Chapter One- Introduction to Computer Graphics

# Introduction to Computer Graphics

Computer Graphics simplify the process of displaying pictures of any size on a computer screen. Various algorithms and techniques are used to generate graphics in computers. This chapter will help you understand how all these graphics are processed by the computer to give a rich visual experience to the user.

In this chapter we will introduce some basic concepts of computer graphics and review some of the possible applications. We will also examine the technology behind the most common type of graphics display device: the Cathode Ray Tube (CRT) and the Liquid Crystal Display (LCD).

# A survey of computer Graphics

We can define computer graphics as the algorithms and hardware required to produce realistic and/or useful images on a computer.

# Background to Computer Graphics

It is quite obvious that one picture is worth a thousand words. It is believed that a single picture can express than lots of words. The motive of computer graphics is the need for expressing things, events, data and many others. For example, assume a company wants to analyze the progress of its production annually. If this is going to be studied in every year as a production of different items, you may result on thousands of records which is not easy to understand. But using charts, all the burdens can be illustrated simply. Hence the pictorial display (the chart) is very advantageous.

# Definition of Computer Graphics

It is the use of computers to create and manipulate pictures on a display device. It comprises of software techniques to create, store, modify, represents pictures.

# Definition of basic concepts in Computer Graphics

# Pixel, Scan Conversion, Resolution, Aspect Ratio, DPI

**Pixel**

Pixel (also known as picture element) is the fundamental building block of all computer images or color spot that can be displayed and addressed on a monitor. Image displayed on the monitor is made up of thousands of pixels. Each pixel has particular color and brightness value.

**Scan Conversion**

It is a process of representing graphics objects a collection of pixels. The graphics objects are continuous. The pixels used are discrete. Each pixel can have either on or off state. The circuitry of the video display device of the computer is capable of converting binary values (0, 1) into a pixel on and pixel off information. 0 is represented by pixel off. 1 is represented using pixel on. Using this ability graphics computer represent picture having discrete dots.

Any model of graphics can be reproduced with a dense matrix of dots or points. Most human beings think graphics objects as points, lines, circles, ellipses. For generating graphical object, many algorithms have been developed.

# Resolution

The maximum number of points that can be displayed without overlap on a screen is referred to as **resolution. Image resolution** refers to the pixel spacing, i.e., the distance from one pixel to the next pixel. A typical PC monitor displays screen images with a resolution somewhere between 25 **pixels per inch** and 80 **pixels per inch** (ppi). A full screen image with resolution 800 × 600 means that there are 800 columns of pixels, each column comprising 600 pixels, i.e., a total of 800 × 600 = 480000 pixels in the image area.

# Aspect Ratio

The aspect ratio of an image is the ratio of the number of X pixels to the number of Y pixels. The standard aspect ratio for PCs is 4:3 and some resolutions even use a ratio of 5:4. Image displayed using 5:4 aspect ratio will cause the image to appear somewhat distorted. The Table 1.1 shows common resolutions, respective number of pixels and standard aspect ratios.

**Table 1.1** Common resolutions, respective number of pixels and standard aspect ratios

|  |  |  |
| --- | --- | --- |
| **Resolution** | **Number of**  **Pixels** | **Aspect**  **Ratio** |
| 320 × 200 | 64,000 | 8:5 |
| 640 × 480 | 307,200 | 4:3 |
| 800 × 600 | 480,000 | 4:3 |
| 1024 × 768 | 786,432 | 4:3 |
| 1280 × 1024 | 1,310,720 | 5:4 |
| 1600 × 1200 | 1,920,000 | 4:3 |

# Dots Per Inch (DPI)

Dots per inch (DPI) is a measure of spatial printing, video, or image scanner dot density, in particular the number of individual dots that can be placed in a line within the span of 1 inch. DPI is used to describe the resolution number of dots per inch in a digital print and the printing resolution of a hard copy print dot gain, which is the increase in the size of the halftone dots during printing. This is caused by the spreading of ink on the surface of the media.

# Application areas of Computer Graphics

Computers graphics is used today in many diverse areas of science, engineering, medicine, business, industry, government, art, entertainment, advertising, education, and training. Here are short looks at some of the computer graphics applications.

# Computer Aided Design (CAD)

A major use of computer graphics is in design processes, particularly for engineering and architectural systems. Computer-aided design methods are now routinely used in the design of buildings, automobiles, aircraft, watercraft, spacecraft, computers, textiles, and many, many other products.

# Presentation Graphics

It is used to produce illustrations for reports or to generate slide for projections. Examples of presentation graphics are bar charts, line graphs, surface graphs, pie charts and displays showing relationships between parameters. 3-D graphics can provide more attraction to the presentation.

# Computer Art

Computer graphics methods are widely used in both fine art and commercial art applications. The artist uses a combination of 3D modeling packages, texture mapping, drawing programs and CAD software. Pen plotter with specially designed software can create “automatic art”. “Mathematical Art” can be produced using mathematical functions, fractal procedures. These methods are also applied in commercial art. Photorealistic techniques are used to render images of a product.

# Entertainment

CG methods are now commonly used in making motion pictures, music videos and television shows. Many TV series regularly employ computer graphics method. Graphics objects can be combined with a live action.

# Education And Training

Computer-generated models of physical, financial and economic systems are often used as educational aids. For some training applications, special systems are designed. For example training of ship captains, aircraft pilots etc. Some simulators have no video screens, but most simulators provide graphics screen for visual operation. Some of them provide only the control panel.

# Visualization

The numerical and scientific data are converted to a visual form for analysis and to study the behavior called visualization. Producing graphical representation for scientific data sets are called scientific visualization. And business visualization is used to represent the data sets related to commerce and industry. The visualization can be either 2D or 3D.

# Image Processing

Computer graphics is used to create a picture. Image processing applies techniques to modify or interpret existing pictures. To apply image processing methods, the image must be digitized first. Medical applications also make extensive use of image processing techniques for picture enhancements, simulations of operations, etc.

# Graphical User Interface

Nowadays software packages provide graphics user interface (GUI) for the user to work easily. A major component in GUI is a window. Multiple windows can be opened at a time. To activate any one of the window, the user needs just to check on that window. Menus and icons are used for fast selection of processing operations. Icons are used as shortcut to perform functions. The advantages of icons are which takes less screen space. And some other interfaces like text box, buttons, and list are also used.

# Graphics primitives

A graphic primitive refers to basic geometric elements such as points, lines, and polygons that are used in the rendering process to create visual representations of data.

# Overview of image representation - geometric representation

Image representation refers to how visual data is translated into a digital format that computers can interpret. It involves capturing information about the color, shape, texture, and other visual characteristics of an image and encoding them in a structured way. The chosen representation format determines how the image is stored, processed, and displayed.

# Overview of image placement - transformation

An image transformation describes an operation that changes or distorts an image. A linear image transformation, in particular, can be used to do things such as rotate, scale, shear, reflect, or project an image.

