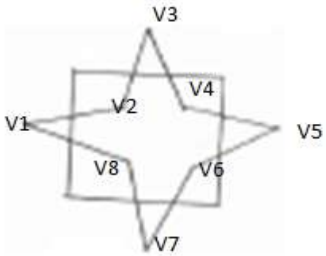
USN | | | | | | | | | | |

Internal Assessment Test 2 – June 2022

Sub:	Computer Graphics and Visualization				Sub Code:	18CS62	Branch:	CSE
Date:	8/6/2022	Duration:	90 mins	Max Marks: 50	Sem/Sec:	6 A,B,C		OBE

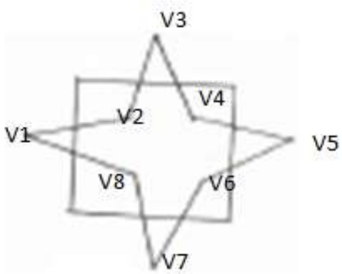
	Answer any FIVE FULL Questions	MARKS	CO	RBT
1	Explain the 2D – Viewing transformation pipeline and demonstrate 2D- normalization and viewport transformations.	[10]	CO4	L2
2	Define clipping. Explain the Cohen Sutherland line clipping with code.	[10]	CO4	L2
3	Demonstrate the working of Sutherland Hodgeman polygon clipping algorithm and find the final clipped vertices of the following figure: 	[10]	CO4	L3
4	Explain 3D translation, Rotations and scaling with relevant transformation matrix. Design transformation matrix to rotate a 3D object about an axis that is parallel to one of the coordinate axis.	[10]	CO3	L2
5	Distinguish between Parallel and perspective projection. Explain orthogonal projection in detail.	[10]	CO4	L2
6	What is perspective projection? Explain the general and special case of perspective projection equation.	[10]	CO4	L2

CI **CCI** **HOD/CSE**

USN | | | | | | | | | | |

Internal Assessment Test 2 – June 2022

Sub:	Computer Graphics and Visualization				Sub Code:	18CS62	Branch:	CSE
Date:	8/6/2022	Duration:	90 mins	Max Marks: 50	Sem/Sec:	6 A,B,C		OBE

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1	Explain the 2D – Viewing transformation pipeline and demonstrate 2D- normalization and viewport transformations.	[10]	CO4	L2
2	Define clipping. Explain the Cohen Sutherland line clipping with code.	[10]	CO4	L2
3	Demonstrate the working of Sutherland Hodgeman polygon clipping algorithm and find the final clipped vertices of the following figure: 	[10]	CO4	L3
4	Explain 3D translation, Rotations and scaling with relevant transformation matrix. Design transformation matrix to rotate a 3D object about an axis that is parallel to one of the coordinate axis.	[10]	CO3	L2
5	Distinguish between Parallel and perspective projection. Explain orthogonal projection in detail.	[10]	CO4	L2
6	What is perspective projection? Explain the general and special case of perspective projection equation.	[10]	CO4	L2

CI **CCI** **HOD/CSE**

CO PO Mapping

Course Outcomes		Modules	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
CO1	Understand basics concepts and applications of Computer Graphics	1,2	2	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-
CO2	Design and implement algorithms for 2D graphics primitives and attributes.	2,3,5	2	2	2	-	-	-	-	-	-	-	-	-	-	-	-	-
CO3	Illustrate Geometric transformations on both 2D and 3D objects.	2,3,4	2	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-
CO4	Understand concepts of clipping and visible surface detection in 2D and 3D viewing, and Illumination Models.	3,4	2	2	2	-	-	-	-	-	-	-	-	-	-	-	-	-
CO5	Design and implement interactive OpenGL graphics programs.	1,2,3,4, 5	3	3	3	3	2	-	-	-	-	-	-	2	2	-	3	-

COGNITIVE LEVEL	REVISED BLOOMS TAXONOMY KEYWORDS
L1	List, define, tell, describe, identify, show, label, collect, examine, tabulate, quote, name, who, when, where, etc.
L2	summarize, describe, interpret, contrast, predict, associate, distinguish, estimate, differentiate, discuss, extend
L3	Apply, demonstrate, calculate, complete, illustrate, show, solve, examine, modify, relate, change, classify, experiment, discover.
L4	Analyze, separate, order, explain, connect, classify, arrange, divide, compare, select, explain, infer.
L5	Assess, decide, rank, grade, test, measure, recommend, convince, select, judge, explain, discriminate, support, conclude, compare, summarize.

