{P}_1$ determines "how long" the curve moves into direction $\mathbf{P}_2$ before turning towards $\mathbf{P}_3$.

The parametric form of the curve is:

$$\mathbf{B}(t) = (1-t)^3\mathbf{P}_0 + 3(1-t)^2 t\mathbf{P}_1 + 3(1-t)t^2\mathbf{P}_2 + t^3\mathbf{P}_3 \ , \ t \in [0,1].$$

Since the lines and are the tangents of the Bézier curve at and, respectively, cubic Bézier interpolation is essentially the same as cubic Hermite interpolation.

Modern imaging systems like PostScript, Asymptote and Metafont use Bézier splines composed of cubic Bézier curves for drawing curved shapes.

ANTIALIASING

Jaggies

An unwanted artifact of the line/curve drawing algorithms already discussed is that the lines/curves generated have a “jaggy” appearance. Jaggies are an instance of a phenomenon called aliasing. Removal of these artifacts is achieved with the help of anti-aliasing techniques

Aliasing:

In the line drawing algorithms, all rasterzed locations do not match with the true line and have to represent a straight line. This problem is severe in low resolution screens. In such screens line appears like a stair-step. This effect is known as aliasing.

Disadvantages of Alaising:

- Too bad resolution
- Too few colors
- Too few images
- Geometric error
- Numeric error

Antialiasing:

The process of adjusting intensities of the pixels along the line to minimize the effect of aliasing is called antialiasing.

Advantages of Anti-alaising:

1. Improve the devices:
    - High resolution
    - More color levels
    - Faster image sequence
2. Improve the images:
    - Post processing

In computer graphics, *antialiasing* is a software technique for diminishing *jaggies* - stairstep-like lines that should be smooth. Jaggies occur because the output device, the monitor or printer, doesn't have a high enough resolution to represent a smooth line.

Antialiasing reduces the prominence of jaggies by surrounding the stairsteps with intermediate shades of gray (for gray-scaling devices) or color (for color devices). Although this reduces the jagged appearance of the lines, it also makes them fuzzier.

Another method for reducing jaggies is called *smoothing*, in which the printer changes the size and horizontal alignment of dots to make curves smoother. Antialiasing is sometimes called *oversampling.* The undesirable effects of aliasing in computer graphics are known as *artifacts*. Artifacts lower the quality of an image in various ways.

Here is an example of "jaggies", also known as staircasing. The continuous signal on the left becomes rough and jagged in the digital approximation on the right due to low resolution of the sampling grid.

Antialiasing methods were developed to combat the effects of aliasing.

There are three main classes of antialiasing algorithms.

 As aliasing problem is due to low resolution, one easy solution is to increase the resolution , causing sample points to occur more frequently. This increases the cost of image production.

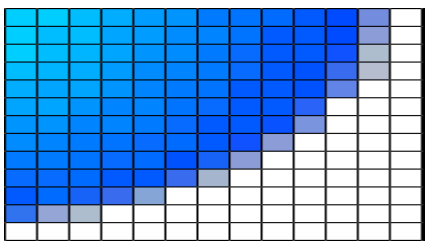
 The image is created at high resolution and then digitally filtered. This method is called supersampling or postfiltering and eliminates high frequencies which are the source of aliases.

 The image can be calculated by considering the intensities over a particular region. This is called prefiltering.

Prefiltering.
Prefiltering methods treat a pixel as an area, and compute pixel color based on the overlap of the scene's objects with a pixel's area. These techniques compute the shades of gray based on how much of a pixel's area is covered by a object.

For example, a modification to Bresenham's algorithm was developed by Pitteway and Watkinson. In this algorithm, each pixel is given an 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 antialiasing is not very prominent, although it still exists.

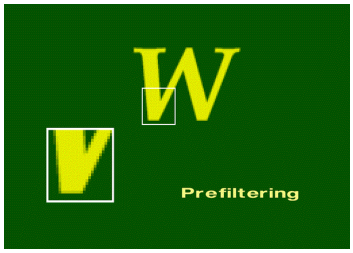
Prefiltering thus amounts to sampling the shape of the object very densely within a pixel region. For shapes other than polygons, this can be very computationally intensive.



Original Image

Without antialiasing, the jaggies are harshly evident.

Prefilteredimage.
Along the character's border, the colors are a mixture of the foreground and background colors.



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.

There are two ***drawbacks*** to this method

 The drawback is that there is a technical and economic limit for increasing the resolution of the virtual image.

 Since the frequency of images can extend to infinity, it just reduces aliasing by raising the Nyquist limit - shift the effect of the frequency spectrum.

Supersampling is basically a three stage process.

 A continuous image I(x,y) is sampled at n times the final resolution. The image is calculated at n times the frame resolution. This is a virtual image.

 The virtual image is then lowpass filtered

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

Algorithm for supersampling

 To generate the origial image, we need to consider a region in the virtual image. The extent of that region determines the regions involved in the lowpass operation. This process is called ***convolution***.

 After we obtain the virtual image which is at a higher resolution, the pixels of the final image are located over superpixels in the virtual image. To calculate the value of the final image at (Si,Sj), we place the filter over the superimage and compute the sum of the filter weights and the surrounding pixels. An adjacent pixel of the final image is calculated by moving the filter S superpixels to the right. Thus the step size is same as the scale factor between the real and the virtual image.

 Filters combine samples to compute a pixel's color. The weighted filter shown on the slide combines nine samples taken from inside a pixel's boundary. Each sample is multiplied by its corresponding weight and the products are summed to produce a weighted average, which is used as the pixel color. In this filter, the center sample has the most influence. The other type of filter is an unweighted filter. In an unweighted filter, each sample has equal influence in determining the pixel's color. In other words, an unweighted filter computes an unweighted average.

 The spatial extent of the filter determines the cutoff frequency. The wider the filter, the lower is the cutoff frequency and the more blurred is the image.

The options available in supersampling are

 The value of S - scaling factor between the virtual and the real images.

 The choice of the extents and the weights of the filter

As far the first factor is concerned, higher the value, the better the result is going to be. The compromise to be made is the high storage cost.

Disadvantages

 It is not a context sensitive technique and thereby results in a lot of wasteful computations.

UNIT – III

Two-dimensional Geometric Transformations – Windowing and Clipping – Clipping of lines and clipping of polygons

TWO-DIMENSIONAL GEOMETRIC TRANSFORMATIONS

Transformation

Transformation is the process of introducing changes in the shape size and orientation of the object .

There are five basic 2D Transformation functions:

- Translation
- Scale
- Rotation
- Shear
- Reflection

Translation

The **Translation** function allows us to move an object in the x-y plane. This is accomplished by adding a translation distance to the x and y coordinates of the original vertex.

$$\begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} tx \\ ty \end{bmatrix} = \begin{bmatrix} x + tx \\ y + ty \end{bmatrix}$$

x' = x + Tx

y' = y + Ty

Scaling

The scale function allows us to change the size of an object. Each of the vertex's original coordinates are multiplied by a scaling factor:

new point: q = s*p = $\begin{bmatrix} sx & 0 \\ 0 & sy \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} sx\,x \\ sy\,y \end{bmatrix}$

x' = x*Sx

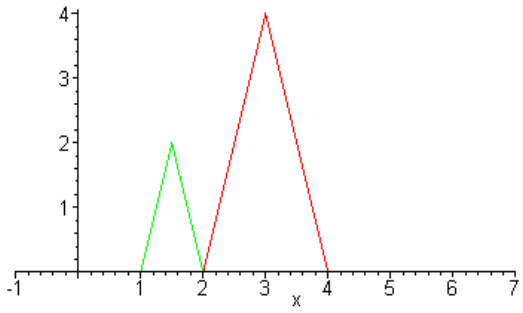
y' = y*Sy

where Sx, Sy > 0

If **Sx = Sy**, the propotions of the object are unchanged

If the scaling factors are less than one, the object will appear smaller and closer to the origin. If the scaling factors are greater than one, the object will appear larger and further from the origin.

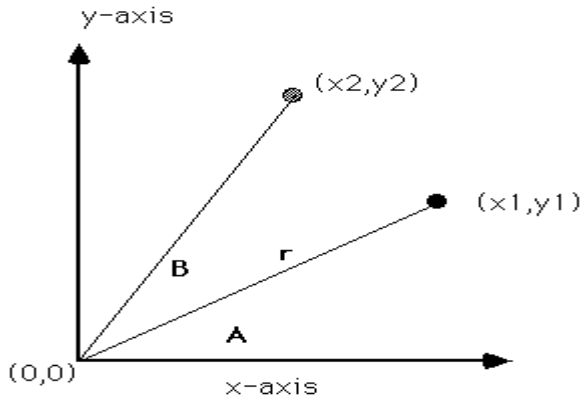
This change in position can be compensated for by scaling from a fixed point, usually a corner of the center.



scale matrix:

Rotation

An object can be rotated about the origin by a specific rotation angle ß. This is accomplished by applying the Rotation transformation to each vertex of the object.



In order to rotate point **(X1,Y1)** to point **(X2,Y2)**, we note the following:

sin(A + B) = Y2/r cos(A + B) = X2/r

sinA = Y1/r cosA = X1/r

From the double angle formulas

sin(A + B) = sinA * cosB + cosA * sinB

Substituting yields

Y2/r = (Y1/r) * cosB + (X1/r) * sinB

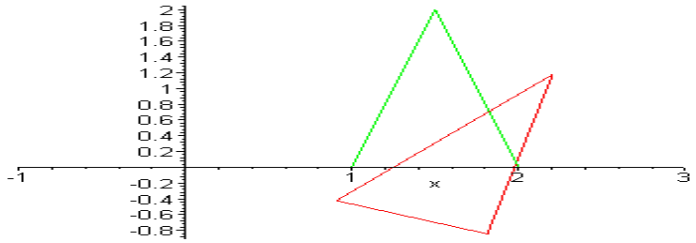
Y2 = Y1cosB + X1sinB

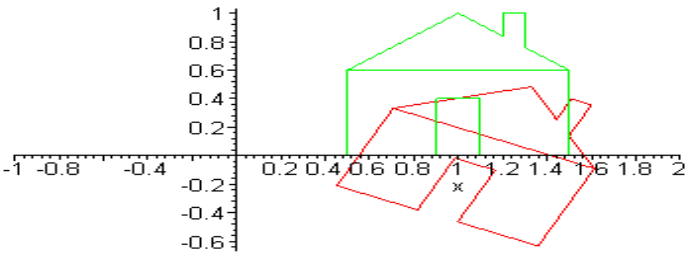
Therefore

X2 = X1cosB - Y1sinB

Y2 = X1sinB + Y1cosB

Below, we see objects that have been rotate by 25 degrees.





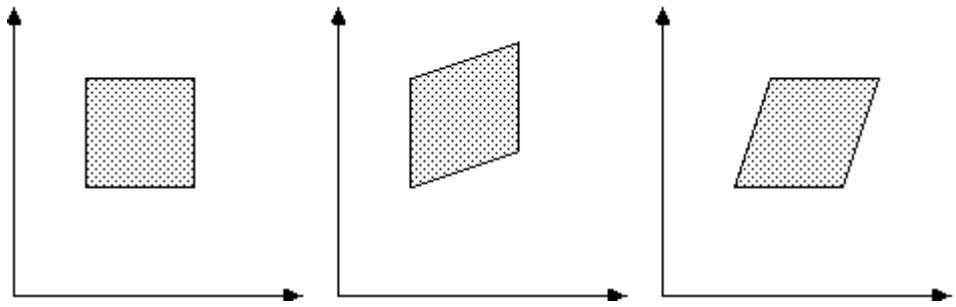
Other Transformations

In addition to the three basic transformations discussed above, there are two additional transformation

- Shear
- Reflection

Shear

The shear transformation distorts an object by scaling one coordinate using the other.



Original Data Y Shear X Shear

The following are the matrix respresentations of the two types of shear:

Matrix/Vector Representation of Translations

shear along x axis = $\begin{bmatrix} 1 & shearx \\ 0 & 1 \end{bmatrix}$

shear along y axis = $\begin{bmatrix} 1 & 0 \\ sheary & 1 \end{bmatrix}$

Reflection

The Reflection transformation produces a mirror image of the object with respect to a specefied axis, point, or line.

The matrix used to reflect about the y-axis is:

(-1 0)
(0 1)

A Reflection about the Y-axis

The matrix used to reflect about the x-axis is:

(1 0)
(0 -1)

A Reflection about the X-axis

In order to reflect about the origin, we use:

(-1 0)
(0 -1)

A Reflection about the Origin

To reflect about the line **Y = X**, we use:

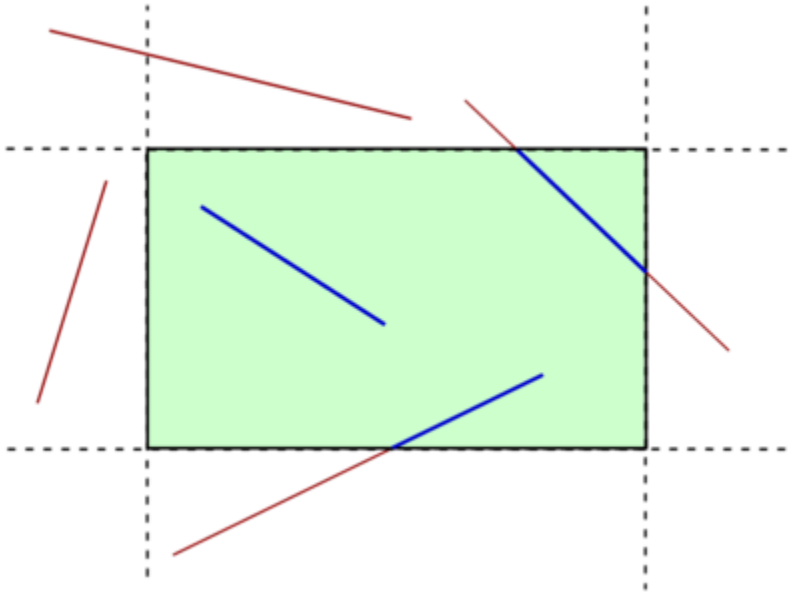
(0 1)
(1 0)

A Reflection about the line Y = X

WINDOWING AND CLIPPING

Line clipping

In computer graphics, **line clipping** is the process of removing lines or portions of lines outside of an area of interest. Typically, any line or part thereof which is outside of the viewing area is removed.



