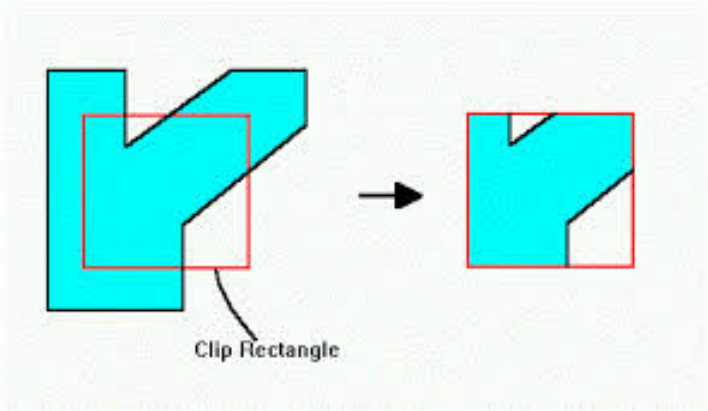
MODULE IV

Polygon Clipping- Sutherland Hodgeman Algorithm- Weiler Atherton Algorithm, Three Dimensional Object Representation-Polygon surfaces, quadric surfaces- Basic 3D Transformations.

Polygon Clipping (Sutherland Hodgman Algorithm)

Polygon is an object having closed outlines bounded by straight edges. An example of polygon clipping is shown below.



A polygon can also be clipped by specifying the clipping window. Sutherland Hodgeman polygon clipping algorithm is used for polygon clipping. In this algorithm, all the vertices of the polygon are clipped against each edge of the clipping window.

Beginning with the initial set of polygon vertices, we could first clip the polygon against the **left rectangle boundary** to produce a new sequence of vertices. The new set of vertices could then be successively passed to a **right boundary clipper, bottom boundary clipper and a top boundary clipper**. There are four possible cases when processing vertices in sequence around the perimeter of a polygon.

