GCA (S5) 03
Exam. Code: CG

Computer Graphics

SEMESTER V

BACHELOR OF COMPUTER APPLICATION

Block 1



KRISHNA KANTA HANDIQUI STATE OPEN UNIVERSITY

Subject Experts

Prof. Anjana Kakati Mahanta, Gauhati University

Prof. (Retd.) Pranhari Talukdar, Gauhati University

Dr. Jyotiprokash Goswami, Associate Professor, Assam Engineering College

Course Co-ordinators

Dr. Tapashi Kashyap Das, Assistant Professor, KKHSOU

Ms. Sruti Sruba Bharali, Assistant Professor, KKHSOU

SLM Preparation Team

UNITS	CONTRIBUTORS
1, 2, 3, 4, 5 & 6	Ms. Jonalee Barman Kakaty, North Eastern Regional Institute of Management (NERIM)
7	Ms. Daisy Kalita, University of Science and Technology, Meghalaya

Editorial Team

Content : Dr. Sanjib Kumar Kalita, Gauhati University

Structure, Format & Graphics: Dr. Tapashi Kashyap Das, KKHSOU

June, 2019 ISBN: 978-93-89123-30-2

This Self Learning Material (SLM) of the Krishna Kanta Handiqui State University is made available under a Creative Commons Attribution-Non Commercial-ShareAlike4.0 License (International): http.//creativecommons.org/licenses/by-nc-sa/4.0.

Printed and published by Registrar on behalf of the Krishna Kanta Handiqui State Open University.

Headquarters : Patgaon, Rani Gate, Guwahati-781 017

City Office : Housefed Complex, Dispur, Guwahati-781 006; Web: www.kkhsou.in

The university acknowledges with thanks the financial support provided by the **Distance Education Bureau, UGC** for the preparation of this study material.

BACHELOR OF COMPUTER APPLICATION

Computer Graphics

Block-1

CONTENTS

		Pages
UNIT 1:	**Introduction to Computer Graphics** Definition, Application Areas of Computer Graphics (CAD, CAM, Education and Training, Entertainment, Image Processing, Computer Art etc.), Graphical User Interfaces (GUI)	**7-12**
UNIT 2:	**Graphics Systems** Cathode Ray Tubes, Random Scan Displays, Raster Scan Displays, Color CRT Monitors, Flat-Panel Displays (Plasma-Panels, Liquid Crystal Displays(LCD), Electroluminescent displays), Graphics Software (GKS, PHIGS)	**13-30**
UNIT 3:	**Line Drawing Algorithms** Points and Lines, Line Drawing Algorithms (DDA Algorithm, Bresenham's Line Algorithm)	**31-42**
UNIT 4:	**Circle and Ellipse Drawing Algorithms** Circle Drawing Algorithms, Ellipse Drawing Algorithms	**43-54**
UNIT 5:	**Filled Area Algorithms** Filling (Scan-Line Polygon Filling, Inside Outside Tests, Boundary-fill and Flood-fill Algorithm)	**55-64**
UNIT 6:	**Transformations** Basic 2-D Transformations (Rotation, Reflection, Shearing, Scaling), Homogeneous Coordinate Representation, Translation, 3-D Transformations	**65-79**
UNIT 7:	**2-D Viewing** 2-D Viewing: The Viewing Pipeline, Viewing Coordinate Reference Frame, Window to View-port Coordinate Transformation, Viewing FunctionsP (Cascading Style Sheet Positioning) and JSSS (JavaScript Assisted Style Sheet), Layers of Netscape, The ID Attribute, DHTML Events	**80-90**

COURSE INTRODUCTION

This is a course on ***Computer Graphics***. Graphics provides one of the most natural means of communicating with a computer. Computer graphics is the use of a computer to define, store, manipulate, and present pictorial output. It comprises the creation and representation of simple graphical elements and images, as well as modern techniques for rendering a virtual reality. To apply these techniques correctly, one requires a basic understanding of the fundamental concepts of computer graphics. Computer graphics is used in many different areas of industry, business, government, education, entertainment, and most recently, the home.

This course covers the various basic graphics concepts and techniques for operating with two- and three-dimensional objects and overview a few typical applications of computer graphics. The course is divided into two blocks.

BLOCK INTRODUCTION

This is the first block of the course ***Computer Graphics***. This block describes the basic principles, techniques, algorithms for generating and interacting with simple graphical objects on a display screen. Two-dimentional transformations and two-dimentional viewing are described in this block.

This block comprises the following seven units:

Unit 1 is an introductory unit which provides the definition and application areas of Computer Graphics. It also gives the concept of Graphical User Interface.

Unit 2 describes various graphics systems such as Cathod Ray Tube, Raster Scan and Random Scan Display system and Graphics Softwares.

Unit 3 presents important line drawing algorithms. It mainly focuses on DDA and Bresenham's line drawing algorithm.

Unit 4 presents Circle and Ellipse drawing algorithms.

Unit 5 discusses Filling algorithms. Boundary-fill, Flood-fill algorithms are described in this unit.

Unit 6 focuses on two-dimentional transformations. Homogeneous coordinates, composition of 2D transformations are described in this unit.

Unit 7 is on two-dimentional viewing. The viewing pipeline, window-to-viewport coordinate transformation in matrix form, viwing functionsa re described in this unit.

Each unit of this block includes some along-side boxes to help you know some of the difficult, unseen terms. Some "EXERCISES" have been included to help you apply your own thoughts. You may find some boxes marked with: "LET US KNOW". These boxes will provide you with some additional interesting and relevant information. Again, you will get "CHECK YOUR PROGRESS" questions. These have been designed for you to self-check your progress of study. It will be helpful for you if you solve the problems put in these boxes immediately after you go through the sections of the units and then match your answers with "ANSWERS TO CHECK YOUR PROGRESS" given at the end of each unit.

UNIT 1: INTRODUCTION TO COMPUTER GRAPHICS

UNIT STRUCTURE

1.1 Learning Objectives

1.2 Introduction

1.3 Definition of Computer Graphics

1.4 Application Areas of Computer Graphics

1.5 Graphical User Interface

1.6 Let Us Sum Up

1.7 Further Reading

1.8 Answers to Check Your Progress

1.9 Model Questions

1.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- describe where Computer Graphics comes into use
- learn about Graphical User Interface (GUI).

1.2 INTRODUCTION

Computer Graphics is a vast field that is rapidly evolving. It encompasses almost any graphical aspect in a computer. It has become the integral part of all computer user interfaces. It is considered as a branch of Computer Science that deals with the production of images.

In this unit, we will introduce the basics of Computer Graphics.

1.3 DEFINITION OF COMPUTER GRAPHICS

There is a saying that "*A Picture is worth a thousand words*". **Computer Graphics** deals with the creation of pictures or images in a computer. The pictures can be anything like photographs, drawings, moving pictures or internal organ pictures of human body which cannot be seen directly. It involves graphical or pictorial representation and manipulation of

data. Everything in a computer is graphics along with the text displayed except the sound. A computer image is usually composed of picture elements or pixels, which are arranged in a grid-like structure.

Computer graphics is an important area of Computer Science and Engineering. It plays a vital role in almost every application of computer.

The computer graphics system consists of a computer with a fast processor, large memory, output devices and the input devices. Figure 1.1 shows some graphics display on a computer.



Figure 1.1: Example of Computer Graphics

Graphics can be categorized as *Raster or Bitmap Graphics* and *Vector Graphics*. The raster images are made up of pixels in a grid. Vector images are made up of one or many scalable objects which are generally defined by mathematical equations rather than pixels. Computer graphics system can be active or passive. In both the input to the system is the picture description and the output is the static or animated picture displayed. In active systems, the user can control the picture display with the help of a GUI, using an input device.

1.4 APPLICATION AREAS OF COMPUTER GRAPHICS

Computer graphics is a rapidly evolving field and is now used in almost every field. The major areas of computer graphics are:

- **Modeling:** Modeling deals with the mathematical specification of shape and appearance properties of objects in a way that can be stored in a computer.

- **Rendering:** Rendering deals with the creation of shaded images from 3D computer models.
- **Animation:** Animation is a technique to create illusion of motion through sequences of images. It uses both modeling and rendering.

There are many other application areas of computer graphics which are as follows:

1) The GUI (Graphical User Interface): The typical components used in the GUI are the *menus*, *icons*, *cursors*, *dialog boxes*, *scroll bars*, *buttons* etc, which are all graphics.
2) In Business where one has to produce the charts of the business and economical data, etc.
3) In the computer softwares like the Office Automation (MS Word, Excel, Power Point presentations), computer art like Paint Brush etc.
4) Plotting 2D and 3D graphs in Science and Technology related subjects which otherwise would have been impossible without computer graphics.
5) In Engineering fields like construction designs, circuit designs, etc.
6) In entertainment like the movies, television, cartoons, etc.
7) Publishing advertisements in the Web and Television.
8) Cartography which includes design, creation and the drawing of maps.
9) CAD (Computer Aided Designs) and CAM (Computer Aided Manufacturing) uses computer technology to design parts and products on the computer and then use those designs to help in the manufacturing process.
10) Digital Image Processing that deals with analyzing, improving and presenting the existing images. It gives machine perception to the visual information. Example, Robotics applications.
11) Education and Training.
12) Simulations like video gaming.
13) Medical Imaging.

1.5 GRAPHICAL USER INTERFACE

The ***Graphical User Interface*** (GUI) exists in between the user of the computer and the graphics system. It helps the user in interacting with

the system. In terms of computer science, the GUI is a visual operating display on the monitor to the user. Mac OS, Microsoft Windows and the X Window System are some examples of GUI.

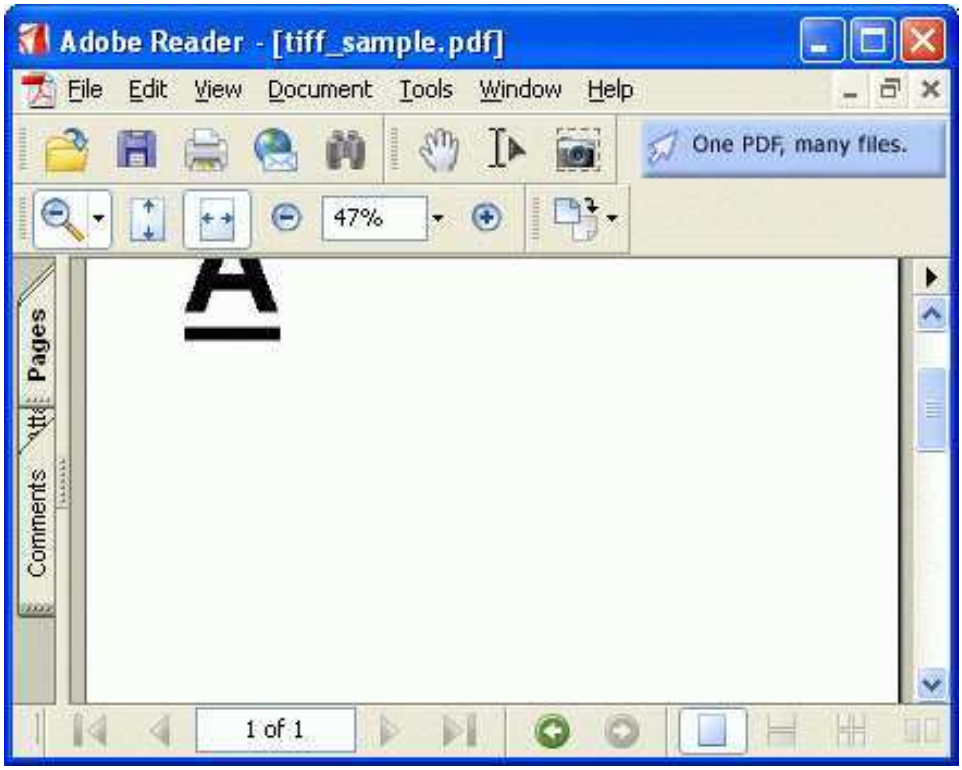


Fig. 1.2: An example of Ghaphical User Interface

The GUI has windows, menus, icons, buttons etc. that helps the user interact with the system rather than memorizing and using the complicated commands. The user issues commands to the computer application via the GUI. The GUI usually has three major components:

a) Windowing system, which creates the windows, menus, dialog boxes tha appears on the screen.

b) Imaging model, which defines the fonts and graphics that appears on the screen.

c) Application Program Interface (API), in which the user specifies how and what windows and graphics appears on the screen.

The first GUI was designed by Xerox Corporation's Palo Alto Research Center in the 1970s, but it was not until the 1980s and the emergence of the Apple Macintosh that graphical user interfaces became popular.

In addition to the visual components, the GUI also makes it easier to move data from one application to another. Figure 1.2 shows an example of a Graphical User Interface.



CHECK YOUR PROGRESS

Q.1: Fill in the blanks:

a) Computer Graphics is the ..... of pictures or images in a computer.

b) Graphics plays a vital role in almost every ..... of computer.

c) Rendering deals with the creation of ..... from 3D computer models

d) The technique to create illusion of motion through a sequence of images is called .....

e) ..... that deals with analyzing, improving and presenting the existing images.

f) Video gaming and cartoons are all ..... with computer graphics.

g) Computer graphics is used to create images of the human body for clinical purposes or medical science and this technique is called .....

h) The Graphical User Interface is the ..... between the computer user and the graphics system

i) The three major components of GUI are ....., ... and .....

j) The first GUI was designed by ..... in 1970s.



1.6 LET US SUM UP

- A computer image is usually composed of picture elements or pixels, which are arranged in a grid-like structure.
- The GUI is the interface between the user and the computer system.

- The GUI usually has three major components : Windowing system, Imaging model and Application Program Interface (API).
- The GUI has windows, menus, icons, buttons etc. that helps the user interact with the system rather than having to memorize the complicated commands and type them.



1.7 FURTHER READING

1) Hearn, Donald and Baker, M. Pauline. *Computer Graphics*. Prentice-Hall.
2) Shirley, Peter; Marschner, Steve. (3rd Edition). *Fundamentals of Computer Graphics*.



1.8 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q. No. 1: a) Creation; b) application; c) shaded images; d) animation; e) Digital Image Processing; f) Created; g) Medical imaging; h) interface; i) Windowing system, imaging model, application program interface; j) Xerox Corporation's Palo Alto Research.



1.9 MODEL QUESTIONS

Q.1: Write the basic difference between Raster and Vector Graphics images.

Q.2: Write the full form of CAD and CAM.

Q.3: Write the name of the three major components of GUI.

Q.4: Write short notes on Graphical User Interface.

Q.5: What are the major areas of Computer Graphics?

Q.6: Briefly describe the application areas of Computer Graphics.

*** ***** ***

UNIT 2: GRAPHICS SYSTEMS

UNIT STRUCTURE

2.1 Learning Objectives

2.2 Introduction

2.3 Cathode Ray Tubes

- 2.3.1 Random Scan Displays
- 2.3.2 Raster Scan Displays

2.4 Color CRT Monitors

2.5 Flat Panel Displays

- 2.5.1 Plasma Panels
- 2.5.2 Liquid Crystal Displays
- 2.5.3 Electroluminescent Displays

2.6 Graphics Software

2.7 Color Models

2.8 Let Us Sum Up

2.9 Further Reading

2.10 Answers to Check Your Progress

2.11 Model Questions

2.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- learn how the CRT monitors works
- learn the different types of Flat-panel displays used in computers.
- learn about different color models.

2.2 INTRODUCTION

In the previous unit we have learnt some basic terminologies associated with Computer Graphics. We have also learnt the applications of graphics and about GUI.

In this unit, we will discuss about the display techniques used in the computer to display images. We will also introduce you how colored pictures

are generated in computer. Graphics devices can be devided into two major areas: (i) the display systems where the graphics are presented and (ii) the hardware components with which the user writes necessary instruction to generate the graphics.

2.3 CATHODE RAY TUBES

The CRT is a specialized vacuum tube that produces images when an electron beam strikes a phosphorescent screen. Most video monitors operates with this CRT technology.

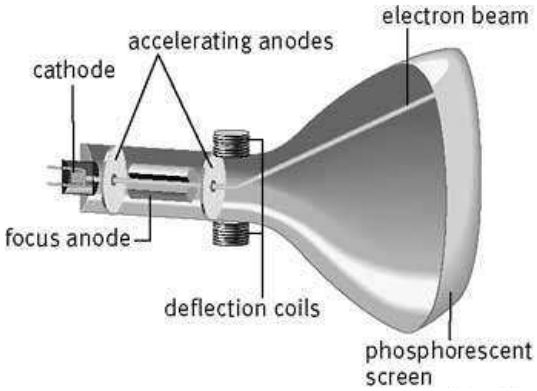


Figure 2.1: Basic design of a CRT

The operation of the Cathode Ray Tube is as follows:

An electron beam emitted by an electron gun, passes through a focusing and deflection systems that direct the beam towards specified positions on the phosphor coated screen. The phosphor then emits light at the positions where the beam has strike. A property of phosphors is that the light emitted fades very rapidly. This property is called ***persistence***.

Persistence is defined as the time it takes the emitted light from the screen to decay to one-tenth of its original intensity.

So to maintain the picture on the screen without flicker, the picture is redrawn repeatedly by quickly directing the electron beam back over the same points. This is called ***Refresh CRT***. The lower persistence phosphors require higher refresh rates to keep the screen picture.

A cathode ray tube consists of several basic components. The electron gun generates an electron beam. The electron gun in a CRT is a heated cathode and a control grid that fits over the cathode. The anodes

accelerate the electrons. The deflecting coils produce a very low frequency electromagnetic field that allows for constant adjustment of the direction of the electron beam. There are two sets of deflecting coils: horizontal and vertical. The figure 2.1 shows only one set of deflection coils.

The electron beam when strikes the phosphor coated screen produces a bright spot on the screen. The intensity of the beam can be controlled by setting voltage levels on the control grid. A smaller negative voltage decreases the number of electrons passing through the small hole at the end of the control grid. A higher negative voltage repels the electrons and stops them from passing through.

The maximum number of points that can be displayed on the screen without overlap on the CRT is called the ***Resolution*** and is usually given as the number of horizontal points versus the number of vertical points. These points are called ***pixels***. For example, a monitor can have a resolution of 1024 × 768 pixels. The number of pixels determines the resolution of an image.

Aspect ratio is the ratio of the vertical points to the horizontal points to produce equal length lines in both directions on the screen.

2.3.1 Random Scan Displays

In Random Scan display, the electron beam is directed only to those parts of the screen where the picture has to be drawn. It draws the picture one line at a time. So it is also called vector, stroke writing or calligraphic displays.

The figure below shows how the random scan display works. The component lines of the picture can be drawn in any specified order.