# Methods of color specification

The additive color model used for computer graphics is represented by the RGB color cube, where R, G, and B represent the colors produced by red, green, and blue phosphors, respectively. To produce blue, one would mix cyan and magenta inks, as they both reflect blue while each absorbing one of green and red.

# Image presentation devices

The display device is an output device used to represent the information in the form of images (visual form). Display systems are mostly called a video monitor or Video display unit (VDU). Display devices are designed to model, display, view, or display information. The purpose of display technology is to simplify information sharing.

# Image files

Image formats are different types of file types used for saving pictures, graphics, and photos. Choosing the right image format is important because it affects how your images look, load, and perform on websites, social media, or in print. Common formats include JPEG, PNG, GIF, and SVG, each with its own strengths. JPEGs are popular for photos because they balance quality and file size. [PNGs](https://www.geeksforgeeks.org/jpg-or-png-which-image-format-is-better/)are great for images that need clear backgrounds, and SVGs are best for designs that need to stay sharp at any size. Knowing which format to use can help make your images look their best and fit your needs perfectly.

# Color representation methods in computer graphics

The RGB color model is one of the most widely used color representation method in computer graphics. It uses a color coordinate system with three primary colors: R(red),G(Green) and Blue(B).Each primary color can take an intensity value ranging from 0(lowest) to 1(highest). Mixing these three primary colors at different intensity levels produces a variety of colors.

The color spaces in image processing aim to facilitate the specifications of colors in some standard way. Different types of color models are used in multiple fields like in hardware, in multiple applications of creating animation, etc.

# Setting the color attributes of pixels

There are two possible protocols for the specification of pixel coordinates and color:

* setPixel(x,y,rgb):where rgb is a three element array with rgb[0]=r, rgb[1]=g and rgb[2]=b. And if the image uses a lookup table then: setPixel(x,y,i):where I is the address of the entry containing (r,g,b).
* setColor(rgb):for direct coding, and setColor(i) for the lookup table representation. Callse to set pixels now need only to provide coordinate information and would look like: setPixel(x,y)

Lookup table entries can be set from application by: setEntry(i,rgb)

Chapter Two- Overview of Graphics systems

# Overview of Graphics systems

Due to the widespread recognition of the power and utility of computer graphics in virtually all fields, a broad range of graphics hardware and software systems is now available.

# Video display Devices

Any computer-generated image must be displayed in some form. The most common graphics display device is the video monitor, and the most common technology for video monitors is the Cathode Ray Tube (CRT). Now we will examine the technology behind CRT displays, and how they can be used to display images.

The display of a computer is responsible for graphic display. It can display text, picture and video. The primary output device in graphics system is a video monitor. The operation of most video monitors is based on the standard Cathode-Ray Tube (CRT) design.

Some of the common types of display systems available in the market are:

* + - Raster Scan Displays
    - Random Scan Displays
    - Direct View Storage Tube
    - Flat Panel Displays
    - 3D Viewing Devices
    - Stereoscopic and Virtual Reality System

# CRT Displays

CRT display devices are very common in everyday life as well as in computer graphics: most television sets use CRT technology.

*The basic mechanism of CRT is as follows:*

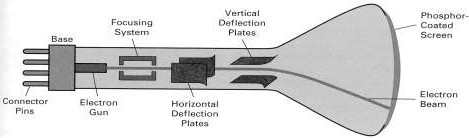
* Beams of electrons are generated by electron guns and fired at a screen consisting of thousands of tiny phosphor dots.
* When a beam hits a phosphor dot it emits light with brightness proportional to the strength of the beam.
* Pictures can be drawn on the display by directing the electron beam to particular parts of the screen. The beam is directed to different parts of the screen by passing it through a magnetic field. The strength and direction of this field, generated by the *deflection yoke*, *[vertical and horizontal deflection plate]* determines the degree of deflection of the beam on its way to the screen.

Displays are often referred to as Video Monitor or Video Display Unit (VDU). The most common monitor that comes with PC is the **Raster Scan type.**

Every display system has three basic parts:

* Display adapter that creates and holds video information,
* Monitor which displays that information, and
* Cable that carries image data between display adapter and monitor Properties of video monitors: pixel, resolution and aspect ratio will be discussed. Cathode-Ray Tube (CRT)

CRT works just like picture tube of television set.



A set of electron guns at the back of CRT produce controlled beam of electrons. Cathode is the primary component of an electron gun. The phosphor material emits light when struck by high- energy electrons.

A beam of electrons (cathode rays) emitted by an electron gun, passes through the focusing and deflection systems that direct the beam toward the phosphor coated screen. The phosphor then emits a small spot of light at each position contacted by the electron beam. Because the light emitted by the phosphor fades rapidly, the picture is redrawn repeatedly and quickly, a method called refresh is used to keep the spot glowing.

**In monochrome CRT** there is only one electron gun, whereas in a **color CRT** there are three electron guns each controlling the display of red, green and blue light, respectively. A color CRT monitor displays color pictures by using a combination of phosphors that emit different colored light.

# Horizontal retrace

The return of the beam direction to the leftmost position one line down is called horizontal retrace during which electron flow is shut off

**Vertical retrace** refers to the movement from bottom to top.

An image in raster scan display is basically composed of a set of dots and lines. Lines are displayed by making these dots bright (with desired color) which lie as close as possible to the shortest path between the end points of the line

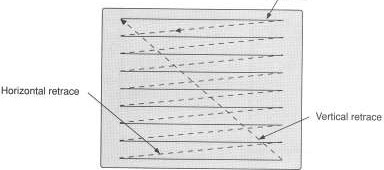
# Refresh Rate

A phosphor material glows for a fraction of second when hit by electron beams and then fades making the image unstable (results in a **flicker** in the image). To make the screen stable, the image must be redrawn a number of times in a second. This process is known as **refreshing** and it is depicted on Figure 1.3

In order to avoid flicker, the image should be redrawn quickly in such a way that our eyes cannot tell what is going on. The number of times per second that the screen is refreshed is known as the

**refresh rate**, and is measured in Hertz (Hz), the unit of frequency. Common refresh rates are: 56, 60, 65, 70, 72, 75, 80, 85, 90, 95, 100, 110 and 120 Hz.

The maximum refresh rate possible depends on the resolution of the image. A higher resolution screen supports less refresh rate than that of lower resolution image due to the more number of pixels to cover with each sweep.



**Figure 1.3** A Schematic diagram of refreshing

# Color of a pixel

The color of a pixel of an image is a result of intersection of three primary colors (red, green and blue) at different intensities. When the intensities of the three electron beams are set to the highest level, then the result is a **white** pixel. When all are set to zero, then the color is **black.** Several millions of colors can be generated by combining the three colors at intermediate intensities. For a mono monitor, a pixel can be black (zero intensity) or white (maximum intensity) or have different shades of gray.