PROGRAM OUTCOMES (PO), PROGRAM SPECIFIC OUTCOMES (PSO)				CORRELATION LEVELS	
PO1	Engineering knowledge	PO7	Environment and sustainability	0	No Correlation
PO2	Problem analysis	PO8	Ethics	1	Slight/Low
PO3	Design/development of solutions	PO9	Individual and team work	2	Moderate/ Medium
PO4	Conduct investigations of complex problems	PO10	Communication	3	Substantial/ High
PO5	Modern tool usage	PO11	Project management and finance		
PO6	The Engineer and society	PO12	Life-long learning		
PSO1	Develop applications using different stacks of web and programming technologies				
PSO2	Design and develop secure, parallel, distributed, networked, and digital systems				
PSO3	Apply software engineering methods to design, develop, test and manage software systems.				
PSO4	Develop intelligent applications for business and industry				

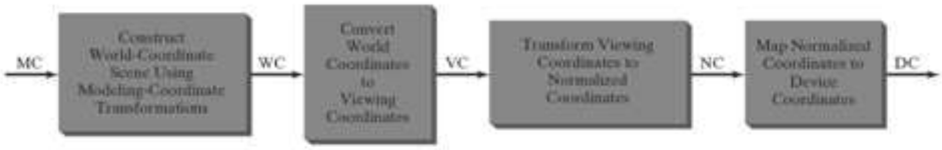
Internal Assessment Test 2 – June 2022
Solution

Sub:	Computer Graphics and Visualization					Sub Code:	18CS62	Branch:	CSE
Date:	8/6/2022	Duration:	90 mins	Max Marks:	50	Sem/Sec:	6 A,B,C		OBE

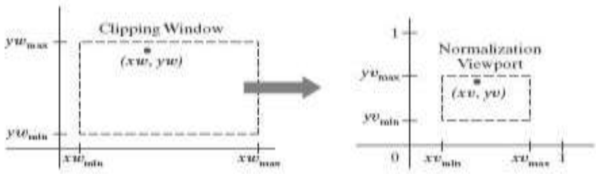
	Answer any FIVE FULL Questions	MARKS	CO	RBT
1	Explain the 2D – Viewing transformation pipeline and demonstrate 2D- normalization and viewport transformations.	[10]	CO4	L2

Solution:

2D – Viewing Transformation Pipeline:



2D- normalization and viewport transformations:



- To transform the world-coordinate point into the same relative position within the viewport, we require that

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

$$\frac{yv - yv_{\min}}{yv_{\max} - yv_{\min}} = \frac{yw - yw_{\min}}{yw_{\max} - yw_{\min}}$$

- We could obtain the transformation from world coordinates to viewport coordinates with the following sequence:

1. Scale the clipping window to the size of the viewport using a fixed-point position of (*xw*min, *yw*min)
2. Translate (*xw*min, *yw*min) to (*xv*min, *yv*min).

- Solving these expressions for the viewport position (*xv*, *yv*), we have *xv* = *sxxw* + *tx*, *yv* = *syyw* + *ty*

Scaling:

$$\mathbf{S} = \begin{bmatrix} s_x & 0 & xw_{\min}(1 - s_x) \\ 0 & s_y & yw_{\min}(1 - s_y) \\ 0 & 0 & 1 \end{bmatrix}$$

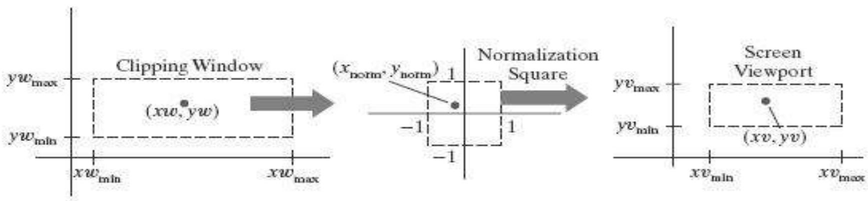
Translation:

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & xv_{\min} - xw_{\min} \\ 0 & 1 & yv_{\min} - yw_{\min} \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{M}_{\text{window, normviewp}} = \mathbf{T} \cdot \mathbf{S} = \begin{bmatrix} s_x & 0 & t_x \\ 0 & s_y & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

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

$$s_y = \frac{yv_{\max} - yv_{\min}}{yw_{\max} - yw_{\min}} \qquad t_y = \frac{yw_{\max}yv_{\min} - yw_{\min}yv_{\max}}{yw_{\max} - yw_{\min}}$$



- The matrix for the normalization transformation is obtained by **substituting −1 for** *xv*min and *yv*min and substituting +1 for *xv*max and *yv*max.

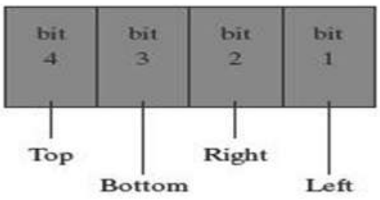
$$M_{window,\,normsquare} = \begin{bmatrix} \frac{2}{xw_{max} - xw_{min}} & 0 & -\frac{xw_{max} + xw_{min}}{xw_{max} - xw_{min}} \\ 0 & \frac{2}{yw_{max} - yw_{min}} & -\frac{yw_{max} + yw_{min}}{yw_{max} - yw_{min}} \\ 0 & 0 & 1 \end{bmatrix}$$

- This **time, we get the transformation matrix by substituting −1 for** *xw*min and *yw*min and substituting +1 for *xw*max and *yw*max

$$M_{normsquare,\,viewport} = \begin{bmatrix} \frac{xv_{max} - xv_{min}}{2} & 0 & \frac{xv_{max} + xv_{min}}{2} \\ 0 & \frac{yv_{max} - yv_{min}}{2} & \frac{yv_{max} + yv_{min}}{2} \\ 0 & 0 & 1 \end{bmatrix}$$

2 Define clipping. Explain the Cohen Sutherland line clipping with code. [10] CO4 L2

Clipping:



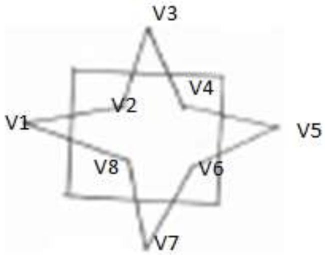
Top left 1001	Top 1000	Top right 1010
Left 0001	0000	Right 0010
Bottom left 0101	Bottom 0100	Bottom right 0110

```
const int RIGHT = 2
const int LEFT = 1
const int TOP = 8
const int BOTTOM = 4
```

```
outcode ComputeOutCode (double x, double y)
{
        outcode code = 0;
        if (y > ymax)           //above the clip window
                code |= TOP;
        else if (y < ymin)      //below the clip window
                code |= BOTTOM;
        if (x > xmax)           //to the right of clip window
                code |= RIGHT;
        else if (x < xmin)      //to the left of clip window
                code |= LEFT;
        return code;
}
if (!(outcode0 | outcode1))
//logical or is 0 Trivially accept & exit
                {
                accept = true;
                done = true;
                }
        else if (outcode0 & outcode1)
//logical and is not 0. Trivially reject and exit
                done = true;
if (outcodeOut & TOP)
{
                y = ymax;
                x = x0 + (y - y0)*(x1 - x0) /( y1 - y0);
}
else if (outcodeOut & BOTTOM)
{
                y = ymin;
                x = x0 + (y - y0)*(x1 - x0) /( y1 - y0);
        }
else if (outcodeOut & RIGHT)
        {
```

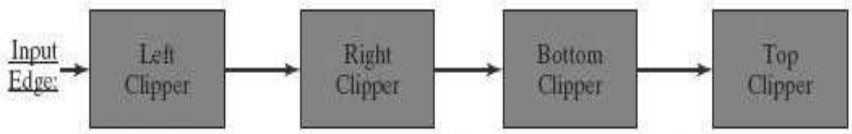
```
                x = xmax;
                y = y0 +  (x - x0)* (y1 - y0) /(x1 - x0);
        }
        else
{
                 x = xmin;
                 y = y0 +  (x - x0)* (y1 - y0) /(x1 - x0);
        }
if (outcodeOut == outcode0)
        {
                x0 = x;
                y0 = y;
                outcode0 = ComputeOutCode (x0, y0);
        }
        else
        {
                x1 = x;
                y1 = y;
                outcode1 = ComputeOutCode (x1, y1);
        }
```