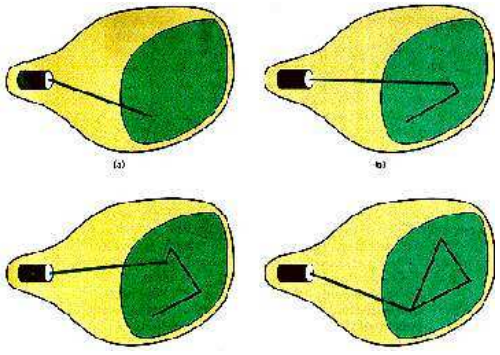


Figure 2.2: Lines are drawn in any order in a random scan system

The picture definition, in random scan displays is stored as a set of line drawing commands in a memory area called the refresh display file. The refresh rate depends on the number of lines in the picture and generally the lines of the pictures are drawn 30 to 60 times per second.

The advantages of Random Scan displays are:

- Very high resolution as the picture definition is stored as a set of line commands and not as a set of intensity values.
- Produces smooth line drawings as the electron beam directly follows the line path
- Requires little memory.

The disadvantages are:

- Cannot draw complex pictures or realistic scenes.
- Cost is high.

A pen plotter and a laser light show are also random displays.

A simple architecture of the random display systems is shown below:

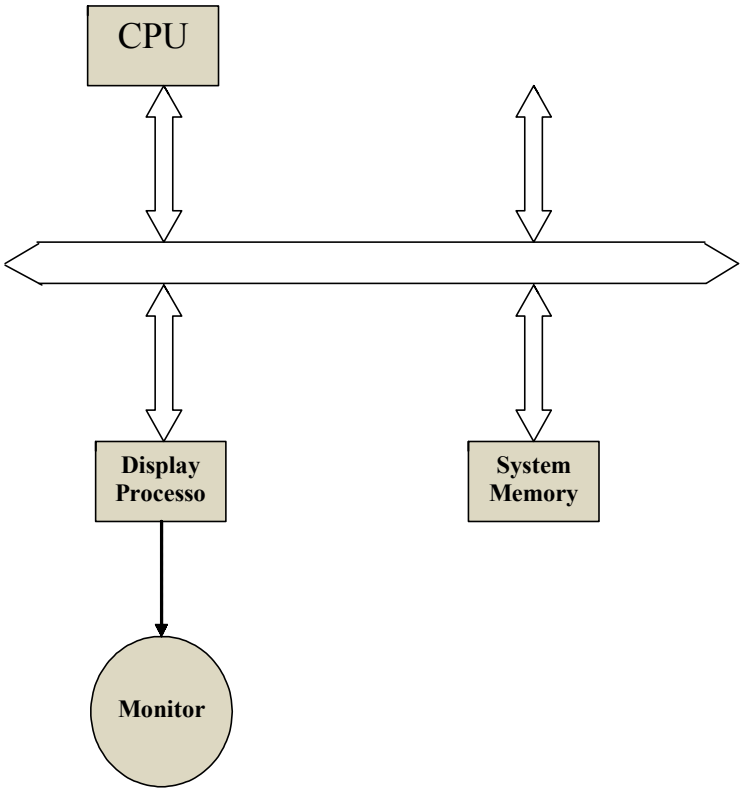


Figure 2.3: Architecture of a Random display system

2.3.2 Raster Scan Displays

In raster scan display, the electron beam sweeps the screen from top to bottom one row at a time in a regular pattern. The television uses this type of technology.

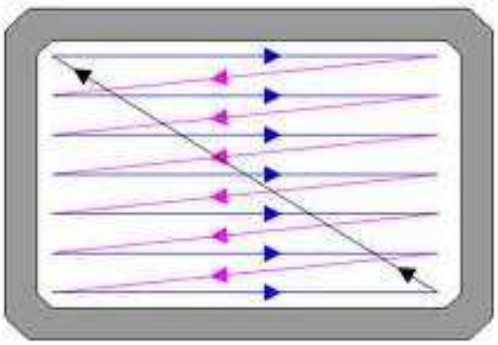


Figure 2.4: Raster Scan system

Unlike Random scan systems which were a line drawing system, the raster scan CRTs is a point plotting device. The electron beam is turned on and off to create a collection of bright spots. The picture definition is stored as a set of pixel intensity values in an area of memory called the *frame buffer*. Frame buffer is also called bit map or refresh buffer. The rows in a raster display are called scan lines. Figure 2.5 shows how the beam sweeps across each scan line to produce a picture as a set of discrete points.

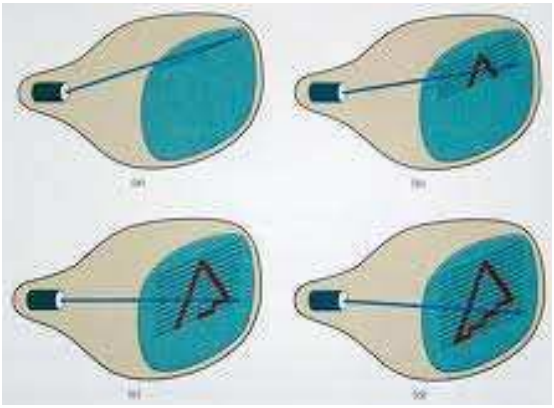


Figure 2.5: Raster scan

The raster scan displays are refreshed at a rate of 60 to 80 frames per second. At the end of each scan line, the electron beam returns to the left side of the screen to scan the next line. The return

to the left side of the screen, after refreshing each scan line, is called the *horizontal retrace* of the electron beam. At the end of each frame, the beam returns to the top left corner of the screen to begin the next frame. This is called the *vertical retrace.*

On some raster systems, each frame is displayed in two passes using an interlaced refresh process. In the first pass, the beam sweeps across every other scan line from top to bottom. Then the vertical retrace sweeps the remaining lines.

A simple architecture of the Raster Scan systems is shown in figure 2.6.

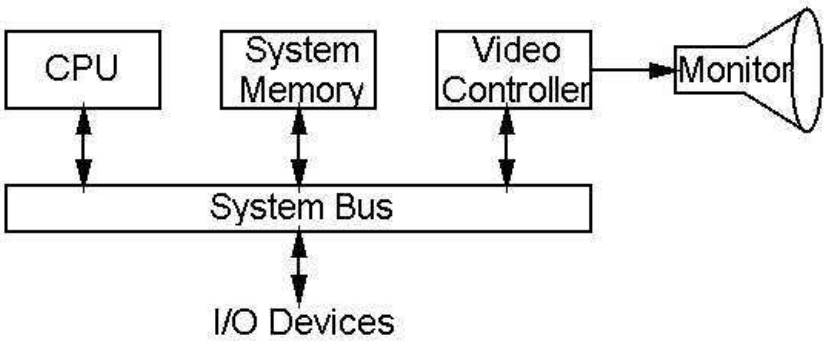


Figure 2.6: Architecture of a raster display system

The advantages of Raster Scan displays are:

- Realistic scenes can be drawn.
- Decreasing memory costs has made raster systems popular.

The disadvantages are:

- They have less resolution
- The lines produced are zig-zag as the plotted values are discrete.

2.4 COLOR CRT MONITORS

A color CRT contains a million of red, blue and green phosphors that emits different colored lights by their combination when strike on the screen. Red, green and blue are called primary colors. Any other color can be generated by mixing these three primary color with different intensity.

There are two basic techniques for producing colors displays:

a) Beam Penetration method (Random Scan CRTs)

b) Shadow-mask method (Raster Scan CRTs)

a) Beam Penetration Method: The inside of the CRT screen is coated with two layers of phosphors, usually the outer layer is red and the inner layer is green. The displayed color depends on how far the electron beam has been able to penetrate the phosphor layers.

A beam of slow moving electrons can excite only the outer red layer. A fast moving electron beam can penetrate the outer layer and excite the inner green layer. If the beam speed is not very fast or not very slow, then it displays two colors– yellow and orange.

The disadvantage of this method is:

- It can produce only four colors
- Picture quality is not so good.

b) Shadow-mask Method: The raster scan display systems use the shadow mask method to display color as they can display a broad range of colors compared to the beam penetration method. Each pixel position has three colored phosphor– red, green and blue. This type of CRT has three electron guns, one for each color dot. To create a crisp picture on the CRT screen, it is necessary that the beam for each color strikes only the correct dot intended. This is done by using a fine metal mesh called a shadow mask just behind the phosphor coated screen. The shadow mask contains a series of holes aligned with the phosphor dot pattern. The electron beam strikes the screen phosphors through this mask. When the three beams pass through a hole in the shadow mask, they activate a dot triangle, which appears as a small color spot on the screen. The phosphor dots in the triangles are arranged so that each electron beam can activate only its corresponding color dot when it passes through the shadow mask.

Color variations can be obtained by varying the intensity levels of the three beams. The displayed color depends on the amount of excitation of the red, blue and green phosphors.

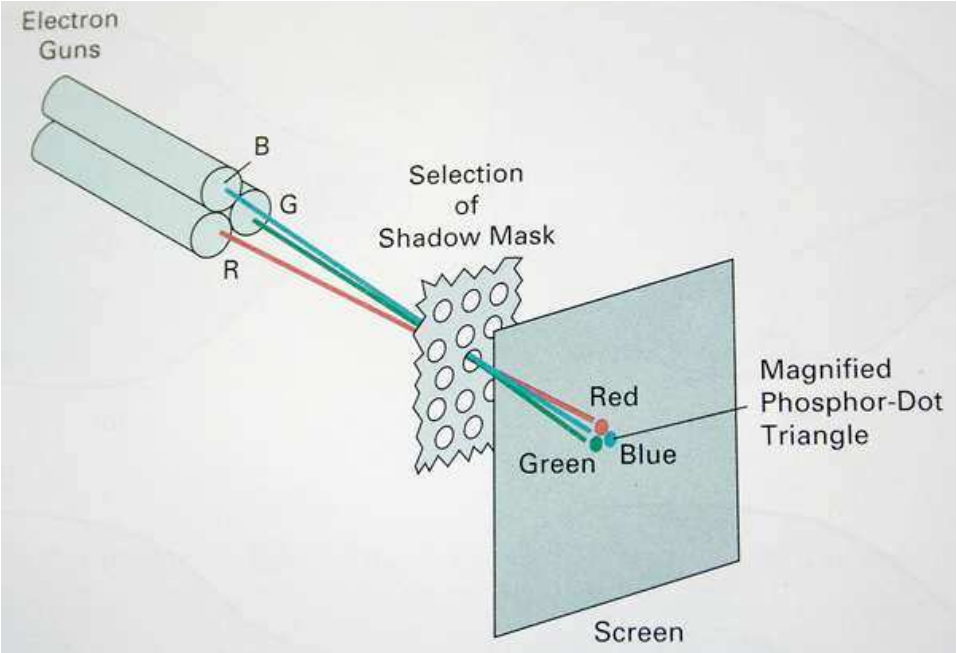


Figure 2.7: Operation of a shadow-mask CRT

So, by combining different intensities from the three electron guns different colors can be obtained. For example,

R	G	B	color
0	0	0	black
0	0	1	blue
0	1	0	green
0	1	1	cyan
1	0	0	red
1	0	1	magenta
1	1	0	yellow
1	1	1	white



CHECK YOUR PROGRESS

Q.1: Say True or False:

a) The CRT is a tube that produces images when an electron beam strikes a phosphorescent screen.

b) Persistence is a property of phosphors that allows the emitted light maintain its glow on the screen.

c) The components of an electron gun in a CRT is a heated cathode and a control grid that fits over the cathode.
d) Resolution of a monitor is independent of the number of pixels that can be displayed on the screen.
e) Aspect ratio is the ratio of the vertical points to the horizontal points to produce equal length lines in both directions on the screen.
f) Random scan displays is also called calligraphic displays.
g) Random Scan displays are very useful in creating realistic scenes.
h) Raster Scan CRTs is a point plotting output device.
i) Horizontal retrace in raster scan displays is the return of the electron beam to the left side of the screen, after refreshing each scan line.
j) The beam penetration method in color CRTs can produce a range of variety colors.

2.5 FLAT PANEL DISPLAYS

The flat panel displays are also called Flat Screens. The technology in flat panel displays are much different from the CRTs. They are much lighter and thinner and are usually less than 4 inches thick. They also consumes much less power than the traditional CRTs.

Flat-panel displays can be divided into two categories:

a) Emissive Displays and
b) Non-emissive Displays

The emissive displays converts electrical energy into light. Examples of emissive displays are Plasma panels, LEDs (Light Emitting Diodes), thin-film electroluminescent displays etc.

Non-emissive displays use optical effects to convert sunlight or light from other sources into graphical patterns. LCD (Liquid Crystal Displays) is an example of non-emissive display.

Flat panel screens have a precise matrix of rows and columns based on the highest resolution supported.



Figure 2.8: Flat Panel vs CRT

2.5.1 Plasma Panels

The plasma panels are also called **gas-discharge displays**. They are called plasma displays because the pixels on the screen rely on plasma cells or florescent lamps.

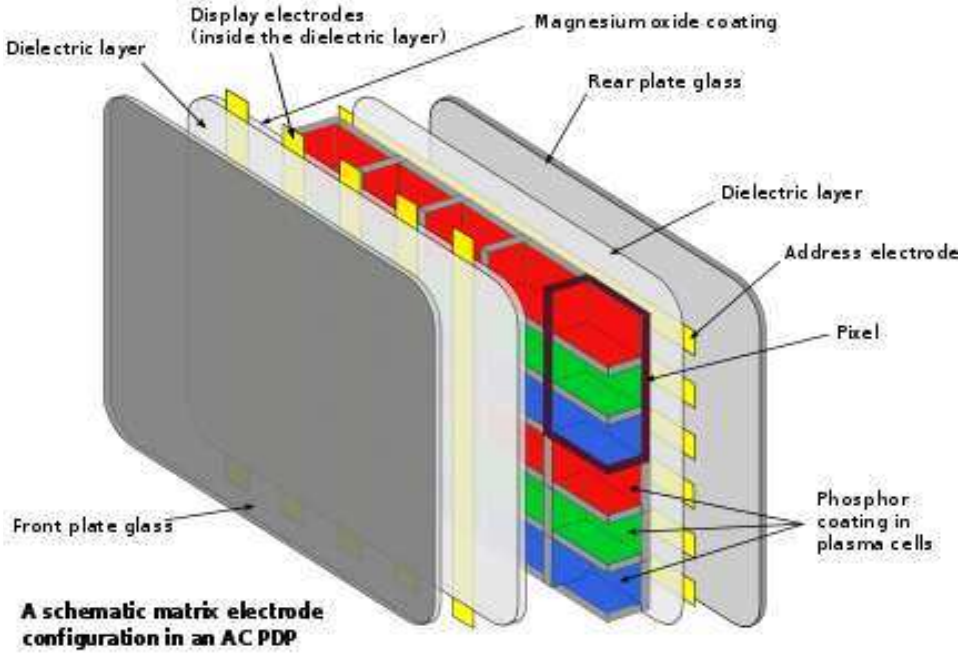


Figure 1.9: Plasma Panels

The plasma panel produces images on the screen pixel by pixel. The above figure shows the construction of a plasma panel. It

consists of two glass plates. The region between the glass plates is filled with neon based gas and is sealed. Long electrodes called the display electrode and address electrode are also put between the glass plates.

By applying voltage between the electrodes, the gas is made to behave as if they are tiny cells which are independent of each other. These cells are called plasma. These cells are made to glow by applying a very high voltage across it by means of the electrodes. Refreshing is not required as the pixel glow can be sustained by maintaining a high frequency alternating voltage across the cells. Each pixel consists of three different colored phosphors– red, green and blue.

Advantages:

1) Produces a flicker free steady image.
2) Refreshing is not required.
3) Slim profile and less bulky than CRTs.
4) Better and more accurate color reproduction.
5) Images do not suffer from degradation at high angles

Disadvantages:

1) Costlier than CRTs.
2) Susceptible to reflection glare in bright rooms.

1.8.2 Liquid Crystal Displays (LCD)

The term liquid crystal refers to the fact that these compounds have a crystalline arrangement of molecules, yet they flow like a liquid.

The basic structure of a LCD can be thought of as two glass plates each containing a light polarizer sandwiching a layer of nematic (thread-like) liquid crystal.

The polarized light passes through the liquid crystals and gets twisted following the direction in which the molecules are arranged. It will then pass through the other polarizer and reflected back to the viewer.

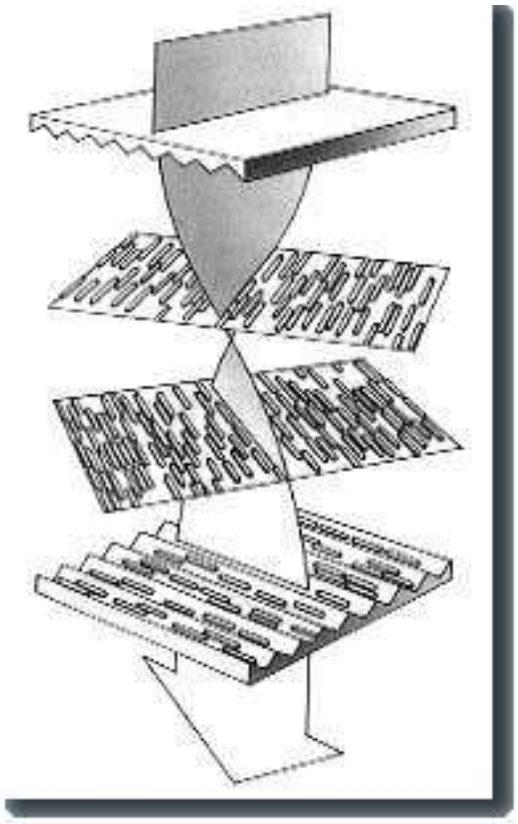


Figure 2.10: Structure of Liquid Crystal Displaysl (LCD)

On applying voltage, the molecules rearrange themselves vertically and the light is not twisted. So it will not pass through the other polarizer and the pixel will thus be turned off.

Thus when no voltage is applied to the liquid crystals, light passes.

Through and when a voltage is applied the light gets blocked and the screen appears black.

Advantages:

1) They are thin and compact.
2) Energy efficient. Consumes much less power.
3) Produces very bright images.
4) Completely flat screen.

Disadvantages:

1) Restricted viewing angles.
2) Contrast is lower than CRTs.
3) More expensive than CRTs.

