——CHAPTER-1

Introduction to Computer Graphics

Graphics is defined as any sketch or a drawing or a special network that pictorially represents some meaningful information.

Computer Graphics is used where a set of image needs to be manipulated or the creation of the image in the form of pixels and is drawn on the computer.

Computer Graphics can be used in digital photography, film, entertainment, electronic gadgets and all other core technologies which are required.

It is a vast subject and area in the field of computer science. Computer Graphics can be used in UI design, rendering, geometric object, animation and many more.

Computer Graphics refers to several things:

- The manipulation and the representation of the image or the data in a graphical manner.
- Various technology required for the creation and manipulation.
- Digital synthesis and its manipulation.

Basically there are two types of computer graphics namely.

1. **Interactive Computer Graphics:**
    - Interactive Computer Graphics involves a two way communication between computer and user. Here the observer is given some control over the image by providing him with an input device for example the video game controller of the ping pong game. This helps him to signal his request to the computer.
    - The computer on receiving signals from the input device can modify the displayed picture appropriately. To the user it appears that the picture is changing instantaneously in response to his commands. He can give a series of commands, each one generating a graphical response from the computer. In this way he maintains a conversation, or dialogue, with the computer.
    - Interactive computer graphics affects our lives in a number of indirect ways. For example, it helps to train the pilots of our airplanes. We can create a flight simulator which may help the pilots to get trained not in a real aircraft but on the grounds at the control of the flight simulator. The flight simulator is a mock-up of an aircraft flight deck, containing all the usual controls and surrounded by screens on which we have the projected computer generated views of the terrain visible on take-off and landing.
    - Flight simulators have many advantages over the real aircrafts for training purposes, including fuel savings, safety, and the ability to familiarize the trainee with a large number of the world’s airports.

2. **Non Interactive Computer Graphics**:

In non-interactive computer graphics otherwise known as passive computer graphics. It is the computer graphics in which user does not have any kind of control over the image. Image is merely the product of static stored program and will work according to the instructions given in the program linearly. The image is totally under the control of program instructions not under the user. Example: screen savers.

Applications of Computer Graphics

Some of the applications of computer graphics are:

1. **Computer Art:**

Using computer graphics we can create fine and commercial art which include animation packages, paint packages. These packages provide facilities for designing object shapes and specifying object motion. Cartoon drawing, paintings, logo design can also be done.

2. **Computer Aided Drawing:**

Designing of buildings, automobile, aircraft is done with the help of computer aided drawing, this helps in providing minute details to the drawing and producing more accurate and sharp drawings with better specifications.

3. **Presentation Graphics:**

For the preparation of reports or summarising the financial, statistical, mathematical, scientific, economic data for research reports, managerial reports, moreover creation of bar graphs, pie charts, time chart, can be done using the tools present in computer graphics.

4. **Entertainment:**

Computer graphics finds a major part of its utility in the movie industry and game industry. Used for creating motion pictures, music video, television shows, cartoon animation films. In the game industry where focus and interactivity are the key players, computer graphics helps in providing such features in the efficient way.

5. **Education:**

Computer generated models are extremely useful for teaching huge number of concepts and fundamentals in an easy to understand and learn manner. Using computer graphics many educational models can be created through which more interest can be generated among the students regarding the subject.

6. **Training:**

Specialised system for training like simulators can be used for training the candidates in a way that can be grasped in a short span of time with better understanding. Creation of training modules using computer graphics is simple and very useful.

7. **Visualisation:**
   Today the need of visualise things have increased drastically, the need of visualisation can be seen in many advance technologies , data visualisation helps in finding insights of the data , to check and study the behaviour of processes around us we need appropriate visualisation which can be achieved through proper usage of computer graphics.

8. Image Processing:

Various kinds of photographs or images require editing in order to be used in different places. Processing of existing images into refined ones for better interpretation is one of the many applications of computer graphics.

9. **Machine Drawing:**

Computer graphics is very frequently used for designing, modifying and creation of various parts of machine and the whole machine itself, the main reason behind using computer graphics for this purpose is the precision and clarity we get from such drawing is ultimate and extremely desired for the safe manufacturing of machine using these drawings.

10. **Graphical User Interface:**

The use of pictures, images, icons, pop-up menus, graphical objects helps in creating a user friendly environment where working is easy and pleasant, using computer graphics we can create such an atmosphere where everything can be automated and anyone can get the desired action performed in an easy fashion.

11. **Printing Technology: Computer** graphics are used in textile designing and flex printing.
12. **Typography:** Use of character pictures to replace the rough form of the past in printing.
13. **Satellite Imaging:** Computer graphics are used to forecast the movement of the cloud and to predict the weather.
14. **Cartography:** Computer graphics are used in map drawing.

CHAPTER-2

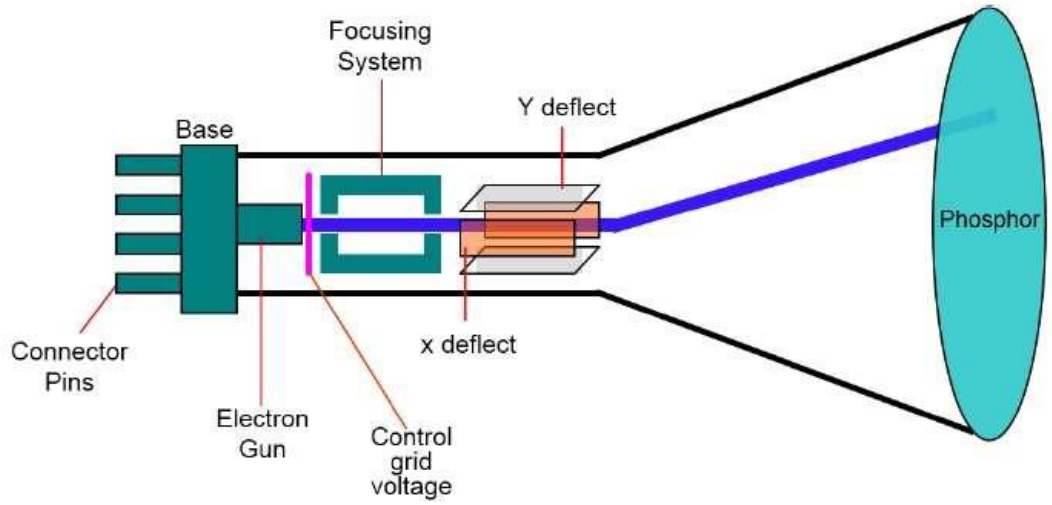
VIDEO DISPLAY DEVICES

Cathode Ray Tube

The primary output device in a graphical system is the video monitor. The main element of a video monitor is the **Cathode Ray Tube** CRT**,** shown in the following illustration.

The operation of CRT is very simple −

- The electron gun emits a beam of electrons cathode rays.
- The electron beam passes through focusing and deflection systems that direct it towards specified positions on the phosphor-coated screen.
- When the beam hits the screen, the phosphor emits a small spot of light at each position contacted by the electron beam.
- It redraws the picture by directing the electron beam back over the same screen points quickly.



There are two ways **Random scan and Raster scan** by which we can display an object on the screen.

Raster Scan Display

Raster can be explained as a rectangular collection of dots or points plotted.

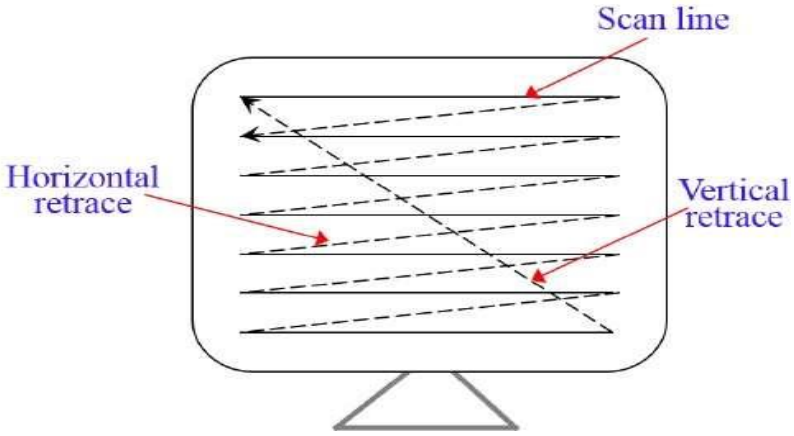
An image is subdivided into various horizontal lines which are referred to as **scan lines** which are then further divided into different **pixels** which helps in the processing of an image.

Basic working of Raster Scan

- In this system, a beam of an electron is moved across the screen. It moves from top to bottom considering one row at a time.
- As the beam of electron moves through each row, its intensity is alternatively turned on and off which helps to create a pattern of spots that are illuminated.

When each scan of the line is refreshed it returns to the left side of the screen. This motion is known as **Horizontal retrace**.

- As a particular frame ends, the beam of electron moves to the left top corner of the screen to move to another frame. This motion is referred to as **Vertical retrace**.
- The picture is then stored in an area of memory which is referred to as the **frame buffer** or **refresh buffer**.
- The buffer in a raster scan is that area that is responsible for containing intensity of the various points on the screen.
- The values stored in the buffer are then fetched and traced over scan lines one by one on the screen.



- The image formed through this raster scan is known as a raster image. The quality of this image is determined by the number of pixels which is termed as the **resolution of the image**.
- The amount of information each pixel represents is known as the **color depth of the image**.
- The raster graphics system of high quality contains 24 bits per pixel in the frame buffer. This is referred to as a **full color** or **true color** system. Refreshing of raster scan displays is carried out at the rate of 60 to 80 frames per second.
- The capability of raster scan system to store intensity information for each screen point makes it well suited for the realistic display of scenes containing subtle shading and colour patterns.

Interlacing

A TV video signal is graphically interlaced, which means every full screen of information is made up of two separate fields which include the odd field and even field. First, the odd lines are printed on the graphics screen. Then, the even lines are printed in between the odd lines before the odd lines fade away. This all happens faster than any human eye can perceive.

Advantages:

1. Realistic image
2. Million Different colors to be generated

3. Shadow Scenes are possible.

Disadvantages:

1. Low Resolution
2. Expensive

Random Scan Display

In Random Scan Display a beam of the electron is directed only to the screen areas where any picture has to be displayed or drawn on the screen. It is also termed as vector display, as it displays or draws a picture in the form of one line at a time. It can draw and refresh lines on the screen of a picture in any sequence not particularly specific.

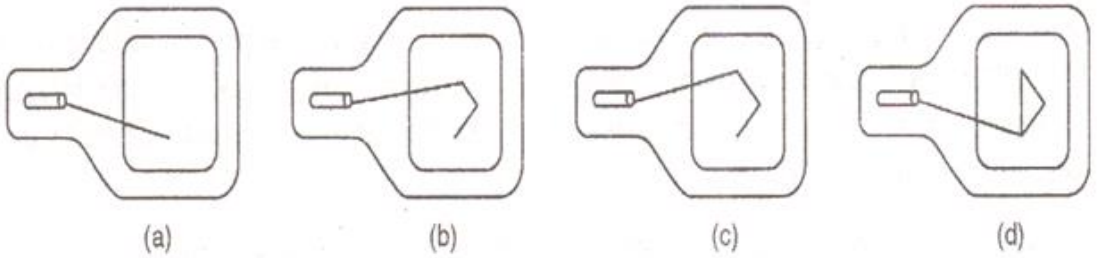


Fig: Random Scan Display

Basic working of random scan display

- Random scan monitors are used to draw a picture in one line at a time and are thus also referred to as **vector displays**, **stroke-writing display,** or **calligraphic display**.
- The cathode ray tube when operates as a random scan display device directs the beam of an electron only to those areas of the screen where display or a picture has to be drawn.
- Picture definition is stored as a set of line-drawing commands in an area of memory referred to as the **refresh display file**.
- To draw a picture or display it on the screen the system goes through a line or set of commands and draws each of them one at a time in a line turn by turn.

The **refresh rate** here depends on the number of lines that are to be displayed on the screen and are designed so that they draw the component lines of the picture 30 to 60 times in a second.

They have a high resolution of pictures and produce smooth line drawing. It's that smooth that while zooming also it doesn't spread.

Advantages:

1. A CRT has the electron beam directed only to the parts of the screen where an image is to be drawn.
2. Produce smooth line drawings.
3. High Resolution

Disadvantages:

1. Random-Scan monitors cannot display realistic shades scenes.

Differentiate between Random and Raster Scan Display:

Random Scan	Raster Scan
1. It has high Resolution	1. Its resolution is low.
2. It is more expensive	2. It is less expensive
3. Any modification if needed is easy	3.Modification is tough
4. Solid pattern is tough to fill	4.Solid pattern is easy to fill
5. Refresh rate depends or resolution	5. Refresh rate does not depend on the picture.
6. Only screen with view on an area is displayed.	6. Whole screen is scanned.
7. Beam Penetration technology come under it.	7. Shadow mark technology came under this.
8. It does not use interlacing method.	8. It uses interlacing
9. It is restricted to line drawing applications	9. It is suitable for realistic display.

CHAPTER-3

GRAPHICS OUTPUT PRIMITIVES

Those functions in a graphics package that we use to describe the various picture components are called the **graphics output primitives**, or simply primitives.

Point positions and straight-line segments are the simplest geometric primitives.

Points and Lines

• Point is the fundamental element of picture representation.
• It is the position in the plan defined as either pair or triplets of number depending upon the dimension.
• Two points represent line or edge and 3 or more points a polygon.
• Curved lines are represented by the short straight lines.

The Line drawing algorithm

"The Line drawing algorithm is a graphical algorithm which is used to represent the line segment on discrete graphical media, i.e., printer and pixel-based media."
A line contains two points. The point is an important element of a line.

Properties of a Line Drawing Algorithm

There are the following properties of a good Line Drawing Algorithm.

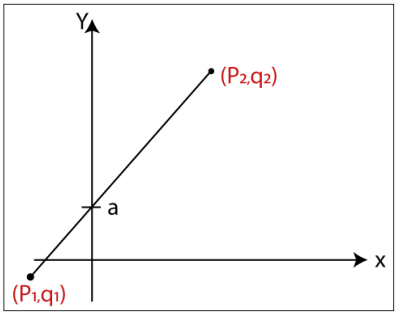
- **An algorithm should be precise:** Each step of the algorithm must be adequately defined.
- **Finiteness:** An algorithm must contain finiteness. It means the algorithm stops after the execution of all steps.
- **Easy to understand:** An algorithm must help learners to understand the solution in a more natural way.
- **Correctness:** An algorithm must be in the correct manner.
- **Effectiveness:** Thesteps of an algorithm must be valid and efficient.
- **Uniqueness:** All steps of an algorithm should be clearly and uniquely defined, and the result should be based on the given input.
- **Input:** A good algorithm must accept at least one or more input.
- **Output:** An algorithm must generate at least one output.

Equation of the straight line

We can define a straight line with the help of the following equation.

y= mx + a

Where,
(x, y) = axis of the line.
m = Slope of the line.
a = Interception point



Let us assume we have two points of the line (p_1, q_1) and (p_2, q_2).
Now, we will put values of the two points in straight line equation, and we get

$y = mx + a$

$q_2 = mp_2$ **…(1)**

$q_1 = mp_1 + a$ **…(2)**

We have from equation (1) and (2)

$q_2 – q_1 = mp_2 – mp_1$

$q_2 – q_1 = m (p_2 – p_1)$

The value of **$m = (q_2 – q_1)/ (p_2 – p_1)$**

$m = \blacktriangle q / \blacktriangle p$

Algorithms of Line Drawing

There are following algorithms used for drawing a line:

- **DDA (Digital Differential Analyzer) Line Drawing Algorithm**
- **Bresenham's Line Drawing Algorithm**
- **Mid-Point Line Drawing Algorithm**

DDA line Drawing Algorithm

DDA (Digital Differential Analyzer) Line Drawing Algorithm

The Digital Differential Analyzer helps us to interpolate the variables on an interval from one point to another point. We can use the digital Differential Analyzer algorithm to perform rasterization on polygons, lines, and triangles.

Digital Differential Analyzer algorithm is also known as an **incremental method** of scan conversion. In this algorithm, we can perform the calculation in a step by step manner. We use the previous step result in the next step.

As we know the general equation of the straight line is:

$y = mx + c$

Here, **m** is the slope of **(x_1, y_1)** and **(x_2, y_2).**

$m = (y_2 – y_1)/ (x_2 – x_1)$

Now, we consider one point **(x_k, y_k)** and **(x_{k+1}, y_{k+1})** as the next point**.**

Then the slope **$m = (y_{k+1} – y_k)/ (x_{k+1} – x_k)$**

Now, we have to find the slope between the starting point and ending point. There can be following three cases to discuss:

Case 1: If **m < 1**

Then **x** coordinate tends to the Unit interval.

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

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

Case 2: If **m > 1**

Then **y** coordinate tends to the Unit interval.

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

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

Case 3: If **m = 1**

Then **x and y** coordinate tend to the Unit interval.

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

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

We can calculate all intermediate points with the help of above three discussed cases.

Algorithm of Digital Differential Analyzer (DDA) Line Drawing

Step 1: Start.

Step 2: We consider Starting point as (x_1, y_1), and ending point (x_2, y_2).

