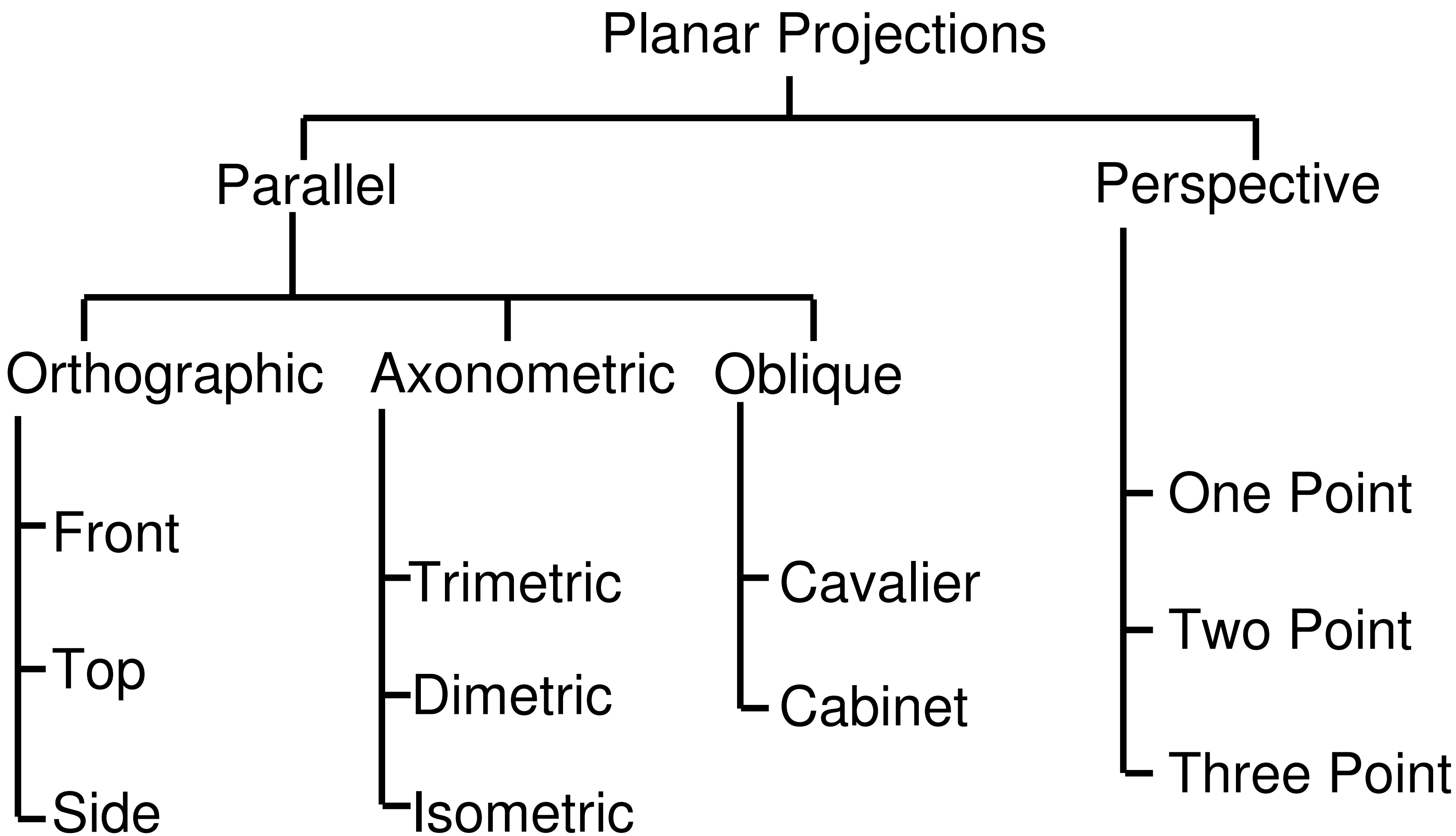
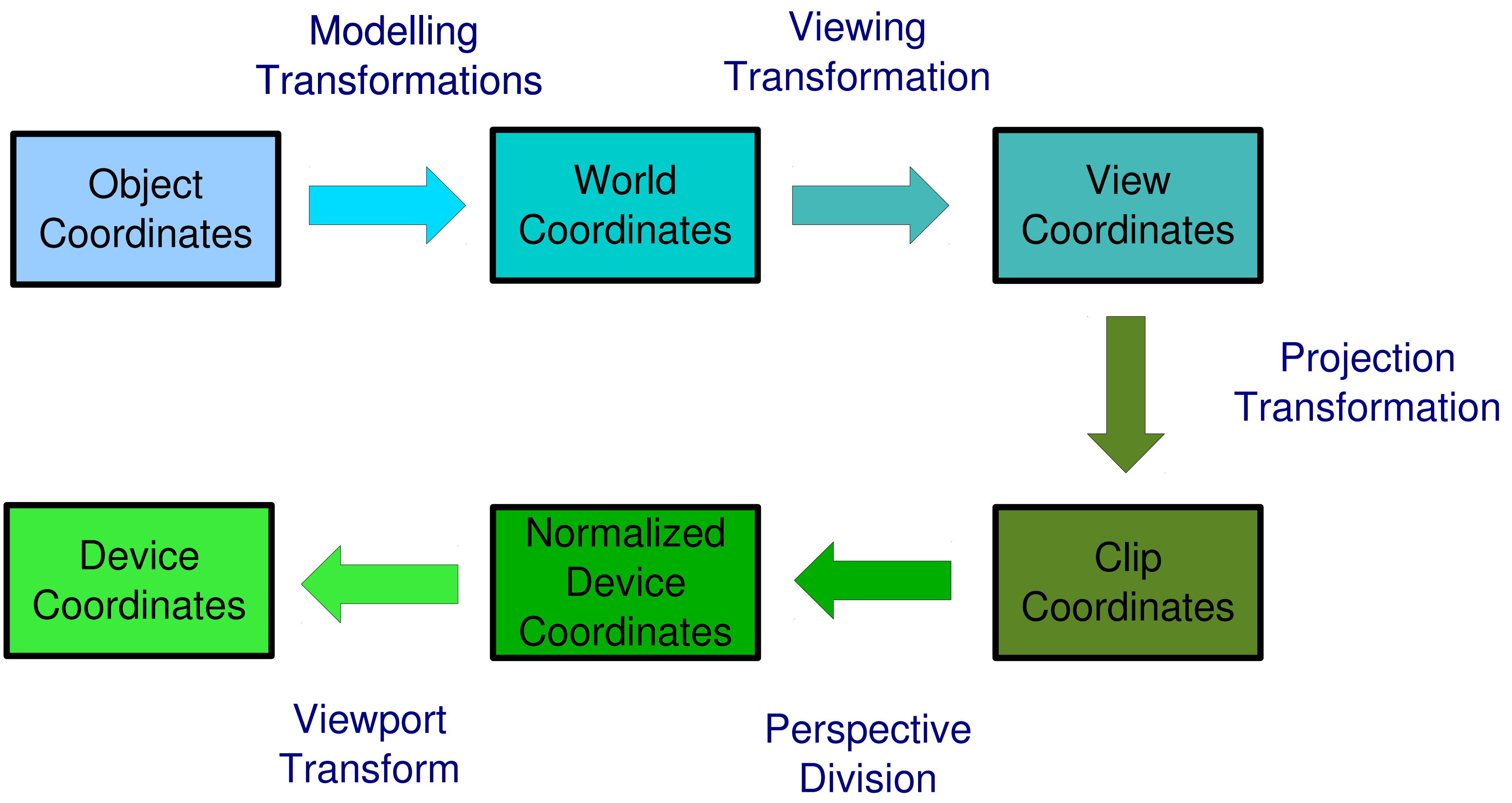
CS475/CS675
Computer Graphics