# Color Depth

The number of memory bits required to store color information (intensity values for all three primary components) about a pixel is called **color depth** or **bit depth.** A minimum of one memory bit (color depth = 1) is required to intensity value either 0 or 1 for every screen pixel. The block of memory which stores (or is mapped with) bi-level intensity values for each pixel of a full screen of a full black and white image is called a **bit plane** or **bitmap.**

Color or gray levels can be achieved using additional bit planes. For example n bits per pixel (color depth = n) is a collection of n-bit planes this allows specifying 2n color or gray shades at each pixel.

The more the number of bits per pixel used, the finer the color detail of the image. However, increased color depth requires more memory for storage, and more data for the video card to process, which reduces the allowable refresh rate. Table 1.2 shows common color depths used in PCs

**Table 1.2** Common color depths used in PCs

|  |  |  |  |
| --- | --- | --- | --- |
| **Color**  **Depth** | **Number of**  **Displayed Colors** | **Bytes of Storage**  **per Pixel** | **Common Name**  **for Color Depth** |
| 4-bit | 16 | 0.5 | Standard VGA |
| 8-bit | 256 | 1 | 256-Color Mode |
| 16-bit | 65,536 | 2 | High Color |
| 24-bit | 16,777,216 | 3 | True Color |

For true color three bytes of information are used, one byte for each primary color. A byte can hold 256 different values and three bytes can hold over 16 million (256 × 256 × 256) color possibilities which are more than the human eye can discern. True color is a necessity for high quality photo editing, graphical design, etc

# Frame Buffer

Frame buffer is the video memory (RAM) that is used to hold or map the image displayed on the screen. It is a memory area which contains an internal representation of an image. It can be implemented as part of the main memory or as separate memory.

The amount of memory depends primarily on the *resolution* of the screen image and also the color depth used per pixel, i.e.

*Memory in MB = (X-resolution × Y-resolution × bits-per-pixel) / (8 × 1024 × 1024)*

Practically you need more memory than this formula compute. This is because of:

* video cards are available in the market in certain memory configurations (in terms of whole megabytes), for example, you can’t order 1.7 MB but you have to use a 2 MB card available in the market, and
* many video cards, especially high end accelerators and 3D cards, use memory for computation as well as for the frame buffer.

**Example**: For a screen of resolution 1024 × 768 and color depth 16 find the size of frame buffer.

**Solution**: Memory in MB = (X-resolution × Y-resolution × bits-per-pixel) / (8 × 1024 × 1024)

= (1024 × 768 × 16 )/( 8 × 1024 × 1024) = 1.50 MB. This is not available in market. You should buy 2MB.

# Type of CRTs

There are two types of CRTs basically: *raster scan type* and *random scan type*. The main difference between the two is the technique with which the image is generated on the screen

# Raster Scan Display

* A Raster Scan CRT works much like a Television set.
* Light occurs when an electron beam stimulates a phosphor.
* In Raster Scan display, images are created as grid of pixels.
* Picture definition is stored in a frame buffer memory or refresh buffer.
* In Raster scan, the electron beam from electron gun is swept horizontally across the phosphor.
* In raster scan method, the entire screen is drawn at a time.
* After the bottom line is swept, the beam returns to the top and sweep process begins again.

# Random Scan Display

* In random scan technique, the electron beam is directed to the particular points of the screen where the image is produced.
* Each image is expressed in the form of a series of {move-to, draw-to} commands.
* The beam could be moved to a specific (x, y) location while turned off and then turned on and draw-to command would move the beam (while on) to a second (x, y) location – the result is a line drawn from the first location to the second.
* Here, the CRT has the electron beam directed only to the parts of the screen where a picture is to be drawn.
* Random-scan monitors draw a picture one line at a time, called as vector display.
* Refresh rates on a random-scan system depends on the number of lines to be displayed.
* Picture definition is stored as a set of line-drawing commands in the refresh display file or refresh buffer.
* To display a specified picture, the system cycles through the set of commands in the display file, drawing each component line.
* These systems are designed for the line-drawing applications and can’t display realistic shaded scenes. It generates the image by drawing a set of random straight lines much in the same way one might move a pencil over a piece of paper to draw an image.
* This method is also referred to as **vector drawing** or **stroke writing** or **calligraphic** display.
  1. **DVST (Direct View Storage Tube)**
* Stores picture information as a charge distribution
* Uses two electron guns. Primary gun and flood gun.
* Primary gun is responsible to store the picture pattern. While the flood gun is responsible to maintain the picture display.
* There is no refreshing system in DVST because of the flood gun.
* Complex picture can be displayed at high resolution.
  1. **Colour displaying systems**
     1. **Beam penetration method**
* Simple modification of CRT
* Uses two phosphor coats: green phosphor and red phosphor.
* Colours are displayed depending on the beam strength.
  + 1. **Shadow mask method**

To achieve a colour display, CRT devices have three electron beams, and on the screen the phosphor dots are in groups of three, which give off red, green and blue light respectively. Because the three dots are very close together the light given off by the phosphor dots is combined, and the relative brightness of the red, green and blue components determines the colour of light perceived at that point in the display.

# Random Scan Devices

In a random scan device the CRT beam is only directed to areas of the screen where parts of the picture are to be drawn. If a part of the screen is blank the electron beam will never be directed at it. In this case, we draw a picture as a set of *primitives*, for example lines or curves. For this reason, random scan devices are also known as *vector graphics displays*.

Random scan displays are not so common for CRT devices, although some early video games did use random scan technology. These days random scan is only really used by some hard-copy plotters.

# Raster Scan Devices

In a raster scan device the primitives to be drawn are first converted into a grid of dots .The brightness of these dots is stored in a data structure known as a *frame buffer*. The CRT electron beam then sweeps across the screen line-by-line, visiting every location on the screen, but it is only switched on when the frame buffer indicates that a dot is to be displayed at that location.

Raster scan CRT devices are the most common type of graphics display device in use today, although recently LCD (Liquid Crystal Display) devices have been growing in popularity. In this course, though, we will concentrate on the details of raster scan devices.

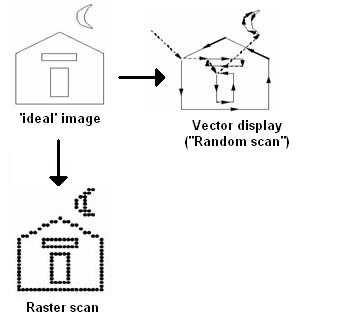


Figure - Random Scan and Raster Scan CRT Displays

* While the frame buffer stores the image data ready for display, the actual display itself is known as the *raster*.
* The raster is a grid of phosphor dots on the display screen.
* Each of these dots is known as a *picture cell*, or *pixel* for short. Each row of pixels in the raster is known as a *scan-line*. The number of pixels in a scan-line is known as the *x-resolution* of the raster, and the number of scan-lines is known as the *y-resolution*. Finally, the ratio of the *y*-resolution to the *x*-resolution (or sometimes the other way around) is known as the *aspect ratio* of the display.