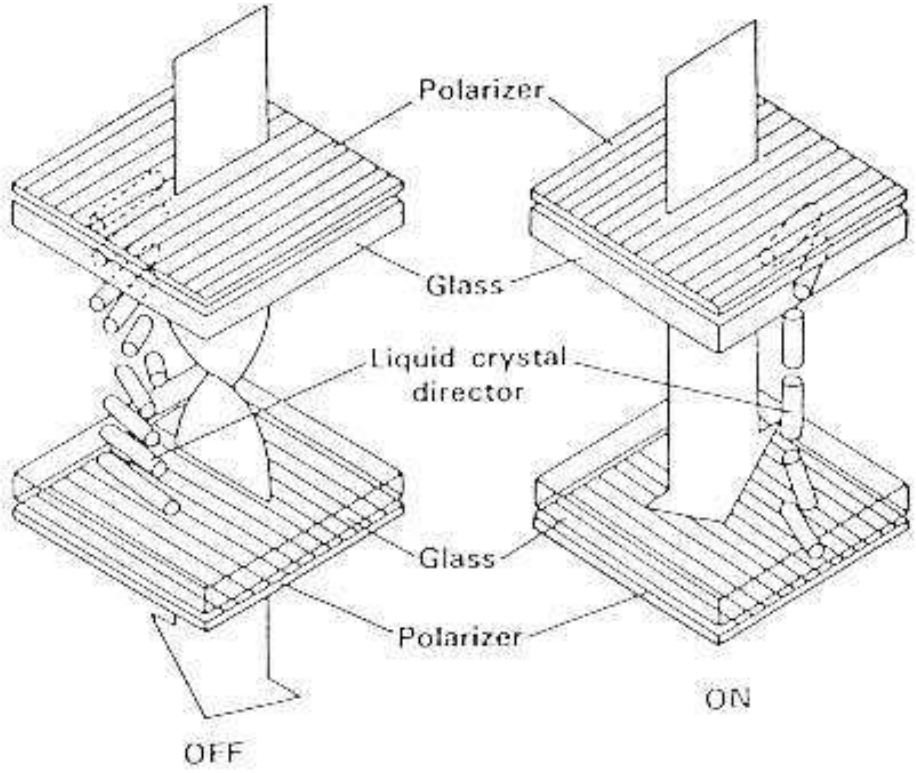


Figure 2.11: Liquid Crystal Display

2.5.3 Electroluminescent Displays

Luminescent is the name given to all forms of visible radiant energy due to causes other than temperature. Electroluminescent is the production of visible light by a substance exposed to an electric field without thermal energy generation.

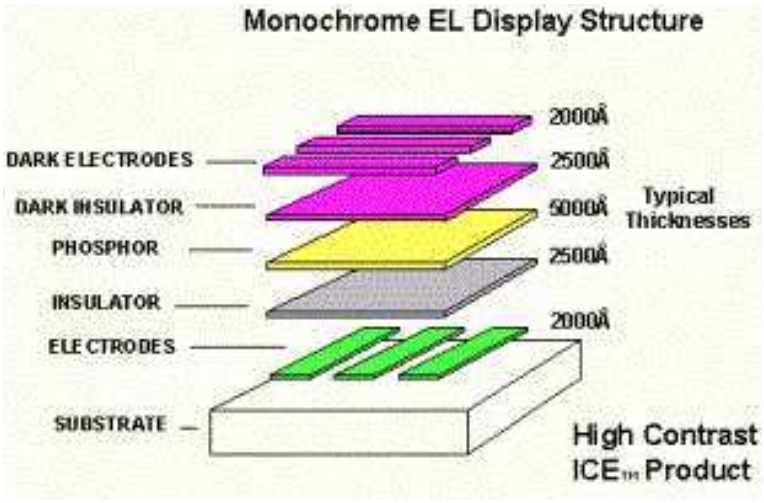


Figure 2.12: Monochrome EL Display Structure

The construction of electroluminescent displays (ELD) is similar to that of the plasma panel. They are created by sandwiching a layer of electroluminescent material between two layers of conductors.

The electroluminescent display consists of a transparent front layer known as the *substrate* carrying over its rear face a transparent electrically-conductive film forming the display's front electrode. Above it is a layer of electroluminescent/phosphor material between two insulators. A conductive film is present over the dark insulator to form the display's back electrodes.

2.6 GRAPHICS SOFTWARE

A graphics software is a software that is used to create, edit and manage computer graphics. The graphics may be any picture, web graphics, logos, headings or any other kind of digital images.

There are two main types of graphics software:

1) pixel-based image editors or raster graphics editor,
2) vector-based image editors or vector graphics editor.

1) Raster graphics editors: This is the most popular type of graphics software. The images created and edited in this type of software are called bitmaps because they are made up of a set of pixels. The pixel based editors are used to edit the digital photos and scanned images as these are always pixel-based. Very realistic pictures can be created in these type of editors. They are resolution dependent. The images are saved in the computer in any of the following bitmap formats: jpeg, gif, png and tiff. Some raster graphics editors are: Adobe Photoshop, Microsoft Paint, Corel Paint Shop.

2) Vector graphics editors: They allow to create images using lines and curves. They can be easily edited by moving the points, adjusting curves and changing the objects color. They are resolution independent. These softwares are not used for creating realistic pictures rather used for creating graphic designs, logos, cartoons etc.

Some vector graphics editors are:

Adobe Illustrator, CorelDRAW, Macromedia Freehand.

2.7 COLOR MODELS

A color model is a system for creating a range of colors using the primary colors. There are two types of color models : additive and subtractive. Additive color models uses light to display the colors and the subtractive color model uses printing inks.

The two most common color models are:

- RGB Color Model,
- CMYK Color Model.

a) RGB Color Model: Computer uses the RGB color model to display colored images. The color displayed results from the light transmitted from the electron guns. The combination of the three colors: red, blue and green produces the white color. The overlapping of the red, blue and green colors produces the subtractive colors (cyan, magenta and yellow).

The RGB color model can produce a broad range of colors. This model is device-dependent. The computer, TV, camera, mobile phones all uses the RGB color model.

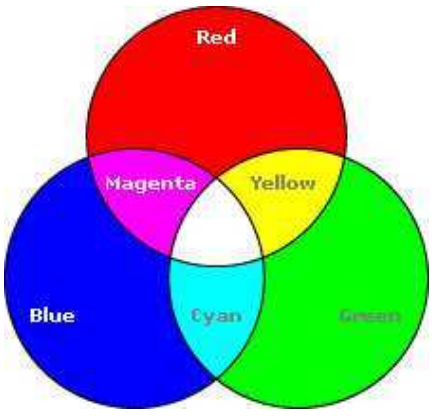


Figure 2.13 : RGB color model

b) CMYK Color Model : This model uses the subtractive primary colors to produce different colors. It is used for printing and therefore uses ink to display color. The combination of the three colors produces the Black color.

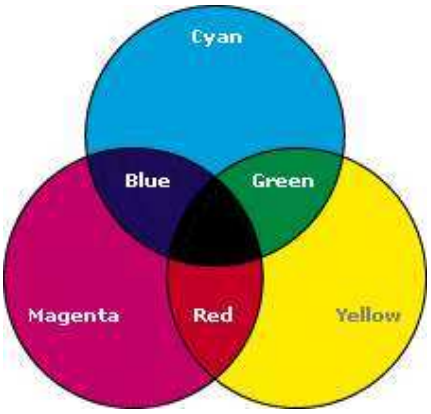


Figure 2.14: CMYK color model

The overlapping of the subtractive colors produces the additive colors that are Red, Green and Blue colors. The colors produced are not as bright as the colors produced in the RGB model.



CHECK YOUR PROGRESS

Q.2: Fill in the Blanks :

a) The Flat panel displays are also called .....

b) The two categories of flat panel displays are ..... and .....

c) The plasma panels are also called .....

d) the region between the two glass plates in plasma panels is filled with a ..... gas.

e) The polarized light in an LCD passes through the liquid crystals and gets twisted following the direction in which the ..... are arranged.

f) ..... is the production of visible light by a substance exposed to an electric field without thermal energy generation.

g) The images created and edited in ..... are called bitmaps because they are made up of a grid of pixels.

h) ..... are resolution independent.

i) the two common color models are ..... and .....

j) The overlapping of the red, blue and green colors in the RGB color model produces the subtractive colors .....



2.8 LET US SUM UP

- The Cathode Ray Tube (CRT) is a type of vacuum tube which displays images when the electron beam through electron guns are strikes on the phosphor coated screen.
- The picture is redrawn repeatedly by quickly directing the electron beam back over the same points so as to maintain the picture on the screen. This is called **Refresh CRT**.
- The electron beam when strikes the phosphor coated screen produces a bright spot on the screen.
- The maximum number of points that can be displayed on the screen without overlap on the CRT is called the **Resolution** and is usually given as the number of horizontal points versus the number of vertical points.
- **Aspect ratio** is the ratio of the vertical points to the horizontal points to produce equal length lines in both directions on the screen.
- In Random Scan display, the electron beam is directed only to those parts of the screen where the picture has to be drawn. It draws the picture one line at a time.
- In Raster scan display, the electron beam sweeps the screen from top to bottom one row at a time in a regular pattern.
- There are two basic techniques for producing color displays Beam Penetration method and Shadow- mask method.
- Flat Panel displays are lighter, thinner and requires less power than CRTs.
- The Plasma Panel consists of two glass plates and the region between the plates is filled with a neon-based gas.
- The electroluminescent displays are created by sandwiching a layer of electroluminescent material between two layers of conductors.

- There are two graphics software – raster graphics editor and vector graphics editor.
- RGB and CMYK are the two most commonly used color models.



2.9 FURTHER READING

1) Hearn, Donald and Baker, M. Pauline. *Computer Graphics*. Prentice-Hall.

2) Shirley, Peter; Marschner, Steve. (3rd Edition). *Fundamentals of Computer Graphics*.



2.10 ANSWER TO CHECK YOUR PROGRESS

Ans. to Q. No. 1: a) True, b) False, c) True, d) False, e) True, f) True, g) False, h) True, i) True, j) False

Ans. to Q. No. 2: a) flat screens; b) emissive displays, non-emissive displays; c) gas-discharge displays; d) Neon-based; e) Crystal molecules; f) Electroluminescent; g) raster graphics editors; h) Vector graphic editors; i) RGB model, CMYK model; j) cyan, magenta and yellow.



2.11 MODEL QUESTIONS

Q.1: Discuss the technology used in Cathode Ray Tubes.

Q.2: What is the difference between raster scan displays and random scan displays?

Q.3: Discuss the color display techniques in color CRT monitors.

Q.4: Explain briefly the flat screen displays.

Q.5: Describe the color models in computer graphics.

Q.6: What is the function of the electron gun in a CRT.

*** ***** ***

UNIT 3: LINE DRAWING ALGORITHMS

UNIT STRUCTURE

3.1 Learning Objectives

3.2 Introduction

3.3 Points and Lines

3.4 Line Drawing Algorithms

- 3.4.1 DDA (Digital Differential Analyser)
- 3.4.2 Bresenham's Algorithm

3.5 Let Us Sum Up

3.6 Further Reading

3.7 Answers to Check Your Progress

3.8 Model Questions

3.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- learn cartesian coordinate system
- learn how the different line drawing algorithms work.

3.2 INTRODUCTION

In the previous unit we have learnt about the different video display devices. The raster scan and random scan display techniques are also elaborated. We have also come across the different color models.

Computer graphics comprises of pictures or images, animations, videos, films created using computers. It refers to computer-generated image with the help of graphical hardware and software. Computer graphics is responsible for displaying art and image data effectively and meaningfully to the consumer. Points and lines are the simplest components of pictures. Others include circles, ellipses, surfaces, polygons and also somewhere character strings.

In this unit we will start with some raster graphics algorithms that will draw straight lines or curves on the display unit. We will also learn how to fill the boundaries and the areas in and out of polygon.

3.3 POINTS AND LINES

The most commonly used display method in a Cathode Ray Tube (CRT) is the Raster Scan Display. It is used for realistic display of scenes containing shading and colour pattern. Each point on the screen is referred to as a *pixel,* shortened for 'picture element'. The picture definition is stored in a memory area called the *refresh buffer* or the frame buffer. The screen positions or we can say the pixel positions are referenced in *Cartesian Coordinates*. The coordinate origin is defined at the lower left screen corner with the positive values of x- coordinate increasing to the right of screen and positive values of y- coordinate increasing from bottom to the top of screen. Points and lines are the fundamental in drawing a picture on the screen. A point is simply a pair of x, y coordinates (x,y) on the display and is graphically represented as a dot. Plotting a point on the screen is nothing but illuminating a phosphor dot on the screen. However a line is an infinite series of points. It connects two points. They can be straight or curved and continuous or broken. In plotting a line, the starting coordinates and the endpoint coordinates of the line segment should be specified.

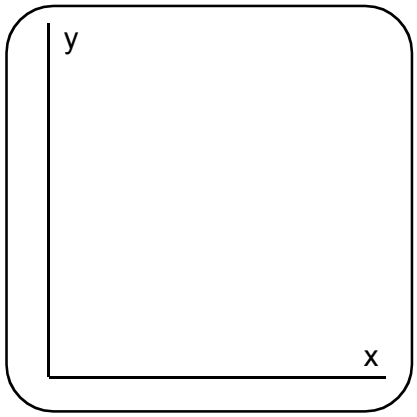


Figure 3.1: Cartesian coordinate

3.4 LINE DRAWING ALGORITHMS

Before going into the details of line drawing algorithms, we have to know some definitions.

Rasterization: It is the process of determining which pixels provides the best approximation to a line on the CRT screen.

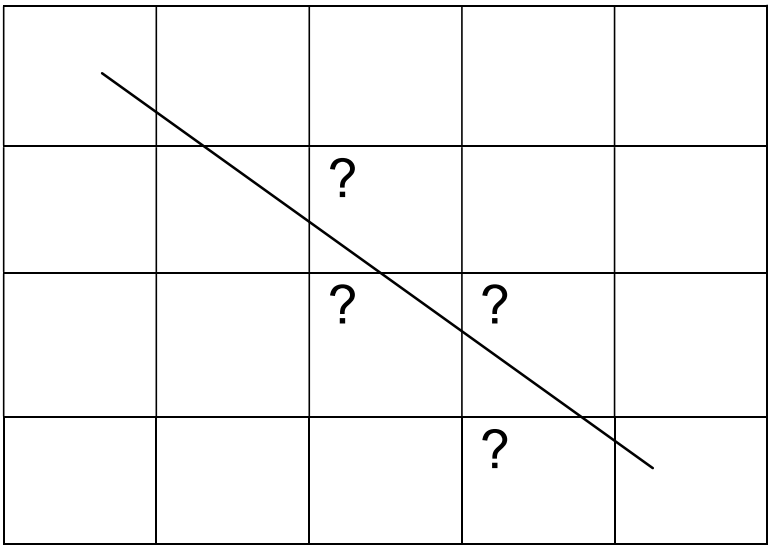


Figure 3.2: Example of one pixel in two coordinate

Scan Conversion: The generation of the picture in scan line order combined with rasterization is called Scan Conversion.

It is always important that the straight lines must appear straight and they should have constant brightness along them.

The goal of any line drawing algorithm is to construct the best possible approximation of a line. Following are the some of the qualities that are often considered while drawing the line on the raster display:

- Continuous appearance of the line
- Uniform thickness and brightness
- Are the pixels nearest to the ideal line turned on or not
- The line generation speed.

3.4.1 Digital Differential Analyser (DDA) Algorithm

A straight line may be defined by two endpoints and an equation. In figure 3.3 the two endpoints are described by (x_1, y_1) and (x_2, y_2). The equation of the line is used to describe the (x, y) coordinates of all the points that lie between the two endpoints. Using the equation of a straight line, $y = mx + b$ where $m = \triangle y/\triangle x$ and b = the y intercept. We can find the values of y by incrementing x from $x = x_1$ to $x = x_2$. By scan converting these calculated x, y values, we represent the line as a sequence of pixels.

The differential equation of a straight line is given by

dy/dx = constant

or, $\triangle y / \triangle x = y_2 - y_1 / x_2 - x_1$

= constant

where (x_1, y_1) and (x_2, y_2) are the end points of a line and are assumed to be not equal.

Now, $\triangle y = m \triangle x$, where m is the slope of the line (3.1)

Similarly, $\triangle x = \triangle y / m$ (3.2)

Equations 3.1 and 3.2 helps in determining the deflection voltages in analog devices.

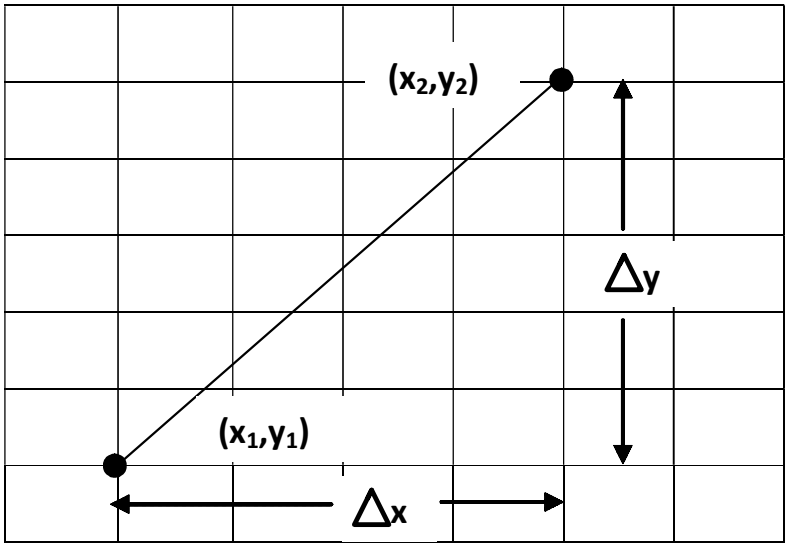


Figure 3.3: Straight line

The DDA Algorithm is a scan conversion algorithm and is based on calculating either $\triangle x$ or $\triangle y$ with the help of equations 3.1 and 3.2.

In raster systems, the lines are made of pixels one after another and the pixel separation forms a constraint in the step sizes in the vertical and horizontal directions. Therefore, sampling of a line at discrete positions and then determine the nearest pixel to the line at each sampled position becomes necessary.

Sampling is done at one coordinate and the corresponding integer values nearest the line path is determined for the other coordinate.

Considering a line with a positive slope, i.e. m<=1, we will sample at unit x intervals ($\triangle x = 1$) and the calculate each of the successive y coordinate values.

$\triangle y / \triangle x = m$

$(y_{k+1} – y_k) / x_{k+1} – x_k) = m$, where k = 1, 2, 3, …, …, …

$y_{k+1} – y_k = m\triangle x$

$y_{k+1} – y_k = m$ (as $\triangle x = 1$)

$y_{k+1} = y_k + m$ (3.3)

The value of k starts at 1 and will go on increasing till it reaches the endpoint. Also, the value of m can be any real number, so the value of y which is calculated must be rounded to the nearest integer value.