Modeling-Viewing Pipeline

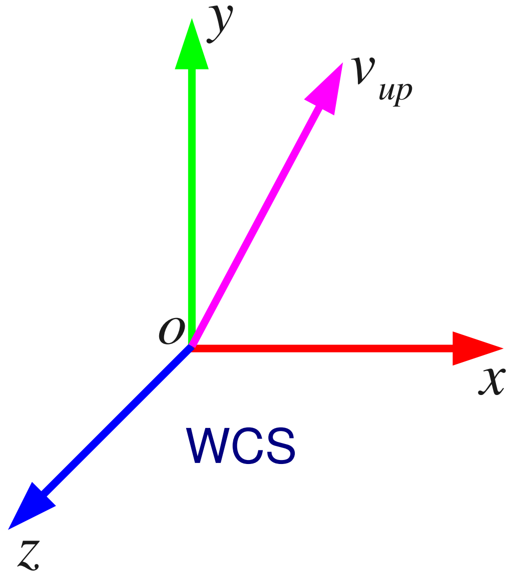
Taxonomy

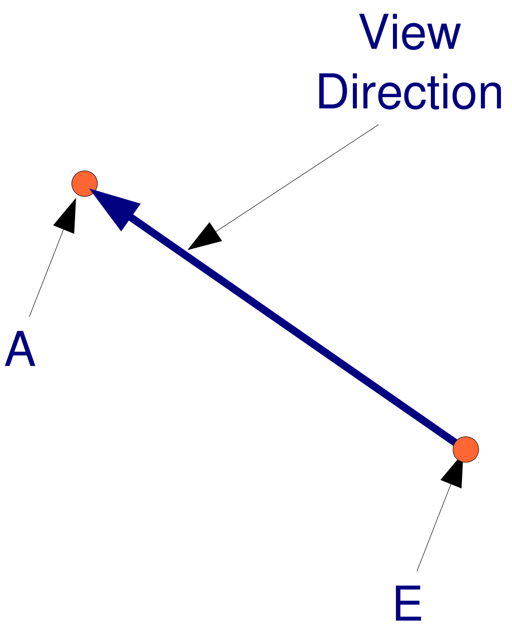

Planar Projections
Parallel
Perspective
Orthographic
Axonometric
Oblique
Front
Top
Side
Trimetric
Dimetric
Isometric
Cavalier
Cabinet
One Point
Two Point
Three Point

The Modeling-Viewing Pipeline



Viewing Transformation

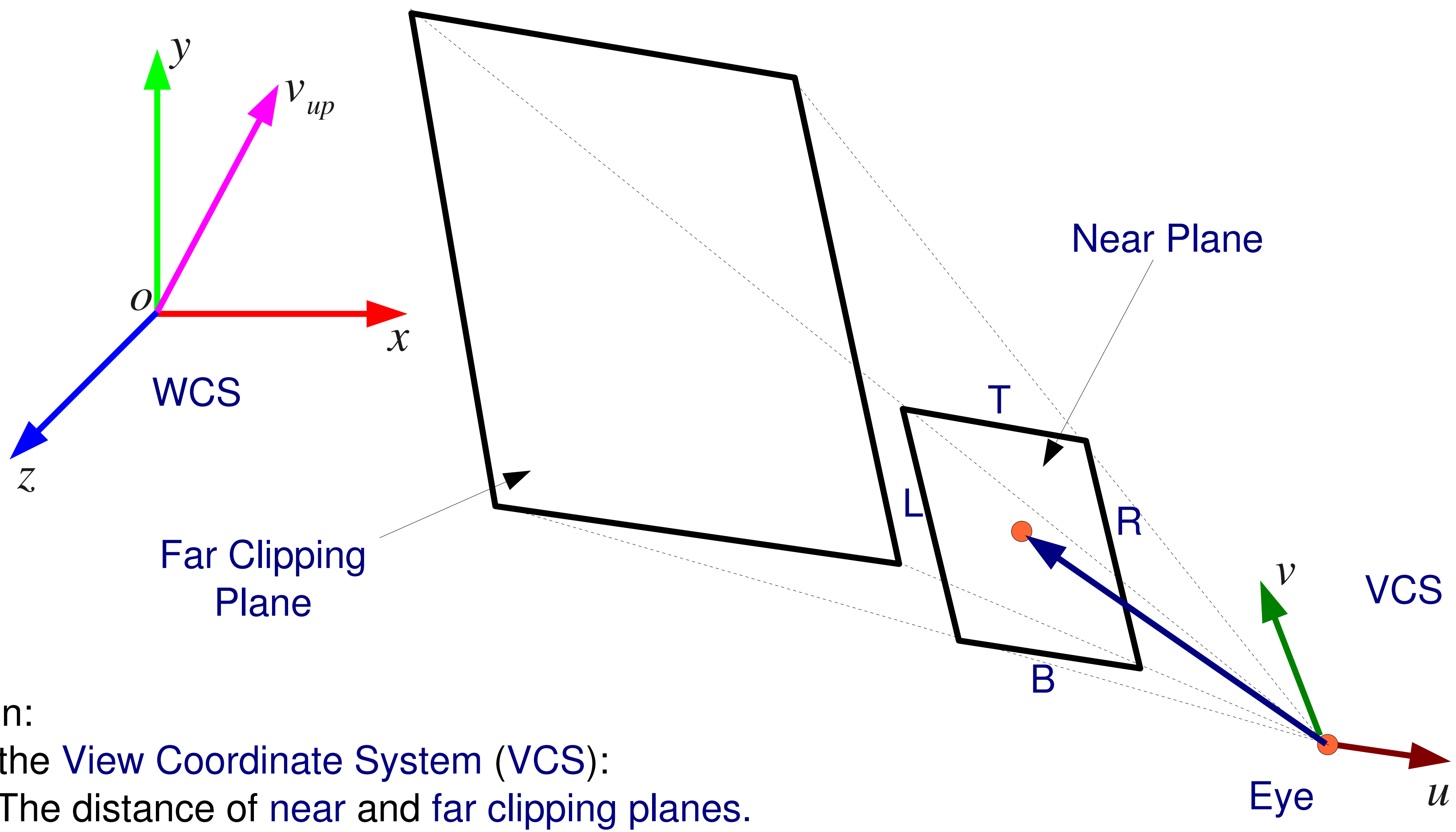




Given:

1. In the World Coordinate System (WCS):
   a) Position of the Eye (E)
   b) The lookat point (A)
   c) The up vector, v_{up} .

Viewing Transformation



Given:

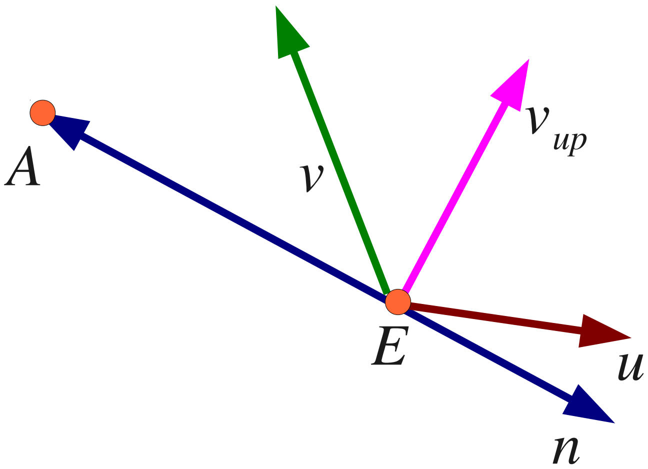
1. In the View Coordinate System (VCS):
   a) The distance of near and far clipping planes.
   b) Extents of the near plane L, R, T, B.