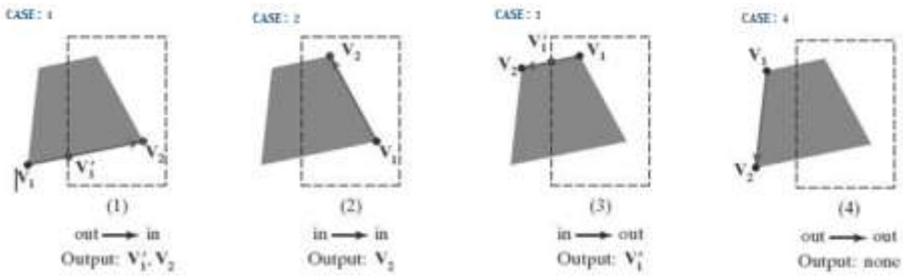
3	Demonstrate the working of Sutherland Hodgeman polygon clipping algorithm and find the final clipped vertices of the following figure:	[10]	CO4	L3

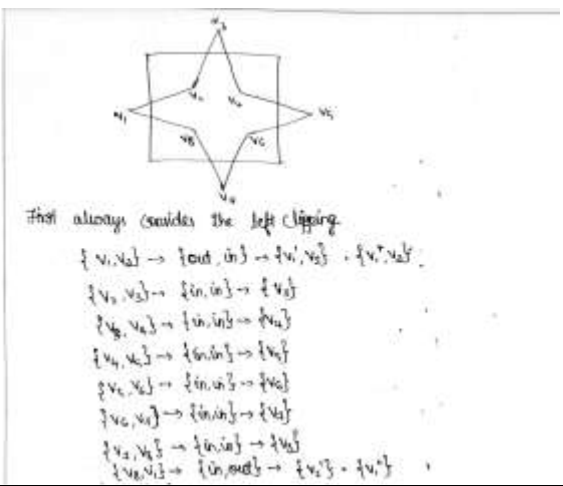


Sutherland Hodgeman polygon clipping algorithm

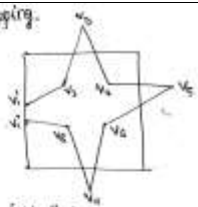
- An efficient method for clipping a convex-polygon fill area, developed by Sutherland and Hodgman
- Send the polygon vertices through each clipping stage so that a single clipped vertex can be immediately passed to the next stage.







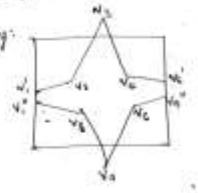
After left clipping.



Now, consider right clipping.

{v4, v5} → {in, out} → {v5'}
{v5, v6} → {out, in} → {v5'', v6}
{v6, v7} → {in→in} → {v7}
{v7, v8} → {in→in} → {v8}
{v8, v1''} → {in→in} → v1''
{v1', v2} → {in→in} → v2
{v2, v3} → {in→in} → v3
{v3, v4} → {in→in} → v4

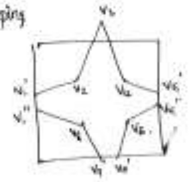
After right clipping:



Consider: bottom clipping:

{v6, v7} → {in→out} → {v7'}
{v7, v8} → {out→in} → {v7'', v8}
{v8, v1''} → {in→in} → {v1''}
{v1' → v2} → {in→in} → {v2}
{v2, v3} → {in→in} → {v3}
{v3, v4} → {in→in} → {v4}
{v4, v5'} → {in→in} → {v5'}
{v5'', v6} → {in→in} → {v6}

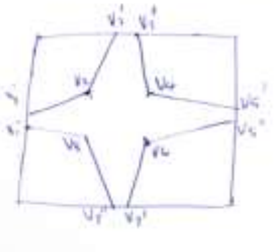
After bottom clipping



Finally top clipping

{v2, v3} → {in→out} → {v3'}
{v3, v4} → {out→in} → {v3'', v4}
{v4, v5'} → {in→in} → {v5'}
{v5'', v6} → {in→in} → {v6}
{v6, v7'} → {in→in} → {v7'}
{v7'', v8} → {in→in} → {v8}
{v8, v1''} → {in→in} → {v1''}
{v1', v2} → {in→in} → {v2}

Final clipped Polygon.