This above calculation was for lines with positive slopes but with lines with negative slopes i.e. when m > 1, we sample at unit y intervals ($\triangle y = 1$) and calculate the corresponding integer x values as:

$x_{k+1} = x_k + 1/m$ (3.4)

If the starting point is at the right side, then we may either have–

$\triangle x = –1$, and

$y_{k+1} = y_k – m$ (3.5)

or, $\triangle y = –1$, and

$x_{k+1} = x_k – 1/m$ (3.6)

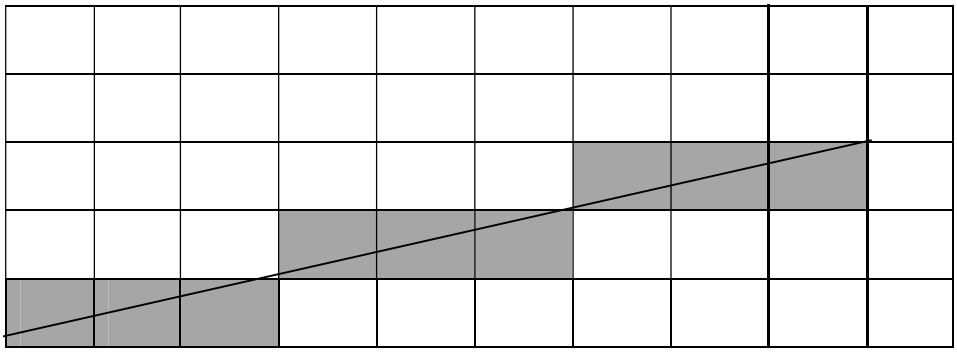


Figure 3.4: Pixels generated by DDA Algorithm

The function for the DDA Algorithm is given as follows:

```
linedda( int x1, int y1, int x2, int y2, int color)
{
    int dx, dy, step, k, roundx, roundy;
    float x, y,xincr, yincr;
    dx = x2-x1;
    dy = y2-y1;
    if(abs(dx) > abs(dy))
        step = abs(dx);
    else
        step = abs(dy);
    xincr = dx / step;
    yincr = dy / step;
    x = x1;
    y = y1;
    roundx = round(x);
    roundy = round(y);
    putpixel(roundx,roundy,color);
    for( k=1;k<=step;k++
    {
        x = x + xincr;
        y = y + yincr;
        roundx = round(x);
        roundy = round(y);
        putpixel(roundx,roundy,color);
    }
}
```

The DDA algorithm runs rather slowly because it requires real arithmetic (floating point operations).

3.4.2 Bresenham's Line Algorithm

The Bresenham's algorithm was developed by Jack Bresenham in 1965 at IBM. It is another incremental scan conversion

algorithm and the advantage is that it uses only integer arithmetic thereby runs significantly faster as integer arithmetic has the advantage of speed and precision. Working with floating point values requires more time and memory and such values would need to be rounded to integers anyway.

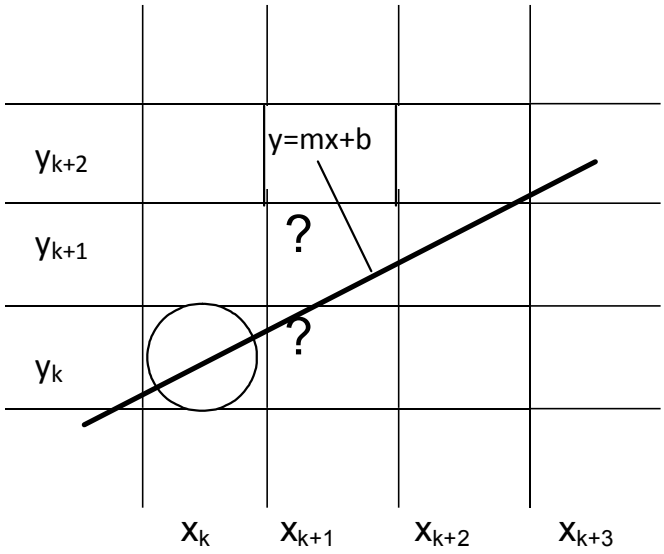


Figure 2.5: Scan line

Let us consider drawing a line with a positive slope, i.e., m<1. The pixel positions along the line are determined by sampling at unit x intervals.

Starting from the left end point of the given line we move across the x-axis in unit intervals and go on plotting the pixel whose scan line y-axis value is closest to the path of the line.

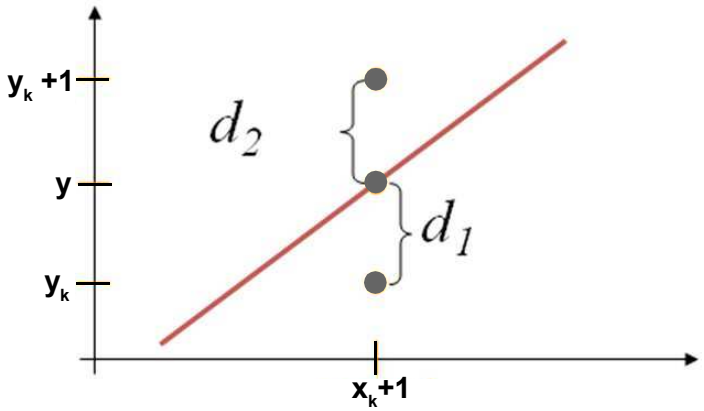


Figure 2.6:

Having plotted a pixel at (x_k, y_k), the line drawing algorithm has a limited range of options as to where to plot the next pixel in the coordinate system.

- Whether to plot the point at (x_k+1, y_k), or
- Whether to plot the point at (x_k+1, y_k+1).

So the algorithm has to decide between such two possibilities at each step.

The above figure shows the situation in which the algorithm finds itself in plotting the k^{th} pixel.

At sample position x_k+1 the vertical separations from the mathematical line are labeled as d_1 and d_2.

The y-coordinate on the mathematical line at x_k+1 is as follows:

$$y = m(x_k+1) + b$$

So, d_1 and d_2 are as given below:

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

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

We can use these to decide which pixel is more closer to the mathematical line and this simple decision is based on the difference between the two pixel positions.

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

Now, substituting m with $\triangle y / \triangle x$ and multiplying both sides by $\triangle x$:

$$x(d_1 - d_2) = \triangle x(2\triangle y / \triangle x(x_k+1) - 2y_k + 2b - 1)$$
$$= 2\triangle y.x_k - 2\triangle x.y_k + 2\triangle y + \triangle x(2b-1)$$
$$= 2\triangle y.x_k - 2\triangle x.y_k + c$$

where c is a constant and $c = 2\triangle y + \triangle x(2b-1)$

A decision parameter p_k for the k^{th} step is given by:

$$p_k = \triangle x(d_1 - d_2) = 2\triangle y.x_k - 2\triangle x.y_k + c$$

The sign of the decision parameter p_k is same as that of $d_1 - d_2$.

If p_k is negative then we choose the lower pixel, otherwise we choose the upper pixel.

At step k+1, the decision parameter p_{k+1} is as:

$$p_{k+1} = 2\triangle y.x_{k+1} - 2\triangle x.y_{k+1} + c$$

Substracting p_k from p_{k+1} we get the following:

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

But x_{k+1} and x_k+1 are same and is equals to 1, so we get,

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

where $y_{k+1} - y_k$ is either 0 or 1 depending on the sign of p_k.

This iteration is repeated until the end point of the mathematical line is reached.

The Bresenham's Line Drawing Algorithm for m<1 is given as follows:

1) Input the two line end points, storing the left end point in (x_0,y_0).
2) Plot the point (x_0, y_0).
3) Calculate the constants $\triangle x$, $\triangle y$, $2\triangle y$ and $(2\triangle y - 2\triangle x)$ and get the first value for the decision parameter as: $p_0 = 2\triangle y - \triangle x$.
4) At each x_k along the line, starting at k = 0, perform the following tests.

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

   Otherwise the next point to plot is:

   (x_k+1, y_k+1) and $p_{k+1} = p_k + 2\triangle y - 2\triangle x$
5) Repeat the step number 4, $\triangle x$ times.



CHECK YOUR PROGRESS

Q.1: Fill in the blanks:

a) The process of determining which pixels provides the best approximation to a line on the CRT screen is called .....

b) The definition of the picture is stored in a memory area called the .....

c) The two things that has to be specified in plotting a line are ..... and .....

Q.2: What are the qualities that are considered while drawing a line on the screen?

.....

.....

.....

.....

.....

Q.3: Suppose a point is plotted at position (x_k, y_k) on the screen coordinate system. What are the possibilities for a line drawing algorithm to plot the next point on the screen?

.....

.....

.....

.....

.....

Q.4: What are the advantages of the Bresenham's Line drawing algorithm over the Digital Differential Analyser (DDA) algorithm.

.....

.....

.....

.....

.....

.....

.....



3.5 LET US SUM UP

- A pixel is a point (x, y) on the display screen and a line is an infinite series of such points.
- Rasterization is the process of determining which pixels provides the best approximation to a line on the CRT screen.

- Scan Conversion is the generation of the picture in scan line order combined with rasterization.
- Two different line drawing algorithms discussed above are DDA (Digital Differential Analyser) and Bresenham's Line drawing algorithms.
- The DDA algorithm works slow as it involves floating point operations. On the other hand, Bresenham's algorithm is faster in terms of speed and precision as it involves only integer arithmetic.



3.6 FURTHER READING

1) Hearn, Donald and Baker, M. Pauline. *Computer Graphics*. Prentice-Hall
2) Newman and Sproull. (1979). *Principles of Interactive Computer Graphics*. McGraw-Hill.
3) Plastock and Kalley. (1986). *Computer Graphics, Schaum's Outlines*. McGraw-Hill.



3.7 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q. No. 1: a) rasterization; b) frame buffer; c) starting coordinates, endpoint coordinates.

Ans. to Q. No. 2: The following are some of the qualities that are often considered while drawing the line on the raster display.

i) Continuous appearance of the line
ii) Uniform thickness and brightness
iii) Are the pixels nearest the ideal line turned on or not
iv) The line generation speed.

Ans. to Q. No. 3: Having plotted a pixel at (x_k, y_k), the line drawing algorithms has a limited range of options as to where to plot the next pixel in the coordinate system. They are:

i) Whether to plot the point at (x_k+1, y_k), or
ii) Whether to plot the point at (x_k+1, y_k+1).

So the algorithm has to decide between such two possibilities at each step.

Ans. to Q. No. 4: The Bresenham's Line drawing algorithm has the following advantages:

i) It is a fast incremental algorithm

ii) It uses only integer calculations

On comparing the Bresenham's algorithm to the DDA algorithm, the DDA algorithm has the following drawbacks:

i) Accumulation of round-off errors can make the pixelated line drift away from what was intended

ii) The rounding operations and floating point arithmetic are time-consuming.



3.8 MODEL QUESTIONS

Q.1: Write the DDA algorithm and the Bresenham's line drawing algorithm. Compare both these two algorithm.

Q.2: The endpoints of a given line are (0, 0) and (6, 18). Compute each value of y as x steps from 0 to 6 and plot the result.

Q.3: What are the steps required to plot a line using the slop method.

Q.4: What are the steps required to plot a line whose slope is between 0 and 45^{o} using Bresenham's method.

*** ***** ***

UNIT 4: CIRCLE AND ELLIPSE DRAWING ALGORITHMS

UNIT STRUCTURE

4.1 Learning Objectives

4.2 Introduction

4.3 Circle Generating Algorithms

4.3.1 Mid-Point Circle Algorithm

4.4 Ellipse Generating Algorithms

4.4.1 Mid-Point Ellipse Algorithm

4.5 Let Us Sum Up

4.6 Further Reading

4.7 Answers to Check Your Progress

4.8 Model Questions

4.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- learn how to draw circle and ellipse using Computer Graphics
- learn about circle generating algorithm
- learn about ellipse generating algorithms.

2.2 INTRODUCTION

In the previous unit we have discussed few line drawing algorithms. We have learnt how to draw lines using Bresenham's line drawing algorithm and DDA.

In this unit we will start with circle drawing algorithms that will draw circle and ellipse on the display unit. A circle is a symmetrical figure. Any circle generating algorithm can take the advantage of the circle symmetry to plot eight points for each value that the algorithm calculate. Eight-way symmetry is used by reflecting each calculated point around each 45^o axis. For example, if point 1 is calculated with a circle drawing algorithm, then seven more points could be found by reflection as shown in figure 4.1.

4.3 CIRCLE GENERATING ALGORITHM

A circle is defined as a set of points that are at a given distance r from a centre position (x_c, y_c).

The equation for a circle is: $x^2 + y^2 = r^2$

where, r is the radius of the circle.

From the above equation we can calculate the y-coordinate values as: $y = \pm\sqrt{r^2 - x^2}$

These equations can be used to calculate the points on a circle circumference at unit x intervals.

However, this method has some disadvantages:

- The resulting circle has large gaps where the slope approaches the vertices.
- It involves considerable calculations.

So a more efficient and accurate solution is required.

The computations required in generating a circle can be reduced by considering the symmetry of the circles. The shape of the circle is similar in each quadrant. The circle section in the first and second quadrants are symmetric with respect to the y-axis. Circle sections in the third and fourth quadrants can be obtained from sections in the first and second quadrants by considering the symmetry about the x-axis. There is also symmetry between octants. Circles thus have eight way symmetry.

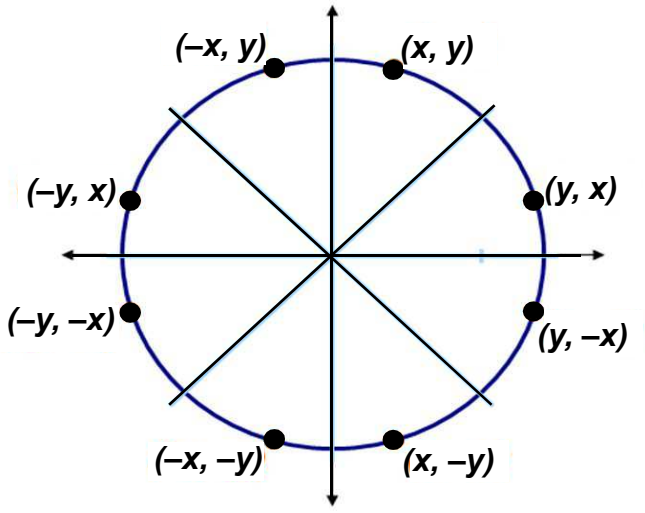


Figure 4.1 : Symmetry of a circle

This symmetry conditions are illustrated in the figure 4.1 where calculation of a circle point at (x, y) in one octant yields the circle points shown for the seven other octants.

The more efficient circle algorithms are based on incremental calculations of decision parameters as in the case with drawing of lines.

4.3.1 Mid-Point Circle Algorithm

In the Mid-Point circle algorithm, the eight-way symmetry of circles is used. In this algorithm, only the points for the top right eighth of a circle are calculated and then use symmetry to obtain the rest of the circle circumference points.

As in the case of line drawing algorithms, we sample at unit intervals and determine the closest pixel to the circle path.

Let us define a circle function, f_{circle} as: $f_{circ}(x, y) = x^2 + y^2 - r^2$

where (x,y) is any point on the boundary of the circle with radius r.

The above equation evaluates as follows:

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

If the point (x,y) is in the circle interior, then the circle function is negative and if it lies outside the circle boundary, then the circle function is positive.

By evaluating this function at the mid-point between the candidate pixels we can make the decision. So this circle function is the decision parameter in the mid-point circle algorithm.

Suppose the pixel at (x_k, y_k) has been plotted. So the next step is to choose a point that is nearest to the actual circle path. Here the two possibilities are: $(x_k + 1, y_k)$ and $(x_k + 1, y_k - 1)$.

The decision parameter can then be defined as:

$$p_k = f_{circ}(x_k + 1, y_k - ½)$$
$$= (x_k + 1)^2 + (y_k - ½)^2 - r^2$$

If $p_k < 0$ then the mid-point is inside the circle and the pixel at y_k is closer to the circle. Otherwise the mid-point is outside and y_k–1 is closer.

The successive decision parameters are obtained using incremental calculations. Now,

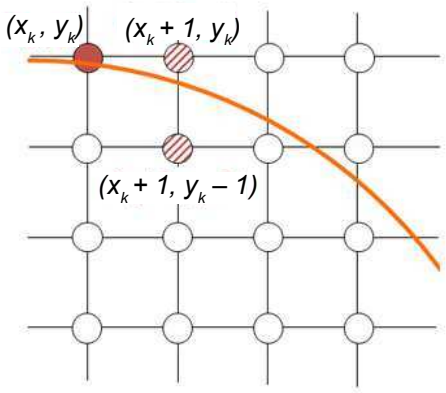


Figure 2.8: Circle scan conversion

$$p_{K+1} = f_{circ}(x_{k+1} + 1, y_{k+1} - ½)$$
$$= [(x_k + 1) + 1]^2 + (y_{k+1} - ½)^2 - r^2$$

Or, $p_{k+1} = p_k + 2(x_k + 1) + (y^2_{k+1} - y^2_k) - y_{k+1} - y_k) + 1$

where y_{k+1} is either y_k or y_k–1 depending on the sign of p_k

The first decision variable is given as: $p_0 = f_{circ}(1, r - ½)$

$$= 1 + (r - ½)^2 - r^2$$
$$= {}^5/_4 - r$$

If $p_k < 0$ then the next decision parameter is as: $p_{k+1} = p_k + 2x_{k+1} + 1$

Otherwise, $p_{k+1} = p_k + 2x_{k+1} + 1 - 2y_k + 1$

The **Mid-Point Circle Algorithm** is as follows:

1) Input the radius r and circle center (x_c, y_c) and obtain the first point on the circumference of a circle centered on the origin as: $(x_0, y_0) = (0, r)$
2) Calculate the initial value of the decision parameter as: $p_0 = {}^5/_4 - r$
3) Starting with k = 0, at each position x_k, perform the following test. If $p_k < 0$ the next point along the circle centered 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 $p_{k+1} = p_k + 2x_{k+1} + 1 - 2y_{k+1}$

4) Determine symmetry points in the other seven octants.
5) Move each calculated pixel position (x, y) onto the circular path centered at (x_c, y_c) to plot the coordinate values .
6) Repeat the steps 3 through 5 until x>=y.

$\sqrt{(x-x_1)^2+(y-y_1)^2} + \sqrt{(x-x_2)^2+(y-y_2)^2}$ = constant

4.4 ELLIPSE GENERATING ALGORITHM

An ellipse is defined as a set of points such that the sum of the distances from two fixed points is a constant.

That is, $d_1 + d_2$ = constant

Let us assume that the foci are at $F_1(x_1, y_1)$ and at $F_2(x_2, y_2)$.

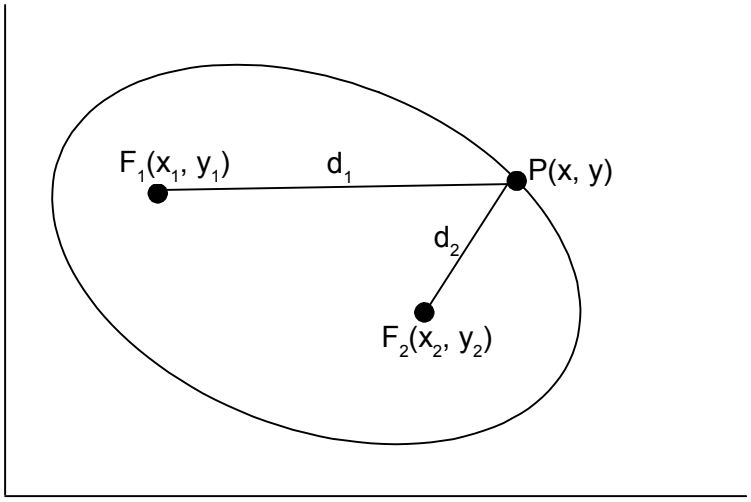


Figure 4.3: Ellipse generated about foci F_1 and F_2

By equating squares, isolating the radical and again equating squares, we will get an equation equivalent to the basic conic form:

$Ax^2 + By^2 + Cxy + Dx + Ey + F = 0$

where the parameters A, B, C, D, E and F depend on where the ellipse is centered and the lengths of its major and minor axes.