Defining the VCS

$$n = \frac{-(A - E)}{\|A - E\|}$$

$$u = \frac{v_{up} \times n}{\|v_{up} \times n\|}$$

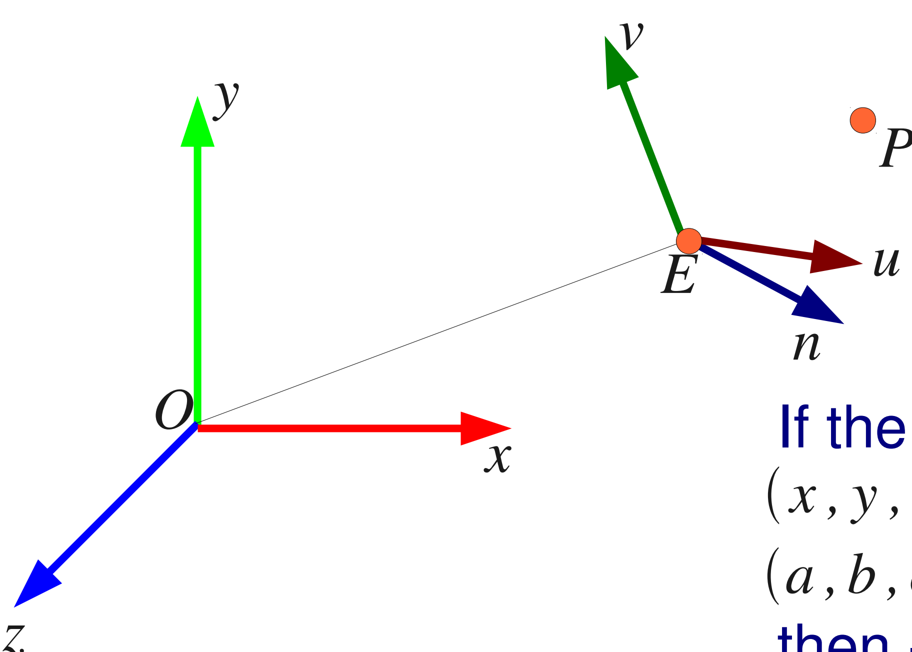
$$v = n \times u$$



From WCS to VCS

The viewing transformation should:

- $x \rightarrow u$
- $y \rightarrow v$
- $z \rightarrow n$
- Map the origin of the WCS, O to the Eye, E.

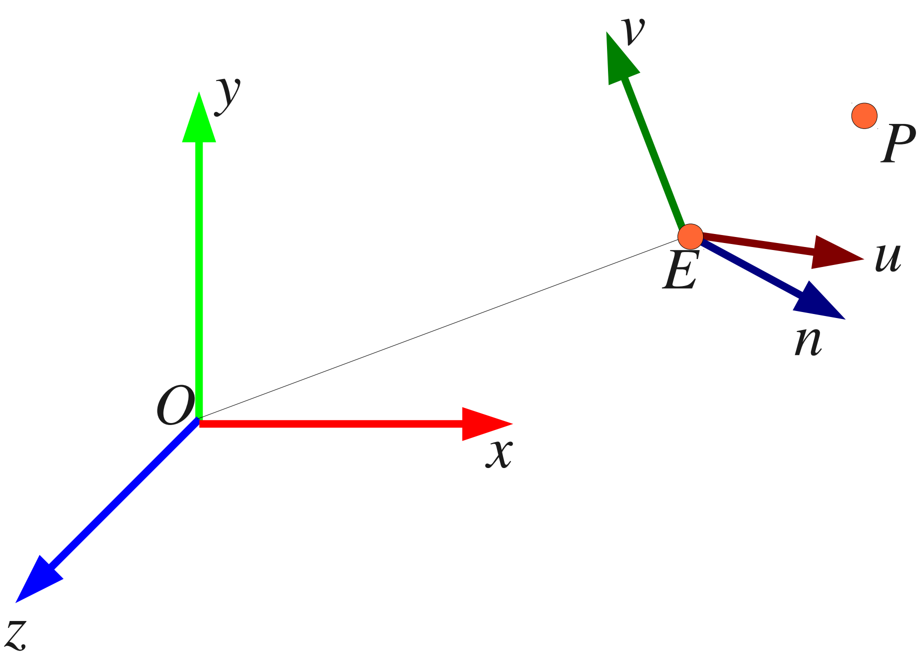


If the point P has coordinates
(x, y, z) in WCS
(a, b, c) in VCS
then -

From WCS to VCS

The viewing transformation should:

- $x \rightarrow u$
- $y \rightarrow v$
- $z \rightarrow n$
- Map the origin of the WCS, O to the Eye, E.

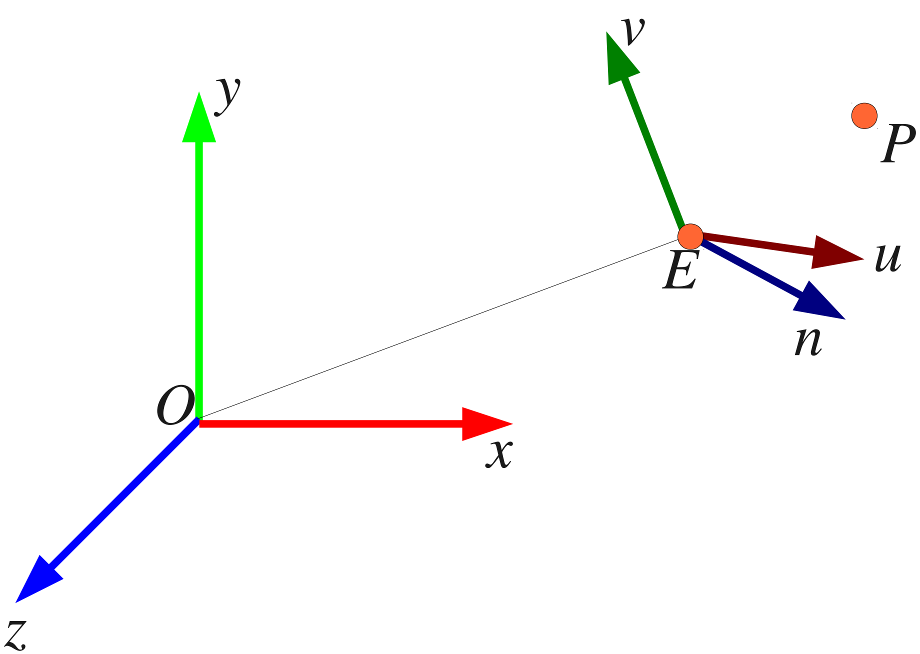


$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \vec{u} \; \vec{v} \; \vec{n} \end{bmatrix} . \begin{bmatrix} a \\ b \\ c \end{bmatrix} + \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix}$$

From WCS to VCS

The viewing transformation should:

- $x \rightarrow u$
- $y \rightarrow v$
- $z \rightarrow n$
- Map the origin of the WCS, O to the Eye, E.

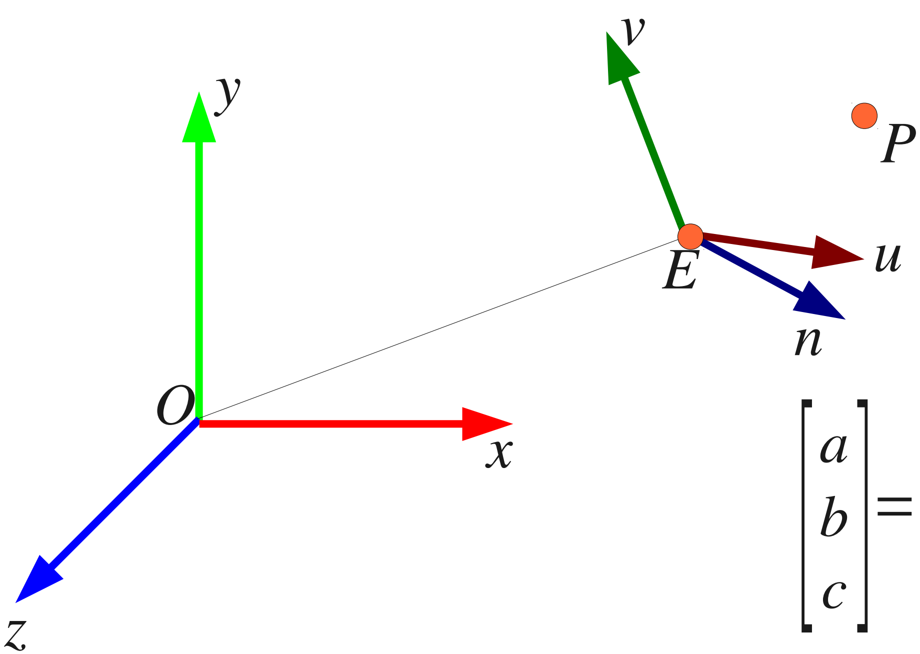


$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} u_x & v_x & n_x \\ u_y & v_y & n_y \\ u_z & v_z & n_z \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ c \end{bmatrix} + \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix}$$

From WCS to VCS

The viewing transformation should:

- $x \rightarrow u$
- $y \rightarrow v$
- $z \rightarrow n$
- Map the origin of the WCS, O to the Eye, E.