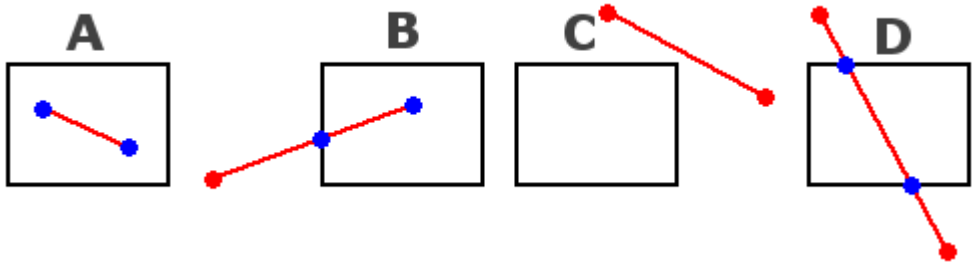
Clipping refers to the removal of part of a scene. Internal clipping removes parts of a picture outside a given region; external clipping removes parts inside a region. We'll explore internal clipping, but external clipping can almost always be accomplished as a by-product.

There is also the question of what primitive types can we clip? We will consider *line* clipping and *polygon* clipping. A line clipping algorithms takes as input two endpoints of line segment and returns one (or more) line segments. A polygon clipper takes as input the vertices of a polygon and returns one (or more) polygons. There are several clipping algorithms. We'll study the Cohen-Sutherland line clipping algorithm to learn some basic concepts. Develop the more efficient Liang-Barsky algorithm and us its insights to culminate with Blinn's line clipping algorithm. The Sutherland-Hodgman polygon clipping algorithm will then be covered and the Weiler-Atherton algorithm, time permitting.

Cohen Sutherland Clipping Algorithm

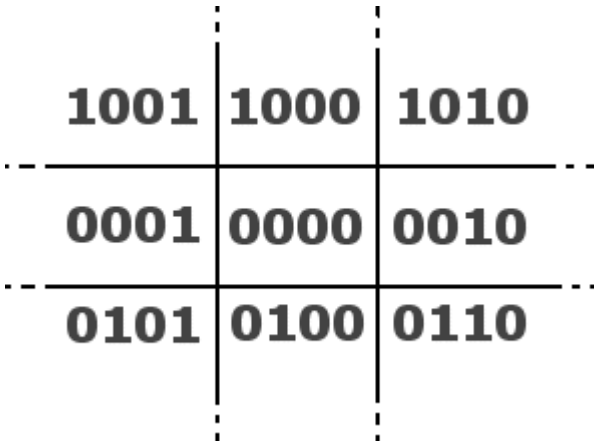
When drawing a 2D line, if one endpoint of the line is outside the screen, and the other inside, you have to clip the line so that only the part of it that's inside the screen remains. Even if both endpoints are outside the screen, it's still possible that a part of the line should be visible. The

clipping algorithm needs to find new endpoints of the lines, that are inside or on the edges of the screen. Here are a few cases, where the black rectangle represents the screen, in red are the old endpoints, and in blue the ones after clipping:



- Case A: both endpoints are inside the screen, no clipping needed.
- Case B: one endpoint outside the screen, that one had to be clipped
- Case C: both endpoint are outside the screen, and no part of the line is visible, don't draw it at all.
- Case D: both endpoint are outside the screen, and a part of the line is visible, clip both endpoints and draw it.

There are tons of different cases, each endpoint can be inside the screen, left of it, right of it, above, below, etc... The Cohen Sutherland Clipping Algorithm can recognize these cases quite efficiently and do the clipping. The algorithm divides the 2D space in 9 regions:



The center region is the screen, and the other 8 regions are on different sides outside the screen. Each region is given a binary number, called an "outcode". The codes are chosen as follows:

- If the region is above the screen, the first bit is 1
- If the region is below the screen, the second bit is 1
- If the region is to the right of the screen, the third bit is 1
- If the region is to the left of the screen, the fourth bit is 1

Obviously an area can't be to the left and the right at the same time, or above and below it at the same time, so the third and fourth bit can't be 1 together, and the first and second bit can't be 1 together. The screen itself has all 4 bits set to 0.

Both endpoints of the line can lie in any of these 9 regions, and there are a few trivial cases:

- If both endpoints are inside or on the edges of the screen, the line is inside the screen or clipped, and can be drawn. This case is the **trivial accept**.
- If both endpoints are on the same side of the screen (e.g., both endpoints are above the screen), certainly no part of the line can be visible on the screen. This case is the **trivial reject**, and the line doesn't have to be drawn.
- These two cases can easily be detected thanks to the outcodes of the regions:
- Trivial Accept: both endpoints have to be in the region with code 0000, so the trivial accept case only happens if code1 | code2 == 0, where code1 and code2 are the codes of both endpoints of the line, and '|' is the binary OR operator, which can only return 0 if both codes are 0.
- Trivial Reject: because of the way the codes of the regions were chosen, only if both endpoints of the line are on the same side of the region, both codes will have two corresponding bits that are both 1. For example, only if both endpoints are on the left of the screen, the fourth bit of both codes is 1. So, the trivial reject case is detected if code1 & code2 != 0, where & is the binary AND operation. The binary AND operation only returns a non zero result if two corresponding bits are 1.

CYRUS-BECK ALGORITHM

Cyrus-beck is the algorithm for line clipping. But in real life the clipping window may not be in the rectangular format. It may be pentagon, hexagon or any polygon. Cyrus beck derived an algorithm in this the clipping window can be of any shape. That's why we call cyrus beck line clipping algorithm as generalized clipping algorithm.

Step 1	Read the vertices of the line (P1 and P2)
Step 2	Read the vertices of the clipping window.
Step 3	calculate the D = P2-P1
Step 4	Take the edge from the clipping window calculate inner normal vector (N)from the taking edge.
Step 5	if(W.N/D.N<0) then t_l=-(W.N/D.N) Else t_u=-(W.N/D.N)
Step 6	Repeat the steps from 4 to 5 until all the edges of the clipping window is processed.
Step 7	Find the maximum lower limit value (t_l) and minimum upper limit value(t_u).
Step 8	if any of the limit i.e t_l or t_u value is more than 1, neglect the line.
Step 9	calculate lower intersection point by using t_l value in the parametric equation $P(t_l)=P1+(P2-P1)t_l$
Step 10	calculate upper intersection point by using t_l value in the parametric equation $P(t_l)=P1+(P2-P1)t_u$
Step 11	Draw the visible line with lower intersection and upper intersection point.
Step_12	Stop

Polygon clipping

An algorithm that clips a polygon must deal with many different cases. The case is particularly note worthy in that the concave polygon is clipped into two separate polygons. All in all, the task of clipping seems rather complex. Each edge of the polygon must be tested against each edge of the clip rectangle; new edges must be added, and existing edges must be discarded, retained, or divided. Multiple polygons may result from clipping a single polygon. We need an organized way to deal with all these cases.

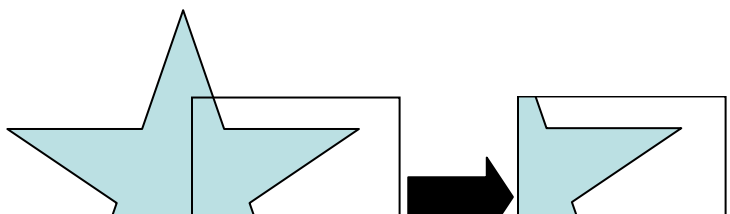
Similarly to lines, areas must be clipped to a window boundary Consideration must be taken as to which portions of the area must be clipped.

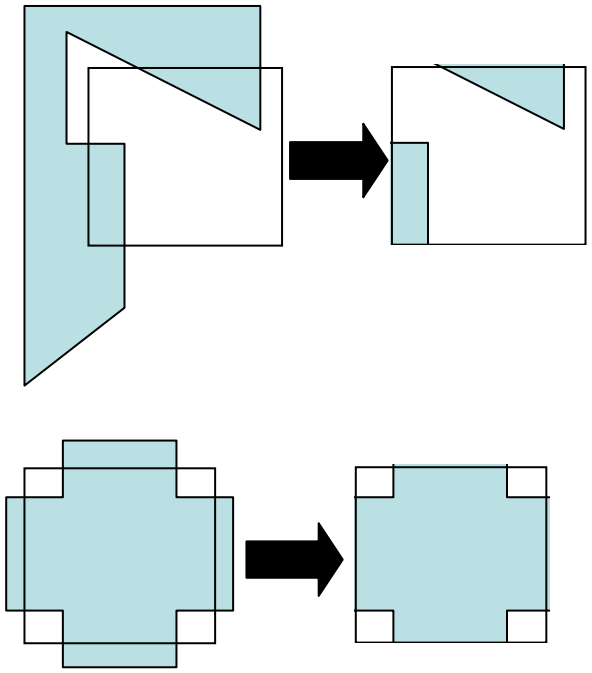
Generalization of polygons:

In a broad sense, a polygon is an unbounded (without ends) sequence or circuit of alternating segments (sides) and angles (corners). An ordinary polygon is unbounded because the sequence closes back in itself in a loop or circuit, while an apeirogon (infinite polygon) is unbounded because it goes on for ever so you can never reach any bounding end point. The modern mathematical understanding is to describe such a structural sequence in terms of an "abstract" polygon which is a partially ordered set (poset) of elements. The interior (body) of the polygon is another element, and (for technical reasons) so is the null polytope or nullitope.

A geometric polygon is understood to be a "realization" of the associated abstract polygon; this involves some "mapping" of elements from the abstract to the geometric. Such a polygon does not have to lie in a plane, or have straight sides, or enclose an area, and individual elements can overlap or even coincide. For example a spherical polygon is drawn on the surface of a sphere, and its sides are arcs of great circles. So when we talk about "polygons" we must be careful to explain what kind we are talking about.

A **digon** is a closed polygon having two sides and two corners. On the sphere, we can mark two opposing points (like the North and South poles) and join them by half a great circle. Add another arc of a different great circle and you have a digon. Tile the sphere with digons and you have a polyhedron called a hosohedron. Take just one great circle instead, run it all the way round, and add just one "corner" point, and you have a monogon or henagon—although many authorities do not regard this as a proper polygon.





An algorithm that clips a polygon must deal with many different cases. The case is particularly note worthy in that the concave polygon is clipped into two separate polygons. All in all, the task of clipping seems rather complex. Each edge of the polygon must be tested against each edge of the clip rectangle; new edges must be added, and existing edges must be discarded, retained, or divided. Multiple polygons may result from clipping a single polygon. We need an organized way to deal with all these cases.

Polygon clipping is one of those humble tasks computers do all the time. It's a basic operation in creating graphic output of all kinds.

There are several well-known polygon clipping algorithms, each having its strengths and weaknesses. The oldest one (from 1974) is called the Sutherland-Hodgman algorithm. In its basic form, it is relatively simple. It is also very efficient in two important cases, one being when the polygon is completely inside the boundaries, and the other when it's completely outside.

A polygon is generally stored as a collection of vertices. Any clipping algorithm takes one collection, and outputs a new collection. A clipped polygon, after all, is also a polygon. Notice that the clipped polygon often will have more vertices than the unclipped one, but it can also have the same number, or less. If the unclipped polygon lies completely outside the clipping boundary, the clipped polygon even has zero vertices.

Sutherland-Hodgman

Sutherland-Hodgman polygon Clipping Algorithm:

>A technique for clipping areas developed by Sutherland & Hodgman

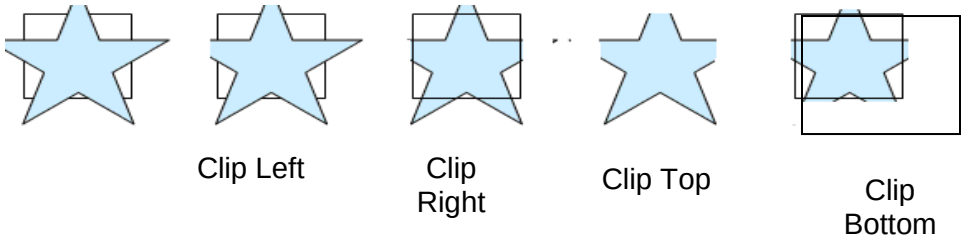
>Put simply the polygon is clipped by comparing it against each boundary in turn

>Beginning with the initial set of polygon vertices. we could first clip the polygon against the left rectangle boundary to produce a new sequence of vertices. The new set of vertices could then be successfully

>Passed to right boundary clipper ,a bottom boundary clipper and a top boundary clipper.

>Sutherland and Hodgman's polygon-clipping algorithm uses a divide-and-conquer strategy: It solves a series of simple and identical problems that, when combined, solve the overall problem. The simple problem is to clip a polygon against a single infinite clip edge. Four clip edges, each defining one boundary of the clip rectangle, successively clip a polygon against a clip rectangle.