Step 3: Now, we have to calculate ▲**x** and ▲**y.**

$\blacktriangle x = x_2 - x_1$

$\blacktriangle y = y_2 - y_1$

$m = \blacktriangle y / \blacktriangle x$

Step 4: Now, we calculate three cases.

If **m < 1**

Then **x** change in Unit Interval

y moves with deviation

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

If **m > 1**

Then **x** moves with deviation

y change in Unit Interval

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

If **m = 1**

Then **x** moves in Unit Interval

y moves in Unit Interval

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

Step 5: We will repeat step 4 until we find the ending point of the line.

Step 6: Stop.

Example: A line has a starting point (1,7) and ending point (11,17). Apply the Digital Differential Analyzer algorithm to plot a line.

Solution: We have two coordinates,

Starting Point = (x_1, y_1) = **(1,7)**

Ending Point = (x_2, y_2) = **(11,17)**

Step 1: First, we calculate ▲**x**, ▲**y** and **m.**

$\blacktriangle x = x_2 - x_1 = 11-1 = 10$

$\blacktriangle y = y_2 - y_1 = 17-7 = 10$

$m = \blacktriangle y / \blacktriangle x = 10/10 = 1$

Step 2: Now, we calculate the number of steps.

$\blacktriangle x = \blacktriangle y = 10$

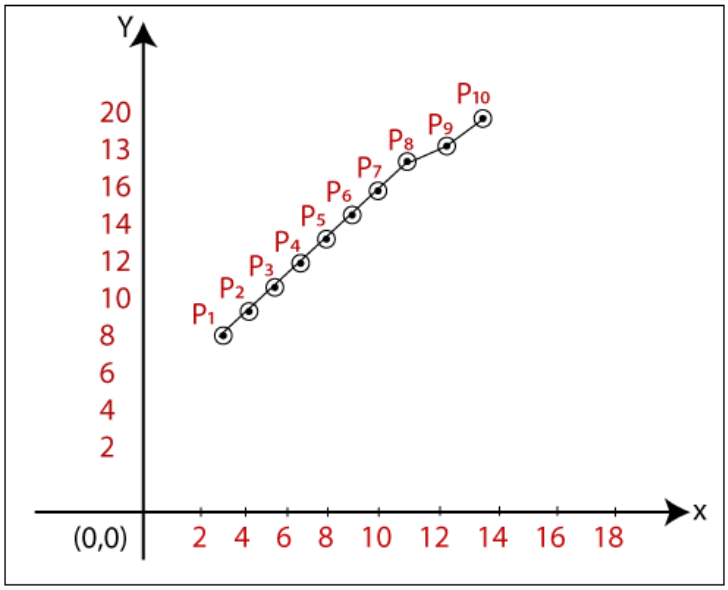
Then, the number of steps = **10**

Step 3: Weget **m = 1,** Third case is satisfied.

Now move to next step.

Step 4: We will repeat step 3 until we get the endpoints of the line.
Step 5: Stop.

x_k	y_k	x_{k+1}	y_{k+1}	(x_{k+1}, y_{k+1})
1	7	2	8	(2, 8)
		3	9	(3, 9)
		4	10	(4, 10)
		5	11	(5, 11)
		6	12	(6, 12)
		7	13	(7, 13)
		8	14	(8, 14)
		9	15	(9, 15)
		10	16	(10, 16)
		11	17	(11, 17)



The coordinates of drawn line are-
P_1 = (2, 8)
P_2 = (3, 9)
P_3 = (4, 10)
P_4 = (5, 11)
P_5 = (6, 12)
P_6 = (7, 13)
P_7 = (8, 14)
P_8 = (9, 15)
P_9 = (10, 16)
P_{10} = (11, 17)

Advantages of Digital Differential Analyzer

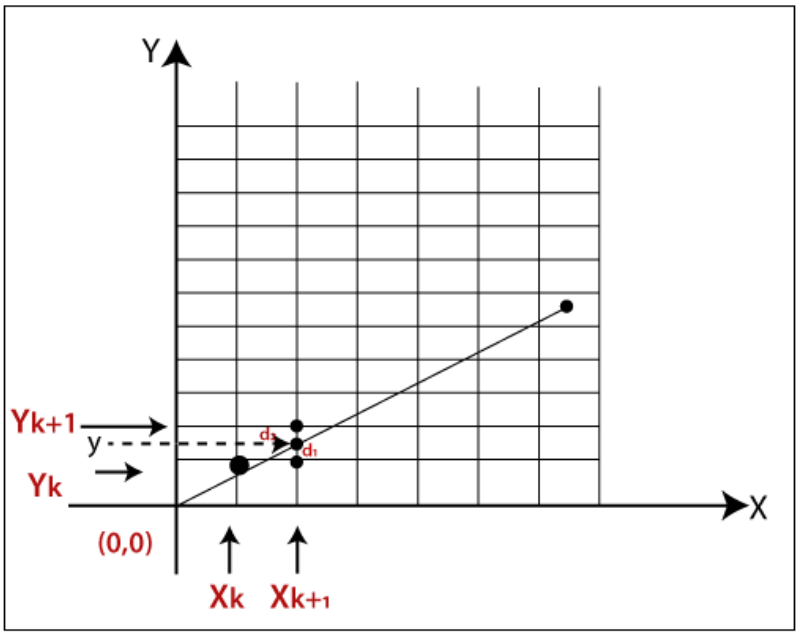
- It is a simple algorithm to implement.
- It is a faster algorithm than the direct line equation.
- We cannot use the multiplication method in Digital Differential Analyzer.
- Digital Differential Analyzer algorithm tells us about the overflow of the point when the point changes its location.

Disadvantages of Digital Differential Analyzer

- The floating-point arithmetic implementation of the Digital Differential Analyzer is time-consuming.
- The method of round-off is also time-consuming.
- Sometimes the point position is not accurate.

Bresenham's Line Drawing Algorithm

This algorithm was introduced by **"Jack Elton Bresenham"** in **1962.** This algorithm helps us to perform scan conversion of a line. It is a powerful, useful, and accurate method. We use incremental integer calculations to draw a line. The integer calculations include addition, subtraction, and multiplication.
In **Bresenham's Line Drawing algorithm**, we have to calculate the slope (**m**) between the starting point and the ending point.



As shown in the above figure let, we have initial coordinates of a line = (x_k, y_k)
The next coordinates of a line **= (x_{k+1}, y_{k+1})**
The intersection point between y_k and y_{k+1} = **y**
Let we assume that the distance between **y** and y_k **= d_1**
The distance between **y** and y_{k+1} **= d_2**
Now, we have to decide which point is nearest to the intersection point.
If **m < 1**
then **x** = x_k**+1** **{** Unit Interval**}**
y = y_k**+1** **{** Unit Interval**}**
As we know the equation of a line-
y = mx +b
Now we put the value of x into the line equation, then
$y = m(x_k+1) +b$ **……. (1)**
The value of **$d_1 = y – y_k$**
Now we put the value of **d_1** in equation (1).
$y = m (x_k+1) +b – y_k$
Now, we again put the value of **y** in the previous equation then we got,
$d_2 = y_{k+1} – y$
$= y_k + 1 – m (x_k+1) – b$
Now, we calculate the difference between **$d_1 – d_2$**
If **$d_1 < d_2$**
Then **$y_{k+1} = y_k$** **{**we will choose the lower pixel as shown in figure**}**
If **$d_1 => d_2$**
Then **$y_{k+1} = y_k+1$** **{**we will choose the upper pixel as shown in figure**}**
Now, we calculate the values of **$d_1 – d_2$**

$(d_1 – d_2)= m (x_k+1) +b – y_k – y_k – 1 + m (x_k+1) + b$
We simplified the above equation and replaced the **m** with **▲y/▲x.**
$(d_1 – d_2) = 2 m (x_{k+1}) -2y_k + 2b-1$
We multiplied **▲x** at both side then we got,
$▲x (d_1 – d_2) = ▲x (2m (x_{k+1}) -2y_k + 2b-1)$
We consider $▲x (d_1 – d_2)$ as a decision parameter (P_k), so
$p_k = ▲x (d_1 – d_2)$
After calculation we got,
$P_k = 2▲yx_k + 2▲y – 2▲xy_k + ▲x (2b-1)$
Then, the next coordinate of p_k
$p_{k+1} = 2▲yx_{k+1} + 2▲y – 2▲xy_{k+1} + ▲x (2b-1)$
Now, the difference between $p_{k+1} – p_k$ then,
$p_{k+1} – p_k = 2▲y (x_{k+1}-x_k) – 2▲x (y_{k+1}-y_k)$
$p_{k+1} = p_k + 2▲y (x_{k+1}-x_k) – 2▲x (y_{k+1}-y_k)$ **{Decision parameter coordinate}**
Now, we put the value of x_{k+1} in above equation then we got,
$p_{k+1} = p_k + 2▲y – 2▲x (y_{k+1} – y_k)$ **{New decision parameter when m <1}**
Similarly, if **m >1**, the new decision parameter for next coordinate will be
$p_{k+1} = p_k + 2▲y – 2▲x (x_{k+1} – x_k)$ **{New decision parameter when m >1}**
If $p_k >= 0$ **{For coordinate y}**
Then,
$y_{k+1} = y_k+1$ **{We will choose the nearest y_{k+1} pixel}**
The next coordinate will be (x_k+1, y_k+1)
If $p_k < 0$
Then,
$y_{k+1} = y_k$ **{We will choose the nearest y_k pixel}**
The next coordinate will be (x_k+1, y_k)
Similarly,
If $p_k >= 0$ **{For coordinate x}**
Then,
$x_{k+1} = x_k+1$ **{We will choose the nearest x_{k+1} pixel}**
The next coordinate will be (x_k+1, y_k+1)
If $p_k < 0$
Then,
$x_{k+1} = x_k$ **{We will choose the nearest x_k pixel}**
The next coordinate will be (x_k, y_k+1)

Algorithm of Bresenham's Line Drawing Algorithm

Step 1: Start.
Step 2: Now, we consider Starting point as (x_1, y_1) and endingpoint (x_2, y_2).
Step 3: Now, we have to calculate **▲x** and **▲y.**
$▲x = x_2-x_1$
$▲y = y_2-y_1$
$m = ▲y/▲x$
Step 4: Now, we will calculate the decision parameter p_k with following formula.
$p_k = 2▲y-▲x$
Step 5: Theinitial coordinates of the line are (x_k, y_k), and the next coordinatesare (x_{k+1}, y_{k+1}). Now, we are going to calculate two cases for decision parameter p_k
Case 1: If
$p_k < 0$
Then
$p_{k+1} = p_k + 2▲y$
$x_{k+1} = x_k + 1$
$y_{k+1} = y_k$

Case 2: If

$p_k >= 0$

Then

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

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

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

Step 6: We will repeat step 5 until we found the ending point of the line and the total number of iterations = **▲x-1.**

Step 7: Stop.

Example: A line has a starting point (9,18) and ending point (14,22). Apply the Bresenham's Line Drawing algorithm to plot a line.

Solution: We have two coordinates,

Starting Point = (x_1, y_1) = **(9,18)**

Ending Point = (x_2, y_2) = **(14,22)**

Step 1: First, we calculate **▲x**, **▲y.**

$\blacktriangle x = x_2 - x_1 = 14-9 = 5$

$\blacktriangle y = y_2 - y_1 = 22-18 = 4$

Step 2: Now, we are going to calculate the decision parameter (p_k)

$p_k = 2\blacktriangle y - \blacktriangle x$

$= 2 \times 4 - 5 = 3$

The value of $p_k = 3$

Step 3: Now, we will check both the cases.

If

$p_k >= 0$

Then

Case 2 is satisfied. Thus

$p_{k+1} = p_k + 2\blacktriangle y - 2\blacktriangle x = 3 + (2 \times 4) - (2 \times 5) = 1$

$x_{k+1} = x_k + 1 = 9 + 1 = 10$

$y_{k+1} = y_k + 1 = 18 + 1 = 19$

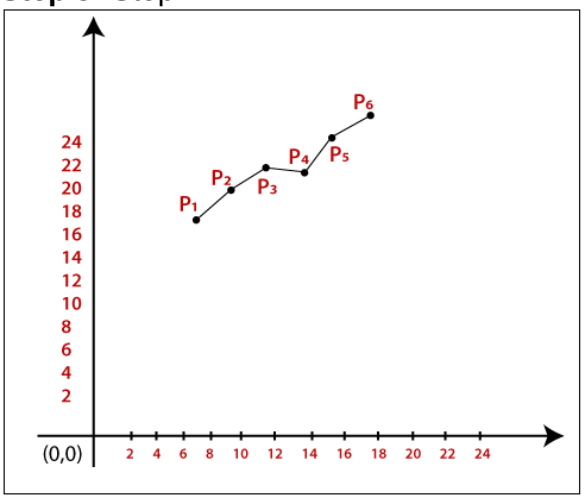
Step 4: Now move to next step. We will calculate the coordinates until we reach the end point of the line.

▲x -1 = 5 – 1 = 4

p_k	p_{k+1}	x_{k+1}	y_{k+1}
		9	18
3	1	10	19
1	-1	11	20
-1	7	12	20
7	5	13	21

5	3	14	22

Step 5: Stop.



The Coordinates of drawn lines are-
P_1 = (9, 18)
P_2 = (10, 19)
P_3 = (11, 20)
P_4 = (12, 20)
P_5 = (13, 21)
P_6 = (14, 22)

Advantages of Bresenham's Line Drawing Algorithm

- It is simple to implement because it only contains integers.
- It is quick and incremental
- It is fast to apply but not faster than the Digital Differential Analyzer (DDA) algorithm.
- The pointing accuracy is higher than the DDA algorithm.

Disadvantages of Bresenham's Line Drawing Algorithm

- The *Bresenham's Line drawing algorithm* only helps to draw the basic line.
- The resulted draw line is not smooth.

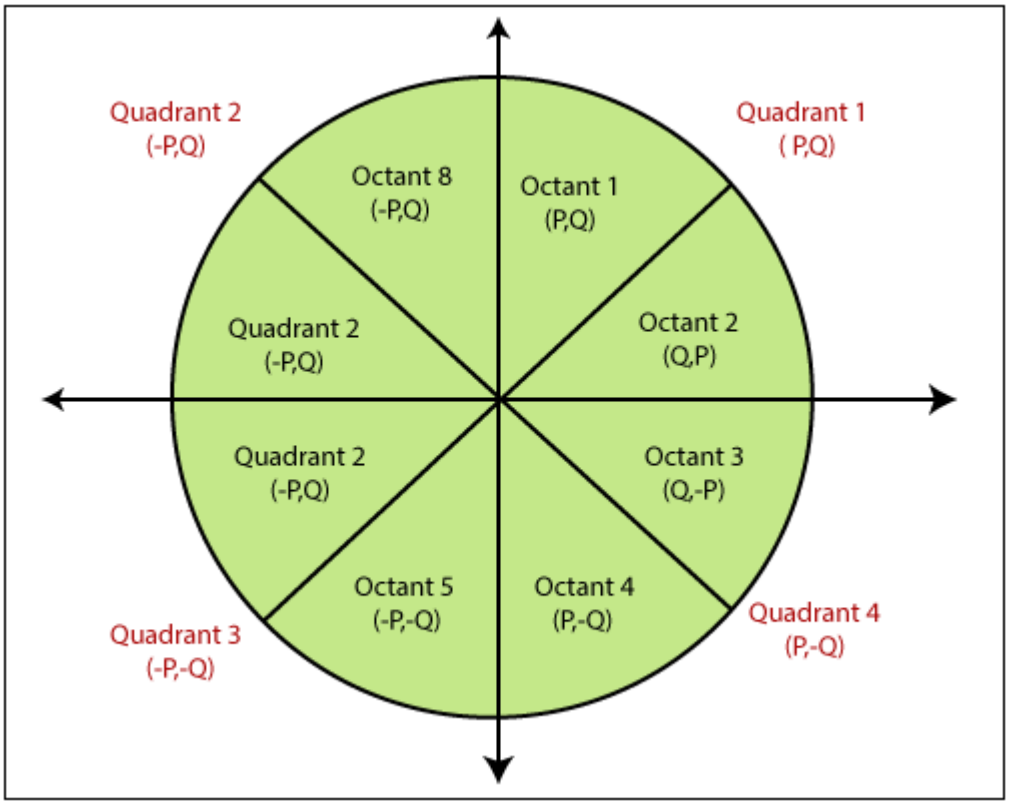
Midpoint Circle Drawing Algorithm

The **midpoint circle drawing algorithm** helps us to calculate the complete perimeter points of a circle for the first octant. We can quickly find and calculate the points of other octants with the help of the first octant points. The remaining points are the mirror reflection of the first octant points.

This algorithm is used in computer graphics to define the coordinates needed for rasterizing the circle. The midpoint circle drawing algorithm helps us to perform the generalization of conic sections. Bresenham's circle drawing algorithm is also extracted from the midpoint circle drawing algorithm. In the algorithm, we will use the 8-way symmetry property.

In this algorithm, we define the unit interval and consider the nearest point of the circle boundary in each step.

Let us assume we have a point **a (p, q)** on the boundary of the circle and with **r** radius satisfying the equation $\mathbf{f_c\,(p, q) = 0}$



As we know the equation of the circle is –

$\mathbf{f_c\,(p, q) = p^2 + q^2 = r^2}$ **…… (1)**

If

$\mathbf{f_c\,(p, q) < 0}$

then

The point is inside the circle boundary.