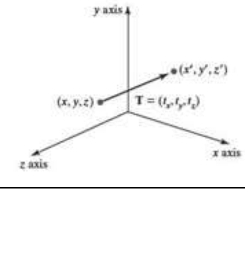
4	Explain 3D translation, Rotations and scaling with relevant transformation matrix. Design transformation matrix to rotate a 3D object about an axis that is parallel to one of the coordinate axis.	[10]	CO3	L2

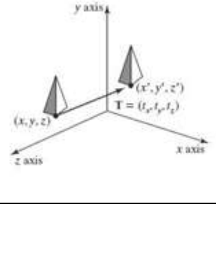
Translation:

- A position **P** = (x, y, z) in three-dimensional space is translated to a location **P'** = (x', y', z') by adding translation distances t_x, t_y, and t_z to the Cartesian coordinates of **P**:

$$x' = x + t_x, \qquad y' = y + t_y, \qquad z' = z + t_z$$

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





Rotation:

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

Along z axis

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

Along x axis

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

Along y axis

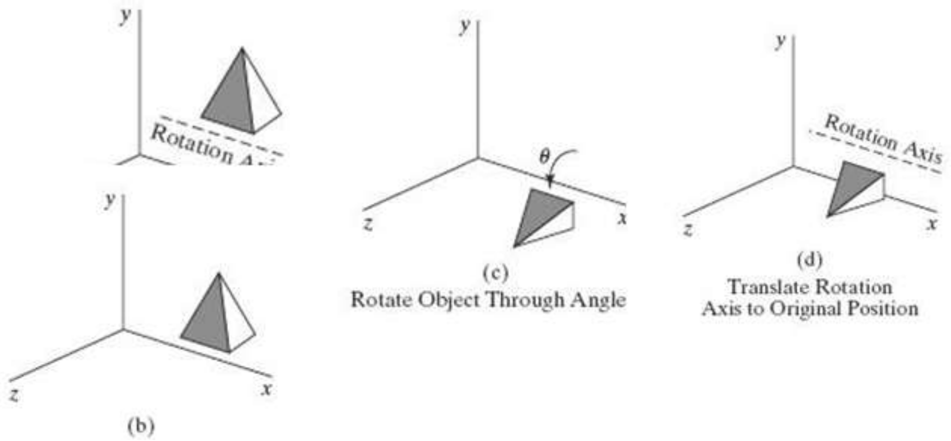
$$z' = z\cos\theta - x\sin\theta$$
$$x' = z\sin\theta + x\cos\theta$$
$$y' = y$$

Scaling:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad \mathbf{P}' = \mathbf{S} \cdot \mathbf{P}$$

$$x' = x \cdot s_x, \qquad y' = y \cdot s_y, \qquad z' = z \cdot s_z$$

Rotate a 3D object about an axis that is parallel to one of the coordinate axis:



1. Translate the object so that the rotation axis coincides with the parallel coordinate axis.
2. Perform the specified rotation about that axis.
3. Translate the object so that the rotation axis is moved back to its original position

$$\mathbf{P}' = \mathbf{T}^{-1} \cdot \mathbf{R}_x(\theta) \cdot \mathbf{T} \cdot \mathbf{P}$$

$$\mathbf{R}(\theta) = \mathbf{T}^{-1} \cdot \mathbf{R}_x(\theta) \cdot \mathbf{T}$$