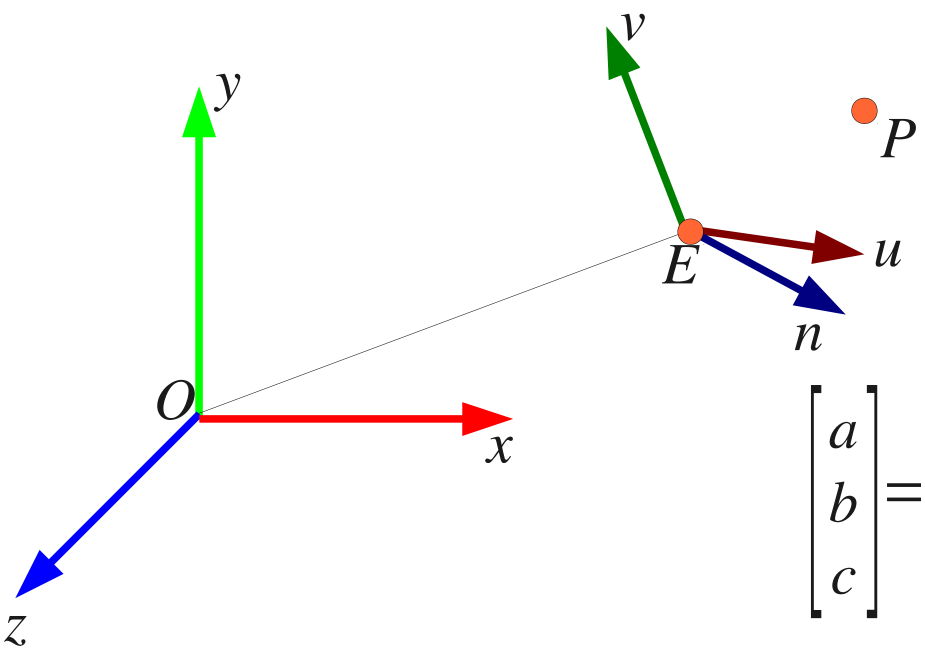


$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} u_x & v_x & n_x \\ u_y & v_y & n_y \\ u_z & v_z & n_z \end{bmatrix}^{-1} \cdot \left(\begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix} \right)$$

From WCS to VCS

The viewing transformation should:

- $x \to u$
- $y \to v$
- $z \to n$
- Map the origin of the WCS, O to the Eye, E.

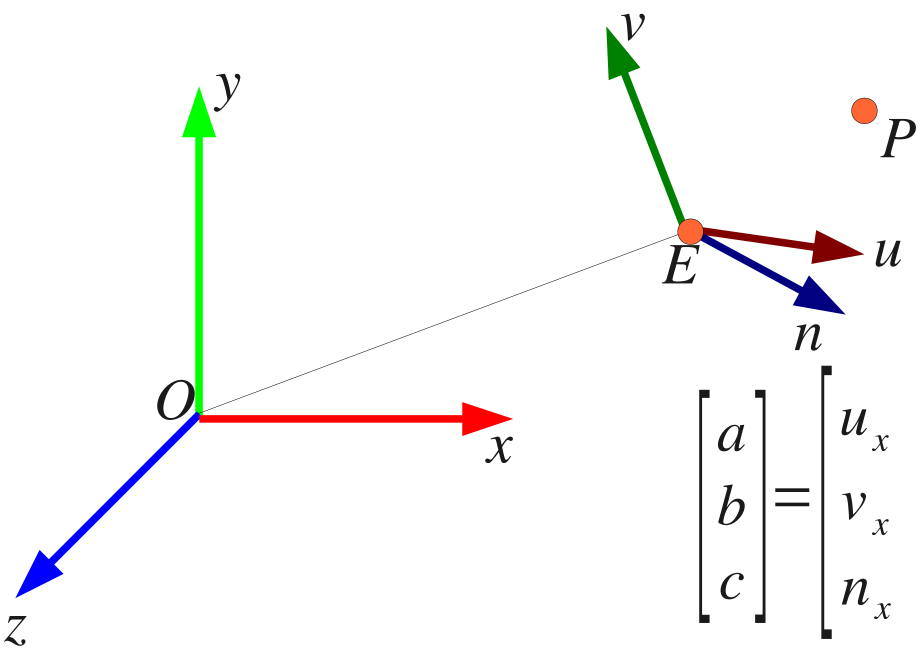


$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} u_x & v_x & n_x \\ u_y & v_y & n_y \\ u_z & v_z & n_z \end{bmatrix}^T \cdot \left(\begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix} \right)$$

From WCS to VCS

The viewing transformation should:

- $x \rightarrow u$
- $y \rightarrow v$
- $z \rightarrow n$
- Map the origin of the WCS, O to the Eye, E .

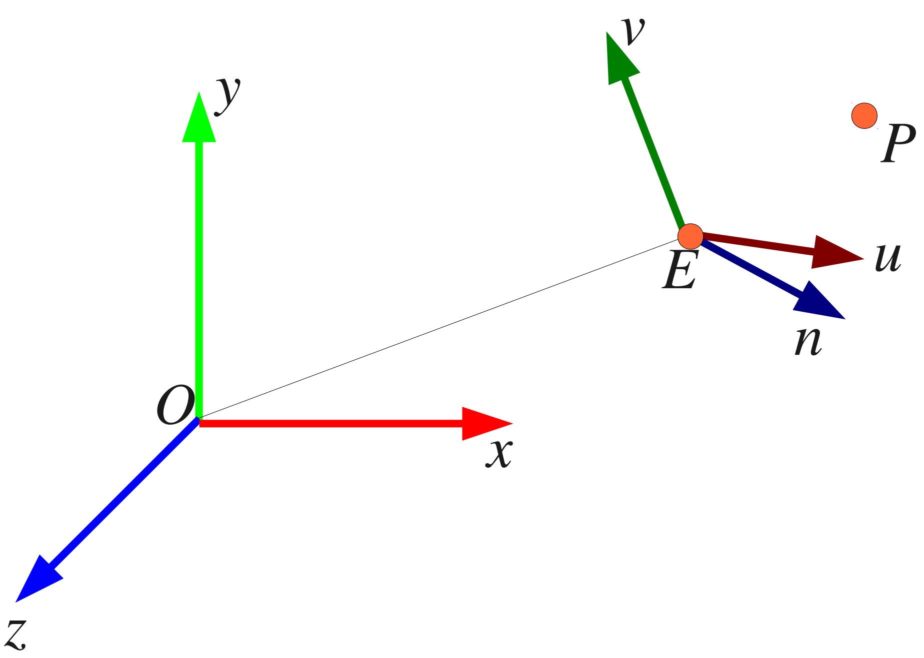


$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} u_x & u_y & u_z \\ v_x & v_y & v_z \\ n_x & n_y & n_z \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} u_x & u_y & u_z \\ v_x & v_y & v_z \\ n_x & n_y & n_z \end{bmatrix} \cdot \begin{bmatrix} e_x \\ e_y \\ e_z \end{bmatrix}$$

From WCS to VCS

The viewing transformation should:

- $x \rightarrow u$
- $y \rightarrow v$
- $z \rightarrow n$
- Map the origin of the WCS, O to the Eye, E.