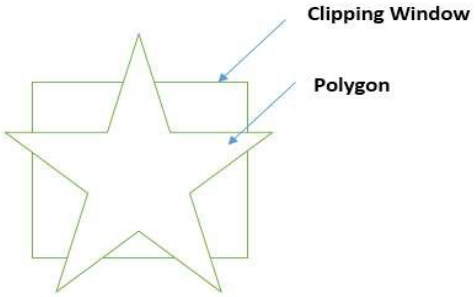


Figure: Polygon before clipping

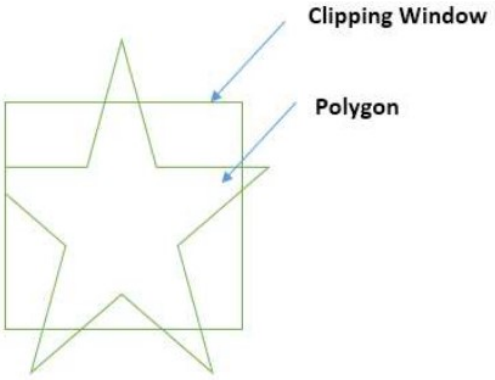


Figure: Clipping Left Edge

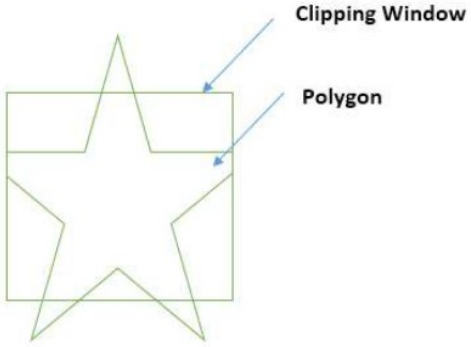


Figure: Clipping Right Edge

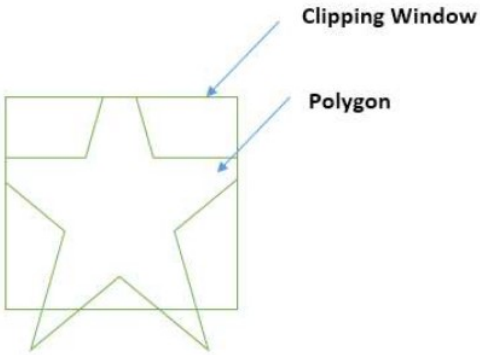


Figure: Clipping Top Edge

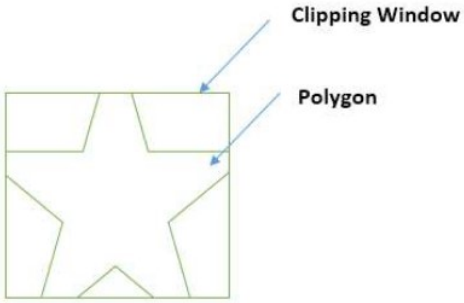


Figure: Clipping Bottom Edge

As each pair of adjacent polygon vertices is passed to a window boundary clipper, we make the following tests:

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

2) If both input vertices are inside the window boundary, only the second vertex is added to the output vertex list.

3) If the first vertex is inside the window boundary and the second vertex is outside, only the edge intersection with the window boundary is added to the output vertex list.

4) If both input vertices are outside the window boundary, nothing is added to the output list.

These 4 cases are shown in the figure.

Disadvantages:

Convex polygons are correctly clipped by the Sutherland-Hodgeman Algorithm.

But, concave polygons cannot be clipped correctly. It may be displayed with extraneous lines. Example shown in the following figure.

Weiler-Atherton polygon clipping

Here the vertex processing procedures for window boundaries are modified so that concave polygons are displayed correctly. This method is useful for identifying visible surfaces and so it can be applied with arbitrary polygon clipping regions.

This algorithm is based on the polygon processing direction (clockwise or counter clockwise).

For clockwise processing of polygons we use the following rules:

(1) For an outside to inside pair of vertices, follow the polygon boundary.

(2) For an inside to outside pair of vertices, follow the window boundary in a clockwise direction.

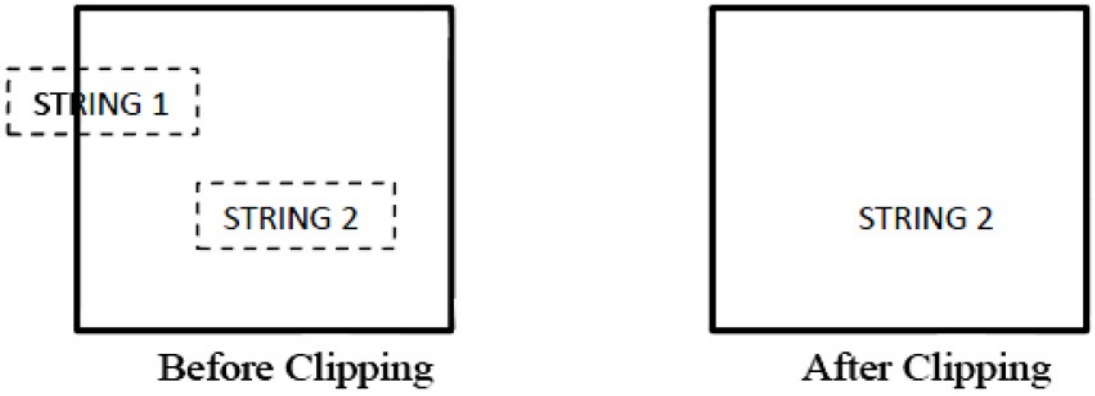
Refer the following example:

Text Clipping

Various techniques are used to provide text clipping in a computer graphics. It depends on the methods used to generate characters and the requirements of a particular application. There are three methods for text clipping which are listed below:

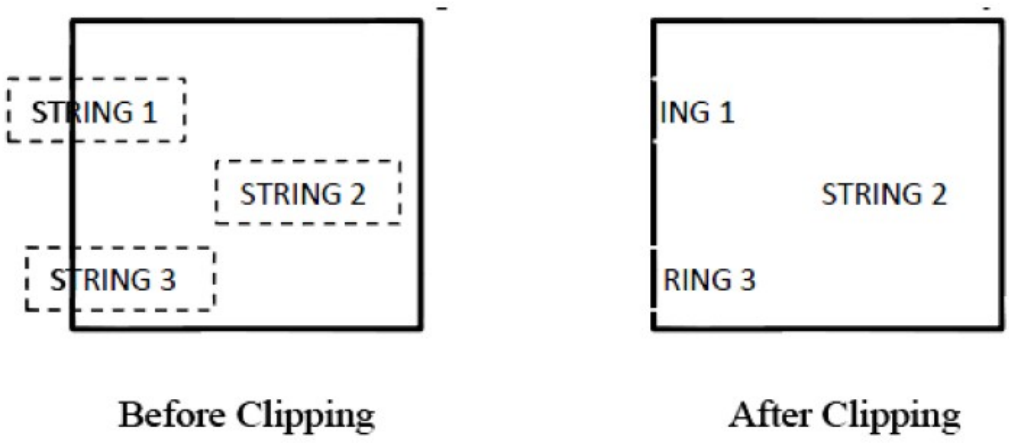
1. All or none string clipping
2. All or none character clipping
3. Text clipping

The following figure shows all or none string clipping:



In all or none string clipping method, either we keep the entire string or we reject entire string based on the clipping window. As shown in the above figure, STRING2 is entirely inside the clipping window so we keep it and STRING1 being only partially inside the window, we reject.

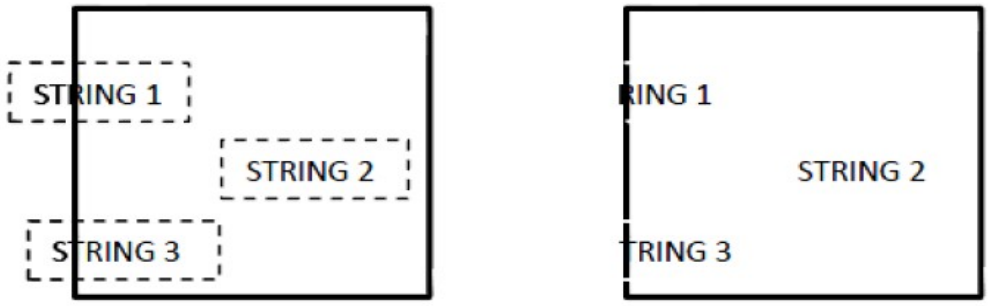
The following figure shows all or none character clipping:



This clipping method is based on characters rather than entire string. In this method if the string is entirely inside the clipping window, then we keep it. If it is partially outside the window, then:

- You reject only the portion of the string being outside
- If the character is on the boundary of the clipping window, then we discard that entire character and keep the rest string.

The following figure shows the text clipping:



Before Clipping After Clipping

This clipping method is based on characters rather than the entire string. In this method if the string is entirely inside the clipping window, then we keep it. If it is partially outside the window, then

- You reject only the portion of string being outside.
- If the character is on the boundary of the clipping window, then we discard only that portion of character that is outside of the clipping window.

Three Dimensional Object Representation

Objects are represented as a collection of surfaces. 3D object representation is divided into two categories.

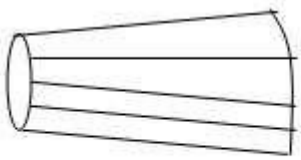
Boundary Representations (B-reps): It describes a 3D object as a set of surfaces that separates the object interior from the environment.

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 boundary representation for a 3D graphics object is a set of surface polygons that enclose the object interior. Many graphics system use this method. 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.

The polygon surfaces are common in design and solid-modelling applications,since their **wireframe display** can be done quickly to give general indication of surface structure. Then realistic scenes are produced by interpolating shading patterns across polygon surface to illuminate.



A 3D object represented by polygons

Polygon Tables

In this method, the surface is specified by the set of vertex coordinates and associated attributes.

Polygon data tables can be organized into two groups:

- **Geometric tables -**Geometric data tables contain vertex coordinates and parameters to identify the spatial orientation of the polygon surfaces.