5	Distinguish between Parallel and perspective projection. Explain orthogonal projection in detail.	[10]	CO4	L2

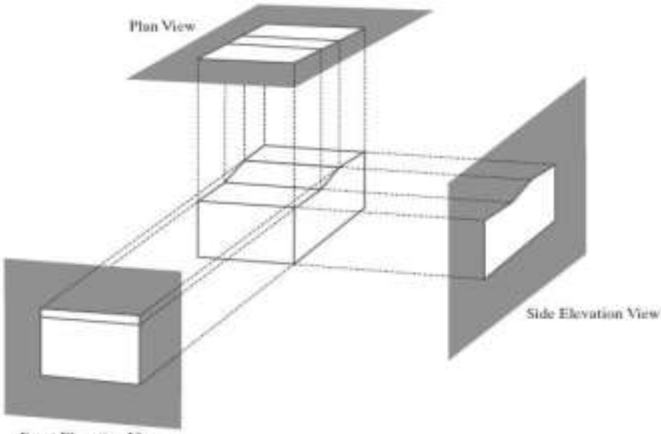
Difference Between perspective projection and parallel projection

Perspective projection	Parallel projection
The center of projection is at a finite distance from the viewing plane	Center of projection at infinity results with a parallel projection
Explicitly specify: center of projection	Direction of projection is specified
Size of the object is inversely proportional to the distance of the object from the center of projection	No change in the size of object

Produces realistic views but does not preserve relative proportion of objects	A parallel projection preserves relative proportion of objects, but does not give us a realistic representation of the appearance of object.
Not useful for recording exact shape and measurements of the object	Used for exact measurement
Parallel lines do not in general project as parallel	Parallel lines do remain parallel

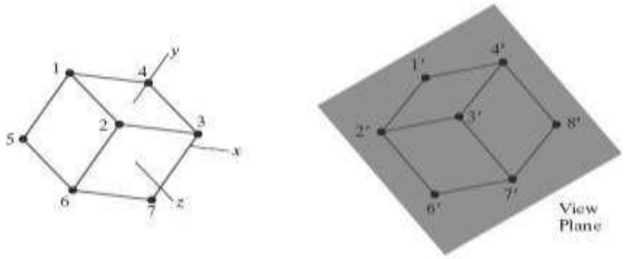
Orthogonal Projection:

- A transformation of object descriptions to a view plane along lines that are all parallel to the view-plane normal vector N is called an orthogonal projection also termed as orthographic projection.
- This produces a parallel-projection transformation in which the projection lines are perpendicular to the view plane.
- Orthogonal projections are most often used to produce the front, side, and top views of an object
- Front, side, and rear orthogonal projections of an object are called *elevations;* and a top orthogonal projection is called a *plan view*



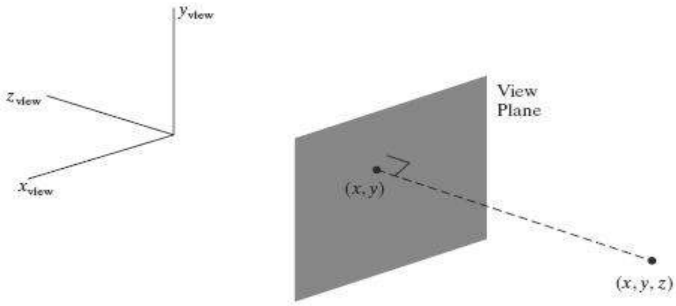
<u>Axonometric and Isometric Orthogonal Projections</u>

- We can also form orthogonal projections that display more than one face of an object. Such views are called axonometric orthogonal projections.



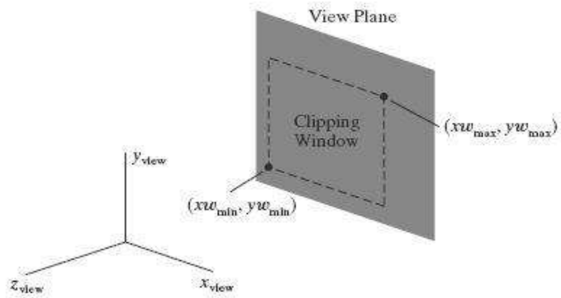
<u>Orthogonal Projection Coordinates:</u>

- With the projection direction parallel to the z_{view} axis, the transformation equations for an orthogonal projection are trivial. For any position (x, y, z) in viewing coordinates, as in Figure below, the projection coordinates are $x_p = x$, $y_p = y$