If

$\mathbf{f_c\,(p, q) = 0}$

then

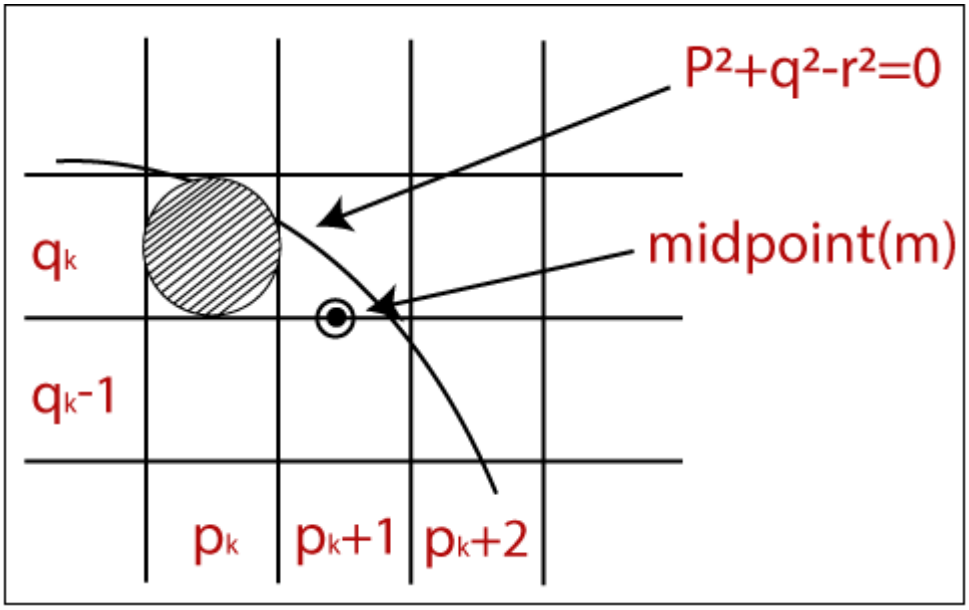
The point is on the circle boundary.

If

$\mathbf{f_c\,(p, q) > 0}$

then

The point is outside the circle boundary.



In the figure, we calculate the mid-point **(m).** The midpoint appears between q_k and q_k **-1.**
The currentposition of the pixel = p_k **+1**
The next position of the pixel = **(p_k+1, q_k)** and **(p_k+1, q_k-1)**
Now, we will calculate the decision parameter **(d_k)**
$d_k = (p_k+1, q_k - 1/2)$
Now, we replace all the values with equation **(1)**
$$d_k = (p_k+1)^2 + (q_k - 1/2)^2 - r^2 \quad \text{..... (2)}$$
Now, there should be two conditions.
Condition 1: If
d_k is negative
then
The midpoint (m) is inside the circle boundary
Condition 2: If
d_k is positive
then
The midpoint (m) is outside the circle boundary
Now, we find the next point of x coordinates. Then,
$$P_{k+1} + 1 = P_k + 2$$
Now, we replace all the value of **equation (2)** with **(k+1)**. We get-
$$d_{k+1} = (p_{k+1}+1)^2 + (q_{k+1} - 1/2)^2 - r^2 \quad \text{..... (3)}$$
Now, we will find the difference between
$$d_{k+1} - d_k = \{(p_{k+1}+1)^2 + (q_{k+1} - 1/2)^2 - r^2\} - \{(p_k+1)^2 + (q_k - 1/2)^2 - r^2\}$$
$$= d_k + (2p_k+1) + (q_{k+1}^2 - q_k^2) - (q_{k+1} - q_k) + 1 \quad \text{..... (4)}$$
Here, If
d_k isnegativethen d_{k+1}
then
$2p_{k+1} + 1$
Otherwise
$2p_{k+1} + 1 - 2q_{k+1}$
Now, the next coordinate for x and y points
$$2p_{k+1} = 2p_k + 2$$
$$2q_{k+1} = 2q_k - 2$$
Now, the initial decision parameter **(d_0)** at the position **(p, q) = (0, r)**
We put **(0, r)** in circle equation and we get-

$d_0 = (1, r - 1/2)$
$= (1 + (r - 1/2)^2 - r^2)$
$= 5/4 - r$
We only take integer value $= 1 - r$

Algorithm of Midpoint Circle Drawing

Step 1: Start.
Step 2: First, we allot the center coordinates (p_0, q_0) as follows-
$P_0 = 0$
$q_0 = r$
Step 3: Now, we calculate the initial decision parameter d_0 –
$d_0 = 1 - r$
Step 4: Assume,the starting coordinates = (p_k, q_k)
The next coordinates will be (p_{k+1}, q_{k+1})
Now, we find the next point of the first octant according to the value of the decision parameter (d_k).
Step 5: Now, we follow two cases-
Case 1: If
$d_k < 0$
then
$p_{k+1} = p_k + 1$
$q_{k+1} = q_k$
$d_{k+1} = d_k + 2\ p_{k+1} + 1$
Case 2: If
$d_k >= 0$
then
$p_{k+1} = p_k + 1$
$q_{k+1} = q_k - 1$
$d_{k+1} = d_k - 2\ (q_{k+1} + 2\ p_{k+1}) + 1$
Step 6: If the center coordinate point (p_0, q_0) is not at the origin **(0, 0)** then we will draw the points as follow-
For x coordinate = $x_c + p_0$
For y coordinate = $y_c + q_0$ {x_c and y_c containsthe current value of x and y coordinate}
Step 7: We repeat step **5** and **6** until we get **x>=y.**
Step 8: Stop.
Example: The center coordinates are **(0, 0),** and the radius of the circle is **10**. Find all points of the circle by using the midpoint circle drawing algorithm?
Solution:
Step 1: The given center coordinates of the circle $(p_0, q_0) = (0, 0)$
Radius of the circle **(r)** = **10**
Step 2: Now, we will determine the starting coordinates (p_0, q_0) as follows-
$P_0 = 0$
$q_0 = r$ (radius) $= 10$
Step 3: Now, we will determine the initial decision parameter (d_0)
$d_0 = 1 - r$
$d_0 = 1 - 10$
$d_0 = -9$
Step 4: The initial parameter $d_0 < 0$ then, case 1 is satisfied.
$p_{k+1} = p_k + 1 = 0 + 1 = 1$
$q_{k+1} = q_k = 10$
$d_{k+1} = d_k + 2\ p_{k+1} + 1 = -9 + 2(1) + 1 = -6$

Step 5: The center coordinates of circle are already **(0, 0).** So, move to next step.
Step 6: We will execute step 4 until we get **x >= y.**

The table for coordinates of octant 1-

d_k	d_{k+1}	(p_{k+1},q_{k+1})
		(0, 10)
-9	**-6**	**(1, 10)**
-6	**-1**	**(2, 10)**
-1	**6**	**(3, 10)**
6	**-3**	**(4, 9)**
-3	**8**	**(5, 9)**
8	**5**	**(6, 8)**

Now, we will determine the coordinates of the octant 2 by swapping the p and q coordinates.

Points of Octant 1	Points of Octant 2
(0, 10)	**(8, 6)**
(1, 10)	**(9, 5)**
(2, 10)	**(9, 4)**
(3, 10)	**(10, 3)**
(4, 9)	**(10, 2)**

(5, 9)	**(10, 1)**
(6, 8)	**(10, 0)**

Thus, we can find all the coordinates of the circle for all quadrants.

Quadrant 1 (p, q)	Quadrant 2 (-p, q)	Quadrant 3 (-p, -q)	Quadrant 4 (p, -q)
(0, 10)	**(0, 10)**	**(0, -10)**	**(0, -10)**
(1, 10)	**(-1, 10)**	**(-1, -10)**	**(1, -10)**
(2, 10)	**(-2, 10)**	**(-2, -10)**	**(2, -10)**
(3, 10)	**(-3, 10)**	**(-3, -10)**	**(3, -10)**
(4, 9)	**(-4, 9)**	**(-4, -9)**	**(4, -9)**
(5, 9)	**(-5, 9)**	**(-5, -9)**	**(5, -9)**
(6, 8)	**(-6, 8)**	**(-6, -8)**	**(6, -8)**
(8, 6)	**(-8, 6)**	**(-8, -6)**	**(8, -6)**
(9, 5)	**(-9, 5)**	**(-9, -5)**	**(9, -5)**
(9, 4)	**(-9, 4)**	**(-9, -4)**	**(9, -4)**
(10, 3)	**(-10, 3)**	**(-10, -3)**	**(10, -3)**
(10, 2)	**(-10, 2)**	**(-10, -2)**	**(10, -2)**

(10, 1)	**(-10, 1)**	**(-10, -1)**	**(10, -1)**
(10, 0)	**(-10, 0)**	**(-10, 0)**	**(10, 0)**

Advantages of Midpoint circle drawing algorithm

- It is a powerful and efficient algorithm.
- The midpoint circle drawing algorithm is easy to implement.
- It is also an algorithm based on a simple circle equation **($x^2 + y^2 = r^2$).**
- This algorithm helps to create curves on a raster display.

Disadvantages of Midpoint circle drawing algorithm

- It is a time-consuming algorithm.
- Sometimes the points of the circle are not accurate.

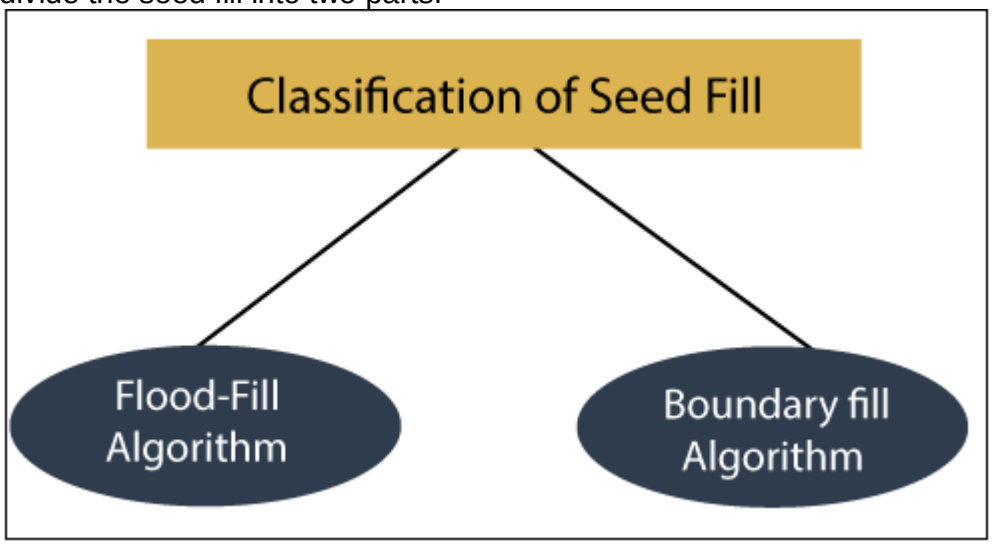
Filled Area Primitives

Filled Area Primitives: Area filling is a method or process that helps us to fill an object, area, or image. We can easily fill the polygon. The polygon filling is defined as filling or highlighting all the pixels. The pixels appear inside the polygon shape with any color other than the background color.
There are two algorithms or methods used to fill the polygon.

- **Seed Fill Algorithm**
- **Scan Line Algorithm**

Seed Fill Algorithm

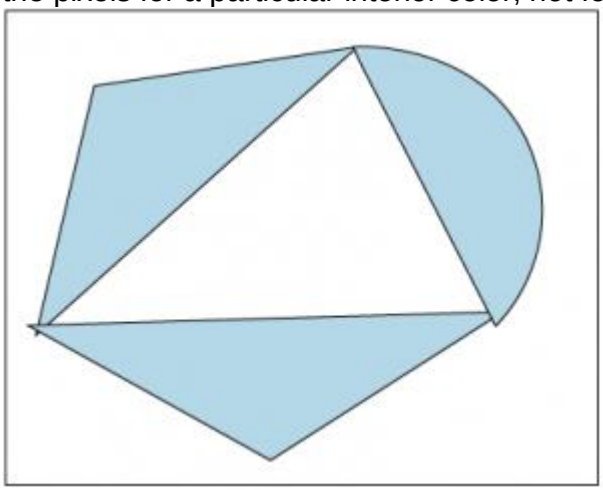
In this method, we will select a seed or starting point inside the boundary. We can further divide the seed fill into two parts.



1. **Flood-fill Algorithm**
2. **Boundary-fill Algorithm**

Flood-fill Algorithm

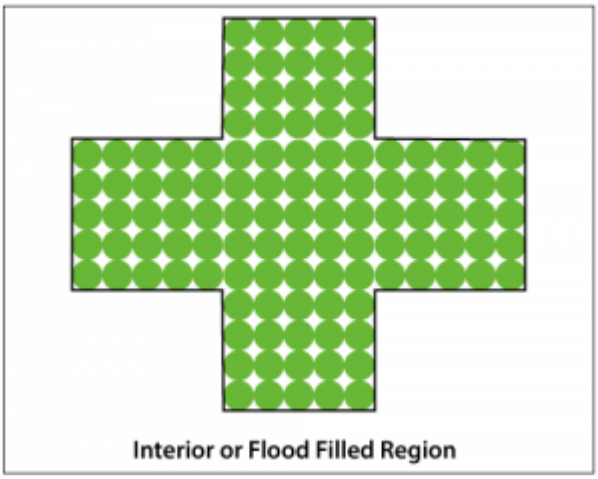
Flood-fill algorithm helps to define a region in the boundary, attached to a point in the multi-dimensional array. It is similar to the bucket tool used in the paint program. The stack-based recursive function is used to implement the algorithm. In flood -fill algorithm, we replace all the associated pixels of the selected color with a fill color. We also check the pixels for a particular interior color, not for boundary color.



Algorithm of Flood-fill

```
Procedure flood_fill (p, q, fill_color, Old_color: Integer)
Var
     Current: Integer
     Current = getpixel (p, q)
 If
      (Current = Old_color)
Then
      Start
setpixel (p, q, fill_color);
         flood_fill (p, q+1, fill_color, Old_color);
         flood_fill (p, q-1, fill_color, Old_color);
         flood_fill (p+1, q, fill_color, Old_color);
         flood_fill (p-1, q, fill_color, Old_color);
    End;
```

Note- Getpixel () defines the color of specified pixel.
Setpixel () set the pixel with the specified color.



Advantages of Flood-fill algorithm

- It provides an easy way to fill color in graphics.
- The Flood-fill algorithm colors the whole area through interconnected pixels by a single color.
- The algorithm fills the same color inside the boundary.

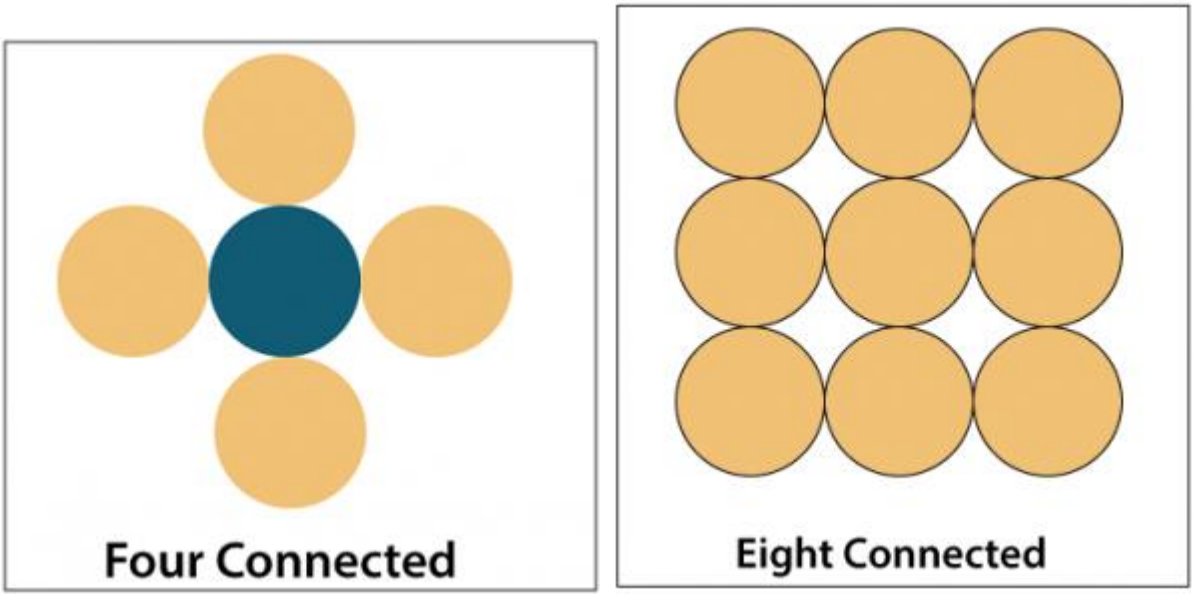
Disadvantages of Flood-fill Algorithm

- It is a more time-consuming algorithm.
- Sometimes it does not work on large polygons.

Boundary-fill Algorithm

It is also known as the **"Edge-fill algorithm."** The boundary fill algorithm is used for area filling. We can perform boundary fill where we want to create an attractive painting. In this, we can easily select the interior points. If the object has a particular boundary in a single color, then the algorithm travels each pixel until it reaches the boundary.