- **Attribute tables-**Attribute information for an object includes parameters specifying the degree of transparency of the object and its surface reflectivity and texture characteristics.

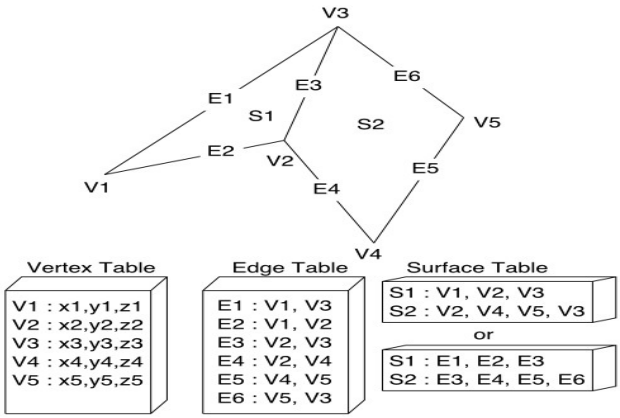
A convenient organization for storing geometric data is to create three lists:

- **a vertex table**
- **an edge table**
- **a polygon table**

Coordinate values for each vertex in the object are stored in the **vertex table**.

The **edge table** contains pointers back into the vertex table to identify the vertices for each polygon edge.

The **polygon table** contains pointers back into the edge table to identify the edges for each polygon.



Plane Equations

To produce a display of a three-dimensional object, we must process the input data representation for the object through several procedures.

These processing steps include transformation of the modelling and world-coordinate descriptions to viewing coordinates, then to device coordinates; identification of visible surfaces; and the application of surface-rendering procedures.

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.

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

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

- $Ax + By + Cz + D \neq 0$ 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.

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.

This method can be used to represent a broad class of solids/surfaces in graphics. A polygonal mesh can be rendered using hidden surface removal algorithms. The polygon mesh can be represented by three ways:

- Explicit representation
- Pointers to a vertex list
- Pointers to an edge list

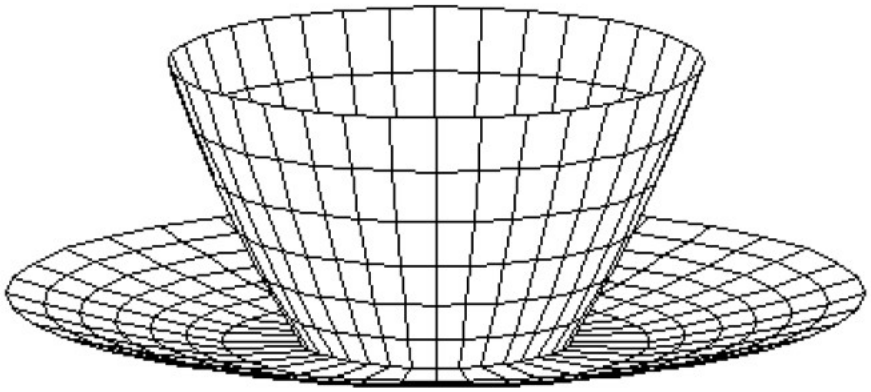


Figure: Polygon Mesh

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.

Quadric surfaces

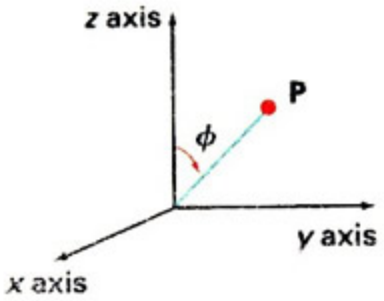
A frequently used class of objects is the quadric surfaces, which are described with second - degree equations (quadratics).

They include spheres, ellipsoids, tori, paraboloids, and hyperboloids.