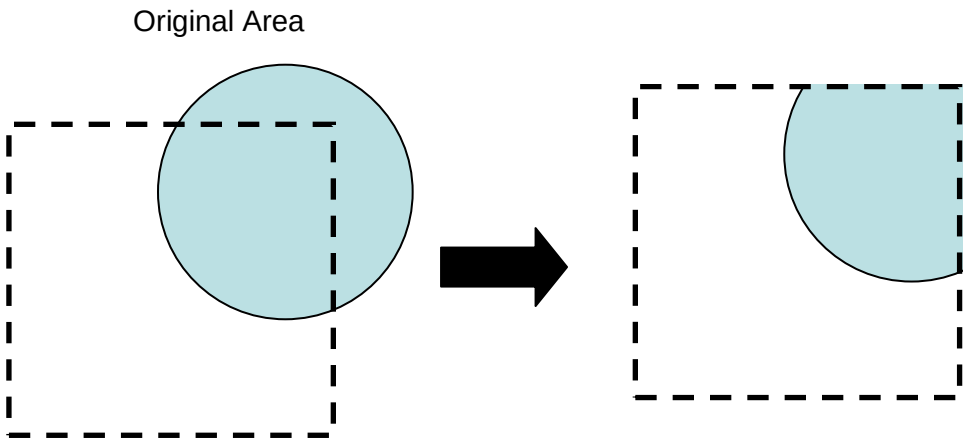
>Note the difference between this strategy for a polygon and the Cohen-Sutherland algorithm for clipping a line: The polygon clipper clips against four edges in succession, whereas the line clipper tests the outcode to see which edge is crossed, and clips only when necessary.



To clip an area against an individual boundary:

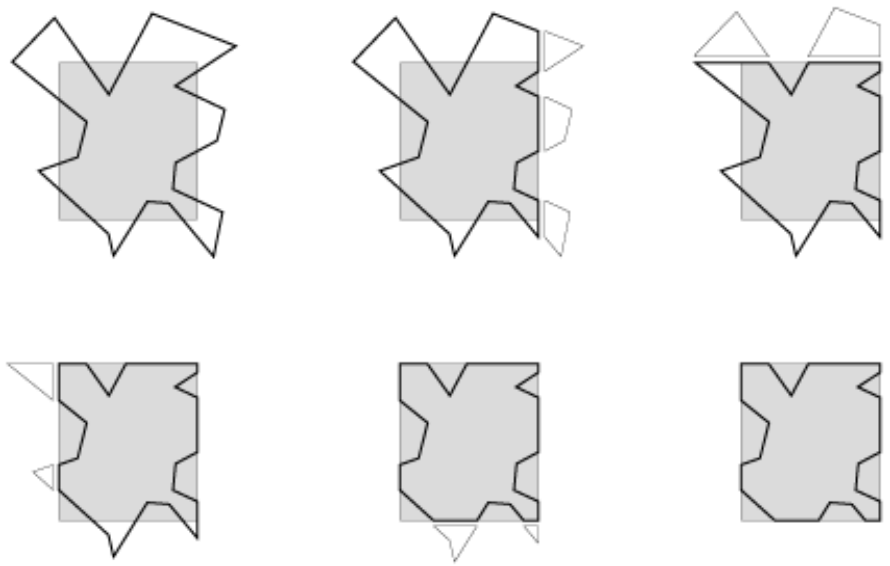
- Consider each vertex in turn against the boundary
- Vertices inside the boundary are saved for clipping against the next boundary
- Vertices outside the boundary are clipped
- If we proceed from a point inside the boundary to one outside, the intersection of the line with the boundary is saved
- If we cross from the outside to the inside intersection point and the vertex are saved

Sutherland-Hodgman Example:



Sutherland-Hodgman uses a divide-and-conquer strategy to attack the problem. First, it clips the polygon against the right clipping boundary. The resulting, partly clipped polygon is then clipped against the top

boundary, and then the process is repeated for the two remaining boundaries. (Of course, it also works in another order.) In a way, it is the most natural thing to do. If you had to clip a paper polygon with a pair of scissors, you would probably proceed the same way.



Sutherland-Hodgman's divide-and-conquer strategy. To clip a polygon against a rectangular boundary window, successively clip against each boundary.

To clip against one boundary, the algorithm loops through all polygon vertices. At each step, it considers two of them, called 'previous' and 'current.' First, it determines whether these vertices are inside or outside the clipping boundary. This, of course, is simply a matter of comparing the horizontal or vertical position to the boundary's position.

Then, it applies the following simple rules:

If 'previous' and 'current' are both inside: Output 'current.'

If 'previous' is inside, and 'current' is outside: Output the intersection point of the corresponding edge and the clipping boundary.

If 'previous' and 'current' are both outside: Do nothing.

If 'previous' is outside, and 'current' is inside: Output the intersection point, and then output 'current.'

UNIT - IV

Three-dimensional concepts – Object representations- Polygon table, quadric surfaces, Splines, Bezier curves and surfaces – Geometric and Modelling transformations – Viewing - Parallel and perspective projections.

Transformations :

Transformation is the process of introducing changes in the shape size and orientation of the object using scaling rotation reflection shearing & translation etc.

Translation

Rotation

Scaling

Reflection

Shear

Translation:

Translation is the process of changing the position of an object in a straight-line path from one coordinate location to another.

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

$$P' = T \cdot P$$

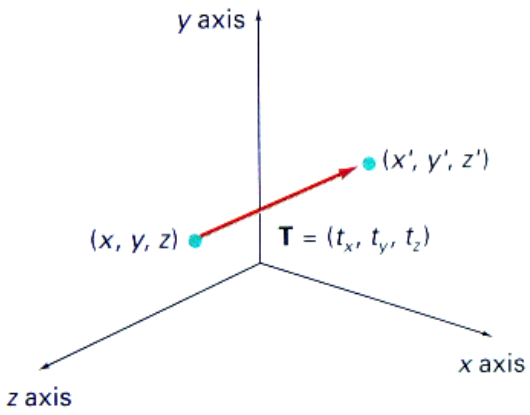
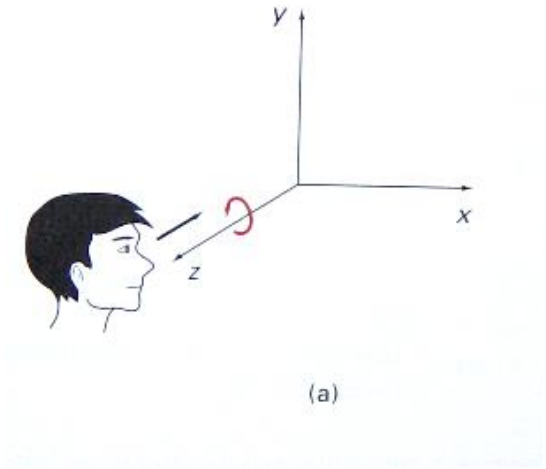
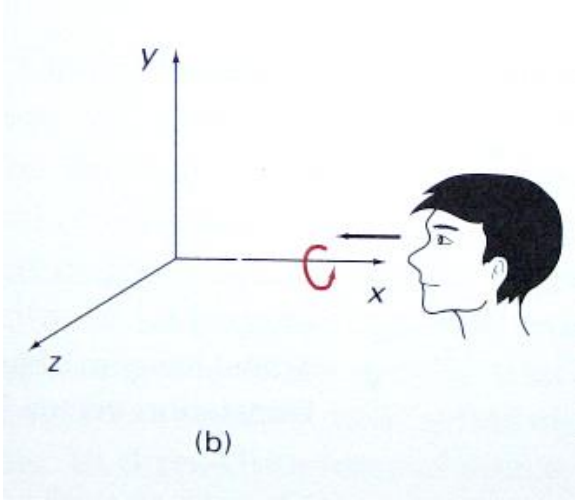


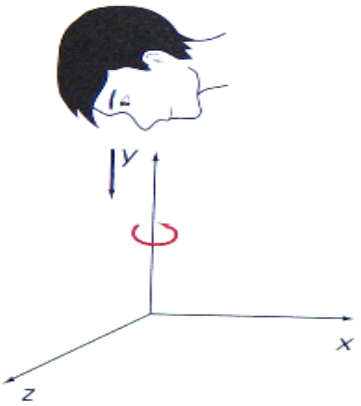
Figure 11-1
Translating a point with translation vector $\mathbf{T} = (t_x, t_y, t_z)$.

Rotation:

Positive rotation angles produce counter clockwise rotations about a coordinate axis







(c)

Figure 11-3
Positive rotation directions about the coordinate axes are counterclockwise, when looking toward the origin from a positive coordinate position on each axis.

Coordinate-Axes Rotations:

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

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

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

$$z' = z$$

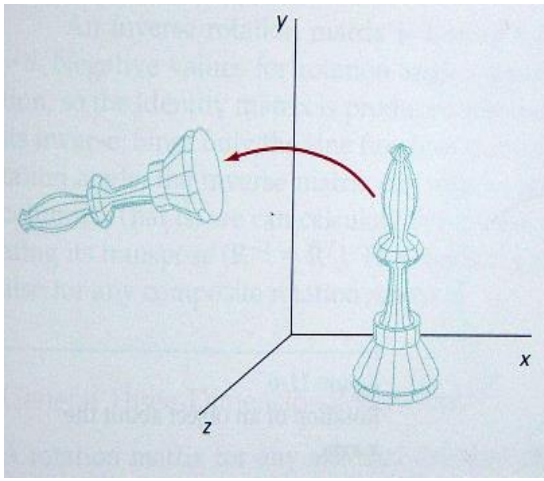


Figure 11-4
Rotation of an object about the z axis.

Scaling ;

The scaling transformations changes the shape of an object and can be carried out by multiplying each vertex (x,y,,z) by scaling factor Sx,Sy,sz where Sx is the scaling factor of x and Sy is the scaling factor of y. Sz is the scaling factor of z.

Scaling with respect to the coordinate origin:

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

$$P' = S \cdot P$$

x’=x*s_x

y’=y*s_y

z’=z*s_z

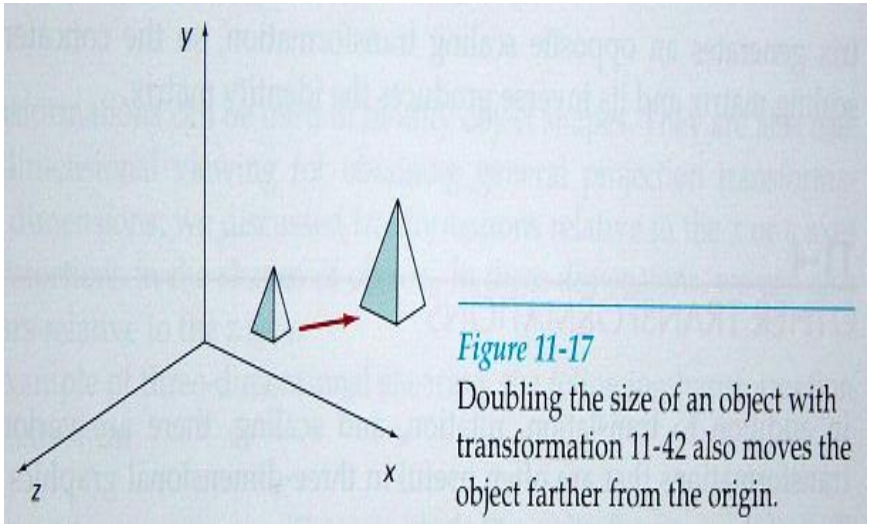


Figure 11-17
Doubling the size of an object with transformation 11-42 also moves the object farther from the origin.

Reflections:

The reflection is actually the transformation that produces a mirror image of an object. For this use some angles and lines of reflection.

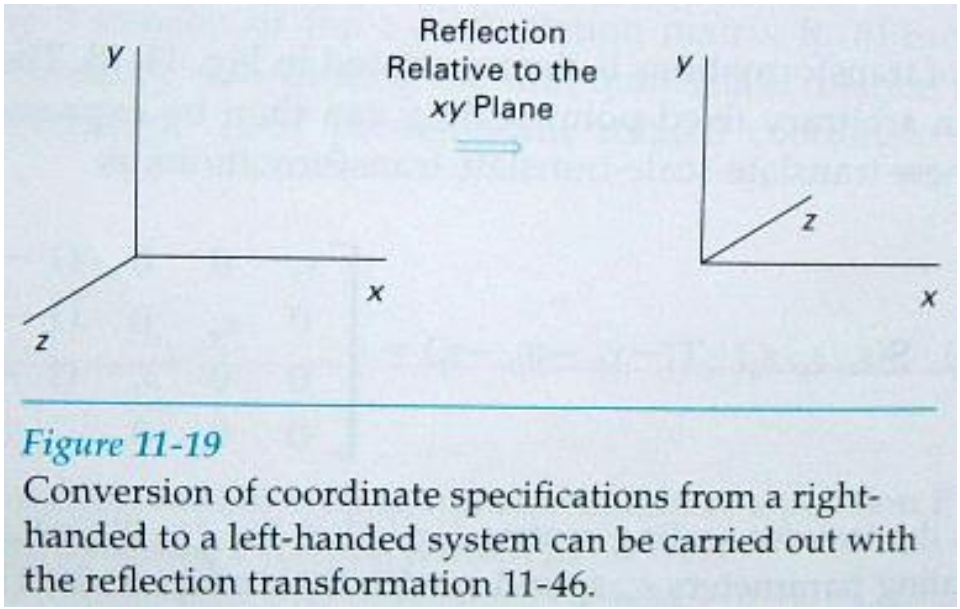


Figure 11-19
Conversion of coordinate specifications from a right-handed to a left-handed system can be carried out with the reflection transformation 11-46.

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

Shears;

The shearing transformation actually slants the object along the X direction or the Y direction or the z direction as required.