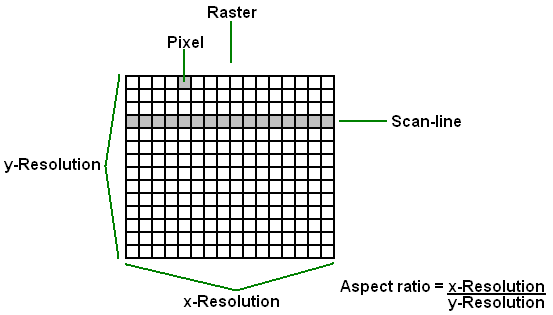


Figure - Raster Scan Terminology

* Remember that for colour CRT displays there are three electron beams: one for red light, one for green light and one for blue light.
* To ensure that these three beams precisely hit the appropriate phosphor dots on the display, after the beams have been deflected by the deflection yoke they pass through a mask containing many tiny holes – one hole for each pixel in the raster. This mask is known as the *shadow mask*. Because the three electron beams originated from slightly different locations (from three separate electron guns), if they all pass through the same hole in the shadow mask they will impact the screen at slightly different locations: the locations of the red, green and blue phosphor dots.

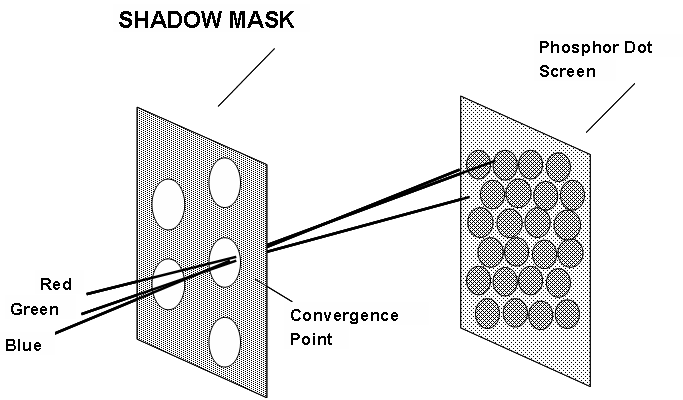


Figure - The Shadow Mask in a Raster Scan CRT Display

* The phosphor dots on a CRT display device only emit light for a very brief period of time after the electron beam has moved on (the length of time that the phosphor emits light is known as its *persistence*).
* Therefore to give the impression of a permanent image on the screen the raster must be continually updated.
* Raster scan systems perform this continual update by ‘sweeping’ the electron beams across the raster scan-line by scan-line, starting from the top and working towards the bottom. When the last scan-line is completed we start again from the top.
* The number of times that the entire raster is *refreshed* (i.e. drawn) each second is known as the *refresh rate* of the device.
* For the display to appear persistent and not to flicker the display must update often enough so that we cannot perceive a gap between frames. In other words, we must refresh the raster when the persistence of the phosphor dots is beginning to wear off. In practise, if the refresh rate is more than 24 frames per second (*f/s*) the display will appear reasonably smooth and persistent.
* Modern graphics displays have high refresh rates, typically in the region of 60 f/s. However, early graphics systems tended to have refresh rates of about 30 f/s.
* Consequently, some flicker was noticeable. To overcome this, a technique known as *interlaced scanning* was employed. Using interlaced scanning alternate scan-lines are updated in each raster refresh. For example, in the first refresh only odd numbered scan-lines may be refreshed, then on the next refresh only even-numbered scan-lines, and so on. Because this technique effectively doubles the screen refresh rate, it has the effect of reducing flicker for displays with low refresh rates. Interlaced scanning was common in the early days of computer graphics, but these days displays have better refresh rates so it is not so common.

The following are the specifications of some common video formats that have been (and still are) used in computer graphics:

* *VGA*: resolution 640x480, 60 f/s refresh rate, non-interlaced scanning.
* *PAL*: resolution 625x480, 25 f/s refresh rate, interlaced scanning
* *NTSC*: resolution 525x480, 30 f/s refresh rate, interlaced scanning

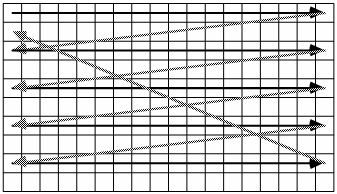


Figure - Interlaced Scanning for Raster Scan Displays

# 

# Frame Buffers

* Frame buffers are used by raster scan display devices to store the pixel values of the image that will be displayed on the raster.
* It is a 2-D array of data values, with each data value corresponding to a pixel in the image.
* The number of bits used to store the value for each pixel is known as the *bit-planes* or *depth* of the frame buffer.
  + For example, a 640x480x8 frame buffer has a resolution of 640x480 and a *depth* of 8 bits; a 1280x1024x24 frame buffer has a resolution of 1280x1024 and a *depth* of 24 bits.
* For colour displays we need to store a value for each component of the colour (red, green and blue), so the bit-planes will typically be a multiple of 3 (e.g. 8 bit-planes each for red, green and blue makes a total of 24 bit-planes).

## Architecture of Raster Graphics Systems

Now we are in a position to examine the basic architecture of raster graphics systems, i.e. what components are required, and how do they interact? shows a block-diagram of a typical raster graphics system.

* Most (non-graphics) processing will occur in the CPU of the computer, which uses the system bus to communicate with the system memory and peripheral devices.
* When graphics routines are to be executed, instead of being executed by the CPU they are passed straight to the *display processor*, which contains dedicated hardware for drawing graphics primitives.
* The display processor writes the image data into the frame buffer, which is a reserved portion of the display processor memory.
* Finally the video controller reads the data in the frame buffer and uses it to control the electron beams in the CRT display.
* The display processor is also known by a variety of other names: *graphics controller*, *display coprocessor*, *graphics accelerator* and *video card* are all terms used to refer to the display processor.
  + Since the display processor contains dedicated hardware for executing graphics routines it must be dedicated to a particular set of routines. In other words, display processors will only be able to handle the graphics processing if the routines used are from a particular graphics package. This is known as *hardware rendering*. Most commercial video cards will support hardware rendering for the OpenGL graphics package, and many PC video cards will also support hardware rendering for DirectX. Hardware rendering is much faster than the alternative, *software rendering*, in which graphics routines are compiled and executed by the CPU just like any other code.
* For the raster graphics architecture to support software rendering the block-diagram shown in the following Figure would need to be modified so that the frame buffer was connected directly to the system bus in order that it could be updated by the CPU.