Sphere

In Cartesian coordinates, a spherical surface with radius r centred on the coordinate origin is defined as the set of points (x, y, z) that satisfy the equation

$$x^2 + y^2 + z^2 = r^2$$

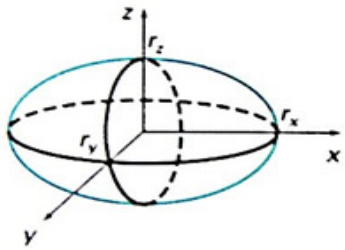


We can also describe the spherical surface in parametric form, using latitude and longitude angles as below:

$$x = r\cos\phi\cos\theta, \qquad -\pi/2 \le \phi \le \pi/2$$
$$y = r\cos\phi\sin\theta, \qquad -\pi \le \phi \le \pi$$
$$z = r\sin\phi$$

Ellipsoid

An ellipsoidal surface can be described as an extension of a spherical surface where the radii in three mutually perpendicular directions can have different values as in figure below.



The Cartesian representation for points over the surface of an ellipsoid centred on the origin is:

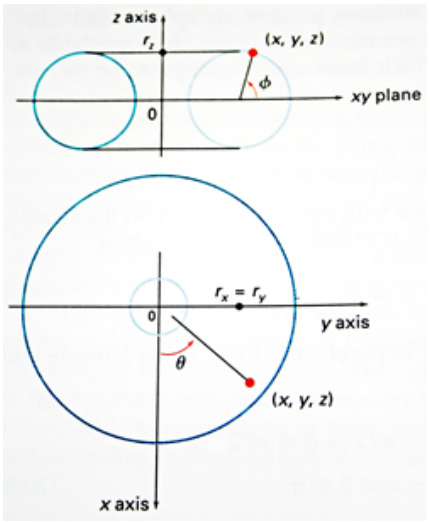
$$\left(\frac{x}{r_x}\right)^2 + \left(\frac{y}{r_y}\right)^2 + \left(\frac{z}{r_z}\right)^2 = 1$$

A parametric representation for the ellipsoid in terms of the latitude angle ϕ and the longitude angle Θ can be written as:

$$x = r_x \cos\phi \cos\theta, \qquad -\pi/2 \le \phi \le \pi/2$$
$$y = r_y \cos\phi \sin\theta, \qquad -\pi \le \phi \le \pi$$
$$z = r_z \sin\phi$$

Torus

A torus is a doughnut-shaped object, as shown in fig. below. It can be generated by rotating a circle or other conic about a specified axis.



The Cartesian representation for points over the surface of a torus can be written in the form:

$$\left[r - \sqrt{\left(\frac{x}{r_x}\right)^2 + \left(\frac{y}{r_y}\right)^2} \right] + \left(\frac{z}{r_z}\right)^2 = 1$$

Where r is any given offset value. Parametric representations for a torus are similar to those for an ellipse, except that angle ϕ extends over 360^0. Using latitude and longitude angles ϕ and Θ, we can describe the torus surface as the set of points that satisfy

$$x = r_x(r + \cos\phi)\cos\theta, \quad -\pi \le \phi \le \pi$$
$$y = r_y(r + \cos\phi)\sin\theta, \quad -\pi \le \phi \le \pi$$
$$z = r_z \sin\phi$$

3D Transformations

Methods for geometric transformations and object modelling in 3D are extended from 2D methods by including the considerations for the z coordinate.

Basic geometric transformations are: Translation, Rotation, and Scaling

Basic Transformations

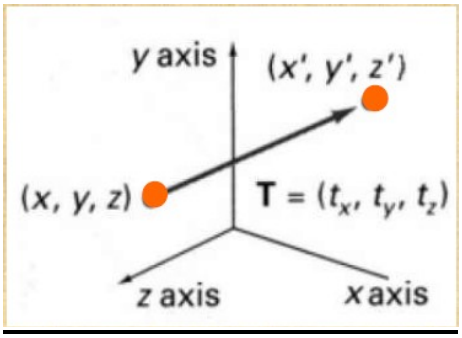
Translation

We translate a 3D point by adding translation distances, tx, ty, and tz, to the original coordinate position (x, y, z):

$$x' = x + tx,\ y' = y + ty,\ z' = z + tz$$

Alternatively, translation can also be specified by the transformation matrix in the following formula:

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



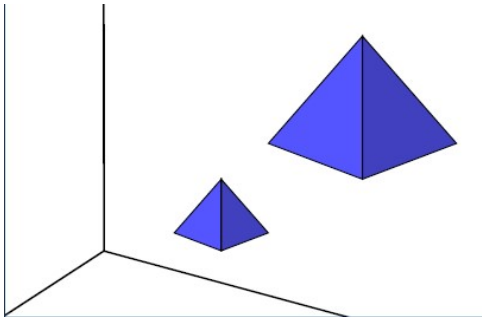
Scaling With Respect to the Origin

We scale a 3D object with respect to the origin by setting the scaling factors sx, sy and sz, which are multiplied to the original vertex coordinate positions (x,y,z):

$$x' = x * sx,\ y' = y * sy,\ z' = z * sz$$

Alternatively, this scaling can also be specified by the transformation matrix in the following formula:

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

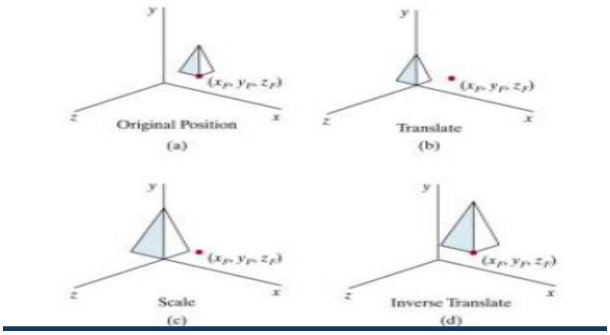


Scaling with respect to a Selected Fixed Position

Scaling with respect to a selected fixed position **(x,** y,z) can be represented with the following transformation sequence:

1. Translate the fixed point to the origin.
2. Scale the object relative to the coordinate origin using the above equation of scaling with respect to origin.
3. Translate the fixed point back to its original position.

This sequence of transformations is demonstrated in the following figure.



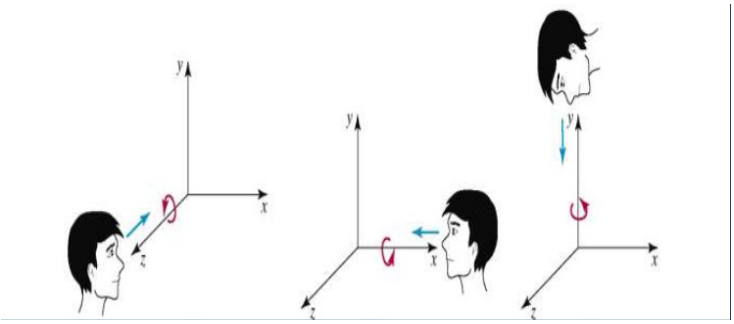
The matrix representation for an arbitrary fixed-point scaling can then be expressed as the concatenation of these translate-scale-translate transformations as

$$\mathbf{T}(x_f, y_f, z_f) \cdot \mathbf{S}(s_x, s_y, s_z) \cdot \mathbf{T}(-x_f, -y_f, -z_f) = \begin{bmatrix} s_x & 0 & 0 & (1-s_x)x_f \\ 0 & s_y & 0 & (1-s_y)y_f \\ 0 & 0 & s_z & (1-s_z)z_f \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

3D Rotation

To perform rotation, we must designate an axis of rotation (about which axis the object is to be rotated) and the amount of angular rotation.

Rotation can be performed along three axes as:



Consider the three dimensional rotation **with respect to Z axis**.