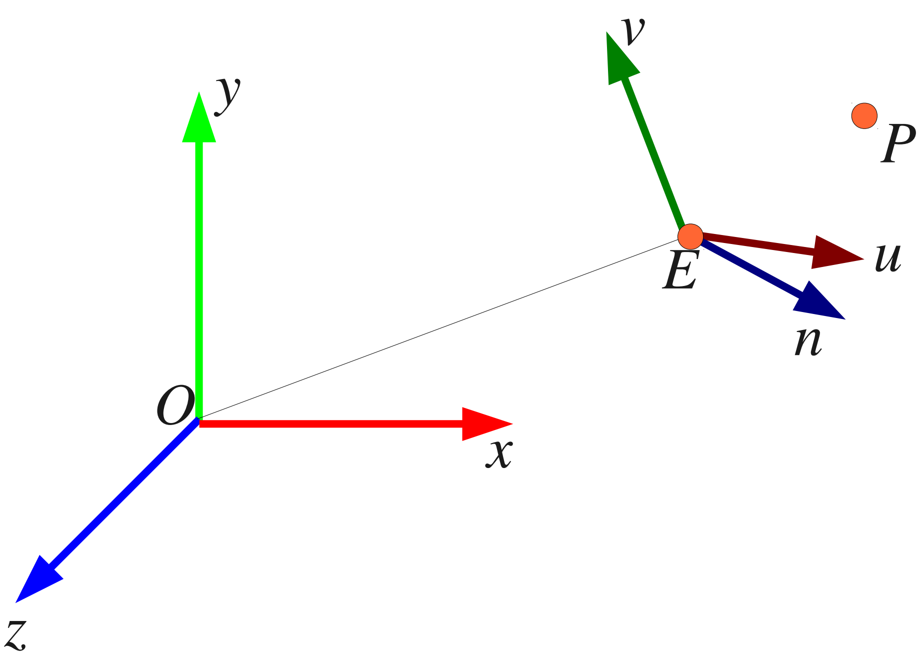


$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} \vec{u} \,.\, \vec{OP} \\ \vec{v} \,.\, \vec{OP} \\ \vec{n} \,.\, \vec{OP} \end{bmatrix} - \begin{bmatrix} \vec{u} \,.\, \vec{OE} \\ \vec{v} \,.\, \vec{OE} \\ \vec{n} \,.\, \vec{OE} \end{bmatrix}$$

From WCS to VCS

The viewing transformation should:

- $x \rightarrow u$
- $y \rightarrow v$
- $z \rightarrow n$
- Map the origin of the WCS, O to the Eye, E.

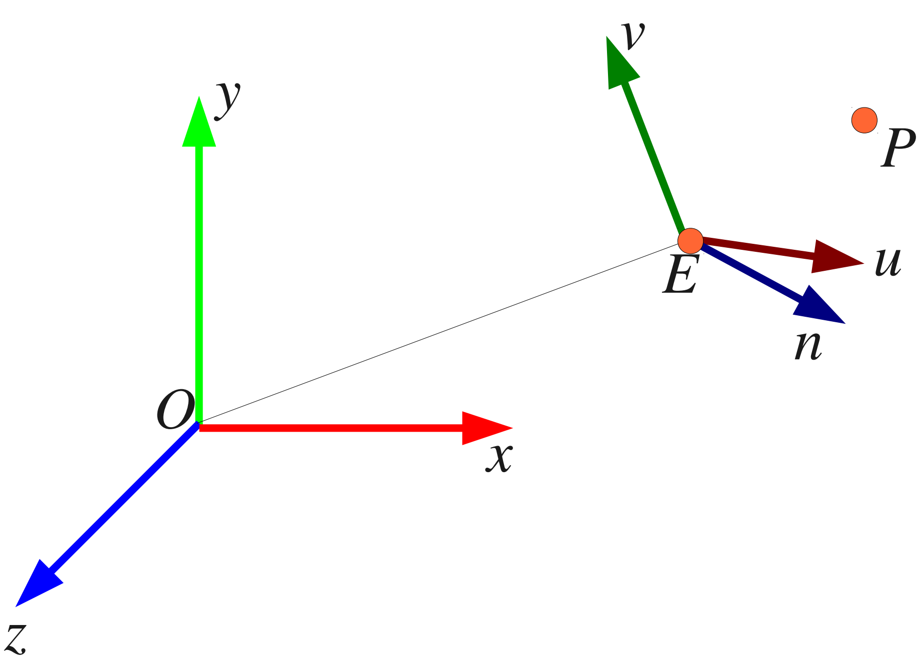


$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} \vec{u} \,.\, \vec{OP} \\ \vec{v} \,.\, \vec{OP} \\ \vec{n} \,.\, \vec{OP} \end{bmatrix} + \begin{bmatrix} e_x{}' \\ e_y{}' \\ e_z{}' \end{bmatrix}$$

From WCS to VCS

The viewing transformation should:

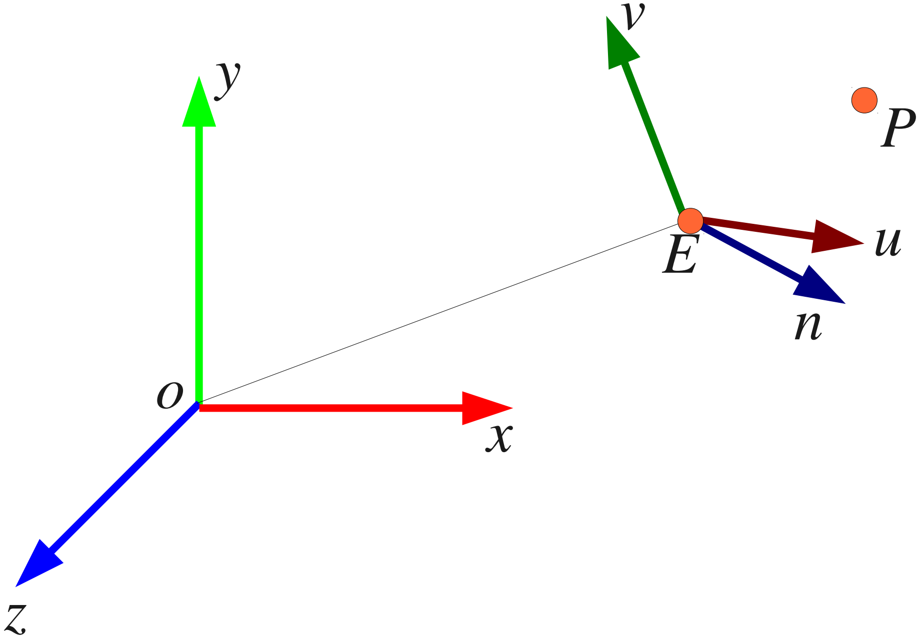
- $x \rightarrow u$
- $y \rightarrow v$
- $z \rightarrow n$
- Map the origin of the WCS, O to the Eye, E.



$$\begin{bmatrix} a \\ b \\ c \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} u_x & u_y & u_z & e_x' \\ v_x & v_y & v_z & e_y' \\ n_x & n_y & n_z & e_z' \\ 0 & 0 & 0 & 1 \end{bmatrix}}_{A_{WV}} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