In the Boundary-fill algorithm, we use the **4-connected** and **8-connected** methods. By the use of a 4-connected or 8-connected method, we can set the new position of the pixels until all the interior points have been filled.



Algorithm of Boundary-fill

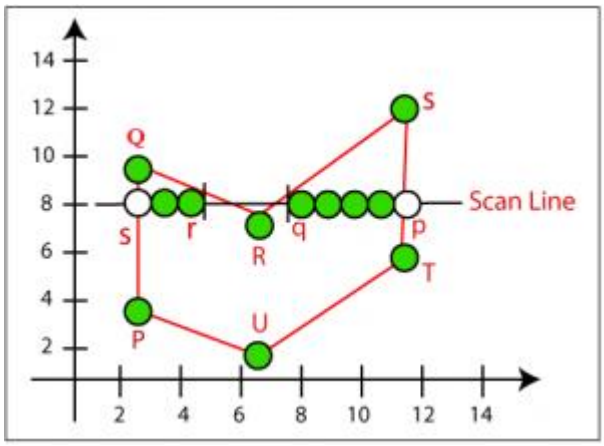
Procedure boundary fill (p, q, fill color, boundary)
Step 1: Initialize the boundary of the region.
Step 2: Get the interior pixel (p, q). Now, define an Integer called current pixel and assign it to (p, q).
Current = getpixel (p, q)
Step 3: If
(current pixel != boundary) and (current pixel != fill)
Then
Setpixel(p, q, fill);
Boundary fill (p+1, q, fill, boundary);
Boundary fill (p-1, q, fill, boundary);
Boundary fill (p, q+1, fill, boundary);
Boundary fill (p, q-1, fill, boundary);
Step 4: End;

Problems with Boundary-fill algorithm

- Sometimes it may not fill all the regions.
- In the 4-connected method, it does not fill corner pixels. Because it only checks the adjacent position of the pixel.

Scan-Line Algorithm

The Scan-Line Algorithm is an area filling algorithm. In this algorithm, we can fill the polygons through horizontal lines or scan lines. The scan-line intersects the edges of the polygon, and the polygon is filled between pairs of the intersection. The main purpose of this algorithm is to fill color in the interior pixels of the polygon.



Special Cases of Polygon Vertices
There are two special cases of polygon vertices which are given below:

1. If both lines intersecting at the vertex lies on the same side of the scan line, then we will consider it as two points.
2. If both lines intersecting at the vertex lies on the other side of the scan line, then we will consider it as a single point only.

Algorithm of Scan line polygon-fill

Step 1: Find the intersection points of the scan line that have edges.
Step 2: Now sort the intersection points by increasing the x coordinate from left to right.
Step 3: Now, we perform the pairing of the intersection points and fill the color inside the pixel pairs.

CHAPTER-4

Transformation

Introduction

The term Transformation is generally referred to as converting a graphic into another graphic by applying some rules or algorithms. Sometimes an image or picture can be a combination of lines, rectangle, circle, and triangle. If we draw the basic and combination of pictures, then there should be a need to transform these images. Now we can perform the following actions to transform the images-

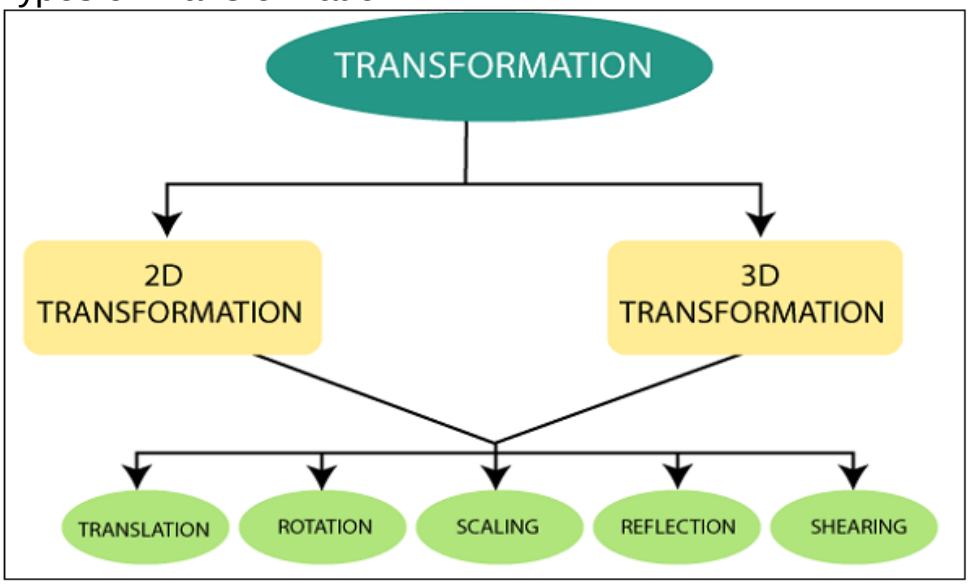
- We can change the position of an image.
- We can increase or decrease the size of an image.
- We can change the angle of the image.

By using the above actions, we will find a new image; this process is called Transformation. We can use some algorithms to produce new pictures.

The object transformation includes two important points-

- **Geometric Transformation:** When we are moving the picture, and the background is fixed, then it is a Geometric Transformation.
- **Coordinate Transformation:** When we are moving the background, and the picture is fixed, then it is Coordinate Transformation.

Types of Transformation



There are two basic kinds of Translation.

1.Two-Dimensional(2D) Transformation:

"When we translate, rotate, and scale object in the two-dimensional plane, then it is called a **Two-Dimensional Transformation."** A two-dimensional plane consists of the x and y-axis.

The Two-Dimensional Transformation includes-

- **2D Translation: "**Translation is a mechanism used to move the object from one position to another position on the screen.**"**
- **2D Rotation: "**Rotation is a process used to rotate the object from origin to a particular angle.**"**
- **2D Scaling: "**Scaling is a process or technique used to resize the object in two-dimensional plane.**"**
- **2D Reflection: "**Reflection is a mechanism or process in which we can rotate the object at the angle of 180°".
- **2D Shearing: "**Shearing is a process that is used to perform slanting on the object.**"** It is also called **"Skewing."**

2.Three-Dimensional Transformation:

"When we translate, rotate, and scale object in the three-dimensional plane then, it is called **Three-Dimensional(3D) Transformation".** A three**–**dimensional plane consists of x, y, and z-axis.

The Three-Dimensional Transformation includes-

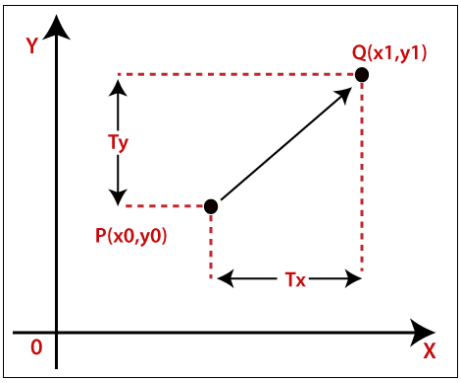
- **3D Translation: "**Translation is a mechanism used to move the object from one position to another position on the three-dimensional plane.**"**
- **3D Rotation: "**Rotation is a process used to rotate the object from origin to a particular angle in three-dimensional plane.**"**
- **3D Scaling: "**Scaling is a process or technique used to resize the object in three-dimensional plane**".**
- **3D Reflection: "**Reflection is a mechanism or process in which we can rotate the object at the angle of 180° in three-dimensional plane.**"**
- **3D Shearing: "**Shearing is a process that is used to perform slanting on the object.**"** It is also called **"Skewing"**. It also includes z-axis.

2D Translation in Computer Graphics

We can move any object from one to another place without changing the shape of the object.

For Example-

Translation of a Point: If we want to translate a point from P (x_0, y_0) to Q (x_1, y_1), then we have to add Translation coordinates (Tx, Ty) with original coordinates.



We can also represent the translation in matrix form-

$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix} = \begin{bmatrix} Tx \\ Ty \end{bmatrix} + \begin{bmatrix} x_0 \\ y_0 \end{bmatrix}$$

We can apply Translation on following object**s-**

- Line
- Rectangle
- Polygon
- Square

Homogeneous Coordinate Representation:

The above Translation is also shown in the form of 3 x 3 matrix-

$$\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & T_x \\ 0 & 1 & T_y \\ 0 & 0 & 1 \end{bmatrix} x \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix}$$

Here, Translation coordinates (T_x, T_y) are also called "**Translation** or **Shift Vector.**"

Example– Given a Point with coordinates (2, 4). Apply the translation with distance 4 towards x-axis and 2 towards the y-axis. Find the new coordinates without changing the radius?

Solution: P = (x_0, y_0) = (2,4)

Shift Vector = (T_x, T_y) = (4, 2)

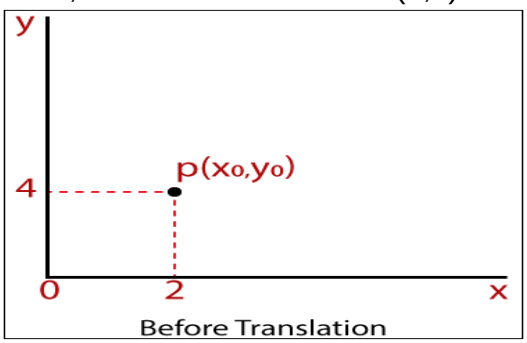
Let us assume the new coordinates of P = (x_1, y_1)

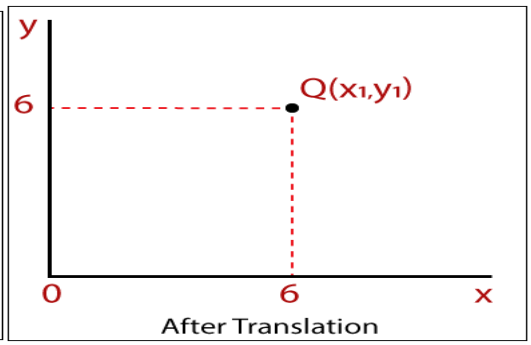
Now we are going to add translation vector and given coordinates, then

$x_1 = x_0 + T_x = (2 + 4) = 6$

$y_1 = y_0 + T_y = (4 + 2) = 6$

Thus, the new coordinates = (6,6)





2D Rotation in Computer Graphics

The Rotation of any object depends upon the two points.

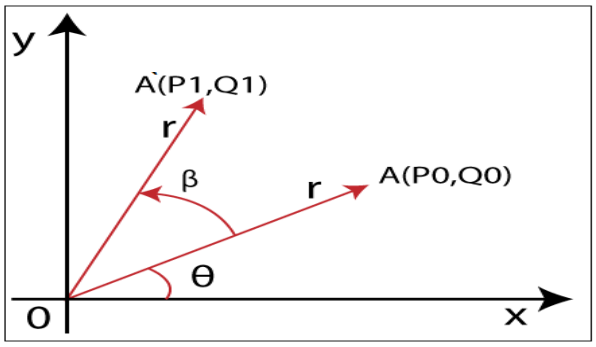
Rotation Point: It is also called **the Pivot point.**
Rotation Angle: It is denoted by **Theta (θ)**.
We can rotate an object in two ways-
Clockwise: An object rotates clockwise if the value of the Rotation angle is negative (-).
Anti-Clockwise: An object rotates anti-clockwise if the value of the Rotation angle is positive (+).
We can apply Rotation on following objects-

- **Straight Lines**
- **Curved Lines**
- **Polygon**
- **Circle**

For Example–
Rotation of a Point: If we want to Rotate a point **A (P_0, Q_0)** that has a Rotation angle with θ distance **r** from origin to **A** (P_1, Q_1) that has a Rotation angle **β.** Then, we can rotate by following Rotation equation-

$$P_1 = P_0 \cos\theta - Q_0 \sin\theta$$

$$Q_1 = P_0 \sin\theta + Q_0 \cos\theta$$



We can represent the coordinates of point A (P_0, Q_0) by using standard trigonometry-

$P_0 = r \cos\theta$..... (1)
$Q_0 = r \sin\theta$..... (2)

We can also define point A (P_1, Q_1) in the same way-

$P_1 = r \cos(\theta+\beta) = r\cos\theta\cos\beta - r\sin\theta\sin\beta$ ..... (3)
$Q_1 = r \sin(\theta+\beta) = r\cos\theta\sin\beta + r\sin\theta\cos\beta$ ..... (4)

By using equation (1) (2) (3) (4), we will get-

$P_1 = P_0 \cos\theta - P_0 \sin\theta$
$Q_1 = P_0 \sin\theta + P_0 \cos\theta$

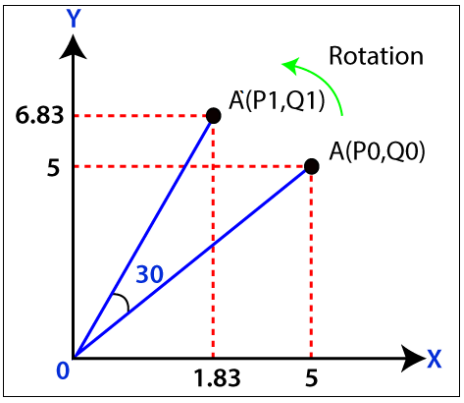
We can also represent the Rotation in the form of matrix–

$$\begin{bmatrix} P_1 \\ Q_1 \end{bmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} x \begin{bmatrix} P_0 \\ Q_0 \end{bmatrix}$$

Homogeneous Coordinates Representation: The Rotation can also be represented in the form of 3 x 3 Rotation matrix-

$$\begin{bmatrix} P_1 \\ Q_1 \\ 1 \end{bmatrix} = \begin{pmatrix} \cos\theta & \sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{pmatrix} x \begin{bmatrix} P_0 \\ Q_0 \\ 1 \end{bmatrix}$$

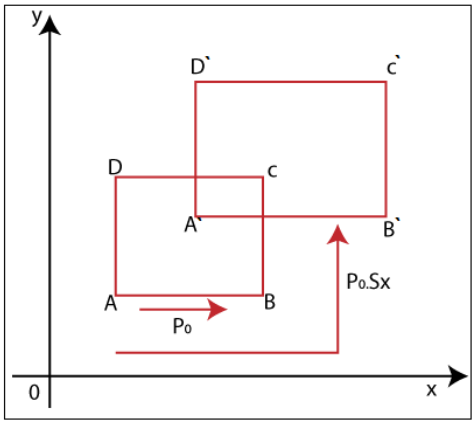
Example– A line segment with the starting point (0, 0) and ending points (5, 5). Apply 30-degree rotation anticlockwise direction on the line. Find the new coordinates of the line?
Solution– We can rotate the straight line by its endpoints with the same angle.
We have,
$(P_0, Q_0) = (0, 0)$
Rotation Angle $(\theta) = 30°$
Let the new coordinates of line = (P_1, Q_1)
We can apply the rotation matrix, then,
$P_1 = P_0 \times \cos\theta - Q_0 \times \sin\theta$
$= 5 \times \cos30 - 5 \times \sin30$
$= 5 \times (\sqrt{3}/2) - 5 \times (1/2)$
$= 4.33 - 2.5$
$= 1.83$
$Q_1 = P_0 \times \sin\theta + Q_0 \times \cos\theta$
$= 5 \times \sin30 + 5 \times \cos30$
$= 5 \times (1/2) + 5 \times (\sqrt{3}/2)$
$= 2.5 + 4.33 = 6.83$



Thus, the new endpoint coordinates of the line are = $(P_1, Q_1) = (1.83, 6.83)$

2D Scaling in Computer Graphics

In scaling, we can expend or compress the size of any object. We can apply scaling on the object by multiplying the original coordinates with scaling factors.
The term scaling factor is used to define whether the size of an object is increased or decreased. We can represent the scaling factor by **'Sx'** for the x- axis and **'Sy'** for the y-axis.
For Example– If we want to scale an object that has **R (P_0, Q_0)** coordinate and the new coordinates of an object are **R` (P_1, Q_1)** then the equation will be-
$P_1 = P_0 . S_x$
$Q_1 = Q_0 . S_y$



We can also represent Scaling in the form of Matrix-

$$\begin{bmatrix} P_1 \\ Q_1 \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} x \begin{bmatrix} P_0 \\ Q_0 \end{bmatrix}$$

Homogeneous Coordinates Representation: The scaling is also represented in the form of 3 x 3 matrix-

$$\begin{bmatrix} P_1 \\ Q_1 \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} x \begin{bmatrix} P_0 \\ Q_0 \\ 1 \end{bmatrix}$$

Example: A Square object with the coordinate points P (1, 4), Q (4, 4), R (4, 1), T (1,1). Apply the scaling factor 3 on the X-axis and 4 on the Y-axis. Find out the new coordinates of the square?