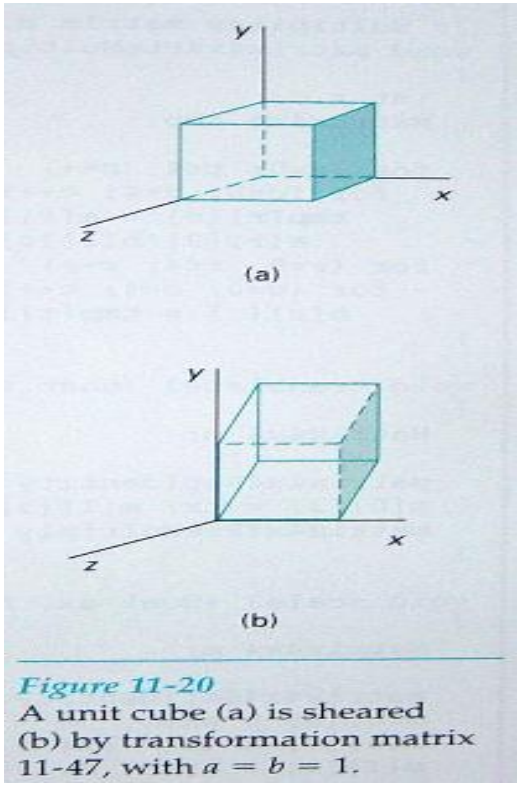


Figure 11-20
A unit cube (a) is sheared (b) by transformation matrix 11-47, with $a = b = 1$.

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

Object Representation

To recognize objects, one must have an internal representation of an object suitable for matching its features to image descriptions. There are several approaches to modelling, and this research uses a surface-based object representation distinguished by its use of shape segment able surfaces organized in a subcomponent hierarchy.

Object Representation Approaches

five criteria for evaluating object representations:

1. accessibility - needed information should be directly available from the model rather than derivable through heavy computation,
2. scope - a wide range of objects should be representable,
3. uniqueness - an object should have a unique representation,
4. stability - small variations in an object should not cause large variations in the model and
5. sensitivity - detailed features should be represented as needed.

Modeling schemes in computer vision mainly belong to two families:

- Property representations - that define objects by properties or constraints (without recourse to an explicit geometric model) the satisfaction of which should lead to unique identification.
- Geometric representations - that represent object shape and structure.

The representations may be expressed implicitly in a computer program or explicitly as an identifiable defined model. The implicit model is not different in competence from the explicit model, but is ignored here because of its lack of generality. We discuss the two families in greater detail in the following subsections

Property Representations

A typical property representation associates lists of expected properties with each object. Some examples of this are:

- color, size and height for image regions in office scenes
- rough object sizes, colors and edge shapes for desk top objects and

- face shape, edge lengths and two dimensional edge angles for identifying polyhedra.

Geometric Representations

Early geometric models were based on three dimensional point or line descriptions. Point models specify the location of significant points relative to the whole object. This method is simple but problematic, because of difficulties in correctly establishing model-to-data correspondences. Edge models specify the location, orientation and shape of edges (typically orientation discontinuities). These characterize the wire-frame shape of an object better than the point models and have stronger correspondence power, but lead to difficulties because of the ambiguity of scene edges and the difficulty of reliably extracting the edges. Further, it is difficult to define and extract intrinsic linear features on curved surfaces. Point and edge models have trouble meeting Marr's scope and sensitivity criteria.

Surface models describe the shape of observable surface regions and their relationship to the whole object (and sometimes to each other). Key questions include how to describe the shape of the surfaces and what constitutes an identifiable surface primitive.

Polygon Table

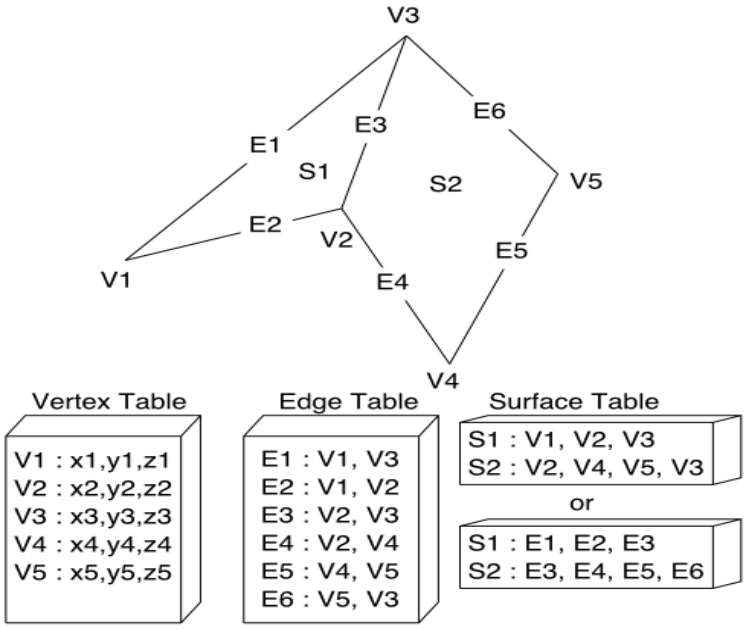
We specify a polygon surface with a set of vertex coordinates and associated attribute parameters.

Objects : set of vertices and associated attributes.

Geometry : stored as three tables : vertex table, edge table, polygon table.

Edge table ?

Tables also allow to store additional information



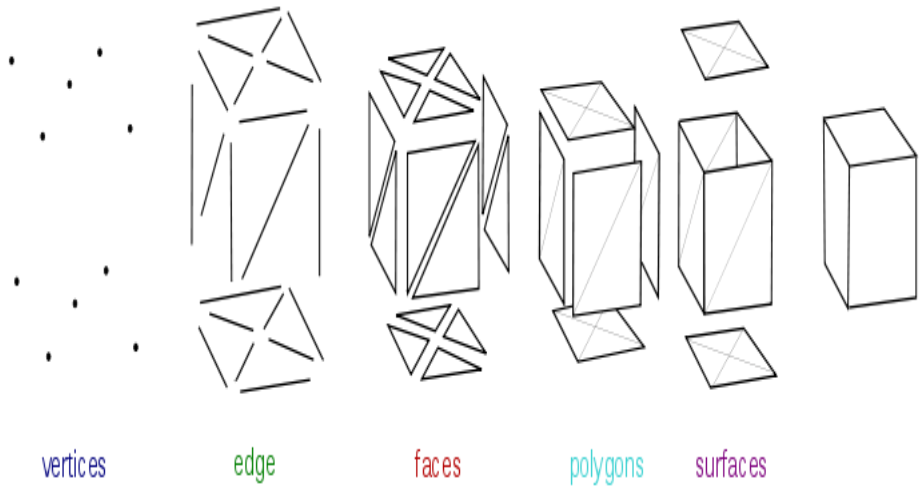
- We specify objects as a set of vertices and associated attributes. This information can be stored in tables, of which there are two types: geometric tables and attribute tables.
- The geometry can be stored as three tables: a vertex table, an edge table, and a polygon table. Each entry in the vertex table is a list of coordinates defining that point. Each entry in the edge table consists of a pointer to each endpoint of that edge. And the entries in the polygon table define a polygon by providing pointers to the edges that make up the polygon.
- We can eliminate the edge table by letting the polygon table reference the vertices directly, but we can run into problems, such as drawing some edges twice, because we don't realize that we have visited the same set of points before, in a different polygon. We could go even further and eliminate the vertex table by listing all the coordinates explicitly in the polygon table, but this wastes space because the same points appear in the polygon table several times.
- Using all three tables also allows for certain kinds of error checking. We can confirm that each polygon is closed, that each point in the vertex table is used in the edge table and that each edge is used in the polygon table.
- Tables also allow us to store additional information. Each entry in the edge table could have a pointer back to the polygons that make use of it. This would allow for quick look-up of those edges which are

shared between polygons. We could also store the slope of each edge or the bounding box for each polygon--values which are repeatedly used in rendering and so would be handy to have stored with the data.

Polygon mesh

A **polygon mesh** or unstructured grid is a collection of vertices, edges and faces that defines the shape of a polyhedral object in 3D computer graphics and solid modeling. The faces usually consist of triangles, quadrilaterals or other simple convex polygons, since this simplifies rendering, but may also be composed of more general concave polygons, or polygons with holes.

Elements of Mesh Modeling



Objects created with polygon meshes must store different types of elements. These include **vertices**, **edges**, **faces**, **polygons** and **surfaces**. In many applications, only vertices, edges and either faces or polygons are stored. A renderer may support only 3-sided faces, so polygons must be constructed of many of these, as shown in the Figure 1. However, many renderers either support quads and higher-sided polygons, or are able to triangulate polygons to tris on the fly, making unnecessary to store a mesh in a triangulated form. Also, in certain applications like head modeling, it is desirable to be able to create both 3 and 4-sided polygons.

A **vertex** is a position along with other information such as color, normal vector and texture coordinates. An **edge** is a connection between two vertices. A **face** is a closed set of edges, in which a *triangle face* has three edges, and a *quad face* has four edges. A **polygon** is a set of **faces**. In systems that support multi-sided faces, polygons and faces are equivalent. However, most rendering hardware supports only 3 or 4-sided faces, so polygons are represented as multiple faces. Mathematically a polygonal mesh may be considered an unstructured grid, or undirected graph, with addition properties of geometry, shape and topology.

Representations

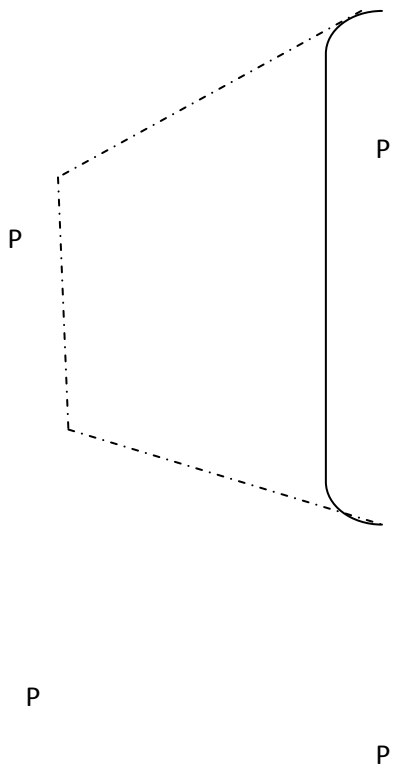
Polygon meshes may be represented in a variety of ways, using different methods to store the vertex, edge and face data. These include:

- Face-Vertex Meshes – A simple list of vertices, and a set of polygons that point to the vertices it uses.
- Winged-Edge Meshes – In which each edge points to two vertices, two faces, and the four (clockwise and counterclockwise) edges that touch it. Winged-Edge meshes allow constant time traversal of the surface, but with higher storage requirements.
- Half-Edge Meshes – Similar to Winged-Edge meshes except that only half the edge traversal information is used.
- Quad-Edge Meshes – A quad-edge mesh stores edges, half-edges, and vertices without any reference to polygons. The polygons are implicit in the representation, and may be found by traversing the structure. Memory requirements are similar to half-edge meshes.
- Corner-Table – A corner-table stores vertices in a predefined table, such that traversing the table implicitly defines polygons. This is in essence the "triangle fan" used in hardware graphics rendering. The representation is more compact, and more efficient to retrieve polygons, but operations to change polygons are slow. Furthermore, - Corner-Tables do not represent meshes completely. Multiple corner-tables (triangle fans) are needed to represent most meshes.
- Vertex-Vertex Meshes – A *vv* mesh represents only vertices, which point to other vertices. Both the edge and face information is implicit in the representation. However, the simplicity of the representation allows for many efficient operations to be performed on meshes.

BEZIER CURVE

Bezier curve is another approach for the construction of the curve. It is a different way of specifying the curve than B-spline curve. Bezier curves have a number of properties that make them highly useful and convenient for curve and surface design. They are also easy to implement. Therefore Bezier curves are widely available in various CAD systems and in general graphic packages. In this section we will discuss the cubic Bezier curves. The reason for choosing cubic Bezier curve is that they provide reasonable design flexibility and also avoid the large no of calculation.

In cubic Bezier curve four control points are used to specify complete the curve. Unlike the B-spline curve, we do not add intermediate points and smoothly extend Bezier curve, but we pick four more points and construct a second curve which can be attached to the first curve smoothly by selecting appropriate control points.



The Bezier curve begins at the first control point and ends at the fourth control point. This means that if we want to connect two Bezier curves, we have to make the first control points of the second Bezier curve match the last control point of the first curve .we can also observe that the start of the curve, the curve is tangent to the line connecting first and fourth control point. This means that, to join two Bezier curves smoothly we have to place the third and fourth control points of the first curve on the same line specified by the first and second control points of the second curve.

The Bezier matrix for periodic cubic polynomial is;

$$M_{B=} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \quad G_B = \begin{bmatrix} P \\ 1 \\ P \end{bmatrix}$$

$$\mathbf{P(u)=U.\ M_B\ .\ G_B}$$

Examination of cases

Linear Bézier curves

Given points $\mathbf{P}_0$ and $\mathbf{P}_1$, a linear Bézier curve is simply a straight line between those two points. The curve is given by

$$\mathbf{B}(t) = \mathbf{P}_0 + t(\mathbf{P}_1 - \mathbf{P}_0) = (1-t)\mathbf{P}_0 + t\mathbf{P}_1 \ , \ t \in [0,1]$$

and is equivalent to linear interpolation.

Quadratic Bézier curves

A quadratic Bézier curve is the path traced by the function $\mathbf{B}(t)$, given points $\mathbf{P}_0$, $\mathbf{P}_1$, and $\mathbf{P}_2$

$$\mathbf{B}(t) = (1-t)^2\mathbf{P}_0 + 2(1-t)t\mathbf{P}_1 + t^2\mathbf{P}_2 \ , \ t \in [0,1].$$

It departs from $\mathbf{P}_0$ in the direction of $\mathbf{P}_1$, then bends to arrive at $\mathbf{P}_2$ in the direction from $\mathbf{P}_1$. In other words, the tangents in $\mathbf{P}_0$ and $\mathbf{P}_2$ both pass through $\mathbf{P}_1$. This is directly seen from the derivate of the Bezier curve:

$$\mathbf{B}'(t) = 2(1-t)(\mathbf{P}_1 - \mathbf{P}_0) + 2t(\mathbf{P}_2 - \mathbf{P}_1)\,.$$

A quadratic Bézier curve is also a parabolic segment.