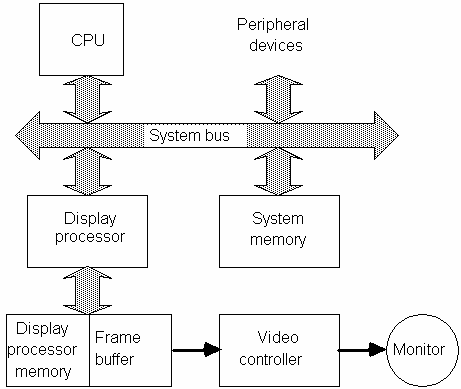


Figure - The Architecture of a Raster Graphics System with a Display Processor

# Interactive Devices

# Graphical Input Devices

In the interactive graphics environment the use of conventional alphanumeric keyboard is tedious, error-prone and sometimes impossible. To overcome this problem, there are a number of interactive input devices. Examples of these interactive input devices are

* + Graphic tablet (digitizer) – used to digitize pictures by decoding signal received to x-y positions.
  + Mouse – used to control movement of the pointer on the screen tasked on relative movement of a wheel.
  + Scanners – used to scan graphic pictures from paper (e.g. OCR application)
  + Light pen – detects presence of light on surface.

# Graphics Software

There are two general classifications for graphics software: general programming packages and special-purpose applications packages. A general graphics programming package provides an extensive set of graphics functions that can be used in a high-level programming language, such as C, FORTRAN, C++, Java etc. An example of a general graphics programming package is the Graphics Library (GL). Basic functions in a general package include those for generating picture components (straight lines, polygons, circles, and other figures), setting color and intensity values, selecting views, and applying transformations. By contrast, application graphics packages

are designed for nonprogrammers, so that users can generate displays without worrying about how graphics operations work. The interface to the graphics routines in such packages allows users to communicate with the programs in their own terms. Examples of applications packages are the artist's painting programs and various business, medical, and CAD systems.

# Graphics Software Standards

The primary goal of standardized graphics software is portability. When packages are designed with standard graphics functions, software can be moved easily from one hardware system to another and used in different implementations and applications. The generally accepted standards for computer graphics are:

1. Graphical Kernel System (GKS) - the first graphics software standard by the International Standards Organization (ISO) and by various national standards organizations, including the American National Standards Institute (ANSI). Although GKS was originally designed as a 2-D graphics package, a 3-D GKS extension was subsequently developed.
2. Programmer's Hierarchical Interactive Graphics Standard (PHIGS) – was the second software standard to be developed and approved by the standards organizations, which is an extension of GKS. Increased capabilities for object modeling, color specifications, surface rendering, and picture manipulations are provided in PHIGS.
3. PHIGS+ - an extension of PHIGS, was developed to provide 3-D surface-shading capabilities not available in PHIGS.
4. Computer Graphics Interface (CGI) & Computer Graphics Metafile (CGM) systems - Although PHIGS presents a specification for basic graphics functions; it does not provide a standard methodology for a graphics interface to output devices. And also does not specify methods for storing and transmitting pictures. Separate standards have been developed for these areas. Standardization for device interface methods is given in the CGI system. And the CGM system specifies standards for archiving and transporting pictures.

# Summary Questions

* 1. **Say true or false**

1. 3-D graphics can provide more attraction to the presentation.
2. Image processing applies techniques to interpret any pictures.
3. The primary output device in graphics system is a video monitor.
4. Aspect ratio of an image is the ratio of the number of Y pixels to the number of X pixels.
5. In Raster scan the electron beam from electron gun is swept vertically across the phosphor.
6. PHIGS is the first graphics software standard recognized by ISO.

# Fill in the blank space

1. The most common monitor that comes with PC is  **.**
2. is the fundamental building block of all computer images.
3. The maximum number of points that can be displayed without overlap on a screen is
4. The number of times per second that the screen is refreshed is
5. The number of memory bits required to store color information about a pixel is
6. is used to hold or map the image displayed on the screen.

# Answer Accordingly

1. For a collection of n bit planes if it is allowed to use 65536 colors to shade a single pixel what is the color depth?
2. Differentiate raster scan and random scan systems
3. Determine the resolution in the x and y directions for the video monitor in use on your system. Determine the aspect ratio, and explain how relative proportions of objects can be maintained on your system.
4. Consider three different raster systems with resolutions of 640 by 400, 1280 by 1024, and 2560 by 2048. What size frame buffer (in bytes) is needed for each of these systems to store 12 bits per pixel? How much storage is required for each system if 24 bits per pixel are to be stored?
5. Suppose an RGB raster system is to be designed using an 8-inch by 10-inch screen with a resolution of 100 pixels per inch in each direction. If we want to store 6 bits per pixel in the frame buffer, how much storage ( in bytes) do we need for the frame buffer?
6. Why PHIGS+ was developed?

# Answer for Summary Questions

* 1. 1. True 2. False 3. True 4. False 5. False 6. False
  2. 1. Raster Scan 2. Pixel 3. Screen resolution 4. Refresh rate 5. Color depth 6. Frame buffer
  3. Self trial

Chapter Three -Scan Conversion

# Scan Conversion

Scan conversion or scan converting rate is a video processing technique for changing the vertical / horizontal scan frequency of video signal for different purposes and applications. The device which performs this conversion is called a scan converter.

It is a process of representing graphics objects a collection of pixels. The graphics objects are continuous. The pixels used are discrete. Each pixel can have either on or off state.

# The concept of Scan-conversion or rasterization

The circuitry of the video display device of the computer is capable of converting binary values (0, 1) into a pixel on and pixel off information. 0 is represented by pixel off. 1 is represented using pixel on. Using this ability graphics computer represent picture having discrete dots.

Any model of graphics can be reproduced with a dense matrix of dots or points. Most human beings think graphics objects as points, lines, circles, ellipses. For generating graphical object, many algorithms have been developed

## Advantage of developing algorithms for scan conversion

1. Algorithms can generate graphics objects at a faster rate.
2. Using algorithms memory can be used efficiently.
3. Algorithms can develop a higher level of graphical objects.

## Examples of objects which can be scan converted

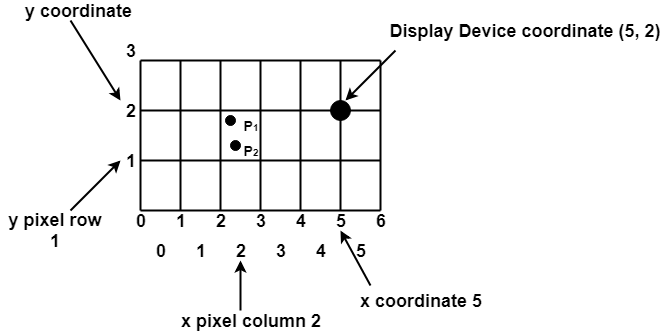
1. Point
2. Line
3. Sector
4. Arc
5. Ellipse
6. Rectangle
7. Polygon
8. Characters
9. Filled Regions

The process of converting is also called as rasterization. The algorithms implementation varies from one computer system to another computer system. Some algorithms are implemented using the software. Some are performed using hardware or firmware. Some are performed using various combinations of hardware, firmware, and software.

# Scan-Converting a Point

Each pixel on the graphics display does not represent a mathematical point. Instead, it means a region which theoretically can contain an infinite number of points. Scan-Converting a point involves illuminating the pixel that contains the point.