Solution: We have,

Coordinates Points for Square = P (1, 4), Q (4, 4), R (4, 1), S (1, 1)

Scaling factor along with X axis (Sx) = 3

Scaling factor along with Y axis (Sy)= 4

Applying the equation to find the new coordinates-

For Coordinate P (1, 4)–

Let the new Coordinate for P **=** (P_1, Q_1)

$P_1 = P_0$. Sx **=** 1 x 3 = 3

$Q_1 = Q_0$. Sy = 4 x 4 = 16

The new Coordinates = (3, 16)

For Coordinate Q (4, 4)–

Let the new Coordinate for Q **=** ($P_{,1}$ Q_1)

$P_1 = P_0$. Sx **=** 4 x 3 = 12

$Q_1 = Q_0$. Sy = 4 x 4 = 16

The new Coordinates = (12, 16)

For Coordinate R (4, 1)–

Let the new Coordinate for R **=** (P_1, Q_1)

$P_1 = P_0$. Sx **=** 4 x 3 = 12

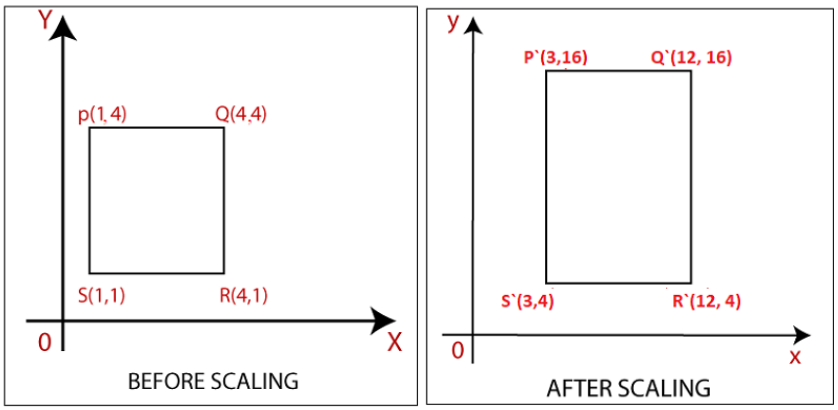
$Q_1 = Q_0$. Sy = 1 x 4 = 4

The new Coordinates = (12, 4)

For Coordinate S (1, 1)–

Let the new Coordinate for S **=** (P_1, Q_1)

$P_1 = P_0$. Sx = 1 x 3 = 3
$Q_1 = Q_0$. Sy = 1 x 4 = 4
The new Coordinates = (3, 4)



Thus, the new coordinates for square after scaling = P (3, 16), Q (12, 16), R (12, 4), S (3, 4).

2D Reflection in Computer Graphics

The Reflection is a mirror image of the original object. In the Reflection process, the size of the object does not change.

We can represent Reflection by using four ways-

1. **Reflection along X-axis:** In this kind of Reflection, the value of X is positive, and the value of Y is negative.

We can represent the Reflection along x-axis by following equation-

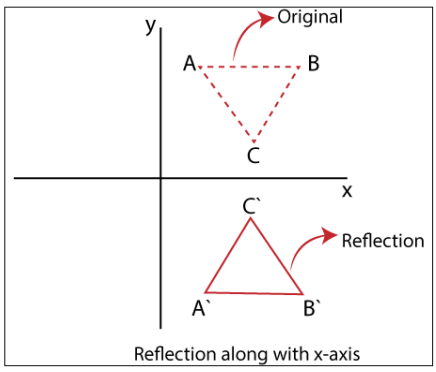
$X_1 = X_0$
$Y_1 = -Y_0$

We can also represent Reflection in the form of matrix–

$$\begin{bmatrix} X_1 \\ Y_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} x \begin{bmatrix} X_0 \\ Y_0 \end{bmatrix}$$

Homogeneous Coordinate Representation: We can also represent the Reflection along x-axis in the form of 3 x 3 matrix-

$$\begin{bmatrix} X_1 \\ Y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} x \begin{bmatrix} X_0 \\ Y_0 \\ 1 \end{bmatrix}$$

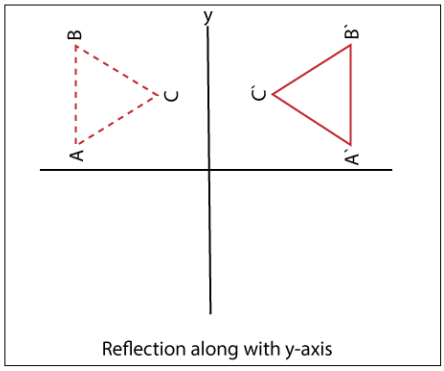


2. **Reflection along Y-axis:** In this kind of Reflection, the value of X is negative, and the value of Y is positive.
We can represent the Reflection along y-axis by following equation-
$X_1 = -X_0$
$Y_1 = Y_0$
We can also represent Reflection in the form of matrix–

$$\begin{bmatrix} X_1 \\ Y_1 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} x \begin{bmatrix} X_0 \\ Y_0 \end{bmatrix}$$

Homogeneous Coordinate Representation: We can also represent the Reflection along x-axis in the form of 3 x 3 matrix-

$$\begin{bmatrix} X_1 \\ Y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} x \begin{bmatrix} X_0 \\ Y_0 \\ 1 \end{bmatrix}$$

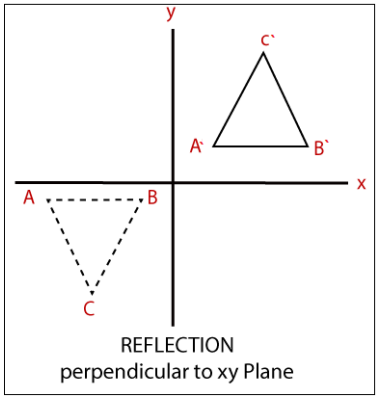


3. Reflection perpendicular to XY plane: In this kind of Reflection, the value of both X and Y is negative.
We can represent the Reflection along y-axis by following equation-
$X_1 = -X_0$
$Y_1 = -Y_0$
We can also represent Reflection in the form of matrix–

$$\begin{bmatrix} X_1 \\ Y_1 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} x \begin{bmatrix} X_0 \\ Y_0 \end{bmatrix}$$

Homogeneous Coordinate Representation: We can also represent the Reflection along with x-axis in the form of 3 x 3 matrix-

$$\begin{pmatrix} X_1 \\ Y_1 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} x \begin{pmatrix} X_0 \\ Y_0 \\ 1 \end{pmatrix}$$

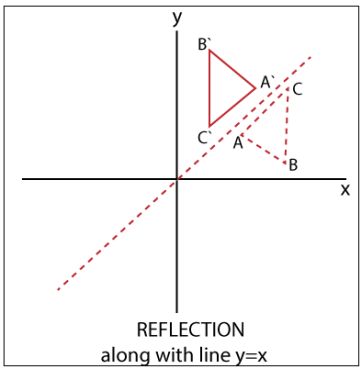


4. **Reflection along with the line:** In this kind of Reflection, the value of X is equal to the value of Y.
We can represent the Reflection along y-axis by following equation-
Y=X, then the points are **(Y, X)**
Y= –X, then the points are **(–Y, –X)**
We can also represent Reflection in the form of matrix–

$$\begin{pmatrix} X_1 \\ Y_1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} x \begin{pmatrix} X_0 \\ Y_0 \end{pmatrix}$$

Homogeneous Coordinate Representation: We can also represent the Reflection along with x-axis in the form of 3 x 3 matrix-

$$\begin{pmatrix} X_1 \\ Y_1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix} x \begin{pmatrix} X_0 \\ Y_0 \\ 1 \end{pmatrix}$$



2D Shearing in Computer Graphics

We can denote shearing with **'SH_x'** and **'SH_y.'** These 'SH_x' and **'SH_y'** are called **"Shearing factor."**

We can perform shearing on the object in two ways-

1. **Shearing along x-axis:** In this, wecan store the y coordinate and only change the x coordinate. It is also called **"Horizontal Shearing."**

We can represent Horizontal Shearing by the following equation-

$$X_1 = X_0 + SH_x . Y_0$$

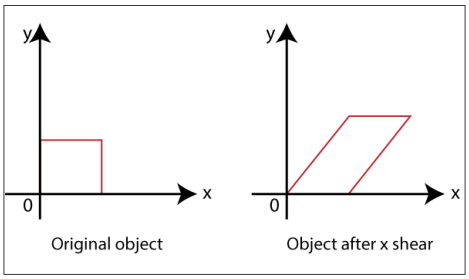
$$Y_1 = Y_0$$

We can represent Horizontal shearing in the form of matrix–

$$\begin{pmatrix} X_1 \\ Y_1 \end{pmatrix} = \begin{pmatrix} 1 & SH_x \\ 0 & 1 \end{pmatrix} x \begin{pmatrix} X_0 \\ Y_0 \end{pmatrix}$$

Homogeneous Coordinate Representation: The 3 x 3 matrix for Horizontal Shearing is given below-

$$\begin{pmatrix} X_1 \\ Y_1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ SH_x & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} x \begin{pmatrix} X_0 \\ Y_0 \\ 1 \end{pmatrix}$$



2. **Shearing along y-axis:** In this, wecan store the x coordinate and only change the y coordinate. It is also called **"Vertical Shearing."**

We can represent Vertical Shearing by the following equation-

$$X_1 = X_0$$

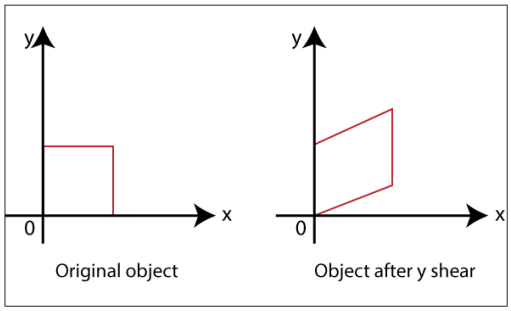
$$Y_1 = Y_0 + SH_y . X_0$$

We can represent Vertical Shearing in the form of matrix–

$$\begin{bmatrix} X_1 \\ Y_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ SH_y & 1 \end{bmatrix} x \begin{bmatrix} X_0 \\ Y_0 \end{bmatrix}$$

Homogeneous Coordinate Representation: The 3×3 matrix for Vertical Shearing is given below-

$$\begin{bmatrix} X_1 \\ Y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & SH_y & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} x \begin{bmatrix} X_0 \\ Y_0 \\ 1 \end{bmatrix}$$



Example: A Triangle with (2, 2), (0, 0) and (2, 0). Apply Shearing factor 2 on X-axis and 2 on Y-axis. Find out the new coordinates of the triangle?

Solution: We have,

The coordinates of triangle = P (2, 2), Q (0, 0), R (2, 0)

Shearing Factor for X-axis = 2

Shearing Factor for Y-axis = 2

Now, apply the equation to find the new coordinates.

Shearing for X-axis:

For Coordinate P (2, 2)-

Let the new coordinate for P = (X_1, Y_1)

$X_1 = X_0 + SH_x . Y_0 = 2 + 2 \times 2 = 6$

$Y_1 = Y_0 = 2$

The New Coordinates = (6, 2)

For Coordinate Q (0, 0)-

Let the new coordinate for Q = (X_1, Y_1)

$X_1 = X_0 + SH_x . Y_0 = 0 + 2 \times 0 = 0$

$Y_1 = Y_0 = 0$

The New Coordinates = (0, 0)

For Coordinate R (2, 0)-

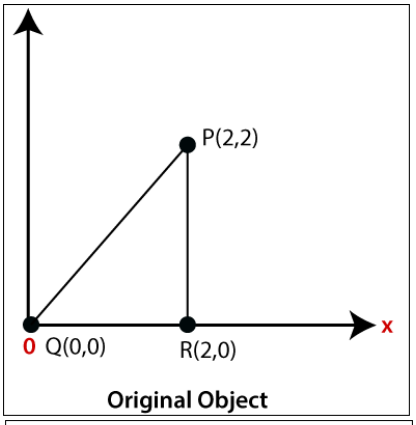
Let the new coordinate for R = (X_1, Y_1)

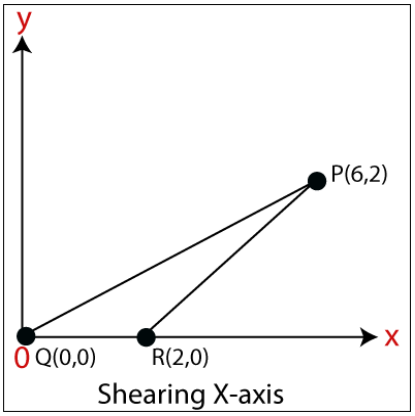
$X_1 = X_0 + SH_x . Y_0 = 2 + 2 \times 0 = 2$

$Y_1 = Y_0 = 0$

The New Coordinates = (2, 0)

The New coordinates of triangle for x-axis = (6, 2), (0, 0), (2, 0)





Shearing for y-axis:

For Coordinate P (2, 2)-

Let the new coordinate for P = (X_1, Y_1)

$X_1 = X_0 = 2$

$Y_1 = Y_0 + Sh_y.X_0 = 2 + 2 \times 2 = 6$

The New Coordinates = (2, 6)

For Coordinate Q (0, 0)-

Let the new coordinate for Q = (X_1, Y_1)

$X_1 = X_0 = 0$

$Y_1 = Y_0 + Sh_y. X_0 = 0 + 2 \times 0 = 0$

The New Coordinates = (0, 0)

For Coordinate R (2, 0)-

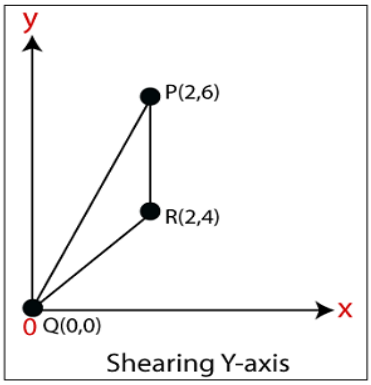
Let the new coordinate for R = (X_1, Y_1)

$X_1 = X_0 = 2$

$Y_1 = Y_0 + Sh_y. X_0 = 0 + 2 \times 2 = 4$

The New Coordinates = (2, 4)

The New coordinates of triangle for y-axis = (2, 6), (0, 0), (2, 4)



Chapter-5

Computer Graphics Window

"**The process of selecting and viewing an image with different views, called windowing.**"

All the objects in the real world have a size. We can measure the size and location of an object by the unit.

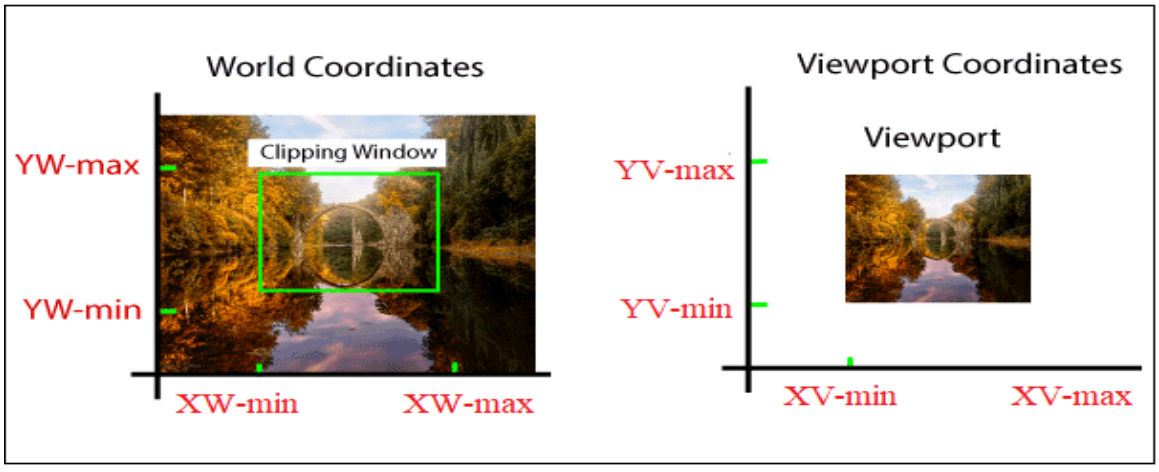
For Example-We use the meter unit to measure both size and the location of the object. When we represent the image of an object on the screen, then we use the screen coordinate system. The screen coordinate system is used to define the location of the object. When we select the screen coordinate system, then the image can be displayed on the screen.
"The Capability to show some part of an object in a window is known as **windowing.**"
"The rectangular area describes in the world coordinate system is called **the window**."

Viewport:

"The viewport can be defined as an area on the screen which is used to display the object." The window is an area space for the object. Viewport surrounds the object. We need the coordinate transformation to display the object on the screen. We can define many viewports on a different area of the screen and also see the same object from a different angle in the viewport.

Window to Viewport transformation:

"Window to viewport transformation is a process of converting two-dimensional or world into a device coordinate."
The object inside the clipping window is mapped to the viewport. The viewport is displayed inside the interface window on the screen. We can use the clipping window to select the part of an object, and the viewport is used to display the selected part of the object on the screen or output device.