TrueType fonts use Bézier splines composed of quadratic Bézier curves.

Cubic Bézier curves

Four points $\mathbf{P}_0$, $\mathbf{P}_1$, $\mathbf{P}_2$ and $\mathbf{P}_3$ in the plane or in three-dimensional space define a cubic Bézier curve. The curve starts at $\mathbf{P}_0$ going toward $\mathbf{P}_1$ and arrives at $\mathbf{P}_3$ coming from the direction of $\mathbf{P}_2$. Usually, it will not pass through $\mathbf{P}_1$ or $\mathbf{P}_2$; these points are only there to provide directional

information. The distance between $\mathbf{P}_0$ and $\mathbf{P}_1$ determines "how long" the curve moves into direction $\mathbf{P}_2$ before turning towards $\mathbf{P}_3$.

The parametric form of the curve is:

$$\mathbf{B}(t) = (1-t)^3\mathbf{P}_0 + 3(1-t)^2 t\mathbf{P}_1 + 3(1-t)t^2\mathbf{P}_2 + t^3\mathbf{P}_3 \ , \ t \in [0,1].$$

Another approach to construct the Bezier curve is called midpoint approach. In this approach the Bezier curve can be constructed simply by taking midpoints. In midpoint approach midpoints of the lines connecting four control points(A, B, C D) are determined (AB, BC, CD). These midpoints are connected by line segments and their midpoints ABC and BCD are determined. Finally these two midpoints are connected by line segments and its midpoint ABCD is determined. The point ABCD on the Bezier curve divides the original curve int two sections. This makes the points A, AB, ABC and ABCD are the control points for the first section and the points ABCD, BCD, CD and D are the control points for the second section. By considering two sections separately we can get two more sections for each separate section i.e. the original Bezier curve gets divided into four different curves. This process can be repeated to split the curve into smaller sections until we have sections so short that they can be replaced by straight lines or even until the sections are not bigger than individual pixels.

Algorithm:

Step1: Get four control points say $A(X_A,Y_A)$, $B(X_B,Y_B)$, $C(X_C,Y_C)$, $D(X_D,Y_D)$.

Step2: Divide the curve represented by points A, B,C and D in two sections,

$X_{AB} = (X_A + X_B) / 2$
$Y_{AB} = (Y_A + Y_B) / 2$
$X_{BC} = (X_B + X_c) / 2$
$Y_{BC} = (Y_B + Y_C) / 2$
$X_{CD} = (X_C + X_D) / 2$
$Y_{CD} = (Y_C + Y_D) / 2$
$X_{ABC} = (X_{AB} + X_{BC}) / 2$
$Y_{ABC} = (Y_{AB} + Y_{BC}) / 2$
$X_{BCD} = (X_{BC} + X_{CD}) / 2$

$Y_{BCD} = (Y_{BC} + Y_{CD}) / 2$
$X_{ABCD} = (X_{ABC} + X_{BCD}) / 2$
$Y_{ABCD} = (Y_{ABC} + Y_{BCD}) / 2$

Step3: Repeat the step 2 for section A, AB, ABC and ABCD and section ABCD, BCD, CD and D.

Step4: Repeat step 3 until we have sections so short that can be replaced by straight lines.

Step5: Replace small sections by straight lines.

Step6: Stop.

Hermite curves

Hermite curves are very easy to calculate but very powerfull to use. They are used to smoothly interpolate data between key-points (like object movement in keyframe animation or camera control). Understanding the mathematical background of hermite curves will help you to understand the entire family of splines.

Maybe you have some experiences with 3D programming and already used them without knowing that.. (the so called kb-splines, curves with control over tension, continuity and bias are just a special form of the hermite curves).

The Math

To keep it simple we first start with some simple stuff. We also only talk about 2 dimensional curves here. If you need a 3d curve just do the same with the z-coordinate what you did with y or x. Hermite curves work in in any dimension.

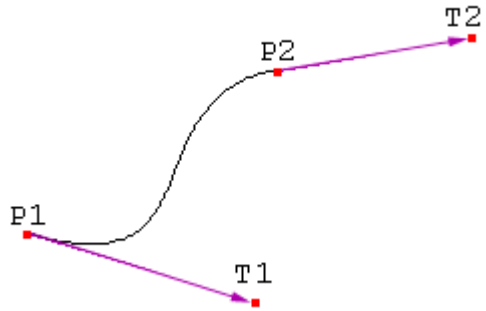
To calculate a hermite curve you need the following vectors:

P1: the startpoint of the curve

T1: the tangent (e.g. direction and speed) how the curve lefts the startpoint

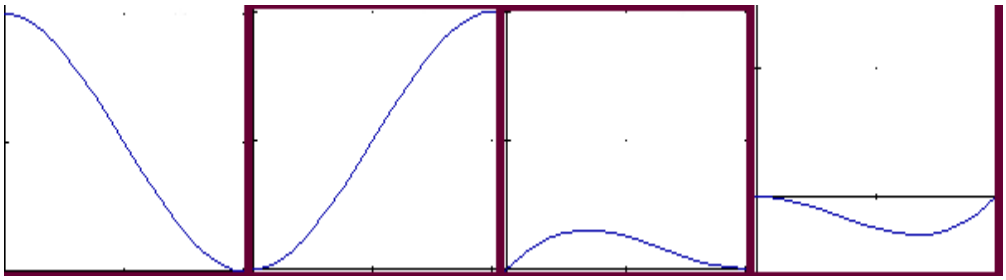
P2: he endpoint of the curve

T2: the tangent (e.g. direction and speed) how the curves enters the endpoint



These 4 vectors are simply multiplied with 4 hermite basis functions and summed together.

h1(s) = 2s^3 - 3s^2 + 1
h2(s) = -2s^3 + 3s^2
h3(s) = s^3 - 2s^2 + s
h4(s) = s^3 - s^2
Below are the 4 graphs of the 4 functions (from left to right: h1, h2, h3, h4)



(all graphs except the 4th have been plotted from 0,0 to 1,1)

Take a closer look at functions h1 and h2:
h1 starts at 1 and goes slowly to 0.
h2 starts at 0 and goes slowly to 1.

If you now multiply the startpoint with **h1** and the endpoint with **h2**. Let **s** go from 0 to 1 to interpolate between start and endpoint.

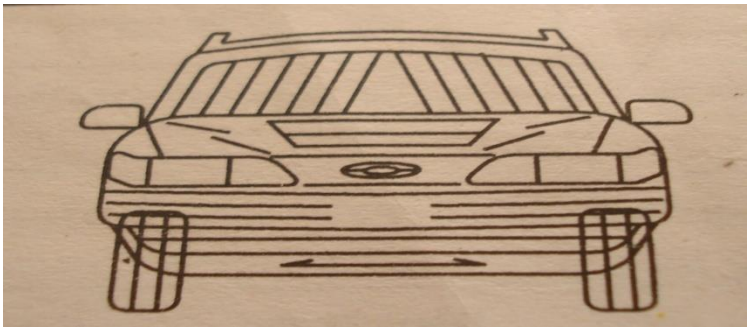
h3 and **h4** are applied to the tangents in the same manner. They take care, that the curve bends into the desired direction at the start and endpoint.

3D geometric modelling:

Wire frame model:

Here the edges of objects are displayed as lines(wires).for curved surface, contour lines are added.The image assumes the appearance of a frame constructed out of wire,hence it is called wire frame model, But this model is inadequate for 3-D representation of objects, to avoid this inadequacy it is supported Hidden line limitation.

Example :



Surface Model:

Here models are developed by representing the shape of its outside surfaces. It does not contain any information about its internal areas (away form visibility).

The several Modeling Techniques are:

Ruled Surface: Surfaces are generated by moving a straight line through space with its end points tracing curves.

Surface of revolution: Surfaces are created by rotating a curve or line through a circular arc.

Sweep surfaces: Surfaces are generated by moving one curve through another.

Sculptured surfaces: Complex curved surface can be created by connecting surface shapes (patches) into a quilt like pattern to approximate the surface.

Interpretation of a surface model can still be ambiguous.

The other techniques are also used in this model such as Hidden surface Removal, Shading, specularity, light sources and Depth – Cue – which provides a depth information such as which part of the object is nearer to the eyes and which are far away, etc.,

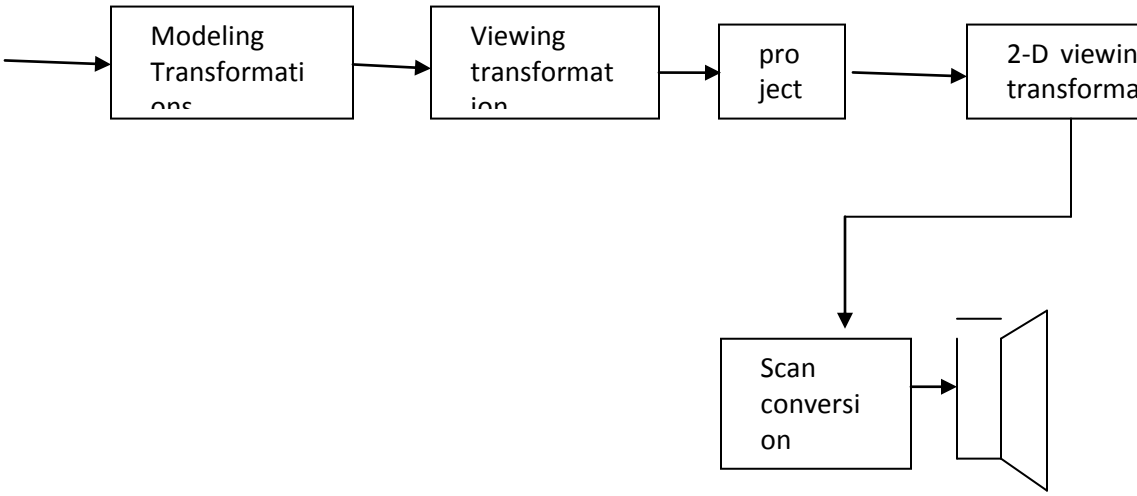
Solid Model: Provides facilities for creating, modifying and inspecting models of 3-D solid objects.

3D Viewing:

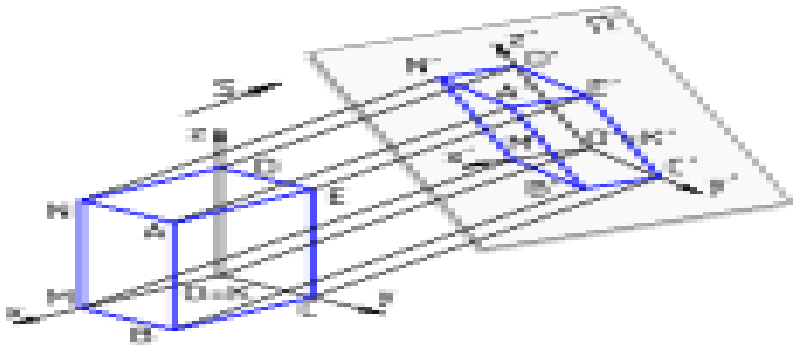
3-D viewing of objects requires the specification of a projection plane(called the view plan),the center of projection (view point).on the direction of projection and a view volume in would coordinates.

The complete 3-D viewing process is described by the following steps:

1. Transform from world coordinates to normalized viewing coordinates by applying the transformation Npar or Nper.
2. clip normalized viewing coordinates against the canonical clipping volume.
3. project on to the screen projection plane using the the projections “par or per”.
4. Apply the appropriate (2-D) viewing transformation.



Graphical projection



.

Graphical projection is a protocol by which an image of an three-dimensional object is projected onto a planar surface without the aid of mathematical calculation, used in technical drawing.

Increasing the focal length and distance of the camera to infinity in a perspective projection results in a parallel projection.

3D projection is method of mapping three-dimensional points to a two-dimensional plane. As most current methods for displaying graphical data

are based on planar two-dimensional media, the use of this type of projection is widespread, especially in computer graphics, engineering and drafting

The projection is achieved by the use of imaginary "projectors". The projected, mental image becomes the technician's vision of the desired, finished picture. By following the protocol the technician may produce the envisioned picture on a planar surface such as drawing paper. The protocols provide a uniform imaging procedure among people trained in technical graphics (mechanical drawing, computer aided design, etc.).

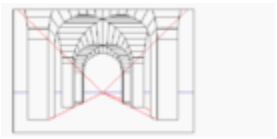
There are two types of graphical projection, categories each with its own protocol:

- parallel projection
- perspective projection

Isometric projection.

Oblique projection.

Oblique projection.



One-point perspective projection.

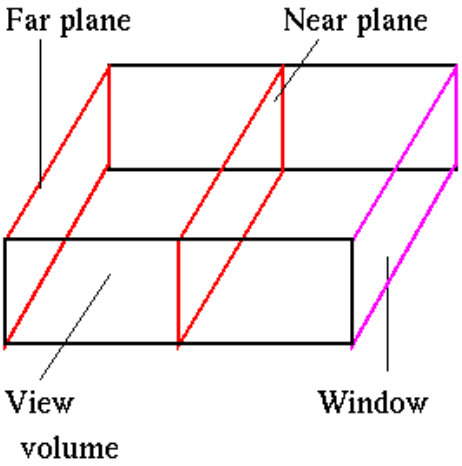
Types of projection

Several types of graphical projection compared.