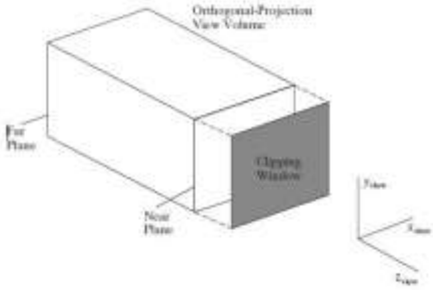


<u>Clipping Window and Orthogonal-Projection View Volume:</u>

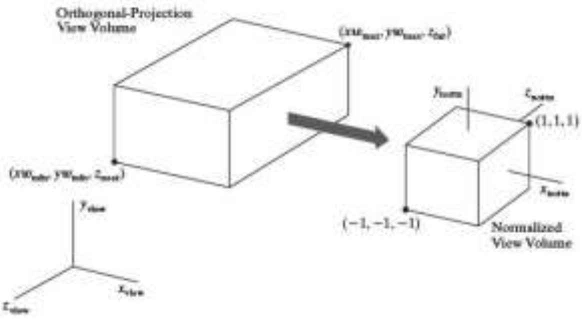
<table>
<tr><td></td><td>



<u>Clipping Window and Orthogonal-Projection View Volume</u>



<u>Normalization Transformation for an Orthogonal Projection</u>



The normalization transformation for the orthogonal view volume is

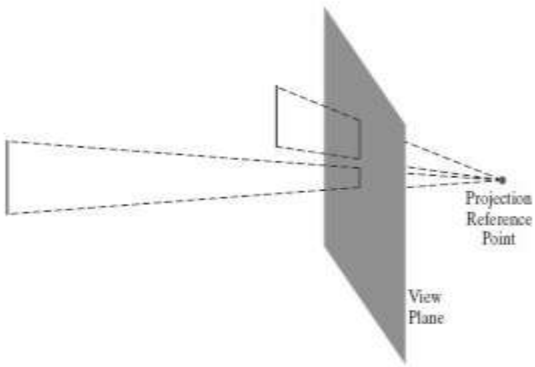
$$\mathbf{M}_{\text{ortho,norm}} = \begin{bmatrix} \frac{2}{xw_{\max} - xw_{\min}} & 0 & 0 & -\frac{xw_{\max} + xw_{\min}}{xw_{\max} - xw_{\min}} \\ 0 & \frac{2}{yw_{\max} - yw_{\min}} & 0 & -\frac{yw_{\max} + yw_{\min}}{yw_{\max} - yw_{\min}} \\ 0 & 0 & \frac{-2}{z_{\text{near}} - z_{\text{far}}} & \frac{z_{\text{near}} + z_{\text{far}}}{z_{\text{near}} - z_{\text{far}}} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

</td><td></td><td></td><td></td></tr>
<tr><td>6</td><td>

What is perspective projection? Explain the general and special case of perspective projection equation.

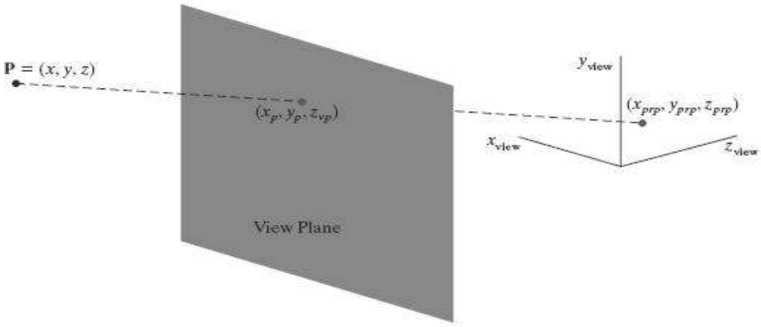
Perspective Projection:

- We can approximate this geometric-optics effect by projecting objects to the view plane along converging paths to a position called the projection reference point (or center of projection).



- projection path of a spatial position (x, y, z) to a general projection reference point at $(x_{prp}, y_{prp}, z_{prp})$.
- The projection line intersects the view plane at the coordinate position (x_p, y_p, z_{vp}), where z_{vp} is some selected position for the view plane on the z_{view} axis.

</td><td>[10]</td><td>CO4</td><td>L2</td></tr>
</table>



projection path of a spatial position (x, y, z) to a general projection
The projection line intersects the view plane at the coordinate position (x_p, y_p, z_{vp}), where z_{vp} is some selected position for the view plane on the z_{view} axis.