Suppose that the major and minor axes are parallel to the x and y axes using r_x for semi-major axis and r_y for semi minor axis, we get an ellipse equation having center as (x_c, y_c)–

$$\left[\frac{x - x_c}{r_x}\right]^2 + \left[\frac{y - y_c}{r_y}\right]^2 = 1$$

4.4.1 Mid-Point Ellipse Algorithm

Mid-Point ellipse algorithm is a method for drawing ellipses in computer graphics. It is a modification of the Bresenham's algorithm and the advantage of this method is that only addition operations are done thereby leading to fast implementation.

Let us consider a quarter of an ellipse as given in the figure 4.4. The curve is divided into two regions.

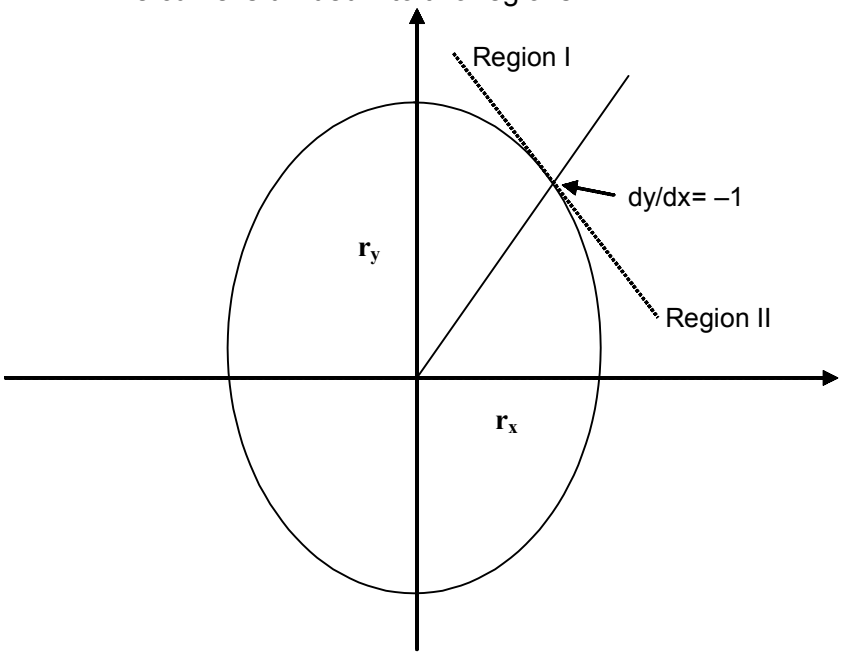


Figure 4.4: Ellipse processing regions

This algorithm is applied to the first quarter in two parts where $r_x < r_y$. The ellipse centre is assumed to be at the origin. The coordinates of the points in the other quarters can be obtained using the ellipse symmetry while implementation. For a pixel (x, y) in the first quarter, the corresponding points in the other three quarters are (x, –y), (–x, y) and (–x, –y) respectively. If the centre is at (x_c, y_c), all calculated coordinate (x, y) should be adjusted by adding the offset (x_C, y_C). The process is by taking unit steps in x-direction where the slope is less than 1 and by taking unit steps in y where the slope is greater than 1.

For region I, we can proceed from $(0, r_y)$ with unit steps in x-direction. For region II we can proceed from $(r_x, 0)$ with unit steps in negative y-direction.

An ellipse function with $(x_c, y_c) = (0, 0)$ is defined as:

$$f_{ellipse}(x,y) = r_y^2x^2 + r_x^2y^2 - r_x^2r_y^2$$

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

This ellipse function is the decision parameter in the mid-point ellipse algorithm. At each sampling position, the next pixel along the ellipse path is selected according to the decision parameter's sign that is evaluated at the mid-point between two candidate pixels. The ellipse symmetry is shown in the figure 4.5.

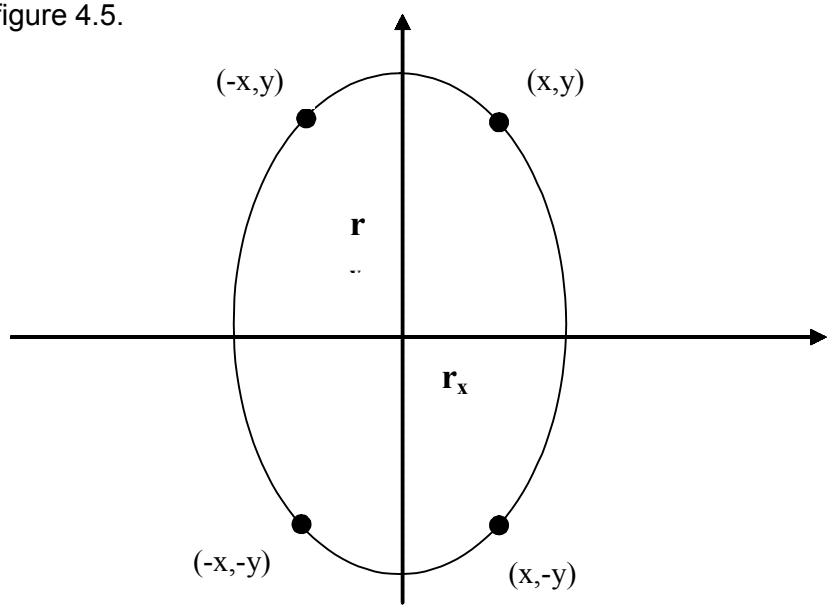


Figure 4.5: Symmetry of an ellipse

For drawing the ellipse we start at the point $(0,r_y)$ and take unit steps in the x-direction until the boundary between region I and region II is reached. Then we move unit steps in the y-direction over the remaining curve in the first quarter. At each step the value of the ellipse slope is calculated.

The ellipse slope is calculated as–

$$\frac{d_y}{d_x} = \frac{2r_y^2x}{2r_x^2y}$$

At the boundary between region I and region II, $d_y/d_x = -1$, so we get,

$$\mathbf{2r_y^2x = 2r_x^2y}$$

Whenever $\mathbf{2r_y^2x >= 2r_x^2y}$, we move out of region I.

$$r^2_y x^2 + r^2_x y^2 - r^2_x r^2_y = 0$$

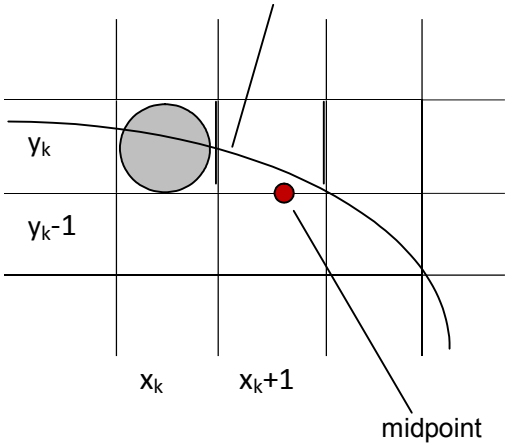


Figure 4.6: Midpoint between candidate pixels at x_k+1 along ellipse path

Suppose the pixel at position (x_k, y_k) has been plotted. We now have to decide which pixel to plot next along the ellipse path. The choices are:

(x_k+1, y_k) and (x_k+1, y_k-1)

The next position is determined by evaluating the decision parameter at the midpoint. The midpoint here is:

$$(x_k+1 + x_k+1)/2 \text{ and } (y_k + y_k - 1)/2$$

$$= (x_k+1, y_k-½)$$

Now, the decision parameter at this midpoint can be defined as–

$$p1_k = f_{ellipse}(x_k+1, y_k-½)$$

$$= r^2_y(x_k+1)^2 + r^2_x(y_k-½)^2 - r^2_x r^2_y$$

If $p1_k < 0$, then the mid-point is inside the ellipse and the pixel at y_k is closer to the ellipse boundary. Otherwise the mid-point is outside or on the ellipse boundary and the pixel y_k-1 is closer.

The decision parameter for the next point p_{k+1} at position $(x_k + 2, y_{k+1} - ½)$ is then evaluated.

As $x_k + 2 = x_{k+1}+1$ we can write as:

$$p1_{k+1} = f_{ellipse}(x_{k+1}+1, y_{k+1}-½)$$

$$= r_y^2[(x_k+1) + 1]^2 + r_x^2[y_{k+1}-½]^2 - r_x^2 r_y^2$$

$$= p1_k + 2r_y^2(x_k+1) + r_y^2 + r_x^2[(y_{k+1} - ½)^2 - (y_k-½)]^2$$

Where y_{k+1} is either y_k or y_k–1 depending on the sign of $p1_{k.}$

For drawing the ellipse the decision parameters are then incremented by the following amounts–

If $p1_k < 0$ then it is incremented by $2r^2_y x_{k+1} + r^2_y$

Otherwise if $p1_k >= 0$ then it is incremented by $2r^2_y x_{k+1} + r_y{}^2 – 2r^2_x y_{k+1}$

After all the ellipse points has been generated in this way, we shift each pixel to be centered at (x_c,y_c) by $x = x + x_c$; $y = y + y_c$

The **Mid-point Ellipse Algorithm** is as given below:

1) Input the radius r_x, r_y and ellipse center (x_c, y_c) and obtain the first point on an ellipse centered on the origin as: $(x_0, y_0) = (0, r_y)$
2) Calculate the initial value of the decision parameter in region I as: $p1_0 = r^2_y – r^2_x r_y + ¼ r^2x$
3) At each x_k position in region I, starting at k = 0 the following tests are performed. If $p1_k < 0$ the next point is (x_{k+1}, y_k). the decision parameter is then, $p1_{k+1} = p1_k + 2r^2_y x_{k+1} + r^2_y$, else the next point is at $(x_k+1, y_k–1)$ and $p1_{k+1} = p1_k + 2r^2_y x_{k+1} – 2r^2_x y_{k+1} + r^2_y$
   where, $2r_y{}^2 x_{k+1} = 2r^2_y x_k + 2r^2_y$ and
   $2r^2_x y_{k+1} = 2r^2_x y_k – 2r^2_x$
4) Calculate the first value of the decision parameter in the region II using the last point of region I.
   $p2_0 = r^2_y(x_0+½)^2 + r^2_x(y_0–1)^2 – r^2_x r^2_y$
5) In region II at each y_k perform the following tests.
   If $p2_k > 0$, the next point in the ellipse path is $(x_k, y_k–1)$ and
   $p2_{k+1} = p2_k – 2r^2_x y_{k+1} + r^2_x$; else the next point is at $(x_k+1, y_k–1)$.
   $p2_{k+1} = p2_k + 2r^2_y x_{k+1} – 2r^2_x y_{k+1} + r^2_x$
6) Determine the symmetry points in the other three quadrants.
7) Shift each of the pixel position (x, y) on the elliptical path centered on (x_c, y_c) to plot the coordinate values $x = x + x_c$, $y = y + y_c$
8) Repeat the steps for region I until $2r^2_y x >= 2r^2_x y$.



CHECK YOUR PROGRESS

Q.1: Why is it not a brilliant solution to write a simple circle drawing algorithm by solving the equation for y at unit x-intervals using $y = \pm\sqrt{r^2 - x^2}$?

.....

.....

.....

.....

.....

Q.2: Fill in the blanks:

a) Circles have a ..... way symmetry and the ellipses have a ..... way symmetry.

b) In the mid-point circle algorithm, only the points for the ..... of a circle are calculated and then uses its ..... to plot the rest of the circle circumference points.

c) If the circle function returns a negative value then we can say that a point (x, y) is in the ..... and if the point lies outside the circle boundary, then the circle function returns a ..... value.

d) If the pixel at (x_k, y_k) has been plotted then the next step is to choose a point that is nearest to the actual circle path. The two possible points are ..... and .....

e) When the ellipse function returns 0 it means that the point (x, y) lies on the .....

f) The mid-point circle and ellipse algorithms considers the centre to be at the origin (0, 0). If the actual centre is at (x_c, y_C), then all the calculated coordinates (x, y) should be adjusted by adding the offset .....



4.5 LET US SUM UP

- A circle is a symmetrical figure. Circles have eight way symmetry.
- The equation for a circle is: $x^2 + y^2 = r^2$, where, r is the radius of the circle.
- The eight-way symmetry of circles is used in Mid-Point circle algorithm.
- An ellipse is defined as a set of points such that the sum of the distances from two fixed points is a constant.
- Mid-Point ellipse algorithm is a modification of the Bresenham's algorithm and the advantage of this method is that only addition operations are done thereby leading to fast implementation.
- In case of circle and ellipse drawing algorithms, their symmetry is taken into consideration.



4.6 FURTHER READING

1) Hearn, Donald and Baker, M. Pauline. *Computer Graphics*. Prentice-Hall.
2) Newman and Sproull. (1979). *Principles of Interactive Computer Graphics*. McGraw-Hill.
3) Plastock and Kalley. (1986). *Computer Graphics, Schaum's Outlines*. McGraw-Hill.



4.7 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q. No. 1: Using the given equation for y and writing an algorithm for drawing circles is not a brilliant idea because

- Firstly, the resulting circle has large gaps where the slope approaches the vertical
- Secondly, the calculations are not very efficient. Basically the square operations and the square root operations.

Ans. to Q. No. 2: a) 8, 4; b) top right eighth, symmetry; c) circle interior, positive; d) (x_k+1, y_k) and (x_k+1, y_k-1); e) ellipse boundary; f) (x_C, y_C).



4.8 MODEL QUESTIONS

Q.1: Write the Mid-Point circle algorithm.

Q.2: Describe the Ellipse generating procedure.

Q.3: Write the equation of circle and ellipse.

*** ***** ***

UNIT 5: FILLED AREA ALGORITHMS

UNIT STRUCTURE

5.1 Learning Objectives

5.2 Introduction

5.3 Filling Polygons

- 5.3.1 Scan-converting Polygon Filling
- 5.3.2 Inside Outside Tests
- 5.3.3 Boundary Fill Algorithm
- 5.3.4 Area Fill Algorithm
- 5.3.5 Flood Fill Algorithm

5.4 Let us Sum Up

5.5 Further Reading

5.6 Answers to Check Your Progress

5.7 Model Questions

5.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- learn how to fill a region
- learn about different filling techniques
- describe different polygon filling algorithms.

5.2 INTRODUCTION

In the previous unit we have learnt about the circle and ellipse drawing algorithms. In this unit we will learn filling process. Filling is the process of coloring a definite area or region. Regions may be described at the pixel or geometric level. In the first case the region is called interior defined and the collection of algorithms used for filling such a region. The other type of region is called aboundary defined region. For both the region different algorithms are used.

5.3 FILLING POLYGONS

In raster scan systems, there are two approaches in filling polygons.

First Approach:

1) **Polygon Scan Conversion:** Rasterize a polygon scan line by scan line, determining which pixels to fill on each of the line (figure 5.1).

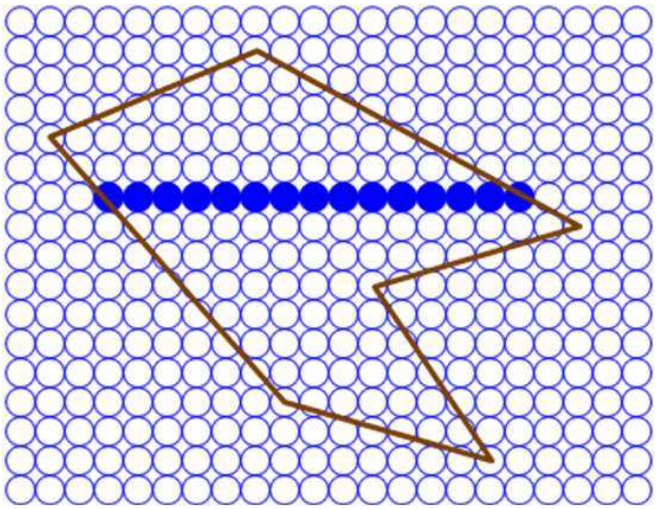


Figure 5.1: Filling using line scan conversion

Second Approach:

2) **Polygon Fill:** Select a pixel inside the polygon and grow in outward direction until the whole polygon is filled. This is depicted in figure 5.2.

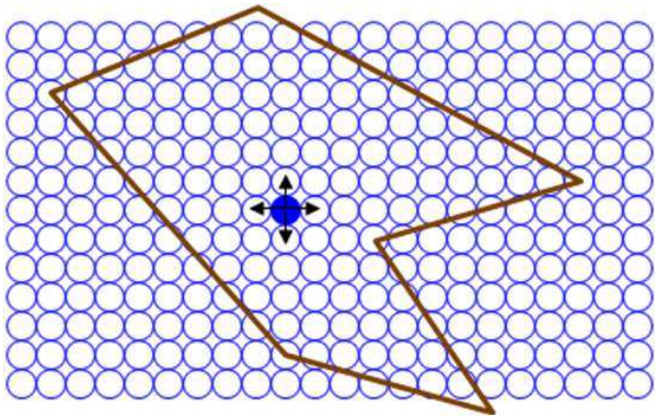


Figure 5.2: Filling using pixel enhancing

5.3.1 Scan-Converting Polygon Filling

Figure 5.3 illustrates the scan-line procedure for filling of polygon areas.

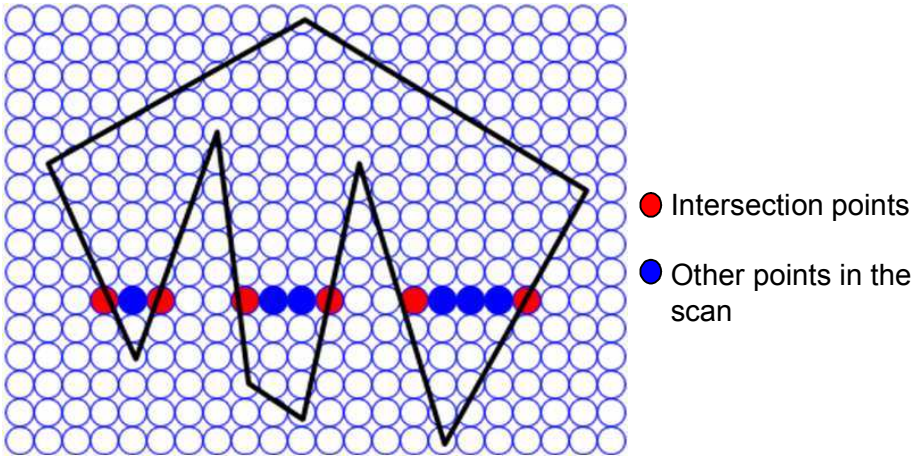


Figure 5.3: Scan line converting filling

The scan conversion algorithm works as follows:

A) Process each of the scan line

- For each scan line crossing a polygon, find out the intersections of the scan line with all the polygon edges.
- Sort the intersections from left to right, that is, sort by the x coordinate.
- The corresponding frame buffer positions between each intersection pair are set to the specified color

B) Some scan line intersections at the polygon vertices require special handling.

- A scan line passing through a vertex intersects two polygon edges at that position, so two points area added to the list of intersection points.