Parallel Projection
Perspective projection

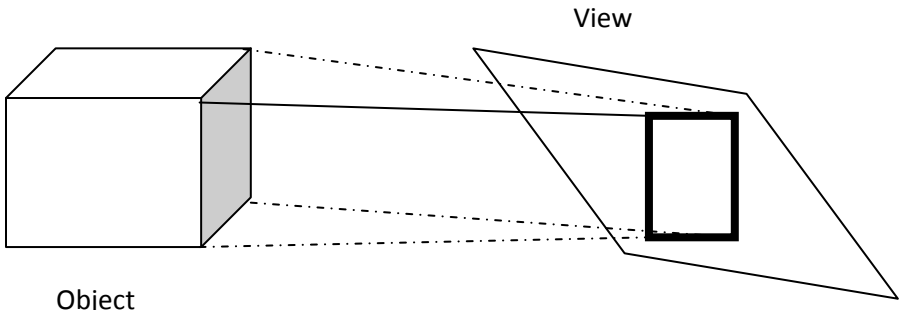
Introduction to parallel projection

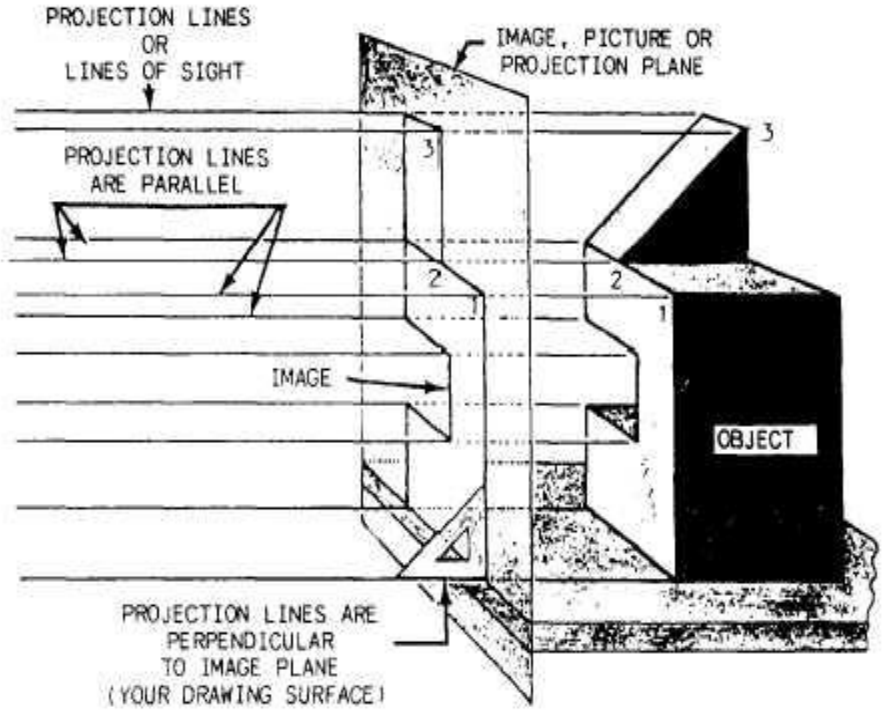
Parallel projections are very useful for engineers, and draftsmen to create working drawings, blueprints, schematics or working drawings of objects that preserve its shape and scale.



Parallel projection:

In parallel projection, z coordinates is discarded and parallel lines from each vertex on the object are extended until they intersect the view plane. The point of intersection is the projection of the vertex. We connect the projected vertices by line segments which correspond to connections on the original object.





Types of parallel projection:

Parallel projections are basically categorized into two types, depending on the relation between the direction of projection and the normal to the view plane. When the direction of the projection is normal (perpendicular) to the view plane, we have an orthographic parallel projection. Otherwise, we have an oblique parallel projection.

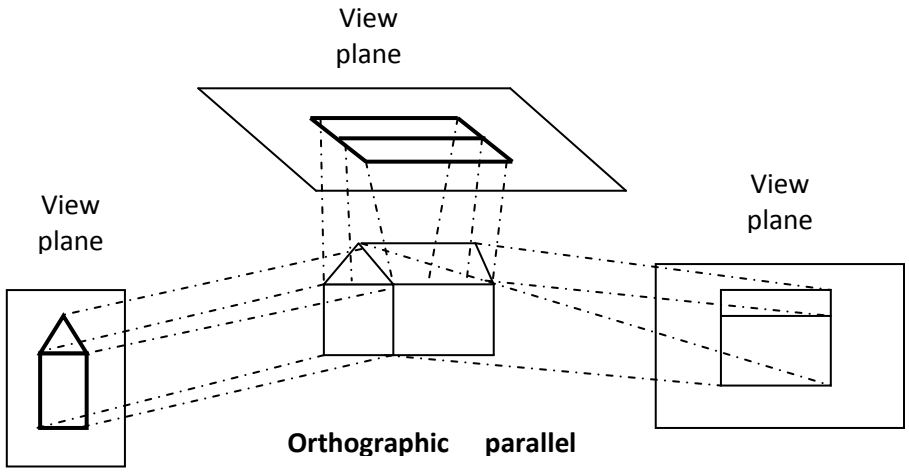
Orthographic projection:

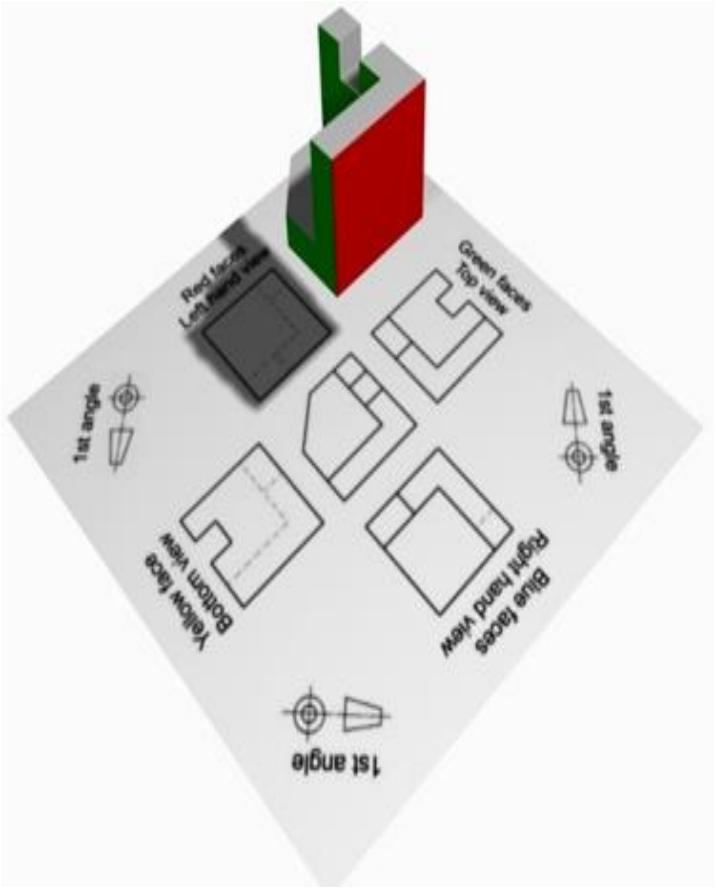
The most common types of orthographic projections are the front projection, top projection and side projection. In all these, the projection plane (view plane) is perpendicular to the principle axis. These projections are often used in engineering drawing to depict machine parts, assemblies, buildings and so on.

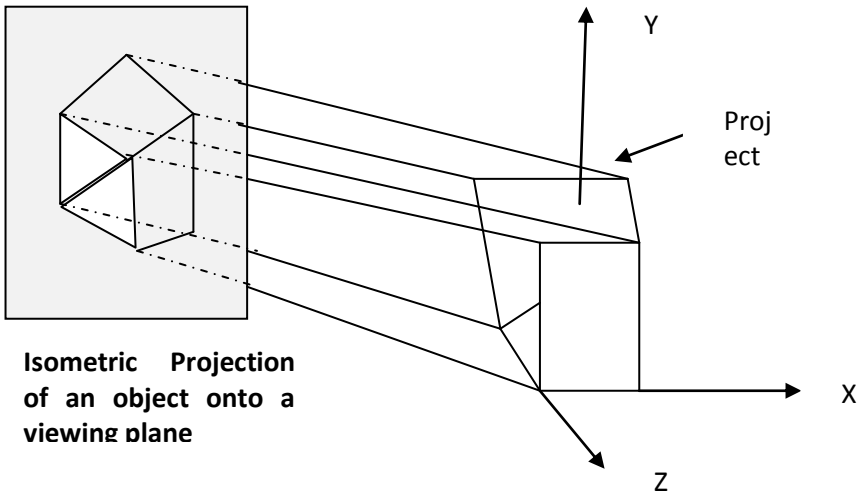
The orthographic projection can display more than one face of an object. Such an orthographic projection is called axonometric orthographic projection. It uses projection planes (view plane) that are not normal to a principle axis. They resemble the perspective projection in this way, but

differ in that the foresightening is uniform rather than being related to the distance from the center of projection. Parallelism of lines is preserved but angles are not. The most commonly used axonometric orthographic projection is the isometric projection.

The isometric projection can be generated by aligning the view plane so that it intersects each coordinates axis in which the object is defined at the same distance from the origin. The isometric projection is obtained by aligning the projection vector with the cube diagonal. It uses an useful property that all three principle axes are equally foresightened, allowing measurements along the axes to be made to the same scale.



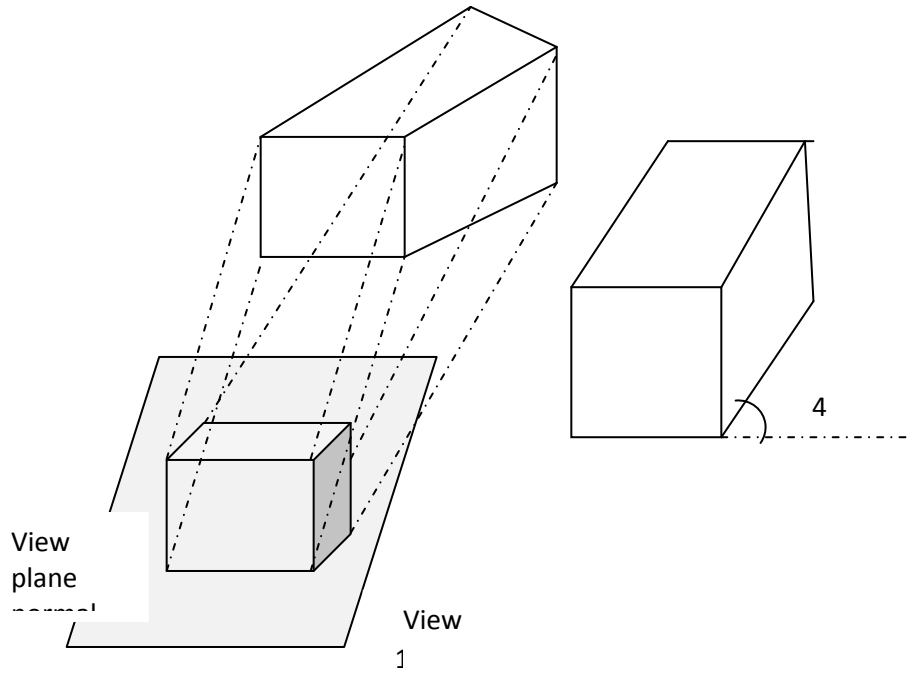




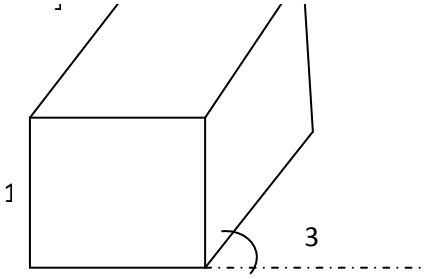
Isometric Projection of an object onto a viewing plane

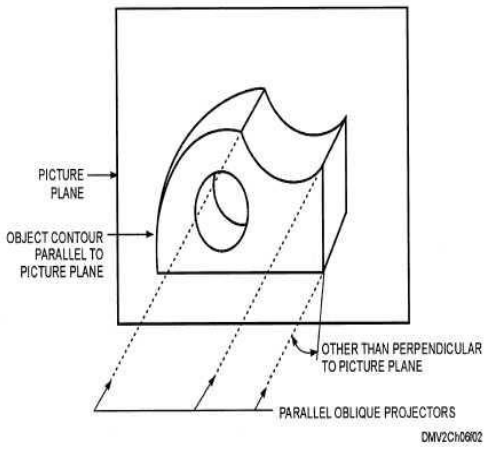
Oblique Projection:

An oblique projection is obtained by projecting points along parallel lines that are not perpendicular to the projection plane. Notice that the view plane normal and the direction of projection are not the same. The oblique projections are further classified as the cavalier and cabinet projections. For the cavalier, the direction of projection makes a 45^o angle with the view plane. As a result, the projection of a line perpendicular to the view plane has the same length as the line itself; that is, there is no foreshorting. In the following figure the cavalier projections of a unit cube with $\propto = 45^o$ and $\propto = 30^o$

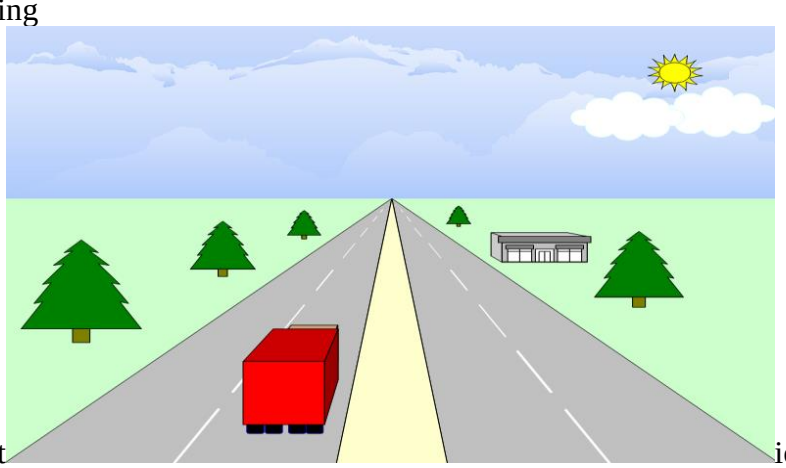


Oblique parallel projection





Perspective projection:

The perspective projection, on the other hand, produces realistic views but does not preserve relative proportions. In perspective projection, the lines of projection are not parallel. Instead, they all converge at a single point called the centre of projection or projection reference point. The object positions are transformed to the view plane along these converged projection lines and the projected view of an object is determined by calculating the intersect  ion of the converged projection lines with the view plane.