**Example:** Display coordinates points Scan Converting a Pointas shown in fig would both be represented by pixel (2, 1). In general, a point p (x, y) is represented by the integer part of x & the integer part of y that is pixels [(INT (x), INT (y).



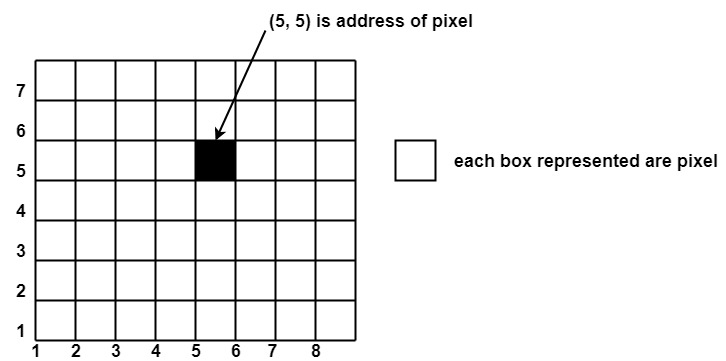
# Define a point

Pixel or Pel:

The term pixel is a short form of the picture element. It is also called a point or dot. It is the smallest picture unit accepted by display devices. A picture is constructed from hundreds of such pixels. Pixels are generated using commands. Lines, circle, arcs, characters; curves are drawn with closely spaced pixels. To display the digit or letter matrix of pixels is used.

The closer the dots or pixels are, the better will be the quality of picture. Closer the dots are, crisper will be the picture. Picture will not appear jagged and unclear if pixels are closely spaced. So the quality of the picture is directly proportional to the density of pixels on the screen.

Pixels are also defined as the smallest addressable unit or element of the screen. Each pixel can be assigned an address as shown in fig:



Different graphics objects can be generated by setting the different intensity of pixels and different colors of pixels. Each pixel has some co-ordinate value. The coordinate is represented using row and column.

P (5, 5) used to represent a pixel in the 5th row and the 5th column. Each pixel has some intensity value which is represented in memory of computer called a frame buffer. Frame Buffer is also called a refresh buffer. This memory is a storage area for storing pixels values using which pictures are displayed. It is also called as digital memory. Inside the buffer, image is stored as a pattern of binary digits either 0 or 1. So there is an array of 0 or 1 used to represent the picture. In black and white monitors, black pixels are represented using 1's and white pixels are represented using 0's. In case of systems having one bit per pixel frame buffer is called a bitmap. In systems with multiple bits per pixel it is called a pixmap.

Point plotting is the process of converting a single coordinate position into operations for the output device in use. With a CRT monitor, for example, the electron beam is turned on to illuminate the screen phosphor at the selected location. How the electron beam is positioned depends on the display technology.

**Example**: drawpixel(x, y, color) in C++

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

Digital devices display a straight line segment by plotting discrete points between the two endpoints. Discrete coordinate positions along the line path are calculated from the equation of the line.

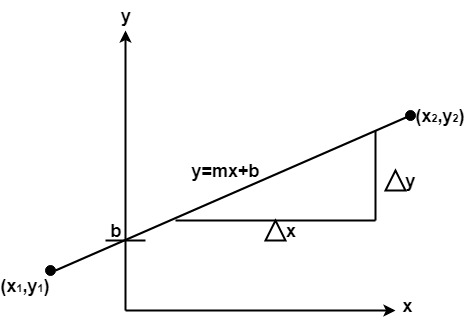
For a raster video display, the line color (intensity) is then loaded into the frame buffer at the corresponding pixel coordinates. Reading from the frame buffer, the video controller then plots the screen pixels. Screen locations are referenced with integer values, so plotted positions may only approximate actual line positions between two specified endpoints.

For example, a computed line position of (10.48, 20.51) would be converted to pixel position (10, 21). Thus rounding of coordinate values to integers causes lines to be displayed with a stair step appearance ("the jaggies").

The characteristic stair step shape of raster lines is particularly noticeable on systems with low resolution, and we can improve their appearance somewhat by displaying them on high resolution systems.

# Scan-Converting a Line

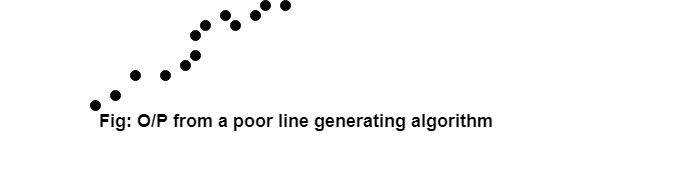
A straight line may be defined by two endpoints & an equation. In fig the two endpoints are described by (x1,y1) and (x2,y2). The equation of the line is used to determine the x, y coordinates of all the points that lie between these two endpoints.



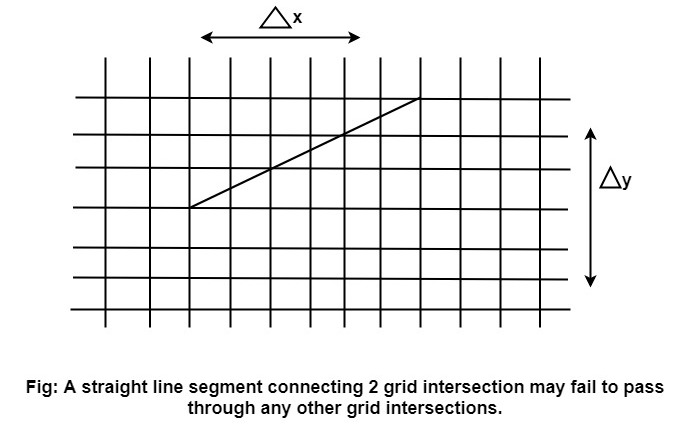
Using the equation of a straight line, y = mx + b where m = Scan Converting a Straight Line & b = the y interrupt, we can find values of y by incrementing x from x =x1, to x = x2. By scan-converting these calculated x, y values, we represent the line as a sequence of pixels.

## Properties of Good Line Drawing Algorithm:

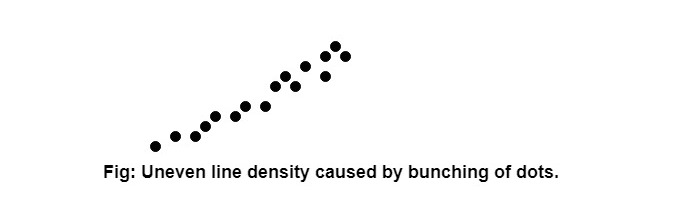
**1. Line should appear Straight:** We must appropriate the line by choosing addressable points close to it. If we choose well, the line will appear straight, if not, we shall produce crossed lines.



The lines must be generated parallel or at 45° to the x and y-axes. Other lines cause a problem: a line segment through it starts and finishes at addressable points, may happen to pass through no another addressable points in between.rward Skip 10s



**2. Lines should terminate accurately:** Unless lines are plotted accurately, they may terminate at the wrong place.



**3. Lines should have constant density:** Line density is proportional to the no. of dots displayed divided by the length of the line.