General Perspective Projection Equation:

$$x_p = x\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right) + x_{prp}\left(\frac{z_{vp} - z}{z_{prp} - z}\right)$$

$$y_p = y\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right) + y_{prp}\left(\frac{z_{vp} - z}{z_{prp} - z}\right)$$

<u>Perspective-Projection Equations: Special Cases</u>

<u>Case 1:</u>

To simplify the perspective calculations, the projection reference point could be limited to positions along the zview axis, the

$x_{prp} = y_{prp} = 0$:

$$x_p = x\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right), \qquad y_p = y\left(\frac{z_{prp} - z_{vp}}{z_{prp} - z}\right)$$

<u>Case 2:</u>

➔ Sometimes the projection reference point is fixed at the coordinate origin, and

$(x_{prp}, y_{prp}, z_{prp}) = (0, 0, 0)$:

$$x_p = x\left(\frac{z_{vp}}{z}\right), \qquad y_p = y\left(\frac{z_{vp}}{z}\right)$$

<u>Case 3:</u>

➔ If the view plane is the uv plane and there are no restrictions on the placement of the projection reference point, then we

have $z_{vp} = 0$:

$$x_p = x\left(\frac{z_{prp}}{z_{prp} - z}\right) - x_{prp}\left(\frac{z}{z_{prp} - z}\right)$$

$$y_p = y\left(\frac{z_{prp}}{z_{prp} - z}\right) - y_{prp}\left(\frac{z}{z_{prp} - z}\right)$$

<u>Case 4:</u>

➔ With the uv plane as the view plane and the projection reference point on the zview axis, the perspective equations are

$x_{prp} = y_{prp} = z_{vp} = 0$:

$$x_p = x\left(\frac{z_{prp}}{z_{prp} - z}\right), \qquad y_p = y\left(\frac{z_{prp}}{z_{prp} - z}\right)$$

CO PO Mapping

Course Outcomes		Modules	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
CO1	Understand basics concepts and applications of Computer Graphics	1,2	2	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-
CO2	Design and implement algorithms for 2D graphics primitives and attributes.	2,3,5	2	2	2	-	-	-	-	-	-	-	-	-	-	-	-	-
CO3	Illustrate Geometric transformations on both 2D and 3D objects.	2,3,4	2	2	-	-	-	-	-	-	-	-	-	-	-	-	-	-
CO4	Understand concepts of clipping and visible surface detection in 2D and 3D viewing, and Illumination Models.	3,4	2	2	2	-	-	-	-	-	-	-	-	-	-	-	-	-
CO5	Design and implement interactive OpenGL graphics programs.	1,2,3,4,5	3	3	3	3	2	-	-	-	-	-	-	2	2	-	3	-

COGNITIVE LEVEL	REVISED BLOOMS TAXONOMY KEYWORDS
L1	List, define, tell, describe, identify, show, label, collect, examine, tabulate, quote, name, who, when, where, etc.
L2	summarize, describe, interpret, contrast, predict, associate, distinguish, estimate, differentiate, discuss, extend
L3	Apply, demonstrate, calculate, complete, illustrate, show, solve, examine, modify, relate, change, classify, experiment, discover.
L4	Analyze, separate, order, explain, connect, classify, arrange, divide, compare, select, explain, infer.
L5	Assess, decide, rank, grade, test, measure, recommend, convince, select, judge, explain, discriminate, support, conclude, compare, summarize.

PROGRAM OUTCOMES (PO), PROGRAM SPECIFIC OUTCOMES (PSO)				CORRELATION LEVELS	
PO1	Engineering knowledge	PO7	Environment and sustainability	0	No Correlation
PO2	Problem analysis	PO8	Ethics	1	Slight/Low
PO3	Design/development of solutions	PO9	Individual and team work	2	Moderate/ Medium
PO4	Conduct investigations of complex problems	PO10	Communication	3	Substantial/ High
PO5	Modern tool usage	PO11	Project management and finance		
PO6	The Engineer and society	PO12	Life-long learning		
PSO1	Develop applications using different stacks of web and programming technologies				
PSO2	Design and develop secure, parallel, distributed, networked, and digital systems				
PSO3	Apply software engineering methods to design, develop, test and manage software systems.				
PSO4	Develop intelligent applications for business and industry				