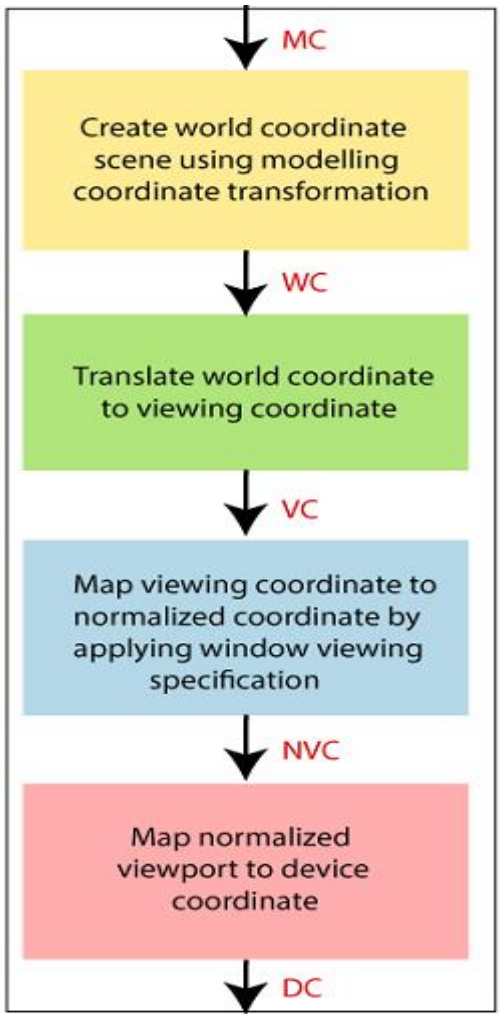


- **Steps for Window to Viewport Transformation:** We can follow the following steps for transform window to viewport:

Step 1: Translation of the window towards the Origin– If we shift the window towards the origin, the upper left, and the lower-left corner of the window will be negative (-). The translation factor also should be negative (-).
Step 2: Resize the window to viewport size– To Convert the size of the window into the size of the viewport, we will use the following formulas:

$$S_x = XV_{max} - XV_{min} / XW_{max} - XW_{min}$$
$$S_y = YV_{max} - YV_{min} / YW_{max} - YW_{min}$$

Step 3: Translation of window (Position of window and viewport must be same)– If the lower-left corner of viewport is (0, 0), then the window lower-left corner is already shifted on origin after following step 1.
If the lower-left corner is not equal to (0, 0), then the translation factor should be positive (+).



It may be possible that sometimes the size of viewport is greater or smaller than the window. **In that situation, we have to enlarge or compress the size of the window according to the viewport. We can perform it using some mathematical calculations.**
A point on window = (XW, YW)

Corresponding point on viewport = (XV, YV)

- **To calculate (XV, YV)** –

Normalized point on window = $(XW - XW_{min} / XW_{max} - XW_{min})$
$(YW - YW_{min} / YW_{max} - YW_{min})$
Normalized point on viewport = $(XV - XV_{min} / XV_{max} - XV_{min})$
$(YV - YV_{min} / YV_{max} - YV_{min})$

- **Now, the position of object in viewport and window are same-**

For Coordinate x:
$(XW - XW_{min} / XW_{max} - XW_{min}) = (XV - XV_{min} / XV_{max} - XV_{min})$
For Coordinate y:
$(YW - YW_{min} / YW_{max} - YW_{min}) = (YV - YV_{min} / YV_{max} - YV_{min})$
Now, we get
$XV = XV_{min} + (XW - XW_{min}) S_x$
$YV = YV_{min} + (YW - YW_{min}) S_y$
Here S_x and S_y are the scaling factor for x and y coordinate.
$\mathbf{S_x = XV_{max} - XV_{min} / XW_{max} - XW_{min}}$
$\mathbf{S_y = YV_{max} - YV_{min} / YW_{max} - YW_{min}}$

Some Important Points:

1. We can select the world coordinate system according to the application program.
2. We can select the screen coordinate system according to the desired design.
3. Viewing transformation is a connection between the world and the screen coordinates.

Advantages:

We can easily represent images on the display screen according to the user's needs.

CHAPTER - 6

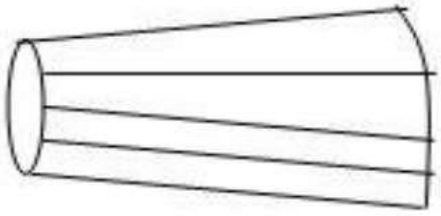
THREE-DIMENSIONAL OBJECT REPRESENTATION

3D object representation is divided into two categories.

- **Boundary Representations (B-reps)** – It describes a 3D object as a set of surfaces that separates the object interior from the environment.
- **Space–partitioning representations –** It is used to describe interior properties, by partitioning the spatial region containing an object into a set of small, non-overlapping, contiguous solids (usually cubes).

Polygon Surfaces

- The most commonly used representation for a 3D graphics object.
- It is a set of surface polygons that enclose the object interior.
- Set of polygons are stored for object description.
- This simplifies and speeds up the surface rendering and display of object since all surfaces can be described with linear equations.

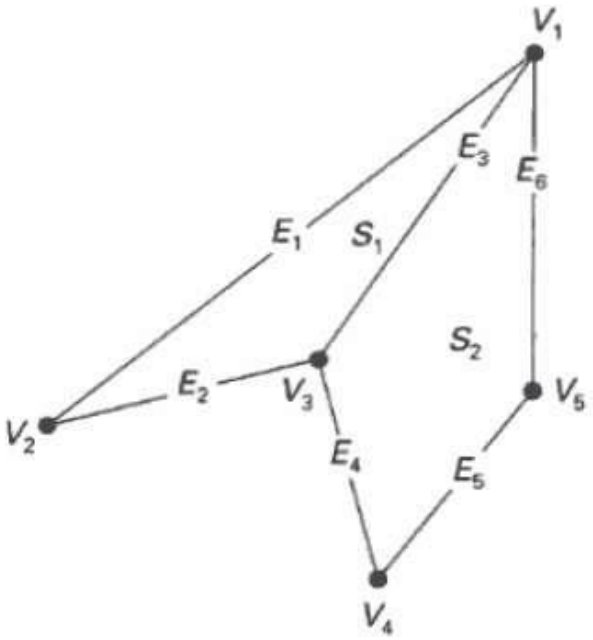


A 3D object represented by polygons

Three ways to represent polygon surfaces

1. Polygon Tables
2. Plane Equations
3. Polygon Meshes

1. **Polygon Tables** The object is store by using three tables
   - Vertex Table
   - Edge table
   - Polygon-Surface table



VERTEX TABLE	
V_1:	x_1, y_1, z_1
V_2:	x_2, y_2, z_2
V_3:	x_3, y_3, z_3
V_4:	x_4, y_4, z_4
V_5:	x_5, y_5, z_5

EDGE TABLE	
E_1:	V_1, V_2
E_2:	V_2, V_3
E_3:	V_3, V_1
E_4:	V_3, V_4
E_5:	V_4, V_5
E_6:	V_5, V_1

POLYGON-SURFACE TABLE	
S_1:	E_1, E_2, E_3
S_2:	E_3, E_4, E_5, E_6

- **Vertex Table**

It store x, y, and z coordinate information of all the vertices as v1 : x1 , y1 , z1 .

- **Edge table**
  - The Edge table is used to store the edge information of polygon.
  - In the following figure, edge E1 lies between vertex v1 and v2 which is represented in the table as E1 : v1 , v2 .
- **Polygon-Surface table**

Polygon surface table stores the number of surfaces present in the polygon.

From the following figure, surface S1 is covered by edges E1 , E2 and E3 which can be represented in the polygon surface table as S1 : E1 , E2 , and E3

1. Plane Equations

The equation for plane surface can be expressed as −

$$Ax + By + Cz + D = 0$$

Where x,y,z is any point on the plane, and the coefficients A, B, C, and D are constants describing the spatial properties of the plane. We can obtain the values of A, B, C, and D by solving a set of three plane equations using the coordinate values for three non collinear points in the plane. Let us assume that three vertices of the plane are (x_1, y_1, z_1), (x_2, y_2, z_2) and (x_3, y_3, z_3).

Let us solve the following simultaneous equations for ratios A/D, B/D, and C/D. You get the values of A, B, C, and D.

$$A/D\,x_1 + B/D\,y_1 + C/D\,z_1 = -1$$
$$A/D\,x_2 + B/D\,y_2 + C/D\,z_2 = -1$$
$$A/D\,x_3 + B/D\,y_3 + C/D\,z_3 = -1$$

To obtain the above equations in determinant form, apply Cramer's rule to the above equations.

$$A = \begin{bmatrix} 1 & y_1 & z_1 \\ 1 & y_2 & z_2 \\ 1 & y_3 & z_3 \end{bmatrix} \quad B = \begin{bmatrix} x_1 & 1 & z_1 \\ x_2 & 1 & z_2 \\ x_3 & 1 & z_3 \end{bmatrix} \quad C = \begin{bmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix} \quad D = -\begin{bmatrix} x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \\ x_3 & y_3 & z_3 \end{bmatrix}$$

For any point (x, y, z) with parameters A, B, C, and D, we can say that −

- Ax + By + Cz + D ≠ 0 means the point is not on the plane.
- Ax + By + Cz + D < 0 means the point is inside the surface.
- Ax + By + Cz + D > 0 means the point is outside the surface.

3 Polygon Meshes

3D surfaces and solids can be approximated by a set of polygonal and line elements. Such surfaces are called **polygonal meshes**. In polygon mesh, each edge is shared by at most two polygons. The set of polygons or faces, together form the "skin" of the object.

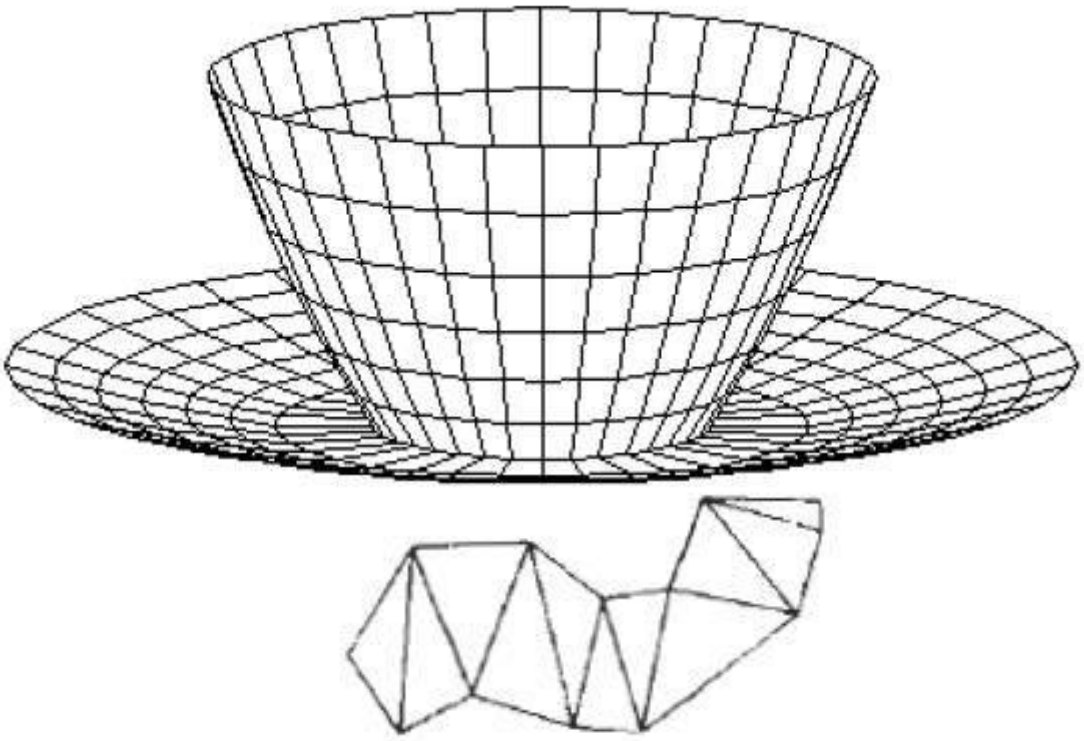


Figure 10-6
A triangle strip formed with 11 triangles connecting 13 vertices.

Advantages

- It can be used to model almost any object.
- They are easy to represent as a collection of vertices.
- They are easy to transform.
- They are easy to draw on computer screen.

Disadvantages

- Curved surfaces can only be approximately described.
- It is difficult to simulate some type of objects like hair or liquid.

CHAPTER-7

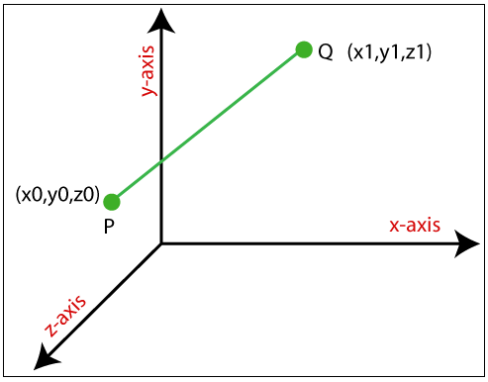
3D Transformation

3D Translation

A 3DTranslation process contains the x-axis, y-axis, and z-axis. We can move any object from one place to another without changing the shape of the object.

For Example-

Translation of a Point: If we want to translate a point from P (x_0, y_0, z_0) to Q (x_1, y_1, z_1), then we have to add Translation coordinates (T_x, T_y, T_z) with original coordinates.



We can also represent the 3D Translation in matrix form-

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \end{bmatrix} = \begin{bmatrix} T_x+x_0 \\ T_y+y_0 \\ T_z+z_0 \end{bmatrix}$$

We can apply Translation on the following object**s-**

- Line
- Rectangle
- Polygon
- Square

3D Translation Matrix Representation:

The above Translation is also shown in the form of 3 x 3 matrix-

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}$$

Here Translation coordinates (**T_x, T_y, T_z**) are also called "**Translation** or **Shift Vector.**"

Example: A Point has coordinates P (1, 2, 3) in x, y, z-direction. Apply the translation with a distance of 2 towards x-axis, 3 towards y-axis, and 4 towards the z-axis. Find the new coordinates of the point?

Solution: We have,

Point P = (x_0, y_0, z_0) = (1,2,3)

Shift Vector = (T_x, T_y, T_z)

Let us assume the new coordinates of P = (x_1, y_1, z_1)

Now we are going to add translation vector and given coordinates, then

$X_1 = x_0 + T_x = (1 + 2) = 3$

$Y_1 = y_0 + T_y = (2 + 3) = 5$
$Z_1 = z_0 + T_z = (3 + 4) = 7$
Thus, the new coordinates are = (3, 5, 7)

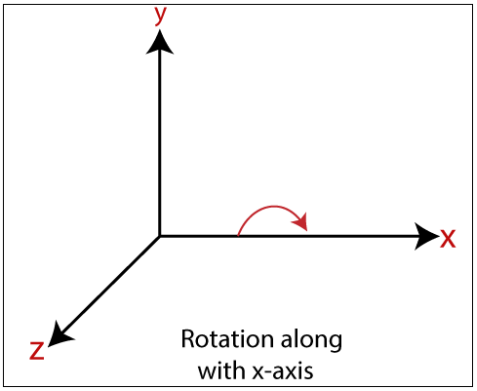
3D Rotation

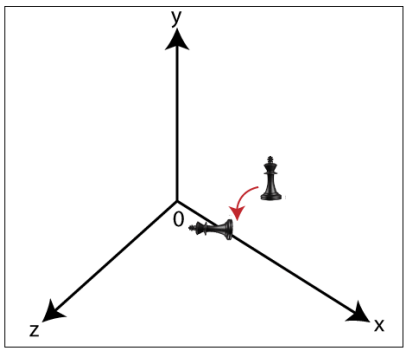
The 3D rotation is different from 2D rotation. In 3D Rotation we also have to define the angle of Rotation with the axis of Rotation.
For Example-Let us assume**,**
The initial coordinates of an object = (x_0, y_0, z_0)
The Initial angle from origin = **β**
The Rotation angle = **θ**
The new coordinates after Rotation = (x_1, y_1, z_1)
In Three-dimensional plane we can define Rotation by following three ways–

1. **X-axis Rotation:** We can rotate the object along x-axis. We can rotate an object by using following equation-

$X_1 = x_0$
$Y_1 = y_0 \cos\theta - z_0 \times \sin\theta$
$Z_1 = y_0 \times \sin\theta + z_0 \times \cos\theta$
We can represent 3D rotation in the form of matrix-

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}$$

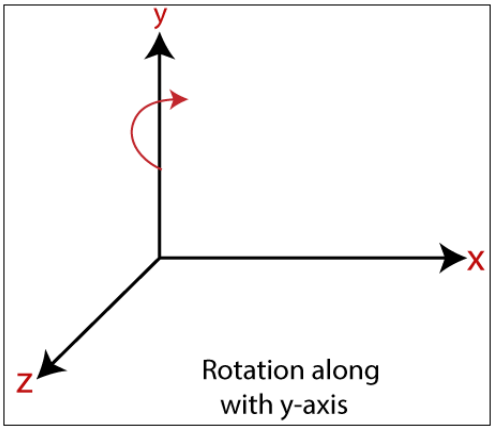


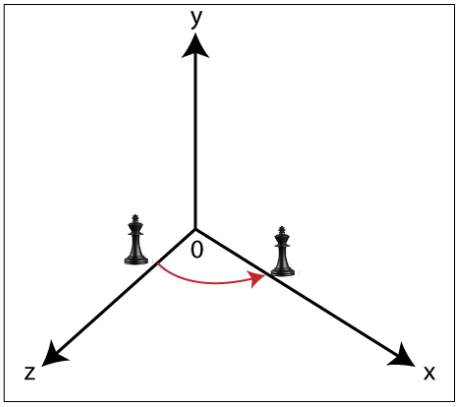


2. Y-axis Rotation: We can rotate the object along y-axis. We can rotate an object by using following equation-

$x_1 = z_0 \times \sin\theta + x_0 \times \cos\theta$
$y_1 = y_0$
$z_1 = y_0 \times \cos\theta - x_0 \times \sin\theta$
We can represent 3D rotation in the form of matrix–

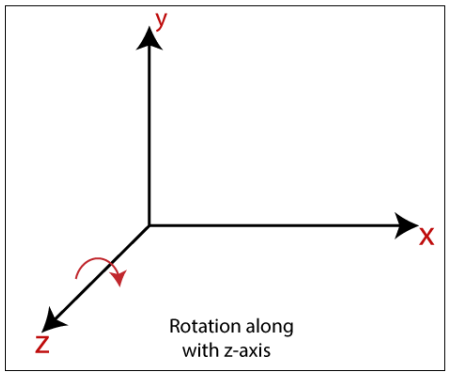
$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix} = \begin{pmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} x \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}$$

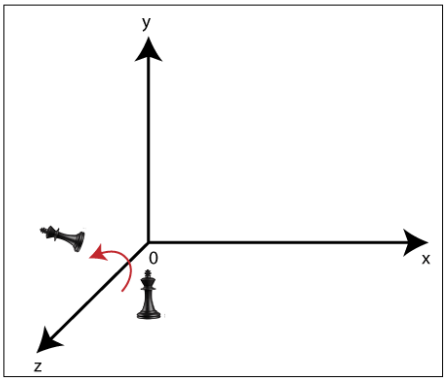




3. Z-axis Rotation: We can rotate the object along z-axis. We can rotate an object by using following equation-
$x_1 = x_0 \times \cos\theta - y_0 \times \sin\theta$
$y_1 = x_0 \times \sin\theta + y_0 \times \cos\theta$
$z_1 = z_0$
We can represent 3D rotation in the form of matrix–

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} x \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}$$





Example: A Point has coordinates P (2, 3, 4) in x, y, z-direction. The Rotation angle is 90 degrees. Apply the rotation in x, y, z direction, and find out the new coordinates of the point?