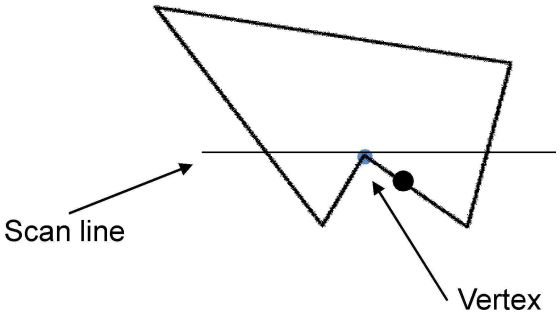


Figure 5.4 : Scan conversion special case

- When a scan line passes through a vertex, it is checked whether the edges meeting at that vertex lie on the same side or on the opposite side to that vertex.

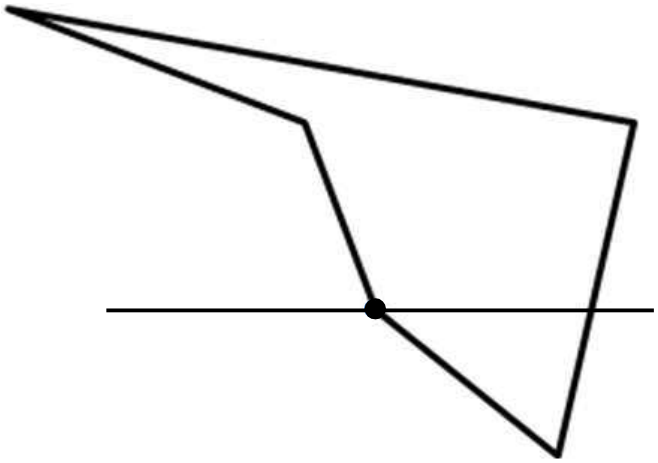
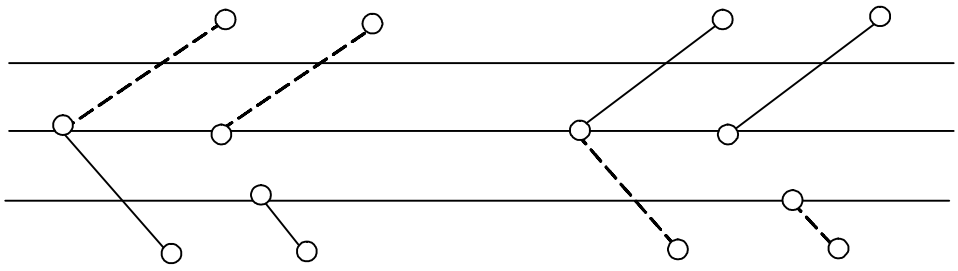


Figure 5.5:

- These vertices can be identified by tracing around the polygon boundary either in clock-wise or anti-clockwise order and then observing the relative changes in the vertex y coordinates as we move from one edge to the next.
  a) If the y coordinate values either decreases or increases only, then the two edges are on the opposite sides. So a single point is added to the list.
  b) If the y coordinate values first decreases and then increases (or vice versa), then the two edges are on the same side. So two points are added to the list.
- An alternative solution to (a) is to shorten some polygon edges to split those vertices.

Proceeding from bottom to top, when the y values are increasing, the y value of upper end of current edge (lower edge) is decreased by 1.



Bottom to top **Top to Bottom**

Figure 5.6:

Proceeding from top to bottom, when the y values are decreasing, the y value of upper end of next edge is decreased by 1.

5.3.2 Inside Outside Tests

To check whether a point lies inside or outside a polygon, there are two rules.

1) **ODD-EVEN RULE (Odd parity rule or Even parity rule):**
   a) A line is drawn from any position P to a distant point outside coordinate extents
   b) Number of edge crossings along that line is counted.
   c) If the number of edge crossed is odd, then P is an interior point. Otherwise P is an exterior point.
   d) For the vertices, the line path chosen should not intersect any vertex of the polygon.
2) **NON-ZERO WINDING NUMBER RULE:** It counts the number of times the polygon edges wind around a particular point in anti-clockwise direction. This count is called *winding number*.
   a) Initialize winding number to 0. Imagine a line drawn from any position P to a distant point beyond coordinate extents. The line must not pass through vertices.
   b) We will trace anti-clockwise the edges of the polygons. As we move along the line from P to the distant point, for every time an edge crosses the line from left to right, we add 1 to the winding number. For every time it crosses from right to left, we substract 1 from the winding number.
   c) If the net value of the winding number is non-zero, P is an interior point. Else it is an exterior point.

      This rule is more effective than the Odd-Even rule.

5.3.3 Boundary Fill Algorithm

A second approach in filling the area of polygons is to use a filling technique, rather than scan-conversion. In this technique, we have to select a point inside the polygon and then fill the neighboring

pixels until the polygon boundary is reached. The boundary is specified in a particular color. The boundary fill algorithm then proceeds outwards pixel by pixel until that boundary color is reached. If the fill color and boundary color is same then a solid color region is displayed.

The **Boundary Fill Algorithm** is as follows :

1) Draw the polygon boundary.
2) Accept as input an interior point (x,y), the fill color and the boundary color.
3) Starting at the interior point, the algorithm tests the neighboring pixels to determine whether they are of the boundary color. If it is not the fill or the boundary color, then,
   - Set this pixel to the fill color
   - Propagate to the pixel's neighbors and continue until all pixels upto the boundary color have been tested.

There are two most frequently used approaches for propagating to neighboring points.

1) Four neighboring points are tested. This method is called 4-connected.

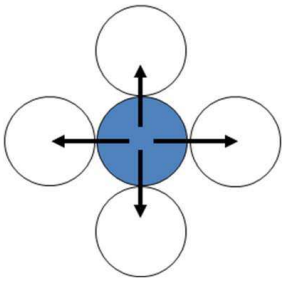


Figure 5.7: Four connected neighbours

2) Eight neighboring points are tested. This method is called 8-connected.

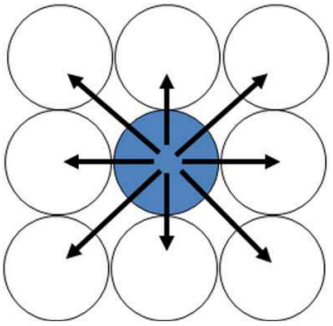


Figure 5.8: Eight connected neighbours

Pseudocode for 4-connected–

```
Procedure boundaryFill4(x,y,fill,boundary : integer)
var current : integer
begin current = getpixel(x,y);// color of the current
pixel
if((current<>boundary) and (current<>fill)) then
    begin
        setpixel(x,y,fill);
        boundaryFill4(x+1,y,fill,boundary);
        boundaryFill4(x-1,y,fill,boundary);
        boundaryFill4(x,y+1,fill,boundary);
        boundaryFill4(x,y-1,fill,boundary);
end     // if statement
end     // end of the procedure
```

For the 8-connected, recursively call the procedure for 8 times.

The 8-connected method is used in more complex figures. The 4-connected algorithm fills the polygons partially, but 8-connected algorithm is better.

The fill algorithm has some problems in case of 4-connected,

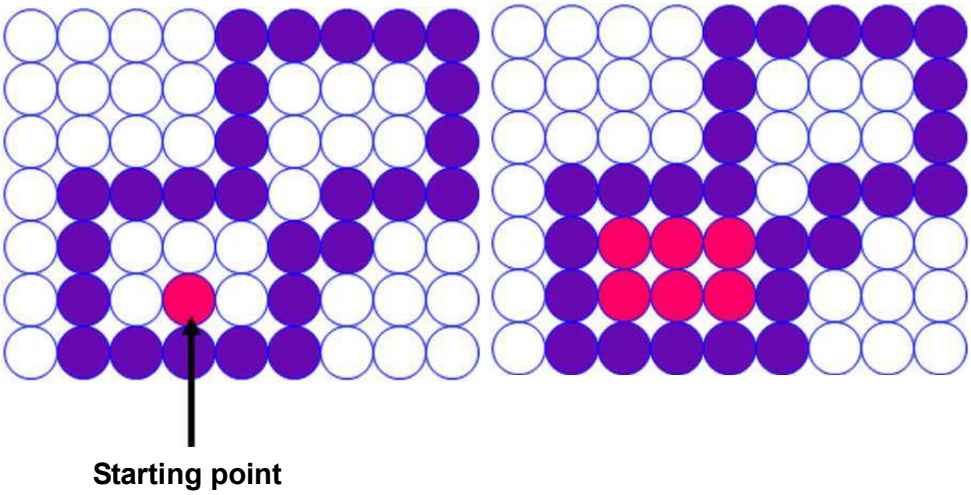


Figure 5.9: Filling using 4-connected

Similarly, the 8-connected can leak over to other polygon.

5.3.5 Flood Fill Algorithm

In the flood filling process, the user will generally privide an initial pixel called the seed. Taking the seed as initial point the algorithm will search each of the sorrunding eight connected pixels to determine whether the extent has been reached. This process is repeated until all pixels inside the region have been processed.

The algorithm is as follows:

1) Select an interior point (x,y) in the polygon.
2) Re-assign all the pixel values currently assigned a given interior color to the specified fill color.
3) If the interior pixels have more than one interior color, then first assign all the pixels to have the same interior color.
4) Using either 4-connected or 8-connected approach, propagate through the pixel positions until all the interior pixels have been repainted with the particular fill color.

The pseudocode for the Flood Fill Algorithm is as follows:

```
Procedure floodFill4(x,y,fillcolor,oldcolor :
integer)
begin
    if  getpixel(x,y) = oldcolor then
    begin
        setpixel(x,y,fillcolor)
        floodFill4(x+1,y,fillcolor,oldcolor);
        floodFill4(x-1,y,fillcolor,oldcolor);
        floodFill4(x,y+1,fillcolor,oldcolor);
        floodFill4(x,y-1,fillcolor,oldcolor);
    end
    end
```



CHECK YOUR PROGRESS

Q.1: State whether the following are True or False:

a) Scan conversion approach is not used for filling areas of polygon.

b) In the scan-converting polygon filling approach, the number of intersecting points of the edges are counted.

c) If the edges lie on the same side of the vertex through which the scan-line passes, then 2 points are added to the list of intersecting points.

d) If the edges lie on the opposite sides of the vertex, then proceeding from bottom to top, if the y values is increasing then the y value of the upper end of the current edge is decreased by 1.

e) In the Odd-Even rule, the line drawn from a point P to a distant point should always intersect the vertices of the polygon.

f) In the Fill algorithms, 5-connected and 9-connected methods are used to propagate to the neighboring points.



5.4 LET US SUM UP

- There are two methods of polygon filling : Polygon Scan Conversion and Polygon Filling
- To test whether a point lies inside or outside the polygon region there are two tests: Odd-Even Rule and the Non-Zero winding number rule.
- In Boundary fill algorithm, a point inside the polygon is selected and then the neighboring pixels are filled with the fill color until the polygon boundary is reached.
- In Flood fill algorithm, first the pixels inside the polygon are set to the same interior color and then these pixels are assigned the specified fill color.



5.5 FURTHER READING

1) Hearn, Donald and Baker, M. Pauline. *Computer Graphics*. Prentice-Hall.
2) Newman and Sproull. (1979). *Principles of Interactive Computer Graphics*. McGraw-Hill.
3) Plastock and Kalley. (1986). *Computer Graphics, Schaum's Outlines*. McGraw-Hill.



5.6 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q. No. 1: a) False; b) True; c) True; d) True; e) False; f) False



5.7 MODEL QUESTIONS

Q.1: Discuss the scan-line procedure for filling of polygon areas.

Q.2: What are the rules for checking whether a point lies inside or outside a polygon boundary?

Q.3: Discuss the Boundary fill and Flood-fill algorithms.

*** ***** ***

UNIT 6: 2-D TRANSFORMATIONS

UNIT STRUCTURE

6.1 Learning Objectives

6.2 Introduction

6.3 2-D Transformation

- 6.3.1 Translation
- 6.3.2 Rotation
- 6.3.3 Scaling
- 6.3.4 Reflection
- 6.3.5 Shear

6.4 Homogeneous Coordinate Representation

6.5 Let Us Sum Up

6.6 Further Reading

6.7 Answers to Check Your Progress

6.8 Model Questions

6.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- learn about different transformations like translation, rotation, scaling, reflection etc.
- learn how the transformations are done in graphics
- define 2D and 3D transformations using homogeneous coordinate system
- learn about projections in computer graphics.

6.2 INTRODUCTION

In the previous unit we have discussed the different polygon filling algorithms. We now know how to use boundary fill algorithm and what the flood fill algorithm does.

In computer graphics, animations are done on the screen using the different transformation techniques. Graphics is now used almost everywhere

and it is rapidly growing. The two dimensional (2D) and three-dimensional (3D) graphics are used extensively by the interactive softwares in business and education.

In this unit, we will introduce you to the transformation techniques of the pictures and graphs. Also we will be discussing how to project the objects of a scene onto another coordinate view plane.

6.3 2D-TRANSFORMATION

A transformation is an operation on a point (x, y) that maps the point's coordinate into a new coordinate (x_1, y_1). Transformations are important in computer graphics.

The basic geometric transformations are:

- Translation
- Rotation
- Scaling

Other transformations are:

- Reflection
- Shear

Let us now discuss how these transformations are applied to reposition and re-size the 2D-objects.

6.3.1 Translation

Repositioning an object along a straight line path from one coordinate location to another coordinate location is called *translation transformation*.

A point (x, y) is translated to a new location (x', y') by adding the translation parameters t_x and t_y to the original coordinates x and y.

$$x' = x + t_x$$

$$y' = y + t_y$$

The distance pair (t_x, t_y) is called transformation vector or shift vector.

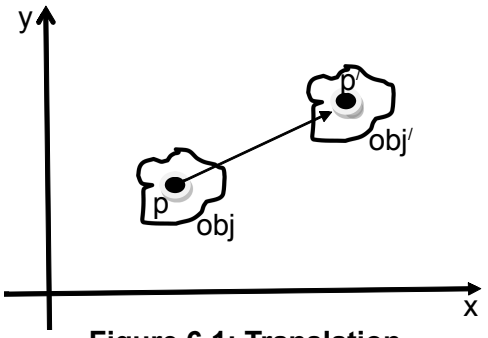


Figure 6.1: Translation

In matrix form, $P^{/} = P + T$

where, $P^{/} = \begin{bmatrix} x^{/} \\ y^{/} \end{bmatrix}$, $P = \begin{bmatrix} x \\ y \end{bmatrix}$, $T = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$

- Translation is a rigid body transformation which moves the objects without deformation.
- To translate a straight line, the translation is first applied to the two endpoints of the line. Then a line is redrawn between the two new endpoints.
- In case of curves or polygons also similar method is used.

3.3.2 Rotation

To reposition an object along a circular path in the xy plane, the 2D-rotation is used. In rotation, the object is rotated through Φ^0 about the origin.

There is a rotation angle θ and a point (x_r, y_r) called the rotation point or the pivot point about which the object is to be rotated.

If θ > 0 (positive angle), then the rotation takes place in anti-clockwise direction about the pivot point.

If θ < 0(negative angle), then the rotation takes place in clockwise direction about the pivot point.

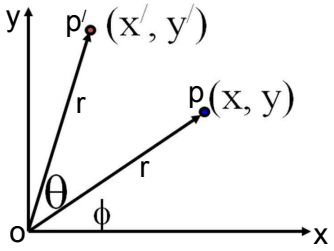


Figure 6.2: Rotation of point p

Transformation of point P when the rotation point is at the origin:

Using trigonometric identities, we get,

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

$= r\cos\Phi\cos\theta - r\sin\Phi\sin\theta$

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

$= r\cos\Phi\sin\theta + r\sin\Phi\cos\theta$

The original coordinates of the point (x,y) making an angle Φ with the x-axis are:

$x = r\cos\Phi$

$y = r\sin\Phi$

Therefore, we get, $x' = x\cos\theta - y\sin\theta$

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

In matrix form, $P' = R.P$

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

Rotation does not change the shape or size of the objects as every point on the object is rotated through the same angle.

6.3.3 Scaling

Scaling means changing the size of an object. It is carried out by multiplying the coordinate values (x, y) of the vertex of the polygons by the scaling factors s_x and s_y. The new transformed coordinates of the object (x', y') are thus obtained as follows:

$$\mathbf{x' = x.s_x}$$

$$\mathbf{y' = y.s_y}$$

The scaling factor s_x scales the objects in the x-direction and the scaling factor s_y scales in the y-direction.

In matrix form we can write as: $P' = S.P$

Where, $P' = \begin{pmatrix} x' \\ y \end{pmatrix}$, $S = \begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix}$, $P = \begin{pmatrix} x \\ y \end{pmatrix}$

- Positive values of s_x and s_y increases the object's size whereas negative values decreases its size.

- When $s_x = s_y$, the scaling is uniform.
- When $s_x \neq s_y$, the scaling is differential.

The objects when are scaled are transformed and repositioned. To make the position of the object not changed after scaling, we can choose a point as the Fixed Point and scale the objects relative to that fixed point. This fixed point can be any vertex of the polygon or any other point. The fixed point is say (x_f, y_f).

The coordinate of the transformed point is then,

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

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

Scaling is not a rigid body transformation as it changes the size of the objects.



(a) Squate (b) transformed into a rectangle with $s_x = 2$ and $s_y = 1$

Figure 6.3: Scaling

6.3.4 Reflection

Reflection produces a mirror image of an object. An object is rotated at a particular angle about the axis of reflection to get the reflected image.

For 2D reflection, the object is rotated about the axis of rotation in 180^0.

A) Reflection about the x-axis (y = 0): This is obtained by keeping the x-coordinate of a point unchanged and inverting the y-coordinate.

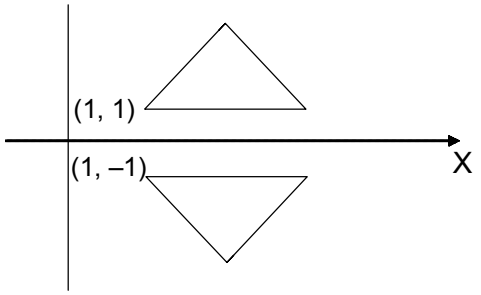


Figure 6.4: Reflection about the x-axis

The transformation matrix is as follows:

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

B) Reflection about the y-axis (x = 0): This is obtained by keeping the y-coordinate of a point unchanged and inverting the x-coordinate.

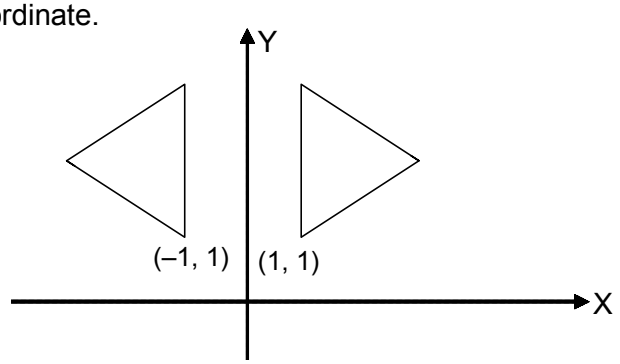


Figure 6.5: Reflection about the y-axis

The transformation matrix is as:

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

C) Reflection about an axis perpendicular to the xy-plane: This is accomplished by inverting both the x and y coordinate values.