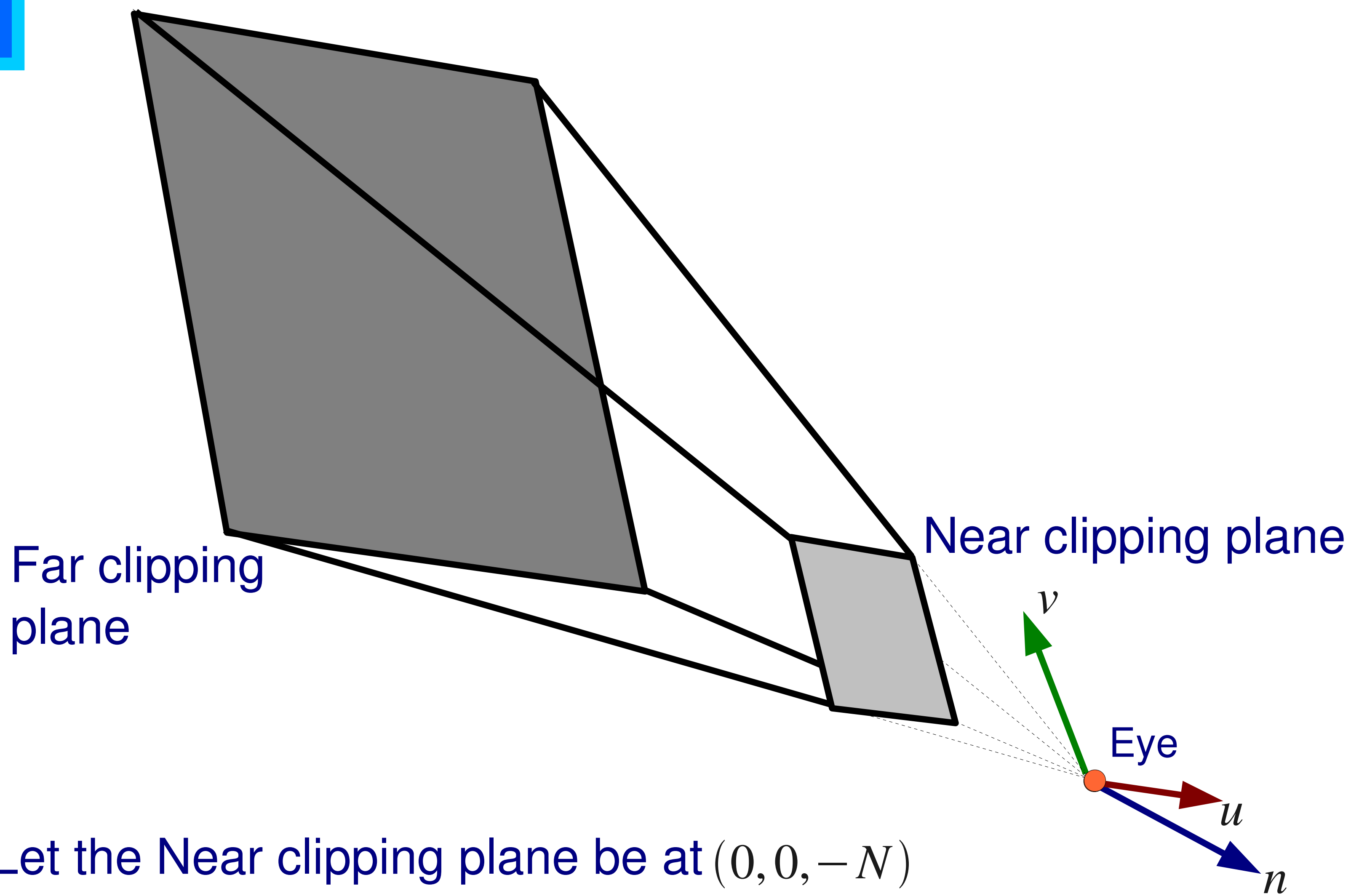
From WCS to VCS - OpenGL

glm::lookAt(glm::vec3 eye, glm::vec3 lookat_pt, glm::vec3 upvec);



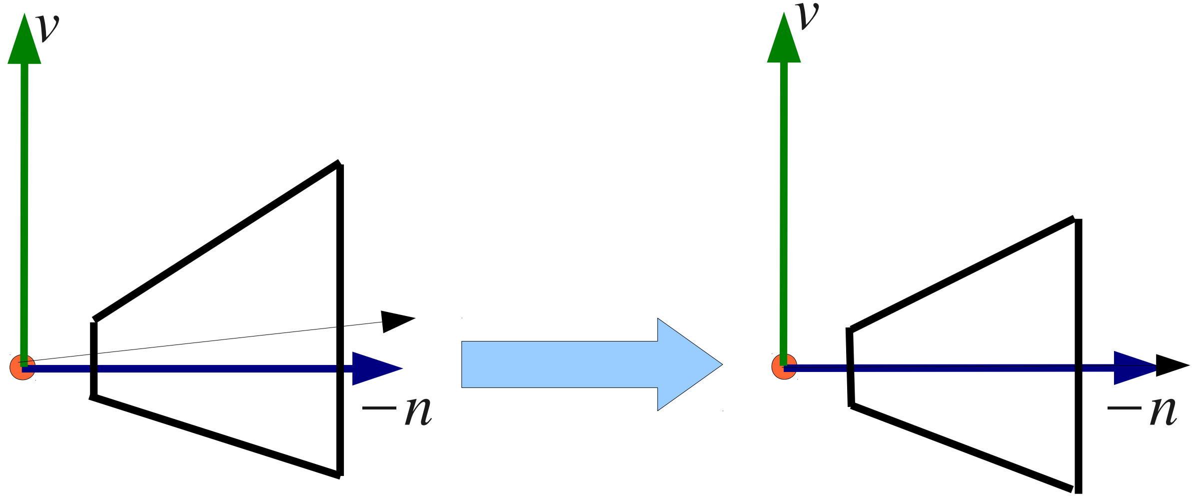
$$\begin{bmatrix} a \\ b \\ c \\ 1 \end{bmatrix} = \overbrace{\begin{bmatrix} u_x & u_y & u_z & e_x' \\ v_x & v_y & v_z & e_y' \\ n_x & n_y & n_z & e_z' \\ 0 & 0 & 0 & 1 \end{bmatrix}}^{A_{WV}} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

From WCS to VCS



Let the Near clipping plane be at $(0, 0, -N)$
and Far clipping plane be at $(0, 0, -F)$

From VCS to CCS

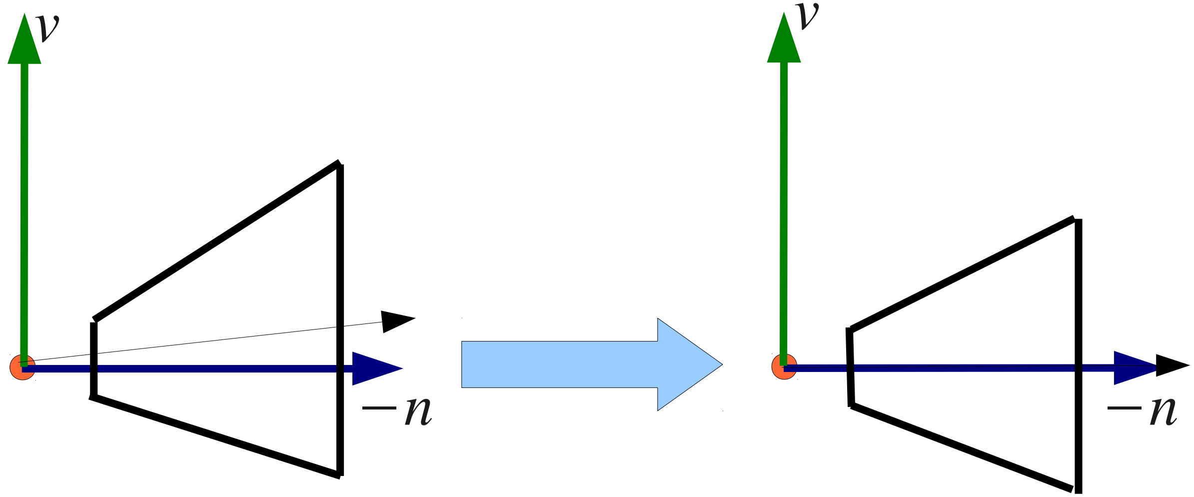


We shear the frustum so that the direction of projection aligns with the -n axis and frustum becomes symmetrically aligned about it.

If the extents of the near plane are given by L, R , T, B then:

$$\begin{bmatrix} 0 \\ 0 \\ -N \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & Sh_{xz} & 0 \\ 0 & 1 & Sh_{yz} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} (R+L)/2 \\ (T+B)/2 \\ -N \\ 1 \end{bmatrix}$$

From VCS to CCS



We shear the frustum so that the direction of projection aligns with the -n axis and frustum becomes symmetrically aligned about it.