To maintain constant density, dots should be equally spaced.

**4. Line density should be independent of line length and angle:** This can be done by computing an approximating line-length estimate and to use a line-generation algorithm that keeps line density constant to within the accuracy of this estimate.

**5. Line should be drawn rapidly:** This computation should be performed by special-purpose hardware.

# Methods of Scan-Converting a Line

## Algorithm for line Drawing:

1. Direct use of line equation
2. DDA (Digital Differential Analyzer)
3. Bresenham's Algorithm

# Using the equation of line

## Direct use of line equation:

It is the simplest form of conversion. First of all scan P1 and P2 points. P1 has co-ordinates (x1',y1') and (x2' y2' ).

Then       m = (y2',y1')/( x2',x1') and b = Scan Converting a Straight Line

If value of |m|≤1 for each integer value of x. But do not consider Scan Converting a Straight Line

If value of |m|>1 for each integer value of y. But do not consider Scan Converting a Straight Line

**Example:** A line with starting point as (0, 0) and ending point (6, 18) is given. Calculate value of intermediate points and slope of line.

**Solution:** P1 (0,0) P7 (6,18)

x1=0  
              y1=0  
              x2=6  
              y2=18  
              Scan Converting a Straight Line

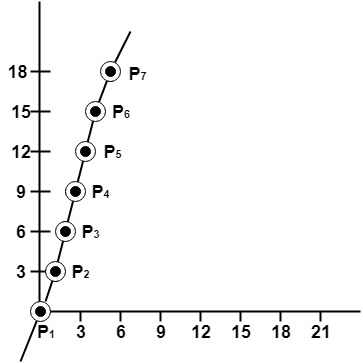
We know equation of line is  
              y =m x + b  
              y = 3x + b..............equation (1)

put value of x from initial point in equation (1), i.e., (0, 0) x =0, y=0  
              0 = 3 x 0 + b  
              0 = b ⟹ b=0

put b = 0 in equation (1)  
              y = 3x + 0  
              y = 3x

Now calculate intermediate points  
    Let x = 1 ⟹ y = 3 x 1 ⟹ y = 3  
    Let x = 2 ⟹ y = 3 x 2 ⟹ y = 6  
    Let x = 3 ⟹ y = 3 x 3 ⟹ y = 9  
    Let x = 4 ⟹ y = 3 x 4 ⟹ y = 12  
    Let x = 5 ⟹ y = 3 x 5 ⟹ y = 15  
    Let x = 6 ⟹ y = 3 x 6 ⟹ y = 18

So points are P1 (0,0)  
              P2 (1,3)  
              P3 (2,6)  
              P4 (3,9)  
              P5 (4,12)  
              P6 (5,15)  
              P7 (6,18)



## Algorithm for drawing line using equation:

**Step1:** Start Algorithm

**Step2:** Declare variables x1,x2,y1,y2,dx,dy,m,b,

**Step3:** Enter values of x1,x2,y1,y2.  
              The (x1,y1) are co-ordinates of a starting point of the line.  
              The (x2,y2) are co-ordinates of a ending point of the line.

**Step4:** Calculate dx = x2- x1

**Step5:** Calculate dy = y2-y1

**Step6:** Calculate m = Scan Converting a Straight Line

**Step7:** Calculate b = y1-m\* x1

**Step8:** Set (x, y) equal to starting point, i.e., lowest point and xendequal to largest value of x.

              If dx < 0  
                  then x = x2  
              y = y2  
                        xend= x1  
        If dx > 0  
              then x = x1  
                  y = y1  
                        xend= x2

**Step9:** Check whether the complete line has been drawn if x=xend, stop

**Step10:** Plot a point at current (x, y) coordinates

**Step11:** Increment value of x, i.e., x = x+1

**Step12:** Compute next value of y from equation y = mx + b

**Step13:** Go to Step9.

# The Digital Differential Algorithm (DDA)

## The DDA Line-Drawing Algorithm

* DDA stands for Digital Differential Analyzer
* Calculates for the vertical and horizontal change of the vector to be generated as dx and dy.
* The algorithm decides which pixel is the next pixel to be highlighted based on the value of the horizontal and vertical changes.

***Algorithm:***

1. Read the end points of the line as (x1,y1) and (x2,y2)

2. calculate the vertical change and the horizontal change as:

dx = x2 - x1

dy = y2 - y1

1. If abs(dx) > abs(dy) then  
    step = abs(dx)  
   otherwise  
    step = abs(dy)
2. xinc = dx/step  
   yinc = dy/step  
   x = x1  
   y = y1
3. putpixel(x,y,color)
4. x = x + xinc  
   y = y + yinc  
   putpixel(x,y,color)
5. Repeat step 6 until x = x2.

The Digital Differential Analyser (DDA) algorithm operates by starting at one end-point of the line, and then using steps (2) - (5) to generate successive pixels until the second end-point is reached. Therefore, first, we need to assign values for dx and dy.

Before we consider the actual DDA algorithm, let us consider a simple first approach to this problem. Suppose we simply increment the value of x at each iteration (i.e. dx = 1), and then compute the corresponding value for y using the steps in the algorithm. This would compute correct line points but, as illustrated by the Figure, it would leave gaps in the line. The reason for this is that the value of dy is greater than one, so the gap between subsequent points in the line is greater than 1 pixel.

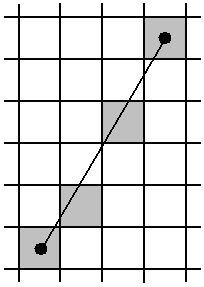


Figure – ‘Holes’ in a Line Drawn by Incrementing x and Computing the Corresponding y-Coordinate

The solution to this problem is to make sure that both dx and dy have values less than or equal to one. To ensure this, we must first check the size of the line gradient. The conditions are:

If |m| ≤ 1:

dx = 1

dy = m

If |m| > 1:

dx = 1/m

dy = 1

Once we have computed values for dx and dy, the basic DDA algorithm is:

Start with (x0,y0)

Find successive pixel positions by adding on (dx, dy) and rounding to the nearest integer, i.e.

xk+1 = xk + dx

yk+1 = yk + dy

For each position (xk,yk) computed, plot a line point at (round(xk),round(yk)), where the round function will round to the nearest integer.

Note that the actual pixel value used will be calculated by rounding to the nearest integer, but we keep the real-valued location for calculating the next pixel position.