Solution: The initial coordinates of point = P (x_0, y_0, z_0) = (2, 3, 4)

Rotation angle (**θ**) = 90°

For x-axis–

Let the new coordinates = (x_1, y_1, z_1) then,

$x_1 = x_0 = 2$

$y_1 = y_0 \times \cos\theta - z_0 \times \sin\theta = 3 \times \cos90° - 4 \times \sin90° = 3 \times 0 - 4 \times 1 = -4$

$z_1 = y_0 \times \sin\theta + z_0 \times \cos\theta = 3 \times \sin90° + 4 \times \cos90° = 3 \times 1 + 4 \times 0 = 3$

The new coordinates of point = (2, -4, 3)

For y-axis–

Let the new coordinates = (x_1, y_1, z_1) then,

$X_1 = z_0 \times \sin\theta + x_0 \times \cos\theta = 4 \times \sin90° + 2 \times \cos90° = 4 \times 1 + 2 \times 0 = 4$

$y_1 = y_0 = 3$

$z_1 = y_0 \times \cos\&\theta - x_0 \times \sin\theta = 3 \times \cos90° - 2 \times \sin90° = 3 \times 0 - 4 \times 0 = 0$

The new coordinates of point = (4, 3, 0)

For z-axis–

Let the new coordinates = (x_1, y_1, z_1) then,

$x_1 = x_0 \times \cos\theta - y_0 \times \sin\theta = 2 \times \cos90° - 3 \times \sin90° = 2 \times 0 + 3 \times 1 = 3$

$y_1 = x_0 \times \sin\theta + y_0 \times \cos\theta = 2 \times \sin90° + 3 \times \cos90° = 2 \times 1 + 3 \times 0 = 2$

$z_1 = z_0 = 4$

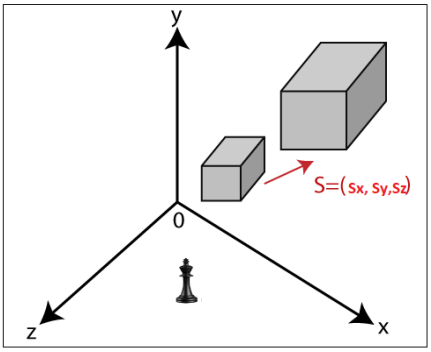
The New Coordinates of points = (3, 2, 4)

3D Scaling

The 2D and 3D scaling are similar, but the key difference is that the 3D plane also includes the z-axis along with the x and y-axis.

In scaling, we can expend or compress the size of any object. We can apply scaling on the object by multiplying the original coordinates with scaling factors.

The term scaling factor is used to define whether the size of the object is increased or decreased. We can represent the scaling factor by **'S_x'** for the x-axis, **'S_y'** for the y-axis, and **'S_z'** for the z-axis.
The increment and decrement of an object is depends on two conditions. They are-
If scaling factor (S_x, S_y, S_z) > 1, then the size of the object increased.
If scaling factor (S_x, S_y, S_z) < 1, then the size of the object decreased.
For Example: Let us assume,
The initial coordinates of object = P (x_0, y_0, z_0)
Scaling factor for x-axis = S_x
Scaling factor for y-axis = S_y
Scaling factor for z-axis = S_z
The coordinates after Scaling = Q (x_1, y_1, z_1)
We can represent the 3D Scaling in the form of equation-
$X_1 = x_0 . S_x$
$Y_1 = y_0 . S_y$
$Z_1 = z_0 . S_z$



Matrix representation of 3D Scaling-

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} = \begin{pmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} x \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix}$$

Example: A 3D object that have coordinates points P(1, 4, 4), Q(4, 4, 6), R(4, 1, 2), T(1, 1, 1) and the scaling parameters are 3 along with x-axis, 4 along with y-axis and 4 along with z-axis. Apply scaling to find the new coordinates od the object?
Solution: we have,
The initial coordinates of object = P (1, 4, 4), Q (4, 4, 6), R (4, 1, 2), S (1, 1, 1)
Scaling factor along with x-axis (S_x) = 3
Scaling factor along with y-axis (S_y) = 4
Scaling factor along with z-axis (S_z) = 4
Let the new coordinates after scaling = **(x_1, y_1, z_1)**
For coordinate P-
$x_1 = x_0 \times S_x = 1 \times 3 = 3$
$y_1 = y_0 \times S_y = 4 \times 4 = 16$
$z_1 = z_0 \times S_z = 4 \times 4 = 16$
The new coordinates = (3, 16, 16)
For coordinate Q-
$x_1 = x_0 \times S_x = 4 \times 3 = 12$
$y_1 = y_0 \times S_y = 4 \times 4 = 16$

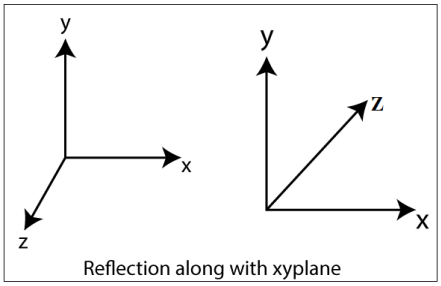
$z_1 = z_0 \times S_z = 6 \times 4 = 24$
The new coordinates = (12, 16, 24)
For coordinate R-
$x_1 = x_0 \times S_x = 4 \times 3 = 12$
$y_1 = y_0 \times S_y = 1 \times 4 = 4$
$z_1 = z_0 \times S_z = 2 \times 4 = 8$
The new coordinates = (12, 4, 8)
For coordinate S-
$x_1 = x_0 \times S_x = 1 \times 3 = 3$
$y_1 = y_0 \times S_y = 1 \times 4 = 4$
$z_1 = z_0 \times S_z = 1 \times 4 = 4$
The new coordinates = (3, 4, 4)
Thus, the new coordinates after scaling = P (3, 16, 16), Q (12, 16, 24), R (12, 4, 8), S (3, 4, 4).

3D Reflection

The Reflection is a mirror image of the original object. We can differentiate 2D and 3D reflection by adding Z-axis. The Z-axis shows the depth of the surface. In the Reflection process, the size of the object does not change.
We can represent Reflection by using the following three ways-

1. **Reflection along with xy Plane:** In the xy plane reflection, the value of z is negative.

$x_1 = x_0$
$y_1 = y_0$
$z_1 = -z_0$

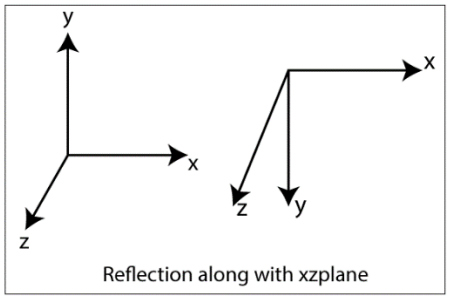


Reflection along with xyplane

Matrix of 3D Reflection-

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} x \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix}$$

2. Reflection along with xz Plane: In the xz plane reflection the value of y is negative.
$x_1 = x_0$
$y_1 = -y_0$
$z_1 = z_0$

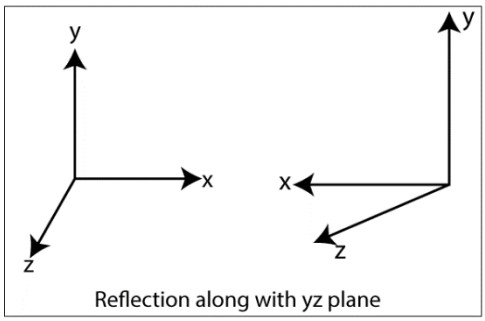


Reflection along with xzplane

Matrix of 3D Reflection-

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} x \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix}$$

3. Reflection along with yz Plane: In the yz plane reflection the value of x is negative.
$x_1 = -x_0$
$y_1 = y_0$
$z_1 = z_0$



Reflection along with yz plane

Matrix of 3D Reflection-

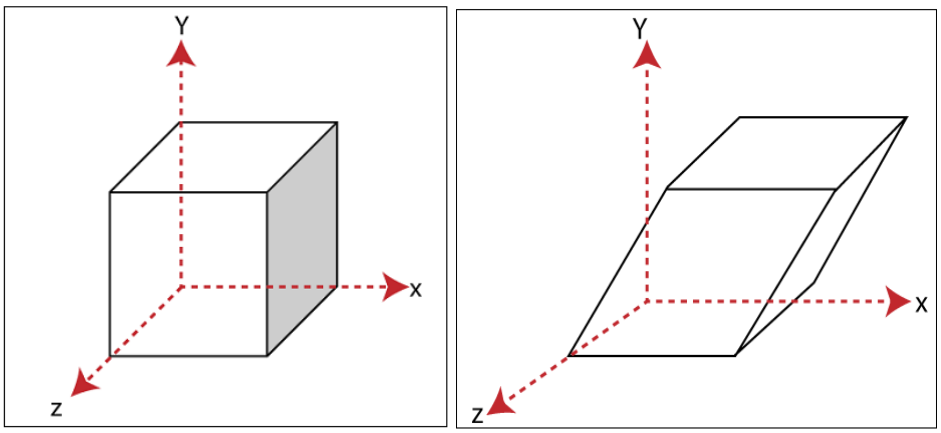
$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} x \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix}$$

Example: A 3D triangle with coordinates points P (4, 5, 2), Q (7, 5, 3), R (6, 7, 4). Apply reflection on xy plane and find the new coordinates of triangle?
Solution: We have,
The initial coordinates of triangle = P (4, 5, 2), Q (7, 5, 3), R (6, 7, 4)
Reflection Plane = xy
Let the new coordinates of triangle = (x_1, y_1, z)
For Coordinate P (4, 5, 2)–
$X_1 = x_0 = 4$
$y_1 = y_0 = 5$
$z_1 = -z_0 = -2$
The new coordinates = (4, 5, -2)
For Coordinate Q (7, 5, 3)–

$X_1 = x_0 = 7$
$Y_1 = y_0 = 5$
$Z_1 = -z_0 = -3$
The new coordinates = (7, 5, -3)
For Coordinate P (6, 7, 4)–
$X_1 = x_0 = 6$
$y_1 = y_0 = 7$
$z_1 = -z_0 = -4$
The new coordinates = (6, 7, 4)

3D Shearing

We can denote shearing with **'SH_x,' 'SH_y,' and 'SH_z.'** These 'SH_x,' 'SH_y,' 'SH_z' are called **"Shearing factor."**
The basic difference between 2D and 3D Shearing is that the 3D plane also includes the z-axis.



We can perform shearing on the object by following three ways-

1. **Shearing along the x-axis:** In this, we can store the x coordinate and only change the y and z coordinate.

We can represent shearing along x-axis by the following equation-
$x_1 = x_0$
$y_1 = y_0 + SH_y . x_0$
$z_1 = z_0 + SH_z . x_0$

3D Shearing Matrix:

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ SH_y & 1 & 0 & 0 \\ SH_z & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} x \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}$$

2. Shearing along the y-axis: In this, we can store the y coordinate and only change the x and z coordinate.
We can represent shearing along with y-axis by the following equation-
$x_1 = x_0 + SH_x . y_0$
$y_1 = y_0$
$z_1 = z_0 + SH_z . y_0$

3D Shearing Matrix:

$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & SH_x & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & SH_z & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} x \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix}$$

3. Shearing along with z-axis: In this, we can store the z coordinate and only change the x and y coordinate.

We can represent shearing along with z-axis by the following equation-

$\mathbf{x_1 = x_0 + SH_x . z_0}$

$\mathbf{y_1 = y_0 + SH_y . Z_0}$

$\mathbf{z_1 = z_0}$

3D Shearing Matrix:

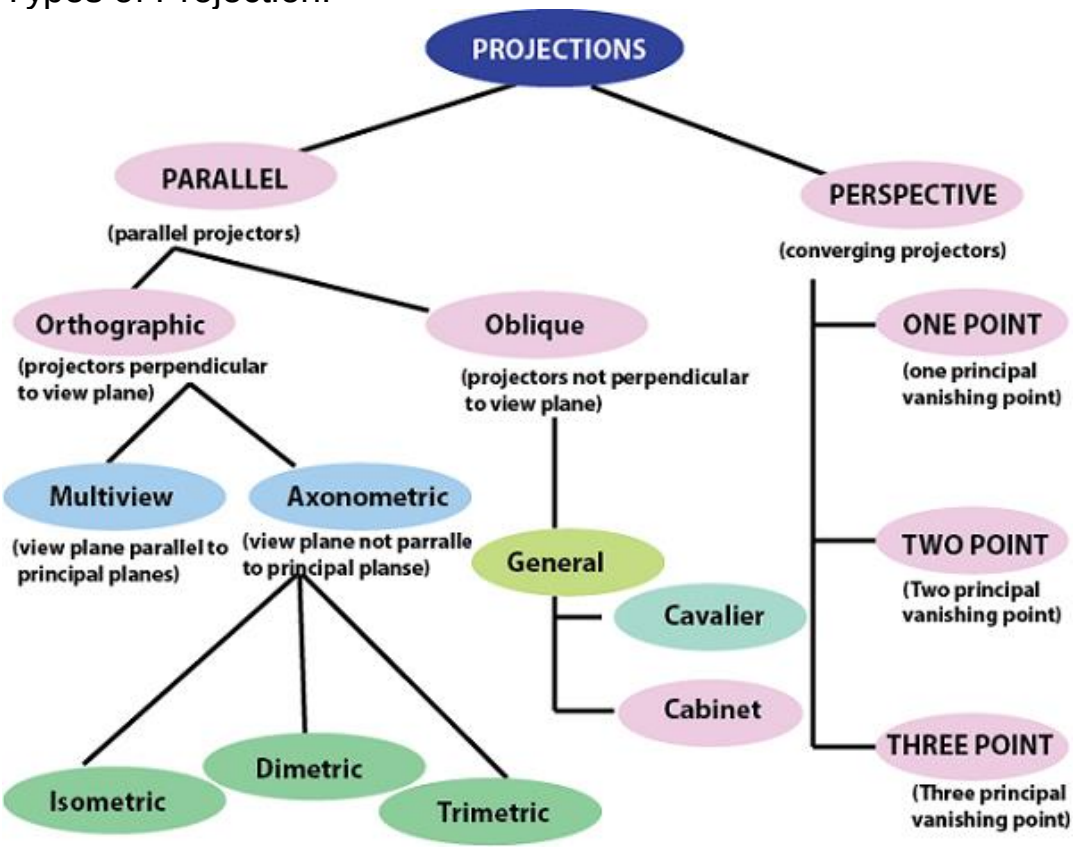
$$\begin{pmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & SH_x & 0 \\ 0 & 1 & SH_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} x \begin{pmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{pmatrix}$$

CHAPTER-8

Projection

Projection : The technique projection was invented by the Swiss mathematician, engineer, and astronomer **"Leonhard Euler Around"** in 1756. The "**Episcope**" was the first projection system.
"Projection is a technique or process which is used to transform a 3D object into a 2D plane." In other words, we can define **"projection as a mapping of points P (x, y, z) on to its image P' (x,' y,'z') in the projection plane or view plane, which create the display surface."**

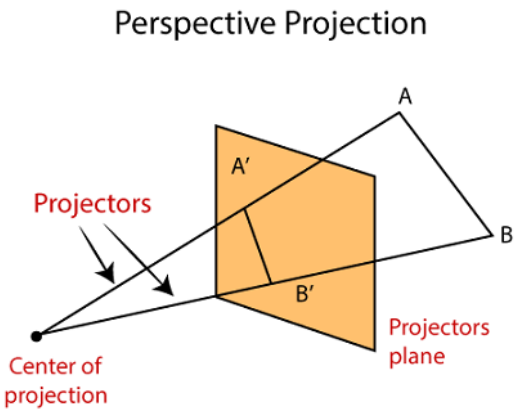
Types of Projection:



There are two types of projection.