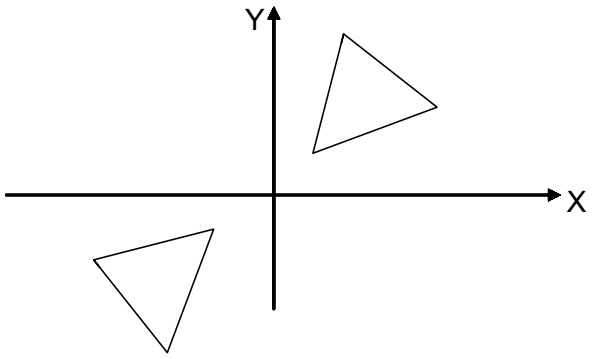


Figure 6.6: Reflection about an axis perpendicular to the xy-plane

The transformation matrix is as:

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

D) Reflection about a line where y = x: This can be obtained by interchanging the x and y coordinate values.

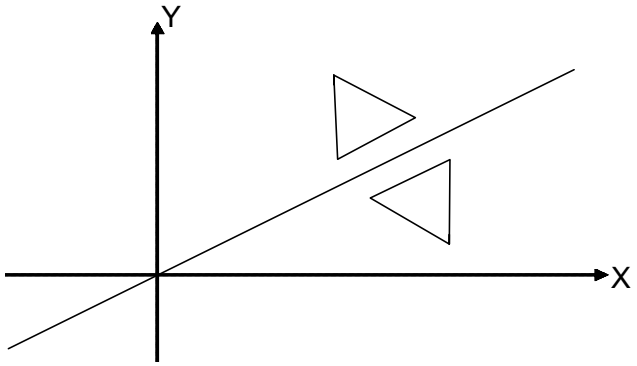


Figure 6.7: Reflection about a line where y = x

The line y = x passes through the origin (0, 0).

The transformation matrix is as:

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

E) Reflection about a line where y = –x. This can be obtained by exchanging and inverting the x and y coordinate values.

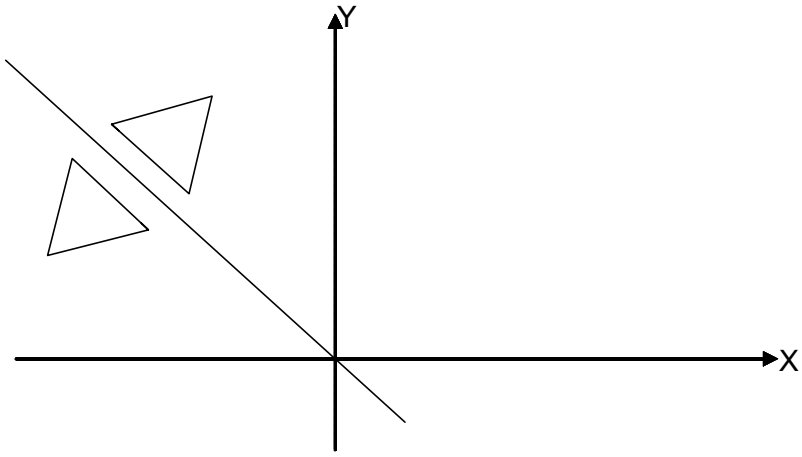


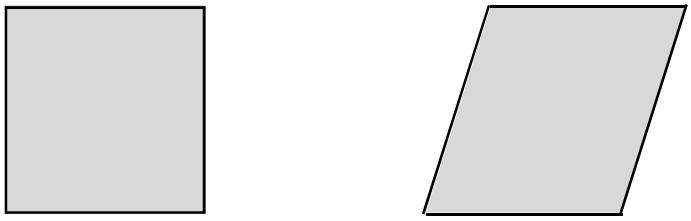
Figure 6.8: Reflection about a line where y = –x

The transformation matrix is as:

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

6.3.5 Shear

This kind of transformation distorts the shape of the objects and they appears to look as if the internal layers are caused to slide over each other. The object is pulled horizontally by different amounts.



Original After Shear

Two common shear operation are those that :

a) Shift the x-coordinate values

b) Shift the y-coordinate values.

- A x-direction shear relative to the x-axis is given by the following matrix:

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

Where $x' = x + sh_x y$

$y' = y$

The sh_x above is the shear parameter whose value can be any real number either positive or negative.

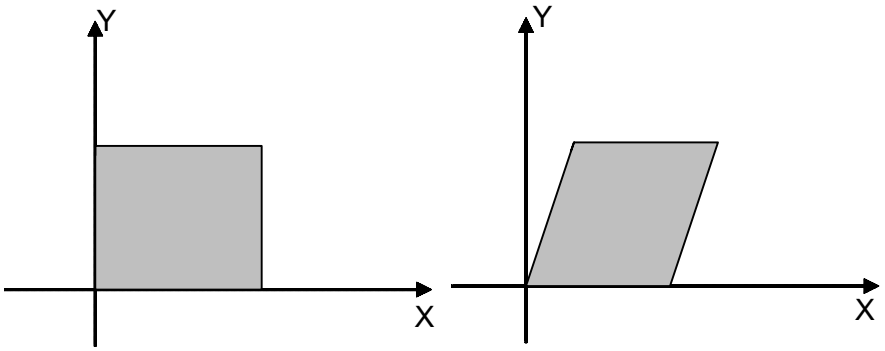


Figure 6.9: Unit Square transformed to a parallelogram

- A x-direction shear relative to other reference lines ($y = y_{ref}$) is as shown below:

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

where the coordinate positions are transformed as:

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

$y' = y$

Example: A square of unit size can be transformed to a parallelogram with $sh_x = 2$

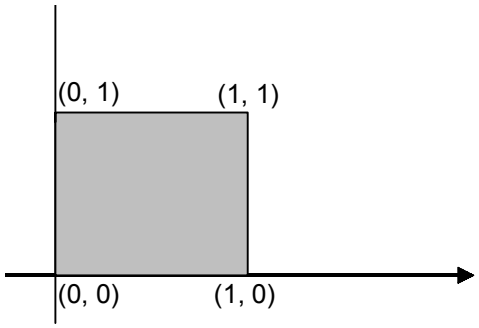


Figure 6.10:

Square: Here, the transformation of the square must be in reference to the x-axis and not to any other reference line. So in this example, $y_{ref} = 0$

Now, in case of x-direction shear relative to the x=axis, we have

$x' = x + sh_x y$

$y' = y$

Therefore, when x = 0, y = 0

$x' = 0 + 2(0) = 0$

$y' = 0$

So after shear the new coordinate value is (0, 0)

for the point (1,0) i.e. when x = 1, y = 0

$x' = 1 + 2(0) = 1$

$y' = 0$

The new coordinate value is (1, 0)

when x = 1, y = 1

$x' = 1 + 2(1) = 3$

$y' = 1$

After shear the new coordinate value is (3, 1)

Similarly for the point (0, 1)

$x^{/} = 0 + 2(1) = 2$

$y^{/} = 1$

After shear the new coordinate value is (2, 1)

Hence from this we can know draw the parallelogram as follows:

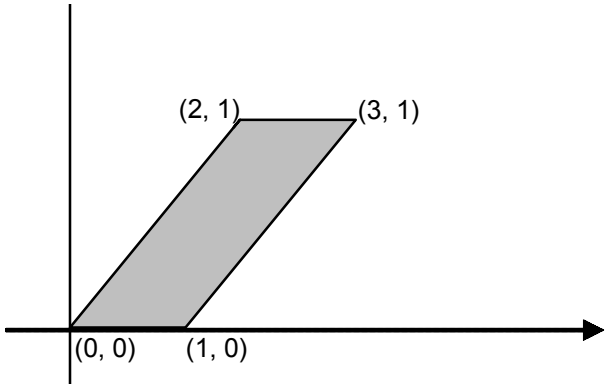


Figure 6.11: Parallelogram

- Any directional shear relative to $x = x_{ref}$ is give by the following matrix:

$$\begin{bmatrix} 1 & 0 & 0 \\ sh_y & 1 & -sh_y x_{ref} \\ 0 & 0 & 1 \end{bmatrix}$$

where, $x^{/} = x$

$y^{/} = sh_y(x - x_{ref}) + y$



CHECK YOUR PROGRESS

Q.1: Fill in the blanks:

a) A transformation is an operation on a point that maps the point's coordinates (x, y) to a new set of .....

b) Reflection is a .....

c) If the rotation angle θ is positive, then the rotation takes place in ..... about the pivot point.

d) For 2D reflection, the object is rotated about the axis of rotation in .....

e) Reflection about the y-axis is obtained by keeping the y-coordinate of a point ..... and inverting the .....

f) In Shear, the object is pulled ..... by different amounts so as to look like the internals layers ..... each other.

g) The common shear operations are shifting the ..... and the .....

h) The scaling factor s_x scales the objects in the ..... and the scaling factor s_y scales in the .....

i) Scaling is not a ..... transformation.

Q.2: Say True or False:

a) Rotation changes the size of objects.

b) Rotation is used to reposition an object along a circular path.

c) Reflection produces a mirror image of an object.

d) Shear changes the shape of an object and distorts it.

e) Scaling retains the original size of the objects.

f) Positive values of s_x and s_y increases the object's size whereas negative values decreases its size.

6.4 HOMOGENEOUS COORDINATE REPRESENTATION

The homogeneous coordinate system is used in projective geometry much as cartesian coordinate system is used in euclidean geometry. Homogeneous coordinates provides a method for doing calculations and proving theorems in projective geometry when used in practical applications.

This homogeneous coordinates provides a method for performing certain operations on points in euclidean space by means of matrix multiplications. They have the advantage that the coordinates of a point even those at infinity can be represented using finite coordinates. They are

used in computer graphics where they allow transformations to be easily represented by a matrix.

The usual cartesian coordinates for a point consists of a list of n points where n is the dimension of the space. Now, the homogeneous coordinates for the same point requires n+1 coordinates. So we add a coordinate to the end of the list, and make it equal to 1.

Thus the two dimensional point (x, y) becomes (x, y, 1) in homogeneous coordinates system. Similarly, the three dimensional point (x, y, z) becomes (x, y, z, 1) in homogeneous coordinates.

The most common use of homogeneous coordinates is for one, two and three dimensional spaces. The final coordinate is the homogeneous parameter called "h" and it will not interfere with the normal x, y and z-coordinates.

In homogeneous coordinates, we can express all transformations as matrix multiplications. The coordinates are represented with three element column vectors and the transformation operations are written as 3x3 matrices.

For Rotation with respect to the origin,

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

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

The rotation parameter, R(θ) is a 3x3 matrix and its inverse is obtained by replacing θ by – θ.

For Scaling,

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

The inverse of scaling is obtained by replacing s_x, s_y by $1/s_x$ and $1/s_y$

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



CHECK YOUR PROGRESS

Q.3: Fill in the blanks:

a) A point (x, y) has the homogeneous coordinates .....

b) ..... is repositioning an object along a straight line path from one coordinate location to another coordinate location.

c) Translation is a rigid body transformation which moves the objects without .....

d) 2D transformations are represented by ..... matrices using homogeneous coordinates.

Q.4: Derive the transformation that rotates an object point θ^0 about the origin. Write the matrix representation for this rotation.

.....

.....

.....

.....

.....

.....

.....



6.5 LET US SUM UP

- A transformation is any operation on a point in space (x, y) that maps the point's coordinates into a new set of coordinates (x1, y1)
- The geometric transformations are *translation*, *rotation*, *scaling*, *reflection* and *shear*.
- Rotation is repositioning an object along a circular path.
- Reflection produces a mirror image of an object. For 2D reflection, the object is rotated about the axis of rotation in 180^0.

- Shear distorts the shape of the objects and it seems the object is pulled horizontally in different amounts.
- Scaling means changing the size of an object. It is carried out by multiplying the coordinate values (x, y) of the vertex of the polygons by the scaling factors s_x and s_y.
- Positive values of s_x and s_y increases the object's size whereas negative values decreases its size.
- When $s_x = s_y$, the scaling is uniform and the original shape of the object is maintained.
- 2D transformations are represented by 3x3 matrices using homogeneous coordinates.



6.6 FURTHER READING

1) Hearn, Donald and Baker, M. Pauline. *Computer Graphics*. Prentice-Hall
2) Newman and Sproull. (1979). *Principles of Interactive Computer Graphics*. McGraw-Hill.
3) Plastock and Kalley. (1986). *Computer Graphics, Schaum's Outlines*. McGraw-Hill.



6.7 ANSWER TO CHECK YOUR PROGRESS

Ans. to Q. No. 1: a) coordinates (x1, y1); b) geometric transformation; c) anti-clockwise direction; d) 180^0; e) unchanged, x-coordinate; f) horizontally, slides over; g) x-coordinate values, y-coordinate values; h) x-direction, y-direction; i) rigid body

Ans. to Q. No. 2: a) False; b) True; c) True; d) True; e) False; f) True

Ans. to Q. No. 3: a) (x, y, 1); b) Translation; c) Deformation; d) 3 × 3; e) 4 × 4;

Ans. to Q. No. 4: From figure 6.11 the definition of the trigonometric functions sin and cos can be written as:

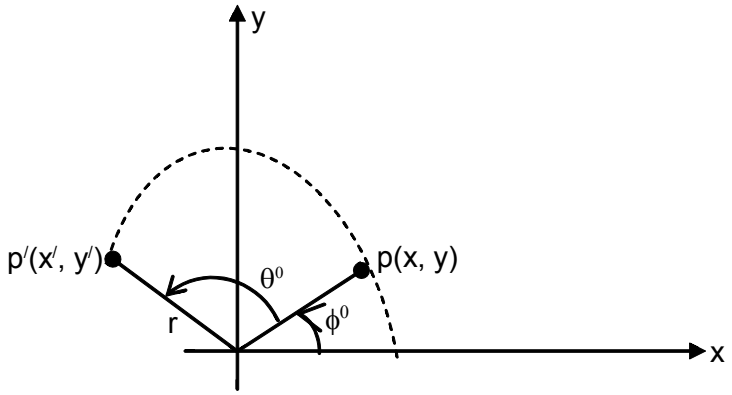


Figure 6.12: Point p is rotated θ^0

$x^/ = r\cos(\theta + \phi)$ $y^/ = r\sin(\theta + \phi)$

and $x = r\cos\phi$ $y = r\sin\phi$

Using trigononometric identities, we obtain,

$r\cos(\theta + \phi) = r(\cos\theta\cos\phi - \sin\theta\sin\phi) = x\cos\theta - y\sin\theta$

and $r\sin(\theta + \phi) = r(\sin\theta\cos\phi + \cos\theta\sin\phi) = x\sin\theta + y\cos\theta$

or $x^/ = x\cos\theta - y\sin\theta$ $y^/ = x\sin\theta + y\cos\theta$

Writing, $P^/ = \begin{pmatrix} x^/ \\ y^/ \end{pmatrix}$, $P = \begin{pmatrix} x \\ y \end{pmatrix}$ and $R_\theta = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$

We can write now $P/ = R_\theta . P$



6.8 MODEL QUESTIONS

Q.1: Discuss the 2D planar geometric transformations.

Q.2: What is 2D reflection? Discuss the x-axis and y-axis reflection.

Q.3: Write the matrix representations of the 2D transformation.

*** ***** ***

UNIT 7: 2-D VIEWING

UNIT STRUCTURE

7.1 Learning Objectives

7.2 Introduction

7.3 2-D Viewing Pipeline

7.4 Viewing Coordinate Reference Frame

7.5 Window-To-Viewport Coordinates Transformation

7.6 Two-Dimensional Viewing Functions

7.7 Let Us Sum Up

7.8 Further Reading

7.9 Answer to Check Your Progress

7.10 Model Questions

7.1 LEARNING OBJECTTIVES

After going through this unit, you will be able to:

- give the definition of window and viewport
- learn about viewing pipelines
- describe viewport transformation
- learn about world coordinates and normalized coordinates
- learn about 2-D viewing functions.

7.2 INTRODUCTION

When we consider the mechanism to display views on an output device, graphics packages allows a user to specify an image to be displayed and where that part of the image to be displayed on the device. For two dimensional displays a user can choose a particular subarea of the whole image. A user can select single area or multiple areas for simultaneous display or for an animated panning sequence across the scene. The image parts within the selected areas are mapped onto specified areas of the device coordinates. When multiple view area are selected, these area can be placed in separate display locations where some areas would be inserted

into other, larger display areas. Transformation from the real world to device coordinates involve translation, scaling and rotation operations and some procedures which delete the parts of the picture which are outside the limits of display device.

Displaying an image of a picture involves mapping the coordinates of the points and lines that form the picture into the appropriate coordinates on the device or computer where the image to be displayed. This is done through the use of coordinate transformations known as ***viewing transformations***.

7.3 2D VIEWING PIPELINE

A world coordinate area that is selected for display is called window and an area of display device where window is mapped is called *viewport*. The window defines 'what' is to be displayed and viewport defines 'where' to be displayed. Normally, we consider window and viewport both are rectangular in shape, with standard position and rectangular edges are parallel to the coordinate axes. Polygon, circles are considered as shape of the window or viewport in some special applications. In general, the mapping of the world-coordinate scene to device coordinates is referred to as a **viewport transformation.** Two-dimensional viewing transformation is simply referred to as **window-to-viewport transformation or windowing transformation.** Figure 7.1 shows the World coordinates and viewport.

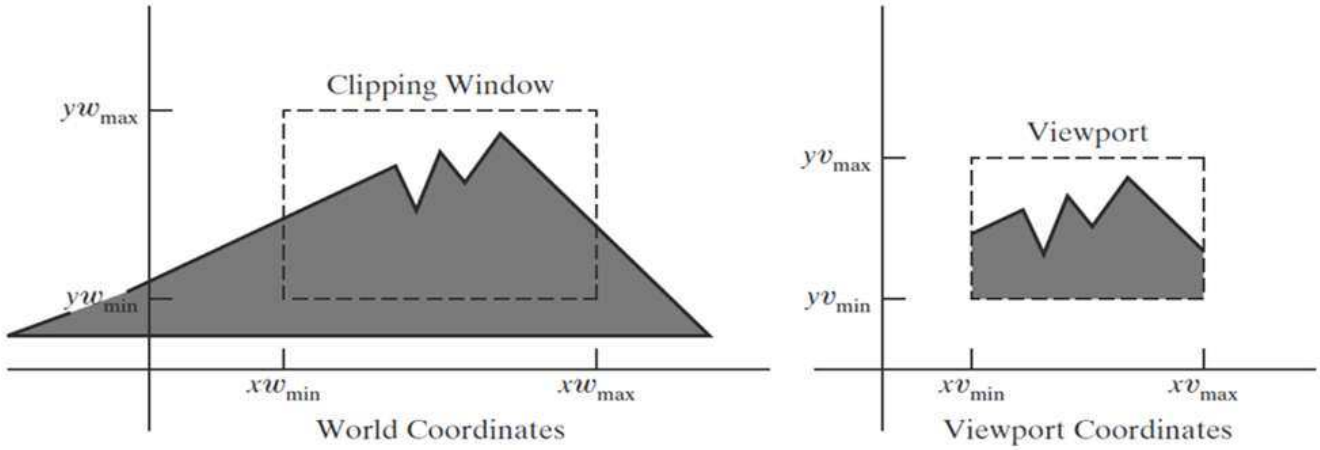


Figure 7.1: World coordinates and viewport

Viewing transformation carry out the several steps. First we construct world coordinate scene using output primitives and attributes and next we can set up a two dimensional coordinate system in the world coordinate plane to obtain the orientation for the window. The viewing reference frame is used to provide a method for setting up arbitrary orientation for rectangular windows. When the reference frame is established, we can transform the object in world coordinates to viewing coordinates. Then we define a viewport in normalized coordinates(in the range from 0 to 1) and mapped the viewport coordinates of the scene to normalized coordinates. And in the final step, the parts of the picture lie outside the viewport are clipped. Then the viewport contents are transferred to the device coordinates.