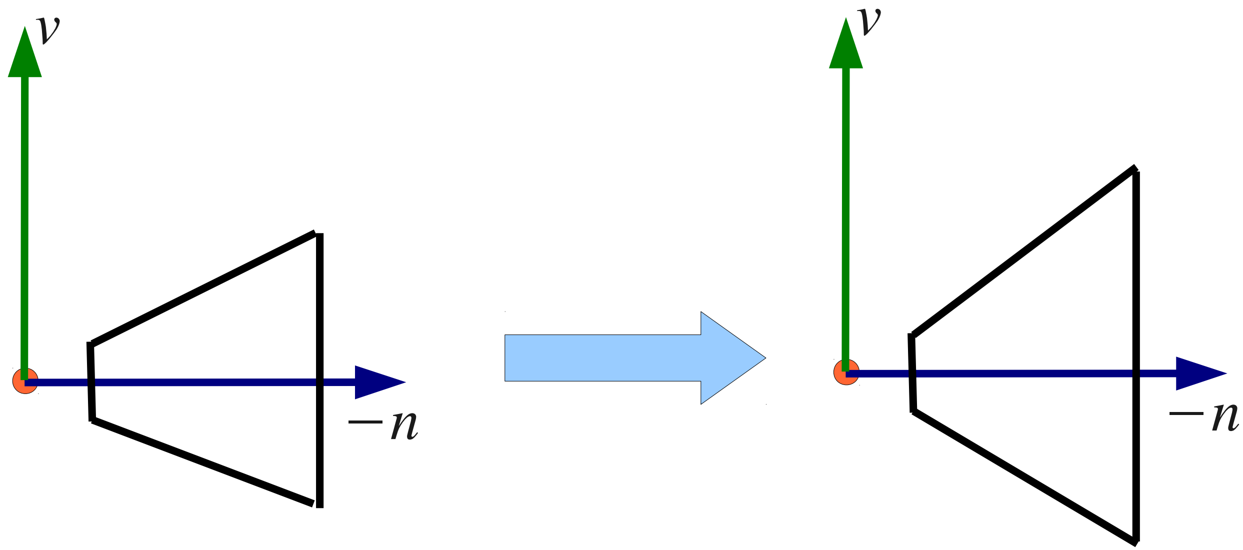
If the extents of the near plane are given by L, R , T, B then:

$$0 = (R+L)/2 - Sh_{xz}.N \Rightarrow Sh_{xz} = \frac{R+L}{2N}$$

$$0 = (T+B)/2 - Sh_{yz}.N \Rightarrow Sh_{yz} = \frac{T+B}{2N}$$

$$Sh = \begin{bmatrix} 1 & 0 & \frac{(R+L)}{2N} & 0 \\ 0 & 1 & \frac{(T+B)}{2N} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

From VCS to CCS

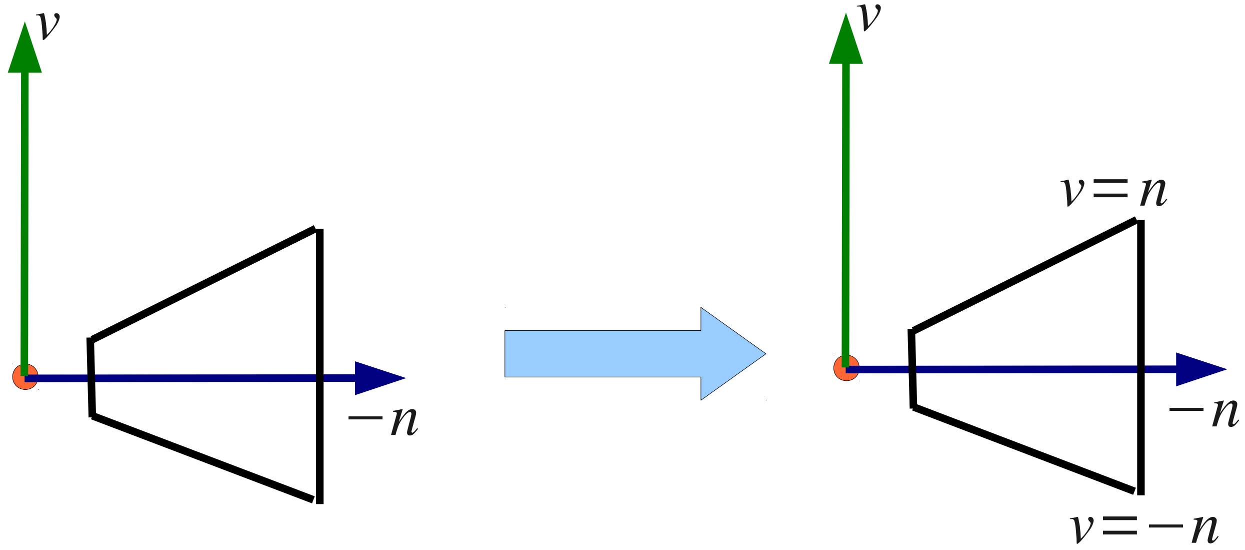


Now we scale along u and v so that we get $u=\pm n, v=\pm n$ as the top side faces of the frustum.

After shearing the point $(L, B, -N)$ becomes $\left(\frac{-(R-L)}{2}, \frac{-(T-B)}{2}, -N\right)$

After shearing the point $(R, T, -N)$ becomes $\left(\frac{(R-L)}{2}, \frac{(T-B)}{2}, -N\right)$

From VCS to CCS



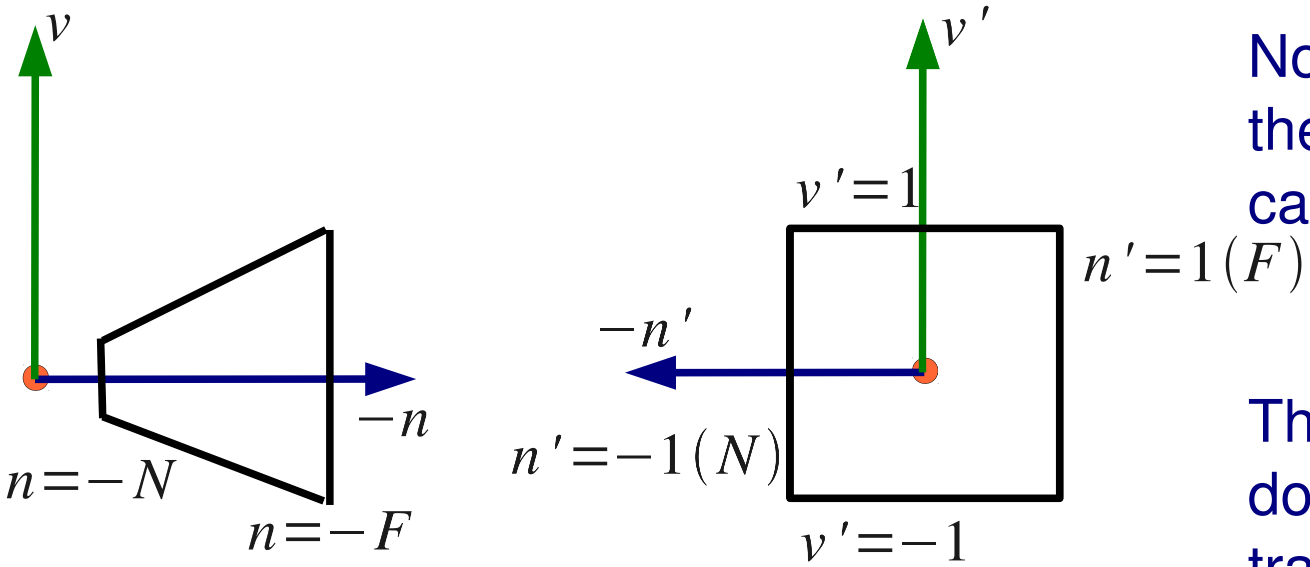
Now we scale along u and v so that we get $u=\pm n, v=\pm n$ as the side faces of the frustum.

So the scaling matrix should map

$$\left(\frac{\pm(R-L)}{2}, \frac{\pm(T-B)}{2}, -N\right) \text{ to } (\pm N, \pm N, -N)$$

$$Sc=\begin{bmatrix} \frac{2N}{(R-L)} & 0 & 0 & 0 \\ 0 & \frac{2N}{(T-B)} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

From VCS to CCS



Now we transform the frustum to a canonical frustum.