Perspective Projection:

In the perspective projection, the distance of the project plane from the center of projection is finite. The object size keeps changing in reverse order with distance.
Perspective projection is used to determine the projector lines come together at a single point. The single point is also called **"project reference point"** or **"Center of projection."**



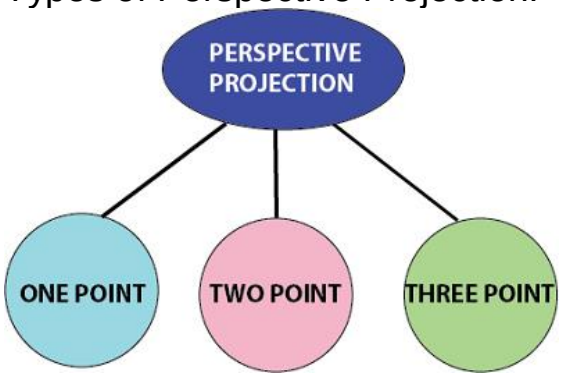
Characteristic of Perspective Projection:

- The Distance between the object and projection center is finite.
- In Perspective Projection, it is difficult to define the actual size and shape of the object.
- The Perspective Projection has the concept of vanishing points.
- The Perspective Projection is realistic but tough to implement.
- **Vanishing Point:** Vanishing point can be defined as a point in image plane where all parallel lines are interlinked. The Vanishing point is also called **"Directing Point."**

Use of Vanishing Point:

- It is used in 3D games and graphics editing.
- It is also used to represent 3D objects.
- We can also include perspective in the background of an image.
- We can also insert the shadow effect in an image.

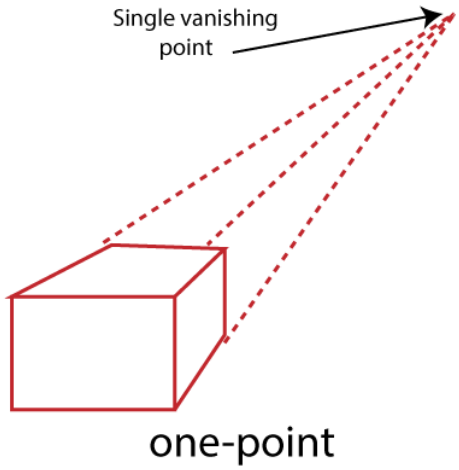
Types of Perspective Projection:



There are three types of Perspective Projection.

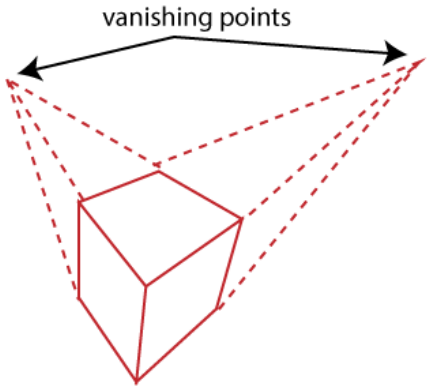
1. **One Point:** A One Point perspective contains only one vanishing point on the horizon line.

It is easy to draw.



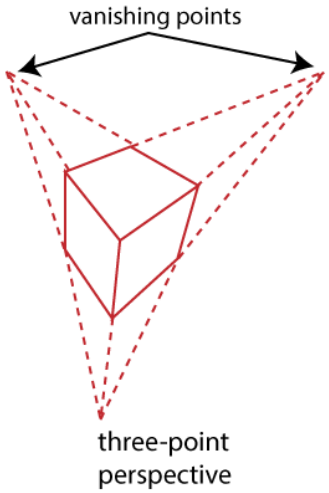
one-point

Use of One Point– The One Point projection is mostly used to draw the images of roads, railway tracks, and buildings.
2.**Two Point:** It is also called **"Angular Perspective.**" A Two Point perspective contains two vanishing points on the line.



two-point
perspective

Use of Two Point– The main use of Two Point projection is to draw the two corner roads.
3.**Three-Point**– The Three-Point Perspective contains three vanishing points. Two points lie on the horizon line, and one above or below the line.
It is very difficult to draw.
When we see an object from above, than the third point is below the ground. If we see an object from the below, than the third point is in the space above.



Use of Three-Point: It is mainly used in skyscraping.

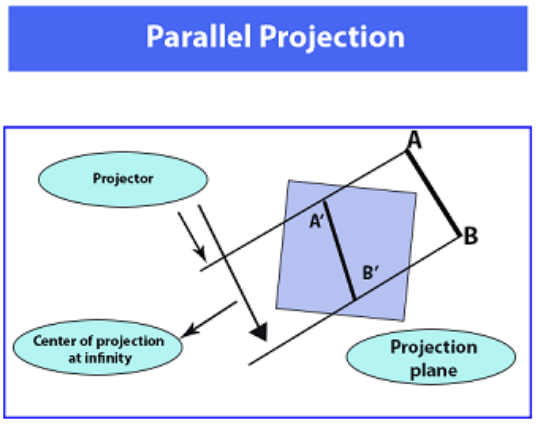
1. Better Look
2. Clear Representation

Disadvantages:

Difficult to Draw
Not Suitable for many-dimensional images

Parallel Projection:

In Parallel Projection, the distance of project plane from the center of projection is infinite. We can specify the direction of projection instead of the center of the projection. We can connect the projected vertices through the line segment.
The parallel Projection eliminates the Z-Coordinate. And the parallel lines from each vertex in the object are enhanced until the lines intersect the view plane.



Characteristic of parallel Projection:

- In parallel Projection, the projection lines are parallel to each other.
- There is the least amount of distortion within the object.
- The lines that are parallel to the object are also parallel to the drawing.
- The view of Parallel Projection is the less realistic cause of no foreshortening.
- The Parallel Projections are good for accurate measurements.

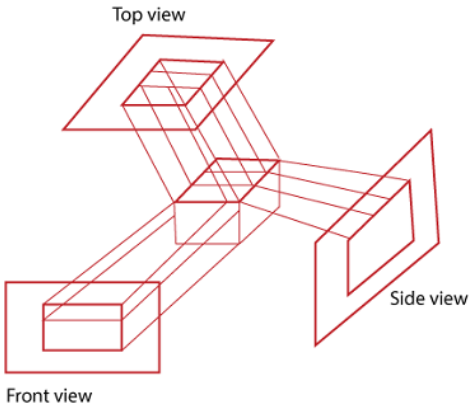
Types of Parallel Projection

There are two kind of parallel Projection:

1. **Orthographic projection:** In the Orthographic Parallel Projection, the Projection is perpendicular to the view plane.

The Orthographic Projection is divided into two parts-

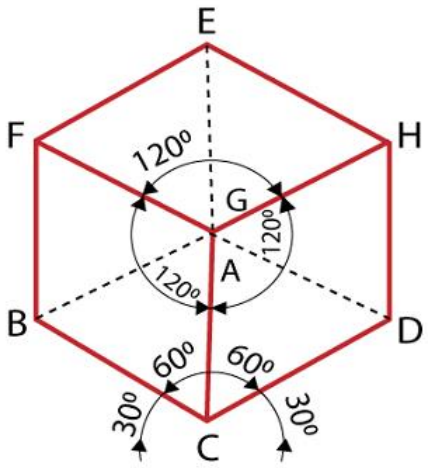
- **Multiview Orthographic Projection:** In Multiview Orthographic Projection, we can represent the two-dimensional Orthographic image into a three-dimensional object. The Multiview Orthographic Projection Includes-
  - **Front View**
  - **Top View**
  - **Side View**



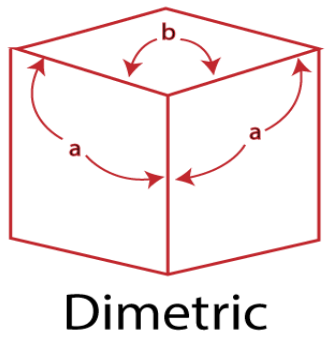
- **Axonometric Orthographic Projection:** The Axonometric Orthographic Projection is used to construct the pictorial representation of an object. The sight lines are perpendicular to the projection plane.

The Axonometric Orthographic Projection includes**–**

1. **Isometric:** In Isometric, we can represent the three-dimensional objects into two-dimensional drawings visually. The Angle between the two co-ordinate is 120 degrees.

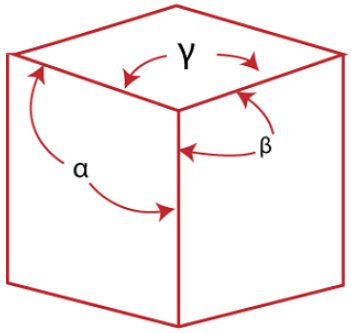


2. Dimetric: In Dimetric Projection, the view direction of the two axes are equal, and the direction of the third axis is defined individually.



Dimetric

3. Trimetric: In the Trimetric Projection, the view direction of all three axes is unequal. The scale of all three angles is defined individually.

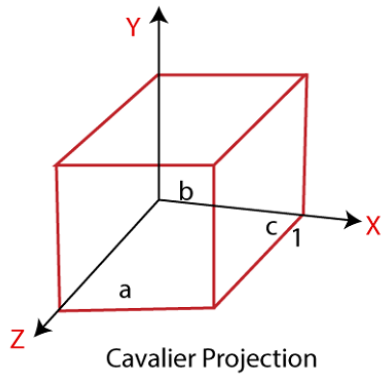


Trimetric

- **Oblique Projection:** In the Oblique Parallel Projection, the direction of projection is not normal to projection of plane. It is a simple technique that is used to construct two-dimensional images of three-dimensional objects.

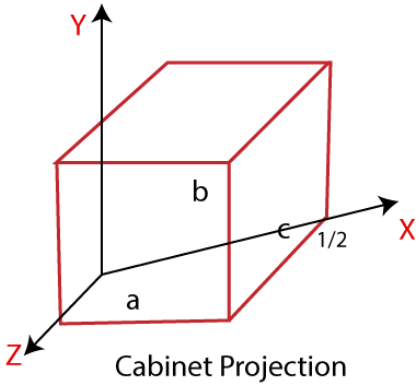
The Oblique Projection is mostly used in technical drawing.
The Oblique Projection is divided into two parts-

- **Cavalier:** In cavalier Projection, there is an angle between the Projection and Projection Plane is 45 degrees.



Cavalier Projection

- **Cabinet:** In Cabinet Projection, there is an angle between Projection and Projection Plane is 63.4 degrees.



Cabinet Projection

Advantages:

2. Good for exact Measurement
3. Parallel lines remain parallel

Disadvantages:

1. Less Realistic Looking
2. Angles are not preserved

CHAPTER - 9

ILLUMINATION MODEL & SURFACE RENDERING METHODS

Illumination model:

Illumination model is also known as Shading model or Lightning model, is used to calculate the intensity of light that is reflected at a given point on surface. There are three factors on which lightning effect depends on:

1. **Light Source :**
   Light source is the light emitting source. There are three types of light sources:
   1. **Point Sources –** The source that emit rays in all directions (A bulb in a room).
   2. **Parallel Sources –** Can be considered as a point source which is far from the surface (The sun).
   3. **Distributed Sources –** Rays originate from a finite area (A tubelight).

   Their position, electromagnetic spectrum and shape determine the lightning effect.
2. **Surface**
   When light falls on a surface part of it is reflected and part of it is absorbed. Now the surface structure decides the amount of reflection and absorption of light. The position of the surface and positions of all the nearby surfaces also determine the lightning effect.
3. **Observer :**
   The observer's position and sensor spectrum sensitivities also affect the lightning effect.

1. Ambient Illumination :

Assume you are standing on a road, facing a building with glass exterior and sun rays are falling on that building reflecting back from it and the falling on the object under observation. This would be **Ambient Illumination**. In simple words, Ambient Illumination is the one where source of light is indirect.
The reflected intensity I_{amb} of any point on the surface is:

$$I_{amb} = K_a I_a$$

Where, I_a : ambient light intensity
K_a : surface ambient reflectivity, value of K_a varies from 0 to 1

2. Diffuse Reflection :

Diffuse reflection occurs on the surfaces which are rough or grainy. In this

reflection the brightness of a point depends upon the angle made by the light source and the surface.
The reflected intensity I_{diff} of a point on the surface is:

$$I_{diff} = K_d I_p \cos(\theta) = K_d I_p (N \cdot L)$$

Where, I_p : the point light intensity
K_d : the surface diffuse reflectivity, value of K_d varies from 0 to 1
N : the surface normal
L : the light direction

3. Specular Reflection :
When light falls on any shiny or glossy surface most of it is reflected back, such reflection is known as Specular Reflection.
Phong Model is an empirical model for Specular Reflection which provides us with the formula for calculation the reflected intensity I_{spec}:

$$I_{spec} = W(\theta)\, I_l \cos^n(\Phi)$$

where, $W(\theta)$: K_s
L : direction of light source
N : normal to the surface
R : direction of reflected ray
V : direction of observer
Θ : Angle between L and R
Φ : angle between R and V

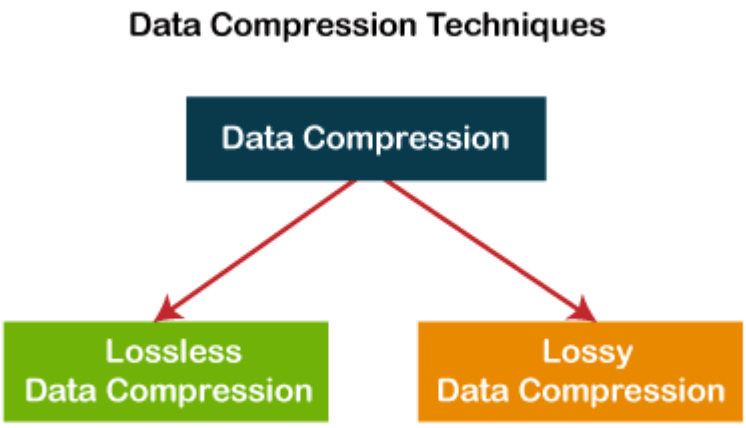
What is Data Compression

Data Compression is also referred to as **bit-rate reduction** or **source coding**. This technique is used to reduce the size of large files.

The advantage of data compression is that it helps us save our disk space and time in the data transmission.

There are mainly two types of data compression techniques -

1. Lossless Data Compression
2. Lossy Data Compression

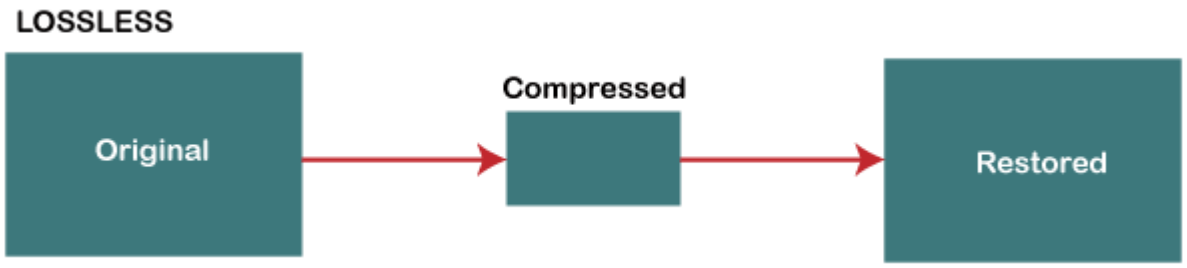


What is Lossless data compressio

Lossless data compression is used to compress the files **without losing an original file's quality and data**. Simply, we can say that in lossless data compression, file size is reduced, but the quality of data remains the same.

The main advantage of lossless data compression is that we can restore the original data in its original form after the decompression.

Lossless data compression mainly used in the sensitive documents, confidential information, and PNG, RAW, GIF, BMP file formats.



Some most important Lossless data compression techniques are -

1. Run Length Encoding (RLE)
2. Lempel Ziv - Welch (LZW)
3. Huffman Coding
4. Arithmetic Coding

Some important Lossy data compression techniques are -

1. Transform coding

2. Discrete Cosine Transform (DCT)
3. Discrete Wavelet Transform (DWT)

.No	Lossless data compression	Lossy data compression
1.	In Lossless data compression, there is no loss of any data and quality.	In Lossy data compression, there is a loss of quality and data, which is not measurable.
2.	In lossless, the file is restored in its original form.	In Lossy, the file does not restore in its original form.
3.	Lossless data compression algorithms are Run Length Encoding, Huffman encoding, Shannon fano encoding, Arithmetic encoding, Lempel Ziv Welch encoding, etc.	Lossy data compression algorithms are: Transform coding, Discrete Cosine Transform, Discrete Wavelet Transform, fractal compression, etc.
4.	Lossless compression is mainly used to compress text-sound and images.	Lossy compression is mainly used to compress audio, video, and images.
5.	As compare to lossy data compression, lossless data compression holds more data.	As compare to lossless data compression, lossy data compression holds less data.
6.	File quality is high in the lossless data compression.	File quality is low in the lossy data compression.
7.	Lossless data compression mainly supports RAW, BMP, PNG, WAV, FLAC, and ALAC file types.	Lossy data compression mainly supports JPEG, GIF, MP3, MP4, MKV, and OGG file types.