UNIT - V
Visible Surface Detection Methods – Computer Animation.

Visible-Surface Detection Methods

Problem definition of Visible-Surface Detection Methods:
To identify those parts of a scene that are visible from a chosen viewing position. Surfaces which are obscured by other opaque surfaces along the line of sighn (projection) are invisible to the viewer.

Classification of Visible-Surface Detection Algorithms:
1. Object-space Methods
Compare objects and parts of objects to each other within the scene definition to determine which surfaces, as a whole, we should label as visible:

For each object in the scene do

Begin
1. Determine those part of the object whose view is unobstructed by other parts of it or any other object with respect to the viewing specification.
2. Draw those parts in the object color.

End
Image-space Methods (Mostly used)
Visibility is determined point by point at each pixel position on the projection plane.

For each pixel in the image do
Begin
1. Determine the object closest to the viewer that is pierced by the projector through the pixel
2. Draw the pixel in the object colour.
End

Application of Coherence in Visible Surface Detection Methods:
- Making use of the results calculated for one part of the scene or image for other nearby parts.
- Coherence is the result of local similarity
- As objects have continuous spatial extent, object properties vary smoothly within a small local region in the scene. Calculations can then be made incremental.

Types of coherence:

1. Object Coherence:

Visibility of an object can often be decided by examining a circumscribing solid (which may be of simple form, eg. A sphere or a polyhedron.)

2. Face Coherence:

Surface properties computed for one part of a face can be applied to adjacent parts after small incremental modification. (eg. If the face is small, we sometimes can assume if one part of the face is invisible to the viewer, the entire face is also invisible).

3. Edge Coherence:

The Visibility of an edge changes only when it crosses another edge, so if one segment of an nonintersecting edge is visible, the entire edge is also visible.

4. Scan line Coherence:

Line or surface segments visible in one scan line are also likely to be visible in adjacent scan lines. Consequently, the image of a scan line is similar to the image of adjacent scan lines.

5. Area and Span Coherence:

A group of adjacent pixels in an image is often covered by the same visible object. This coherence is based on the assumption that a small enough region of pixels will most likely lie within a single polygon. This reduces computation effort in searching for those polygons which contain a given screen area (region of pixels) as in some subdivision algorithms.

6. Depth Coherence:

The depths of adjacent parts of the same surface are similar.

7. Frame Coherence:

Pictures of the same scene at successive points in time are likely to be similar, despite small changes in objects and viewpoint, except near the edges of moving objects.

Most visible surface detection methods make use of one or more of these coherence properties of a scene. To take advantage of regularities in a scene, eg. Constant relationships often can be established between objects and surfaces in a scene.

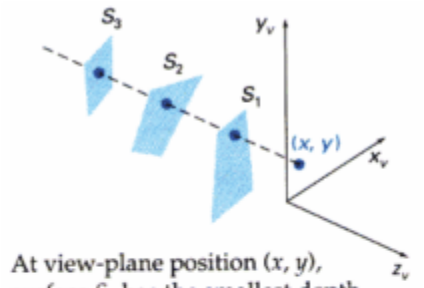
Algorithms :

Depth-Buffer Method (Z-Buffer Method)

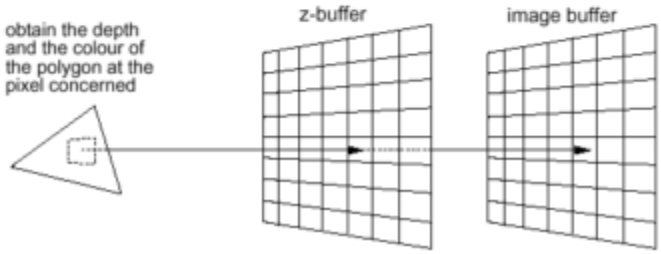
This approach compares surface depths at each pixel position on the projection plane. Object depth is usually measured from the view plane along the z axis of a viewing system.

This method requires 2 buffers: one is the image buffer and the other is called the z-buffer (or the depth buffer). Each of these buffers has the same resolution as the image to be captured. As surfaces are processed, the image buffer is used to store the color values of each pixel position and the z-buffer is used to store the depth values for each (x,y) position.

Algorithm:
1. Initially each pixel of the z-buffer is set to the maximum depth value (the depth of the back clipping plane).
2. The image buffer is set to the background color.
3. Surfaces are rendered one at a time.
4. For the first surface, the depth value of each pixel is calculated.
5. If this depth value is smaller than the corresponding depth value in the z-buffer (ie. it is closer to the view point), both the depth value in the z-buffer and the color value in the image buffer are replaced by the depth value and the color value of this surface calculated at the pixel position.
6. Repeat step 4 and 5 for the remaining surfaces.
7. After all the surfaces have been processed, each pixel of the image buffer represents the color of a visible surface at that pixel.



At view-plane position (x, y), surface S_1 has the smallest depth from the view plane and so is visible at that position.

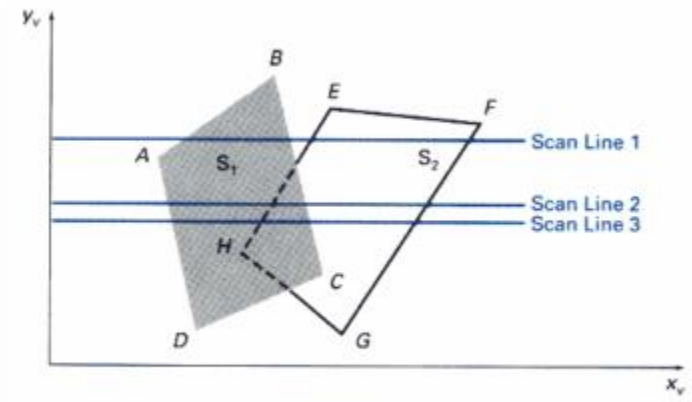


This method requires an additional buffer (if compared with the Depth-Sort Method) and the overheads involved in updating the buffer. So this method

is less attractive in the cases where only a few objects in the scene are to be rendered.
- Simple and does not require additional data structures.
- The z-value of a polygon can be calculated incrementally.
- No pre-sorting of polygons is needed.
- No object-object comparison is required.
- Can be applied to non-polygonal objects.
- Hardware implementations of the algorithm are available in some graphics workstation.
- For large images, the algorithm could be applied to, eg., the 4 quadrants of the image separately,
so as to reduce the requirement of a large additional buffer.

Scan-Line Method

In this method, as each scan line is processed, all polygon surfaces intersecting that line are examined to determine which are visible. Across each scan line, depth calculations are made for each overlapping surface to determine which is nearest to the view plane. When the visible surface has been determined, the intensity value for that position is entered into the image buffer.



Scan lines crossing the projection of two surfaces, S_1 and S_2, in the view plane. Dashed lines indicate the boundaries of hidden surfaces.

This method requires an additional buffer (if compared with the Depth-Sort Method) and the overheads involved in updating the buffer. So this method is less attractive in the cases where only a few objects in the scene are to be rendered.
- Simple and does not require additional data structures.
- The z-value of a polygon can be calculated incrementally.

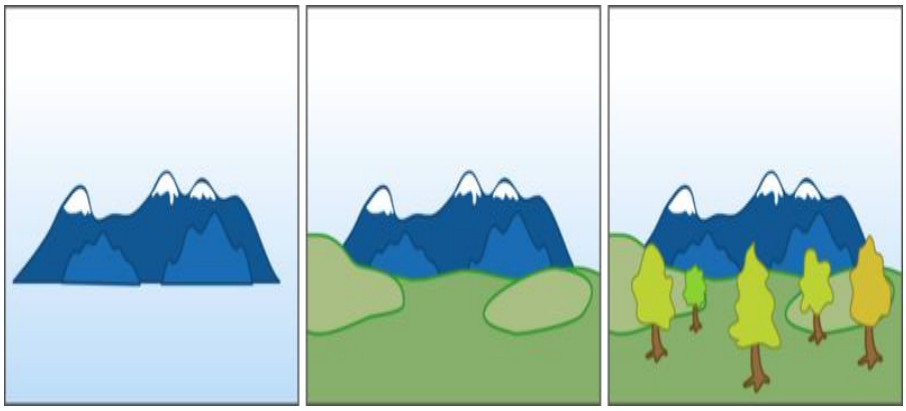
- No pre-sorting of polygons is needed.
- No object-object comparison is required.
- Can be applied to non-polygonal objects.
- Hardware implementations of the algorithm are available in some graphics workstation.
- For large images, the algorithm could be applied to, eg., the 4 quadrants of the image separately, so as to reduce the requirement of a large additional buffer.

Depth-sort

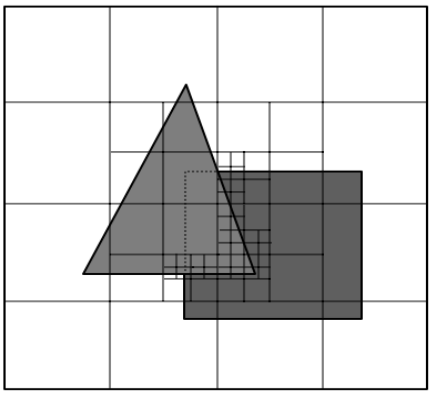
- It is also called "painter's algorithm": how a painter might paint a scene

1. sort all polygons according to smallest z coordinate of each (ie. distance to viewer)
2. resolve overlaps (split polygons if necessary)
3. scan convert polygons from farthest to nearest

- algorithm works for objects that lie in planes of constant z
    - (eg. windowing systems), or those whose z extents do not overlap
    - --> step 2 not required then
- otherwise, step 2 must test whether polygons do not overlap, if any of these tests are true:
    - i) polygons' x extents overlap
    - ii) polygons' y extents overlap
    - iii) their projection extents on x-y plane overlap
    - iv) one polygon is on one side or the other of other's plane and if so, split them up using clipping
- with simple objects and meshes (eg. closed solid volumes), this technique is often quite efficient, because objects can be drawn from furthest to closest without worrying about overlapping polygons.

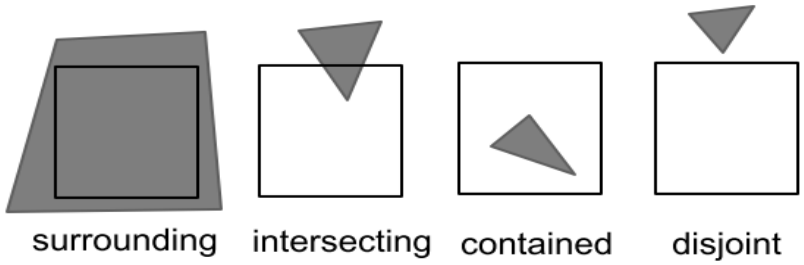


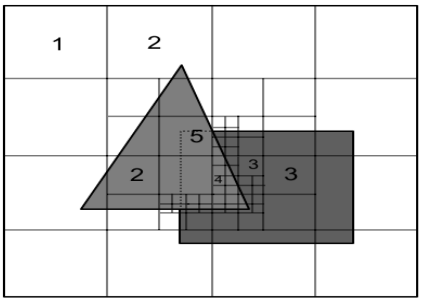
Area subdivision (Warnock's Algorithm)



For each area do the following tests

1. if all polygons disjoint from area? if yes, display background color
2. only one intersecting or contained polygon. if yes, fill with background color, and then draw contained polygon or intersecting portion
3. one single surrounding polygon, no intersecting or contained polygons. if yes, draw area with that polygon's color
4. more than one polygon is intersecting, contained in, or surrounding, but only one polygon is surrounding the area and is in front of others
   if yes draw area with that front polygon's color
5. otherwise, subdivide the area into 4 equal areas and recurse





Computer Animation

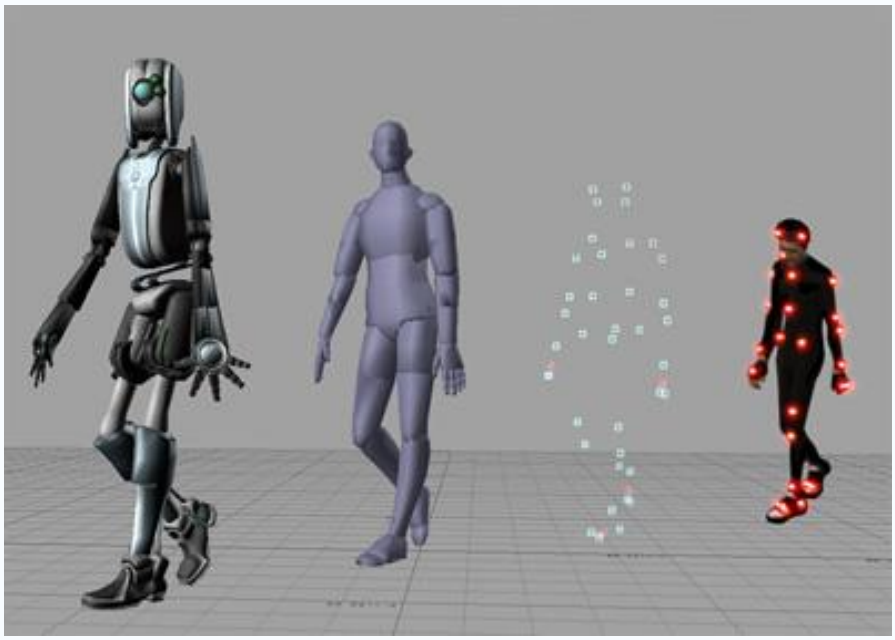
Computer animation refers to any time sequence of visual changes in a scene.

Animation is a sequence of still images, introduces a new dimension, time for generating a sequence of frames to give the impression.