Let us consider an example of applying the DDA algorithm for drawing a straight-line segment. We first compute a value for the gradient(slop) m:



Now, because |m| ≤ 1, we compute dx and dy as follows:

dx = 1

dy = 0.6

Using these values of dx and dy we can now start to plot line points:

Start with (x0,y0) = (10,10) – colour this pixel

Next, (x1,y1) = (10+1,10+0.6) = (11,10.6) – so we colour pixel (11,11)

Next, (x2,y2) = (11+1,10.6+0.6) = (12,11.2) – so we colour pixel (12,11)

Next, (x3,y3) = (12+1,11.2+0.6) = (13,11.8) – so we colour pixel (13,12)

Next, (x4,y4) = (13+1,11.8+0.6) = (14,12.4) – so we colour pixel (14,12)

Next, (x5,y5) = (14+1,12.4+0.6) = (15,13) – so we colour pixel (15,13)

We have now reached the end-point (xend,yend), so the algorithm terminates

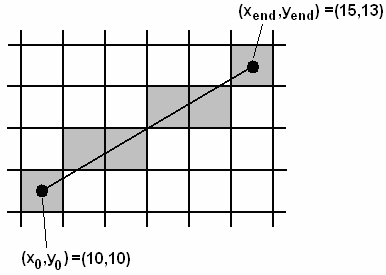


Figure - The Operation of the DDA Line-Drawing Algorithm

* The DDA algorithm is simple and easy to implement but it does involve floating point operations to calculate each new point.
* Floating point operations are time-consuming when compared to integer operations. Since line-drawing is a very common operation in computer graphics, it would be nice if we could devise a faster algorithm which uses integer operations only. The next section describes such an algorithm.

# Bresenham's Line Algorithm

## Bresenham’s Line-Drawing Algorithm

* Bresenham’s line-drawing algorithm provides significant improvements in efficiency over the DDA algorithm.
* These improvements arise from the observation that for any given line, if we know the previous pixel location, we only have a choice of 2 locations for the next pixel. This concept is illustrated in the following Figure .
  + Given that we know (xk,yk) is a point on the line, we know the next line point must be either pixel A or pixel B. Therefore we do not need to compute the actual floating-point location of the ‘true’ line point; we need only make a decision between pixels A and B.

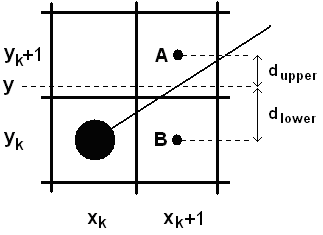


Figure - Bresenham's Line-Drawing Algorithm

Bresenham’s algorithm works as follows. First, we denote by dupper and dlower the distances between the centres of pixels A and B and the ‘true’ line. Using Eqn.

 ……………………………………………………………………… (1)

the ‘true’ y-coordinate at xk+1 can be calculated as:

 …………………………………………………………… (2)

Therefore we compute dlower and dupper as:

 …………………………………………… (3)

 …………………………………… (4)

Now, we can decide which of pixels A and B to choose based on comparing the values of dupper and dlower:

If dlower > dupper, choose pixel A

Otherwise choose pixel B

We make this decision by first subtracting dupper from dlower:

 …………………………………… (5)

* If the value of this expression is positive we choose pixel A; otherwise we choose pixel B. The question now is how we can compute this value efficiently.
* To do this, we define a decision variable pk for the kth step in the algorithm and try to formulate pk so that it can be computed using only integer operations. To achieve this, we substitute  (where Dx and Dy are the horizontal and vertical separations of the two line end-points) and define pk as:

 …………………………… (6)

where d is a constant that has the value . Note that the sign of pk will be the same as the sign of (dlower – dupper), so if pk is positive we choose pixel A and if it is negative we choose pixel B. In addition, pk can be computed using only integer calculations, making the algorithm very fast compared to the DDA algorithm.

An efficient incremental calculation makes Bresenham’s algorithm even faster. (An incremental calculation means we calculate the next value of pk from the last one.) Given that we know a value for pk, we can calculate pk+1 from Eq. (6) by observing that:

Always xk+1 = xk+1

If pk < 0, then yk+1 = yk, otherwise yk+1 = yk+1

Therefore we can define the incremental calculation as:

, if pk < 0 …………………………………………………… (7)

, if pk ≥ 0 …………………………………………… (8)

The initial value for the decision variable, p0, is calculated by substituting xk = x0 and yk = y0 into Eq. (10), which gives the following simple expression:

 …………………………………………………………… (9)

So we can see that we never need to compute the value of the constant d in Eq. (6).

The following is very simple illustration of this method in algorithm.

**The Bresenham’s Algorithm**

**Case1:** for |m| < 1

Step 1. Read (x1,y1) and (x2,y2) as the end points of the line

Step 2. dx = | x2 - x1|

dy = | y2 - y1|

p = 2dy – dx

step 3. If p < 0 then

If(x1<x2)

x1 = x1 + 1

else

x1 = x1 – 1

p = p + 2dy

otherwise

If(y1<y2)

y1 = y1 + 1

else

y1 = y1 - 1

If(x1<x2)

x1 = x1 + 1

else

x1 = x1 – 1

p = p + 2(dy – dx)

step 4. putpixel(x1,y1,color)

step 5. Repeat step 3 and 4 until x1 = x2.

**Case 2:** For slop | m | >= 1.

Step 1. Read (x1,y1) and (x2,y2) as the end points of the line

Step 2. dx = | x2 - x1|

dy = | y2 - y1|

p = 2dx – dy

step 3. If p < 0 then

If(y1<y2)

y1 = y1 + 1

else

y1 = y1 – 1

p = p + 2dx

otherwise

If(y1<y2)

y1 = y1 + 1

else

y1 = y1 - 1

If(x1<x2)

x1 = x1 + 1

else

x1 = x1 – 1

p = p + 2(dx – dy)

step 4. putpixel(x1,y1,color)

step 5. Repeat step 3 and 4 until y1 = y2.

Summary

To summarise, we can express Bresenham’s algorithm as follows:

Plot the start-point of the line (x0,y0)

Compute the first decision variable:



For each k, starting with k=0:

If pk < 0:

Plot (xk+1,yk)



Otherwise:

Plot (xk+1,yk+1)



The steps given above will work for lines with positive |m| < 1. For |m| > 1 we simply swap the roles of x and y. For negative slopes one coordinate decreases at each iteration whilst the other increases.

Exercise

Consider the example of plotting the line shown in Figure using Bresenham’s algorithm:

First, compute the following values:

Dx = 5

Dy = 3

2Dy = 6

2Dy - 2Dx = -4



Plot (x0,y0) = (10,10)

Iteration 0:

p0 ≥ 0, so

Plot (x1,y1) = (x0+1,y0+1) = (11,11)



Iteration 1:

p1 < 0, so

Plot (x2,y2) = (x1+1,y1) = (12,11)



Iteration 2:

p2 ≥ 0, so

Plot (x3,y3) = (x2+1,y2+1) = (13,12)



Iteration 3:

p3 < 0, so

Plot (x4,y4) = (x3+1,y3) = (14,12)



Iteration 4:

p4 ≥ 0, so

Plot (x5,y5) = (x4+1,y4+1) = (15,13)

We have reached the end-point, so the algorithm terminates

We can see that the algorithm plots exactly the same points as the DDA algorithm but it computes them using only integer operations. For this reason, Bresenham’s algorithm is the most popular choice for line-drawing in computer graphics.