The equations are:

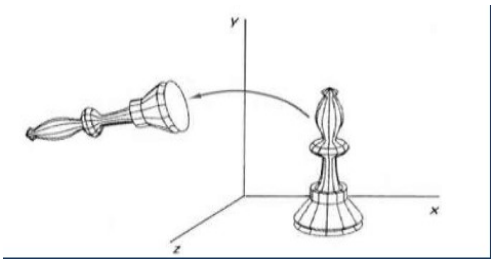
$$x' = x\cos\theta - y\sin\theta$$
$$Y' = x\sin\theta + y\cos\theta$$
$$Z' = z$$

Where parameter θ specifies the rotation angle. It can be represented in matrix form as:

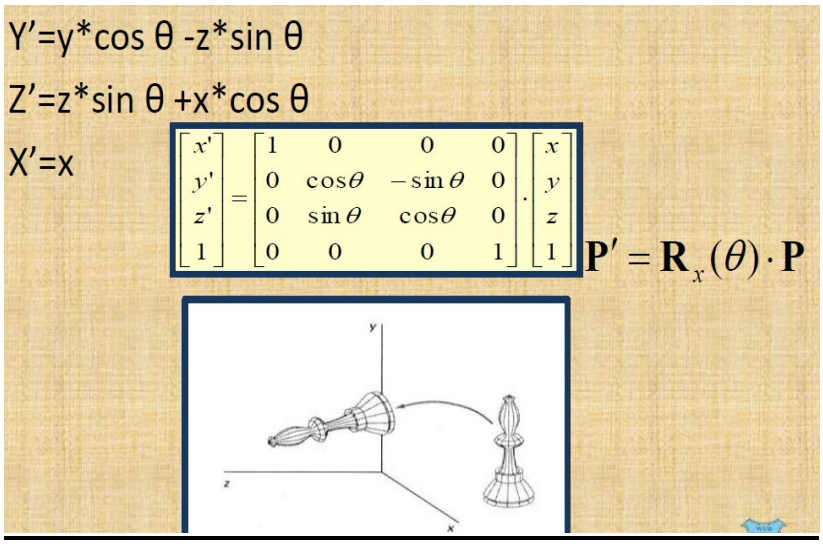
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

This can be written as:

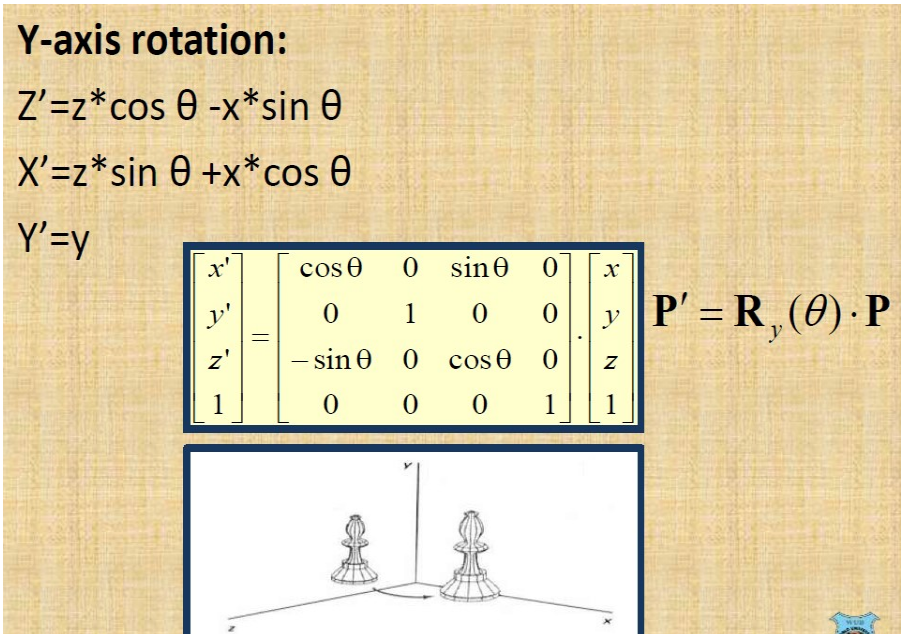
$$\mathbf{P}' = \mathbf{R}_z(\theta) \cdot \mathbf{P}$$



X axis Rotation



Y axis Rotation



3D Reflection

Reflection in computer graphics is used to emulate reflective objects like mirrors and shiny surfaces.