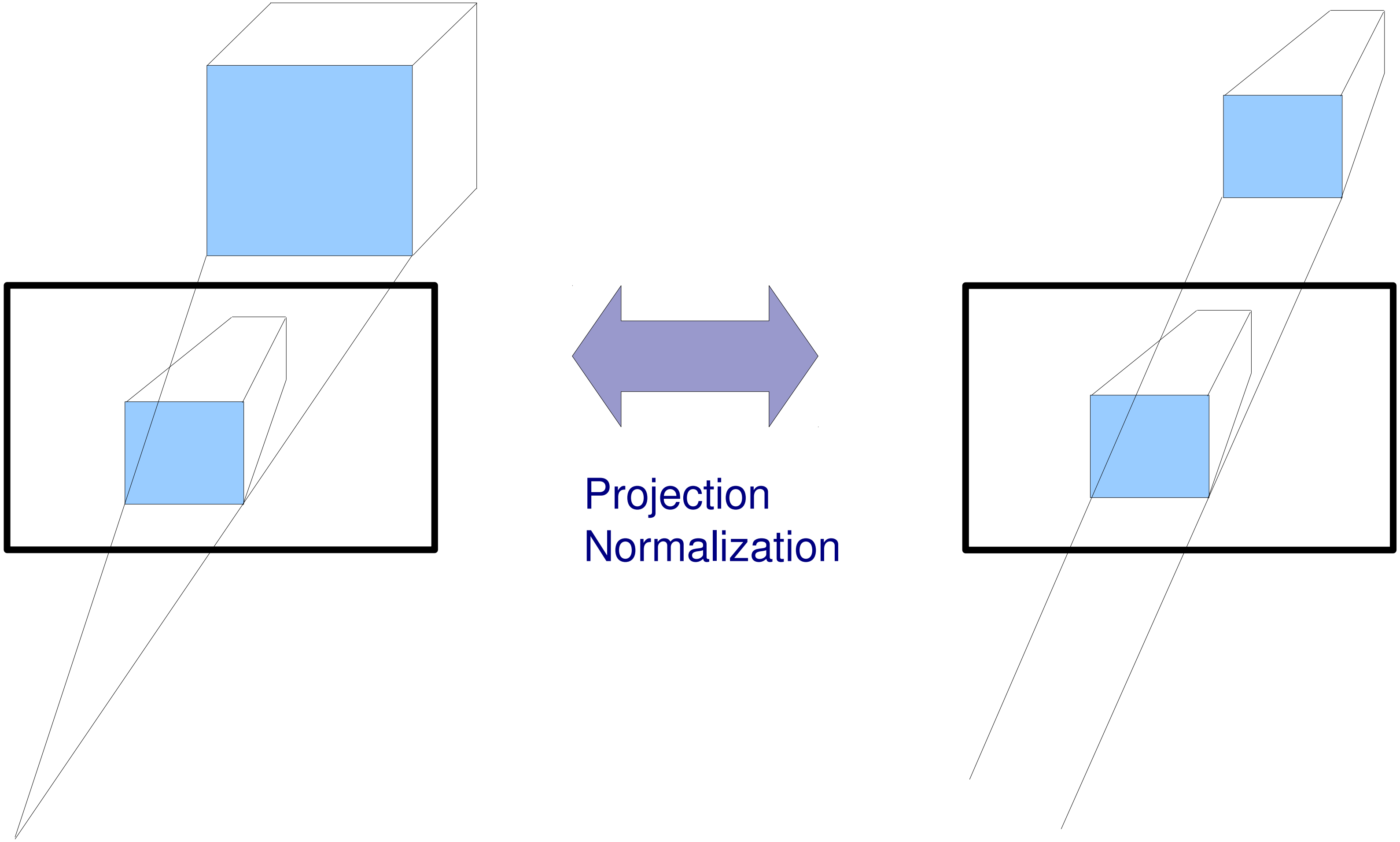
This is equivalent to doing a perspective transform. It is called a **projection normalization**.

$(u, v, -N)$ should map to $(u/N, v/N, -1)$

$(u, v, -F)$ should map to $(u/F, v/F, 1)$

$$Nm = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \frac{-(F+N)}{(F-N)} & \frac{-2\mathrm{FN}}{(F-N)} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

Projection Normalization



An Orthographic projection of a distorted object can be the same as a Perspective projection of the undistorted object.

From VCS to CCS

So the complete transformation is:

$$A_{vc} = Nm \,.\, Sc \,.\, Sh$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \dfrac{-(F+N)}{(F-N)} & \dfrac{-2\mathrm{FN}}{(F-N)} \\ 0 & 0 & -1 & 0 \end{bmatrix} \cdot \begin{bmatrix} \dfrac{2\mathrm{N}}{(R-L)} & 0 & 0 & 0 \\ 0 & \dfrac{2\mathrm{N}}{(T-B)} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & \dfrac{(R+L)}{2\mathrm{N}} & 0 \\ 0 & 1 & \dfrac{(T+B)}{2\mathrm{N}} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

From VCS to NDCS

So the complete transformation is:

$$A_{vc} = Nm\,.\,Sc\,.\,Sh$$

$$= \begin{bmatrix} \frac{2N}{(R-L)} & 0 & \frac{(R+L)}{(R-L)} & 0 \\ 0 & \frac{2N}{(T-B)} & \frac{(T+B)}{(T-B)} & 0 \\ 0 & 0 & \frac{-(F+N)}{(F-N)} & \frac{-2FN}{(F-N)} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

CCS coordinates retain the homogenous coordinate.

This is follwed by a ***perspective divide*** stage where the coordinates are divided by the 'w' coordinate.

That puts all the coordinates within the normalized +/- 1 cube. This is called the Normalized Device Coordinate System.

From VCS to NDCS - OpenGL

```
glm::frustum(L, R, B, T, N, F);
```

or

```
glm::perspective(fovy, aspect, N, F);
```

N and F give the distance of Near and Far clipping planes from the Eye and must be positive numbers and must not be equal.

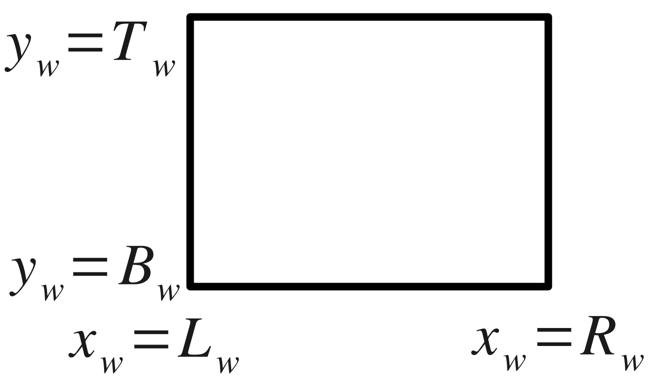
From NDCS to DCS

Now we need to map the NDCs to Device or Window coordinates. This is done by the viewport transformation. If the aspect of the viewport is not the same as that of the view frustum then the image is distorted.

$$x_w = \frac{(x+1).(R_w - L_w)}{2} + L_w$$

$$y_w = \frac{(y+1).(T_w - B_w)}{2} + B_w$$

$$z_w = \frac{(z+1)}{2}$$



From NDCS to DCS- OpenGL

glDepthRange(N_f, F_f); : Default range is 0 to 1

glViewport(x, y, width, height);

From VCS to NDCS - Orthographic

An Orhographic frustum is fully specified by L, R, T, B, N and F as well. Here there is no need to shear the frustum to make it symmetric – a translation can center it on the n-axis. This is followed by a scaling to transform the frustum to the canonical frustum. No projection normalization is required.

$$A_{vc} = Sc\,.\,T$$

$$= \begin{bmatrix} \frac{2}{(R-L)} & 0 & 0 & \frac{(R+L)}{(R-L)} \\ 0 & \frac{2}{(T-B)} & 0 & \frac{(T+B)}{(T-B)} \\ 0 & 0 & \frac{-2}{(F-N)} & \frac{F+N}{(F-N)} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

glm::ortho(L, R, B, T, N, F)

or

glm::ortho(L, R, B, T)

Why carry the depth through?

Intuition: Visibility computation or Hidden Surface Removal

Algorithm: The Z-Buffer Algorithm

OpenGL:
```
glClear(GL_DEPTH_BUFFER_BIT);
glEnable(GL_DEPTH_TEST);
```