Computer animation (or CGI animation) is the art of creating moving images with 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 and repeatedly replaced by a new image that is similar to the previous image, but advanced slightly in the time domain (usually at a rate of

24 or 30 frames/second). This technique is identical to how the illusion of movement is achieved with television and motion pictures.



Computer animation is essentially a digital successor to the art of stop motion animation of 3D models and frame-by-frame animation of 2D illustrations. For 3D animations, objects (models) are built on the computer monitor (modeled) and 3D figures are rigged with a virtual skeleton. For 2D figure animations, separate objects (illustrations) and separate transparent layers are used, with or without a virtual skeleton. Then the limbs, eyes, mouth, clothes, etc. of the figure are moved by the animator on key frames. The differences in appearance between key frames are automatically calculated by the computer in a process known as twining or morphing. Finally, the animation is rendered.

For 3D animations, all frames must be rendered after modeling is complete. For 2D vector animations, the rendering process is the key frame illustration process, while twined frames are rendered as needed. For pre-recorded presentations, the rendered frames are transferred to a different format or medium such as film or digital video. The frames may also be rendered in real time as they are presented to the end-user audience. Low bandwidth animations transmitted via the internet (e.g. 2D Flash, X3D) often

use software on the end-users computer to render in real time as an alternative to streaming or pre-loaded high bandwidth animations.

Multimedia Computer: Music and Sounds

A *multimedia computer* can include sound data and live synthesized music. Various software programs record, play and edit digitized sounds. Sounds are basically analog signals, but they can be easily digitized and stored on disks. This will enhance the performance of computer animation.

Digitized video files are very lengthy. To store video and play it back at a usable rate, the files are heavily compressed using a video codec (compressor-decompressor).

A multimedia kit is a package of hardware and software that adds multimedia capabilities to a computer. Typically a multimedia kit includes a CD-ROM or DVD player, a sound card, speakers, and CD-ROMs.

INTRODUCTION TO 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 low bandwidth and faster real-time rendering needs. Sometimes the target of the animation is the computer itself, but it 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 how the illusion of movement is achieved with television and motion pictures. Computer animation is essentially a

digital successor to the art of stop motion animation of 3D models and frame-by-frame animation of 2D illustrations.

For 3D animations, objects (models) are built on the computer monitor (modeled) and 3D figures are rigged with a virtual frame or outline.

For 2D figure animations, separate objects (illustrations) and separate transparent layers are used, with or without a virtual frame or outline.

Then the limbs, eyes, mouth, clothes, etc. of the figure are moved by the animator on key frames. The differences in appearance between key frames are automatically calculated by the computer in a process known as tweening or morphing. Finally, the animation is rendered. For 3D animations, all frames must be rendered after modeling is complete.

For 2D vector animations, the rendering process is the key frame illustration process, while tweened frames are rendered as needed.

For pre-recorded presentations, the rendered frames are transferred to a different format or medium such as film or digital video.

The frames may also be rendered in real time as they are presented to the end-user audience.Low bandwidth animations transmitted via the internet (e.g.

2D Flash, X3D) often use software on the end-users computer to render in real time as an alternative to streaming or pre-loaded high bandwidth animations.

Animation has historically been produced in two ways. The first is by artists creating a succession of **cartoon frames**, which are then combined into a film. A second method is by using **physical models**, e.g. King Kong, which are positioned, the image recorded, then the model is moved, the next image is recorded, and this process is continued.

Computer animation can be produced by using a rendering machine to produce successive frames wherein some aspect of the image is varied. For a simple animation this might be just moving the camera or the relative motion of rigid bodies in the scene. This is analogous to the second technique described above, i.e., using physical models. More sophisticated computer animation can move the camera and/or the objects

in more interesting ways, e.g. along computed curved paths, and can even use the laws of Physics to determine the behavior of objects.

Animation is used in Visualization to show the time dependent behavior of complex systems.

A major part of animation is motion control. Early systems did not have the computational power to allow for animation preview and interactive control. Also, many early animators were computer scientists rather than artists. Thus, scripting systems were developed. These systems were used as a computer high level language where the animator wrote a script (program) to control the animation. Whereas a high level programming language allows for the definition of complex data types, the scripting languages allowed for the definition of "actors", objects with their own animation rules.

Later systems have allowed for different types of motion control. One way to classify animation techniques is by the level of abstraction in the motion control techniques. A low-level system requires the animator to precisely specify each detail of motion, whereas a high-level system would allow them to use more general or abstract methods. For example, to move a simple rigid object such as a cube, requires six degrees of freedom (numbers) per frame. A more complex object will have more degrees of freedom, for example a bird might have over twenty degrees of freedom. Now think about animating an entire flock of birds.

Therefore, a Control Hierarchy is required, so that high level control constructs can be specified which are then mapped into more detailed control constructs. This is analogous to high level computer languages with complex control structures or data types which are translated at runtime into low level constructs.

Types of Animation Systems

Scripting Systems

Scripting Systems were the earliest type of motion control systems. The animator writes a script in the animation language. Thus, the user must learn this language and the system is not interactive. One scripting system is ASAS (Actor Script Animation Language). ASAS introduced the concept of an actor, i.e., a complex object which has its own animation rules. For example, in animating a bicycle, the wheels will rotate in their own

coordinate system and the animator doesn't have to worry about this detail. Actors can communicate with other actors be sending messages and so can synchronize their movements. This is similar to the behavior of objects in object-oriented languages.

Procedural Animation

Procedures are used that define movement over time. These might be procedures that use the laws of physics (Physically - based modeling) or animator generated methods. An example is a motion that is the result of some other action (this is called a "secondary action"), for example throwing a ball which hits another object and causes the second object to move.

Representational Animation

This technique allows an object to change its shape during the animation. There are three subcategories to this. The first is the animation of articulated objects, i.e., complex objects composed of connected rigid segments. The second is soft object animation used for deforming and animating the deformation of objects, e.g. skin over a body or facial muscles. The third is morphing which is the changing of one shape into another quite different shape. This can be done in two or three dimensions.

Stochastic Animation

This uses stochastic processes to control groups of objects, such as in particle systems. Examples are fireworks, fire, water falls, etc.

Behavioral Animation

Objects or "actors" are given rules about how they react to their environment. Examples are schools of fish or flocks of birds where each individual behaves according to a set of rules defined by the animator.

Animation Techniques:

Key framing

Key frame systems were developed by classical animators such as Walt Disney. An expert animator would design (choreograph) an animation by drawing certain intermediate frames, called Key frames. Then other animators would draw the in-between frames.

The sequence of steps to produce a full animation would be as follows:

1. Develop a script or story for the animation
2. Lay out a storyboard, that is a sequence of informal drawings that shows the form, structure, and story of the animation.
3. Record a soundtrack
4. Produce a detailed layout of the action.
5. Correlate the layout with the soundtrack.
6. Create the "keyframes" of the animation. The keyframes are those where the entities to be animated are in positions such that intermediate positions can be easily inferred.
7. Fill in the intermediate frames (called "inbetweening" or "tweening").
8. Make a trial "film" called a "pencil test"
9. Transfer the pencil test frames to sheets of acetate film, called "cels". These may have multiple planes, e.g., a static background with an animated foreground.
10. The cels are then assembled into a sequence and filmed

Computer-generated imagery

Computer-generated imagery (CGI) is the application of the field of computer graphics (or more specifically, 3D computer graphics) to special effects.

CGI is used in films, television programs and commercials, and in printed media.

Video games most often use real-time computer graphics (rarely referred to as CGI), but may also include pre-rendered "cut scenes" and intro movies that would be typical CGI applications.

CGI is used for visual effects because the quality is often higher and effects are more controllable than other more physically based processes, such as constructing miniatures for effects shots or hiring extras for crowd scenes, and because it allows the creation of images that would not be feasible using any other technology.

It can also allow a single artist to produce content without the use of actors, expensive set pieces, or props. Recent accessibility of CGI software and

increased computer speeds has allowed individual artists and small companies to produce professional grade films, games, and fine art from their home computers.

Morphing

Morphing is a special effect in motion pictures and animations that changes (or morphs) one image into another through a seamless transition.

Morphing software continues to advance today and many programs can automatically morph images that correspond closely enough with relatively little instruction from the user.

This has led to the use of morphing techniques to create convincing slow-motion effects where none existed in the original film or video footage by morphing between each individual frame.

Morphing is used far more heavily today than ever before.

In years past, effects were obvious.

Now, morphing effects are most often designed to be invisible.

3D computer graphics

3D computer graphics (in contrast to 2D computer graphics) are graphics that utilize a three-dimensional representation of geometric data that is stored in the computer for the purposes of performing calculations and rendering 2D images.

Despite these differences, 3D computer graphics rely on many of the same algorithms as 2D computer vector graphics in the wire frame model and 2D computer raster graphics in the final rendered display.

In computer graphics software, the distinction between 2D and 3D is occasionally blurred; 2D applications may use 3D techniques to achieve effects such as lighting, and primarily 3D may use 2D rendering techniques.

Computer Graphics Application :

Computer-aided design (**CAD**), also known as **computer-aided design and drafting** (**CADD**), is the use of computer technology for the process of

design and design-documentation. Computer Aided Drafting describes the process of drafting with a computer. CADD software, or environments, provide the user with input-tools for the purpose of streamlining design processes; drafting, documentation, and manufacturing processes. CADD output is often in the form of electronic files for print or machining operations. The development of CADD-based software is in direct correlation with the processes it seeks to economize; industry-based software (construction, manufacturing, etc.) typically uses vector-based (linear) environments whereas graphic-based software utilizes raster-based (pixelated) environments.

CADD environments often involve more than just shapes. As in the manual drafting of technical and engineering drawings, the output of CAD must convey information, such as materials, processes, dimensions, and tolerances, according to application-specific conventions.

CAD may be used to design curves and figures in two-dimensional (2D) space; or curves, surfaces, and solids in three-dimensional (3D) objects.

Uses

Computer-aided design is one of the many tools used by engineers and designers and is used in many ways depending on the profession of the user and the type of software in question.

CAD is one part of the whole Digital Product Development (DPD) activity within the Product Lifecycle Management (PLM) process, and as such is used together with other tools, which are either integrated modules or stand-alone products, such as:

- Computer-aided engineering (CAE) and Finite element analysis (FEA)
- Computer-aided manufacturing (CAM) including instructions to Computer Numerical Control (CNC) machines
- Photo realistic rendering
- Document management and revision control using Product Data Management (PDM).

CAD is also used for the accurate creation of photo simulations that are often required in the preparation of Environmental Impact Reports, in which computer-aided designs of intended buildings are superimposed into photographs of existing environments to represent what that locale will be

like were the proposed facilities allowed to be built. Potential blockage of view corridors and shadow studies are also frequently analyzed through the use of CAD.

Computer-aided manufacturing (CAM)

Computer-aided manufacturing (**CAM**) is the use of computer software to control machine tools and related machinery in the manufacturing of workpieces. This is not the only definition for CAM, but it is the most common; CAM may also refer to the use of a computer to assist in all operations of a manufacturing plant, including planning, management, transportation and storage.Its primary purpose is to create a faster production process and components and tooling with more precise dimensions and material consistency, which in some cases, uses only the required amount of raw material (thus minimizing waste), while simultaneously reducing energy consumption.

Image processing:

In electrical engineering and computer science, **image processing** is any form of signal processing for which the input is an image, such as a photograph or video frame; the output of image processing may be either an image or, a set of characteristics or parameters related to the image. Most image-processing techniques involve treating the image as a two-dimensional signal and applying standard signal-processing techniques to it.

Image processing usually refers to digital image processing, but optical and analog image processing also are possible. This article is about general techniques that apply to all of them. The *acquisition* of images (producing the input image in the first place) is referred to as imaging.

Applications

- Computer vision
- Optical sorting
- Augmented Reality
- Face detection
- Feature detection
- Lane departure warning system
- Non-photorealistic rendering
- Medical image processing
- Microscope image processing

- Morphological image processing
- Remote sensing

VLSI

Very-large-scale-integration (VLSI) is defined as a technology that allows the construction and interconnection of large numbers (millions) of transistors on a single integrated circuit.
The number of transistors/gates that can fit in to the semiconductor die dictates the complexity of the functionality that the device can perform. The important factors that fuel the research in VLSI technology can be summarized as below:

- Increased functionality
- Higher reliability
- Small footprint
- Very low power consumption
- Increased speed of operation
- Re-programmability(except ASIC devices)
- Mass production
- Low cost

GKS

The **Graphical Kernel System** (**GKS**) was the first ISO standard for low-level computer graphics. GKS provides a set of drawing features for two-dimensional vector graphics suitable for charting and similar duties. The calls are designed to be portable across different programming languages, graphics devices and hardware, so that applications written to use GKS will be readily portable to many platforms and devices.

Features of GKS :

- Devices Independence
- Text
- Display Management
- Graphics Functions

PHIGS :

Programmer's **H**ierarchical **I**nteractive **G**raphics **S**tandard, A graphics system and language used to create 2D and 3D images. Like the GKS standard, PHIGS is a device-independent interface between the application program and the graphics subsystem.

It manages graphics objects in a hierarchical manner so that a complete assembly can be specified with all of its subassemblies. It is a very comprehensive standard requiring high-performance workstations and host processing.

Some Animation Softwares

There more than 100 Animation sowftware.Some list of animation sowtwares are

WidgetCast 2.0 from Reallusion

WidgetCast 2.0 brings the game, but is it worth the cost?

Animation Software Review: FotoMorph Image Animation Software

FotoMorph is free image morphing and animation software that lets you apply changes and effects to image morphs.

Reallusion's WidgetCast

WidgetCast caters to the world of Web 2.0 and constantly-connected social media by providing a tool that allows you to package your content as micro-applications that can easily be shared anywhere Flash can go.

iClone 4.0 Pro

Reallusion's iClone has grown in leaps and bounds from its original incarnation.