By changing the position of the viewport, we can display the object at different position of display device and if we varying the size of viewport, we can change the proportion and size of the displayed object. By successive mapping the different sized window on the fixed sized viewport, we can achieve the zooming effect.

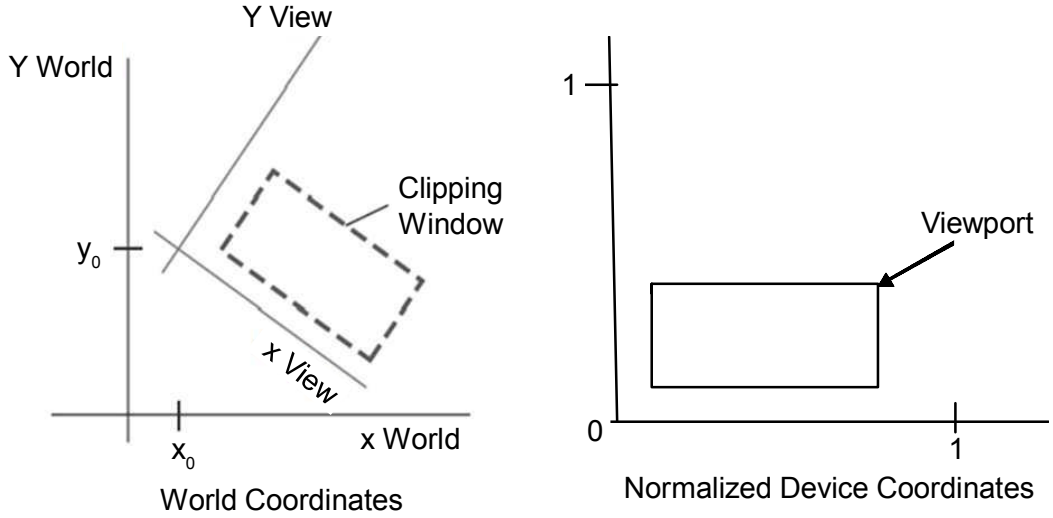


Figure 7.2: Normalized device coordinates

The term viewing pipeline describes a series of transformations, which are passed by data position to end up as image data displayed on a device. The two-dimensional viewing pipeline describes this process for two-dimensional data:

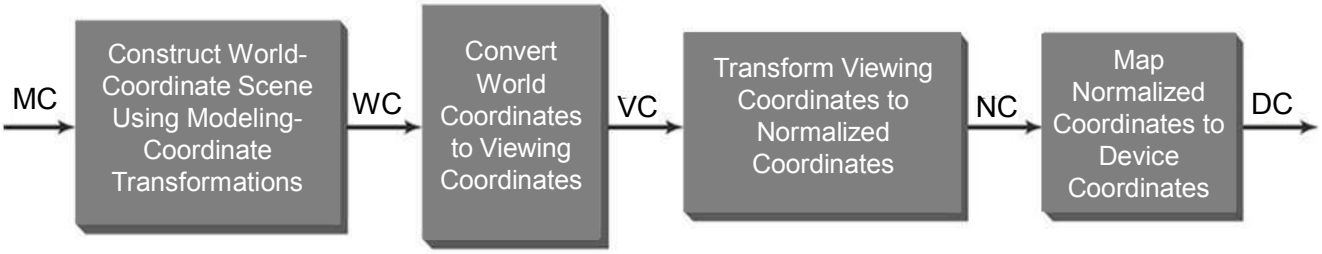


Figure 7.3: Two-dimensional viewing pipeline

The coordinates which describes the position of individual objects (models) are created are called model (or object) coordinates. When more objects are assembled into a scene, they are described by world coordinates. After transformation into the coordinate system of the viewer they become viewing coordinates. Their projection onto a window yields device-independent normalized coordinates. Finally, after mapping those normalized coordinates to a specific device, we get device coordinates.



CHECK YOUR PROGRESS

Q.1: The process of selecting and viewing the picture with different views is called .....

a) Windowing b) clipping

c) both (a) and (b) d) projecting

Q.2: In two dimensional viewing, we have .....?

a) 2D window, 3D viewport c) 2D window, 2D viewport

b) 3D window, 2D viewport d) 3D window, 3D viewport

Q.3: Define viewport and window.

.....

.....

.....

.....

Q.4: What is windowing transformation?

.....

.....

.....

.....

7.4 VIEWING COORDINATE REFERENCE FRAME

It is a reference frame which specifies the world coordinate window. First we select a viewing coordinate origin at some world position P=(x_0,y_0). then we establish the rotation, or orientation, of this reference frame. For this we specify a world vector V that defines the viewing y_v direction. Vector V called as **view up vector**.

We can calculate the components of unit vector v = (v_x, v_y) and u = (u_x, u_y) for the viewing y_v and x_v axes respectively, for the given V. These unit vectors make the first and second rows of the rotation matrix R which is aligns the viewing x_v, y_v axes with the world x_w, y_w axes.

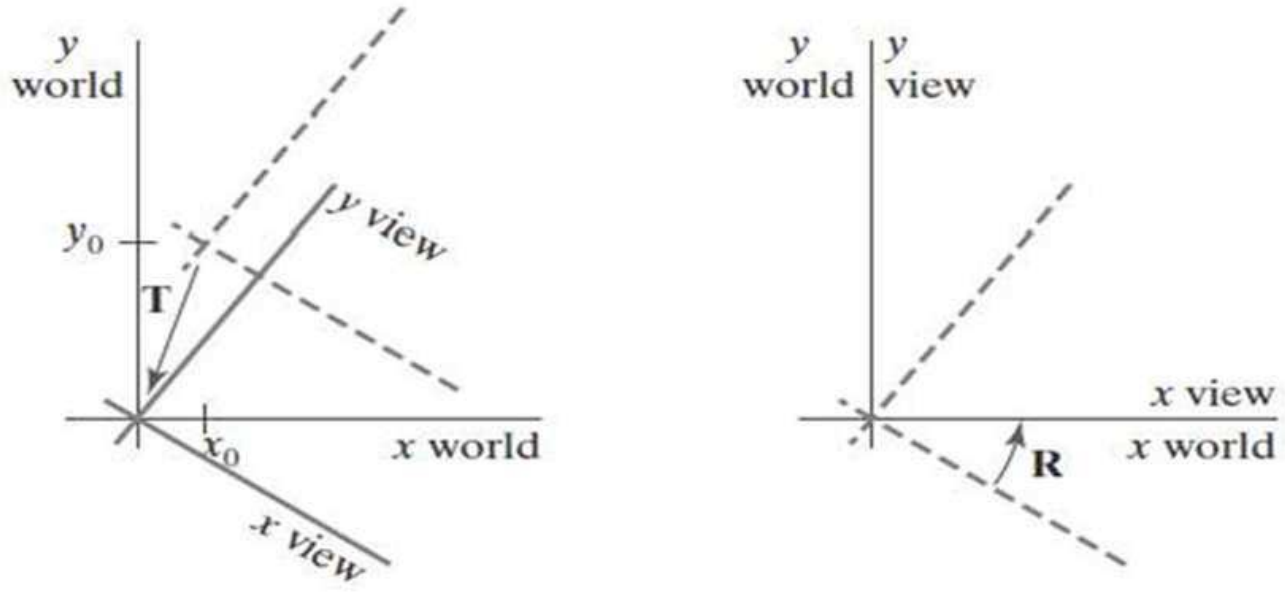


Figure 7.4:

Using two steps composite transformation, we obtain the matrix for converting world coordinate position to viewing coordinates

1) Translate the viewing origin to world origin.
2) Rotate to align two coordinate reference frames..

So composite transformation is:

$$\mathbf{M_{wc,\,vc} = R.T} \qquad (7\text{-}1)$$

Where T is the Translation matrix that takes the viewing origin point P_0 to the world origin and R is the rotation matrix that align the axes of both reference frame.

7.5 WINDOW-VIEWPORT COORDINATE TRANSFORMATION

A window-viewport transformation describes the mapping of a (rectangular) window in one coordinate system into another (rectangular) window in another coordinate system. This transformation is defined by the section of the original image that is transformed (clipping window), the location of the resulting window (viewport), and how the window is translated, scaled or rotated. The following derivation shows how easy this transformation generally is (i.e., without rotation):

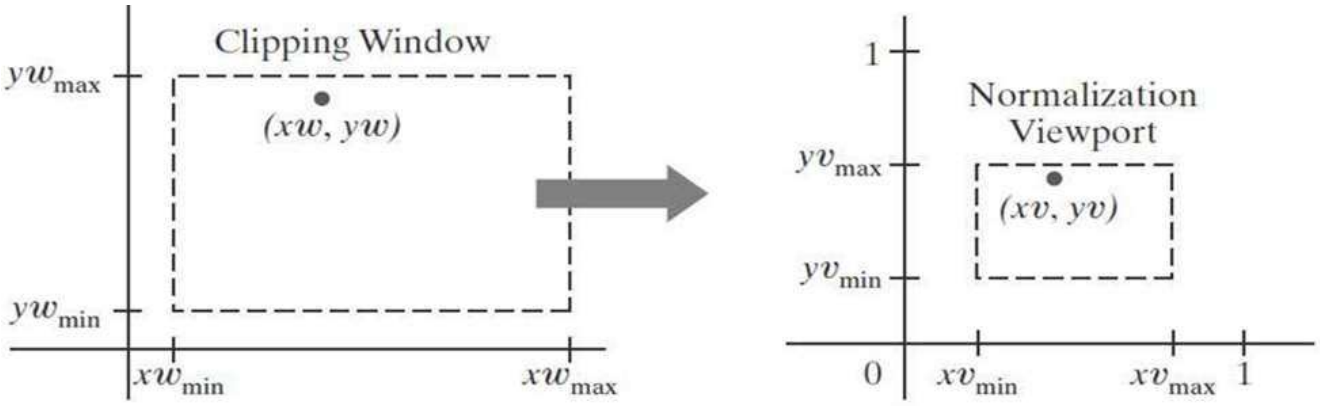


Figure 7.4: Normalized viewport

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

$$\frac{yv - yv_{min}}{yv_{max} - yv_{min}} = \frac{yw - yw_{min}}{yw_{max} - yw_{min}} \qquad (7\text{-}2)$$

By solving this expression for the viewport position (xv,yv), we have,

$$xv = s_x xw + t_x$$

$$yv = s_y yw + t_y \quad (7\text{-}3)$$

Where scaling factors are:

$$s_x = \frac{xv_{max} - xv_{min}}{xw_{max} - xw_{min}}$$

$$s_y = \frac{yv_{max} - yv_{min}}{yw_{max} - yw_{min}} \qquad (7\text{-}4)$$

And translation factors are:

$$t_x = \frac{xw_{max}xv_{min} - xw_{min}xv_{max}}{xw_{max} - xw_{min}}$$

$$t_y = \frac{yw_{max}yv_{min} - yw_{min}yv_{max}}{yw_{max} - yw_{min}} \qquad (7\text{-}5)$$

Equation 7-3 can also be derived with a set of transformations that converts the window area into the viewport area. This conversion is performed with the following sequence of transformations:

1) Perform the scaling transformation using a fixed point position of (xw_{min}, yw_{min}) that scale the window area to the size of the viewport.
2) Translate the scaled window area to the position of the viewport.

If the scaling factors are equal (sx = sy), then the relative proportion of objects are maintained. Otherwise world objects will be constructed in either x or y direction when displayed on output device.

7.6 TWO-DIMENSIONAL VIEWING FUNCTIONS

We define here a viewing reference system in a PHIGS application program with the following function:

1) evaluateViewOrientation Matrix(x0, y0, xv, yv, error, viewMatrix)
it calculate viewmatrix for a world-to-viewport transformation matrix where, x0, y0 are the coordinates of the viewing origin,
xv, yv are the world coordinate position for view up vector
an integer error code is generated if the input parameters are in error otherwise it produce the output.

2) evaluateViewMappingMatrix(xwmin, xwmax, ywmin, ywmax, xvmin, xvmax, yvmin, yvmax,error, viewMappingMatrix)
it sets the elements of the window-to-viewport mapping matrix:
where, xwmin, xwmax, ywmin, ywmax are the parameters for the window limits in viewing coordinates;
xvmin, xvmax, yvmin, yvmax aresets the limits of viewport with normalized coordinate position.

3) set ViewRepresentation (ws, viewIndex, viewMatrix, viewMappinMatrix, xclipmin, xclipmax, yclipmin, yclipmax, clipxy)

with this function, we can store combinations of viewing and window-to viewport mappings for various workstations in a viewing table. Where parameter ws assigned for workstation, parameter viewIndex sets an integer identifier for this particular window-viewport pair. viewMatrix and viewMappingMatrix are concatenated and referenced by viewIndex. Parameter clipxy is assigned to either the value noclip or the value clip. It allows us to turn off the clipping if we want to see the scene outside the viewport. Parameters xclipmin, xclipmax, yclipmin, yclipmax are define the additional clipping limits otherwise viewport boundaries define the clipping limits.

4) setViewIndex (viewIndex)

it select a particular set of options from the viewing table. The viewIndex selection is then applied to subsequently specified output primitives and associated attributes and generates a display on each of the active workstation.

5) At the final stage, we apply a workstation transformation by selecting a workstation window-viewport pair:

setWorkstationwindow(ws, xwsWindmin, xwsWindmax, ywsWindmin, ywsWindmax)

setWorkstationViewport(ws, xwsVPortmin, xwsVPortmax, ywsVPortmin,ywsVPortmax)

where parameter ws gives the workstation number, window coordinate extents are specified in normalized space (form 0 to 1), and the viewport limits are in integer device coordinates.



CHECK YOUR PROGRESS

Q.5: What is view up vector?

.....

.....

.....

.....

Q.6: How can we get normalized viewport?

.....

.....

.....

.....

Q.7: What is the use of setViewIndex() function?

.....

.....

.....

.....



7.7 LET US SUM UP

- The world coordinate system (WCS) is a right handed Cartesian coordinate system where we describe the picture that is to be displayed.
- The normalized device coordinate system is a right handed coordinate system in which the display area of the virtual display device correspond to the unit (1 × 1) square whose lower left hand corner is at the origin of the coordinate system.
- Normalization transformation maps world coordinates to normalized device coordinates.
- Workstation transformation maps normalized device coordinates to physical device coordinates.
- Different types of PHIGS functions used in 2D viewing.



7.8 FURTHER READING

1) Dam, Andries Van; John, F. Hughes; Foley, James D.; Feiner, Steven K. (2002); *Computer Graphics Principles and Practice in C*. Pearson Education.

2) Hearn, Donald; Baker, M. Pauline. (2nd Edition). *Computer Graphics*. Prentice-Hall.



7.9 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q. No. 1: (a)

Ans. to Q. No. 2: (b)

Ans. to Q. No. 3: A world coordinate area that is selected for display is called window and an area of display device where window is mapped is called viewport. The window defines 'what' to be displayed and viewport defines 'where' to be displayed.

Ans. to Q. No. 4: The mapping of the world-coordinate scene to device coordinates is referred to as a viewport transformation. Two-dimensional viewing transformation is simply referred to as window-to-viewport transformation or windowing transformation.

Ans. to Q. No. 5: To establish a reference frame first we select a world coordinate let p = (x, y), then we establish the rotation, or orientation of this reference frame. For this we specify a world vector V that defines the viewing y_v direction. Vector V called as **view up vector**

Ans. to Q. No. 6: This transformation is defined by the section of the original image that is transformed (clipping window), the location of the resulting window (viewport), and how the window is translated, scaled or rotated. If the range of viewport define from 0 to 1, then its called as normalized viewport.

Ans. to Q. No. 7: it select a particular set of options from the viewing table. The viewIndex selection is then applied to subsequently specified

output primitives and associated attributes and generates a display on each of the active workstation.



7.10 MODEL QUESTION

Q.1: Describe the Transformations used in 2-D viewing ?

Q.2: What is normalized coordinates?

Q.3: Define window and viewport. What are the use of these?

Q.4: Explain window-to-viewport transformation.

Q.5: Write a function to combine window-to-viewport and viewing for different workstation.

Q.6: Describe the process to get normalized coordinates from world coordinates.

Q.7: Write a procedure to implement the evaluateViewMappingMatrix function to calculates the elements of a matrix for performing the window-to-viewport transformation.

*** ***** ***

Name of the Paper: .....
Course Code: .....

Centre for Internal Quality Assurance (CIQA)
Krishna Kanta Handiqui State Open University
City Office: Housefed Complex, Guwahati-781006

Learner's Feedback on Course

Dear Learner,

Regarding the course as mentioned above, we would like to know your opinions and comments so as to improve the quality of self learning materials in future. Please respond to the following statements by ticking the number you feel most reflect your opinion. After completion of the additional comments, please detach the page and send/mail the same to us at the address given below.

The Director, Centre for Internal Quality Assurance, KKHandiqui State Open University
Housefed Complex, Dispur, Guwahati-781006
(E-mail id: ciqa@kkhsou.in)

1) Approximately how many hours did you spend for studying the units in the course? ☐☐☐

2) Please give your opinions (by ✓ mark) to the following items based on your reading of the block:

Sl. No.	Statements	Strongly Agree	Agree	Neither Agree nor Disagree	Disagree	Strongly Disagree
I)	The SLMs of the course fulfil the learning objectives set out in the beginning of each unit					
II)	The units of the SLM could satisfy my academic needs and expectations					
III)	The Unit writers have excellent knowledge about the course contents					
IV)	Language and contents of the units were presented at a level which I could readily understand					
V)	Ample opportunity for participation in the activities provided in the units					
VI)	Used enough Illustrations (Diagrams, tables etc.) for conceptual clarity					
VII)	Quality of content is engaging, relevant, and up-to-date					
VIII)	The self check questions are very helpful					
IX)	The Possible/model questions and the answers to check my progress have benefited me a lot					

Additional Comments: *(Please feel free to provide your open comments)*

1) Which aspects of the SLM, according to you, worked well?

.....

.....

2) What sort of changes/improvements do you feel KKHSOU could implement to improve the overall quality of the SLM?

.....

.....

Thank you for taking the time to complete this form.