A 3D reflection can be performed relative to a selected reflection axis or with respect to a selected reflection plane.

Reflections relative to a given axis are equivalent to 180^0rotations about that axis.

Reflection with respect to a plane is equivalent to 180^0 rotation in 4 dimensional space.

Reflection may be an x-axis , y-axis , z-axis. and also in the planes xy-plane, yz-plane , and zx-plane.

Reflection about x-axis:-

$x'=x \quad y'=-y \quad z'=-z$

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

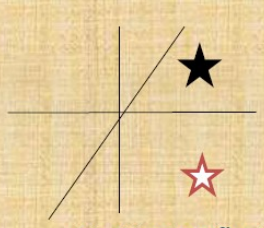


Fig: X axis reflection

Reflection about y-axis:-

$y'=y \quad x'=-x \quad z'=-z$

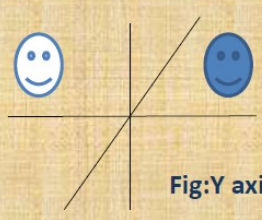


Fig:Y axis reflection

- The matrix for reflection about y-axis:-

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

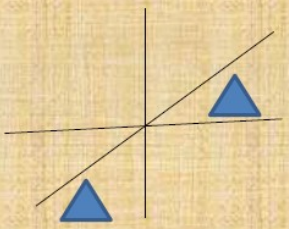


Fig: Z axis reflection

- **Reflection about z-axis:-**

$x'=-x \quad y'=-y \quad z'=z$

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

3D Shearing

A transformation that distorts the shape of an object such that the transformed shape appears as if the object were composed of internal layers that had been caused to slide over each other is called a shearing.

In two dimensions, transformations relative to the x or y axes to produce distortions in the shapes of objects. In three dimensions, we can also generate shears relative to the z axis.

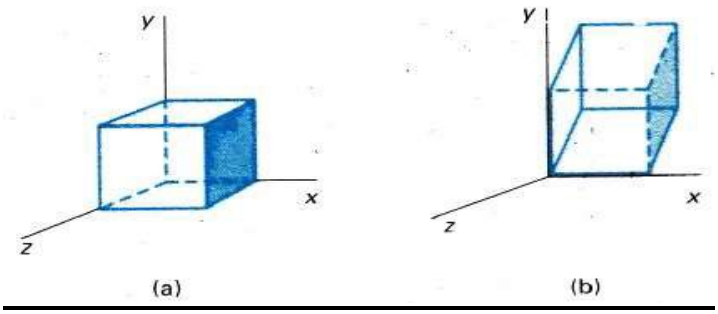
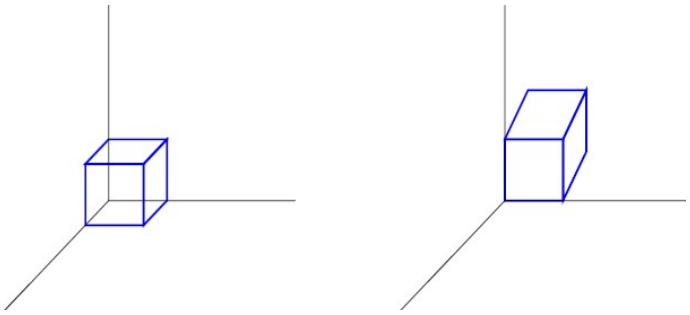


fig: before shearing **fig: after shearing**

Shearing will modify object shapes. It is useful for perspective projections. For Example, draw a cube (3D) on a screen (2D) as shown in below.



Here we are altering the values for **x** and **y** by an amount proportional to the distance from zref.

Shearing about XY axis

Parameters a and b can be assigned any real values. The effect of this transformation matrix is to alter x and y-coordinate values by an amount that is proportional to the z value, while leaving the z coordinate unchanged.

Boundaries of planes that are perpendicular to the z axis are thus shifted by an amount proportional to z. An example of the effect of this shearing matrix on a unit cube is shown in the above figure for shearing values a=b=1. Shearing matrices for the x axis and y axis are defined similarly.

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