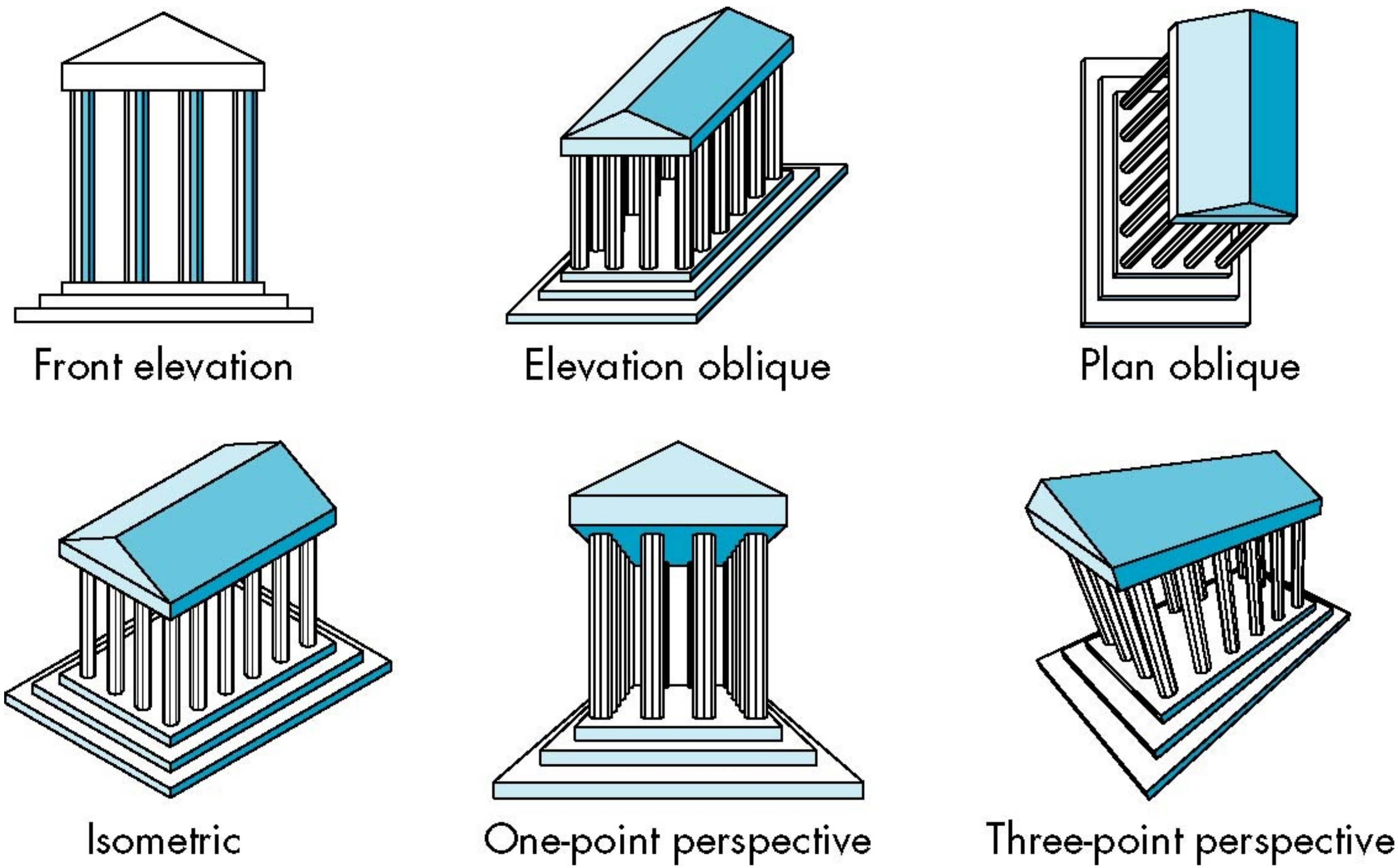
Viewing and Projections

What are Projections?

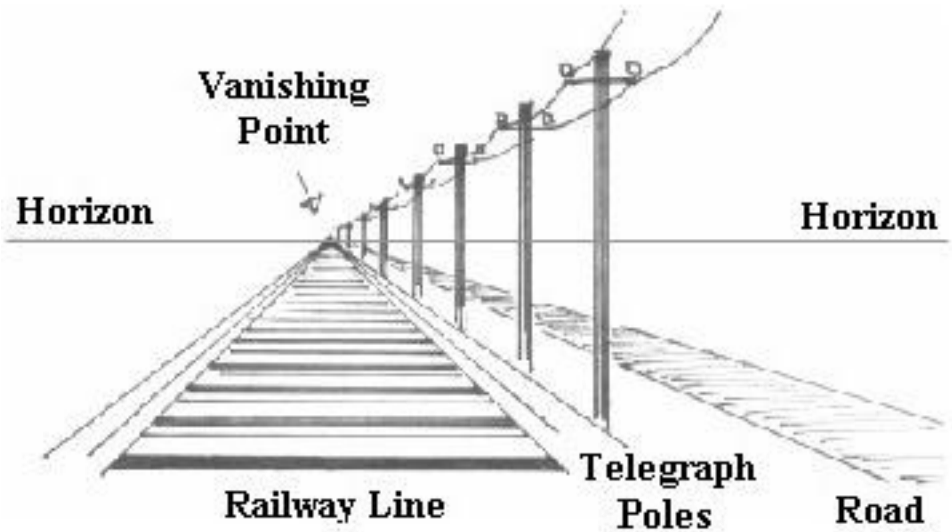
Classical Projections

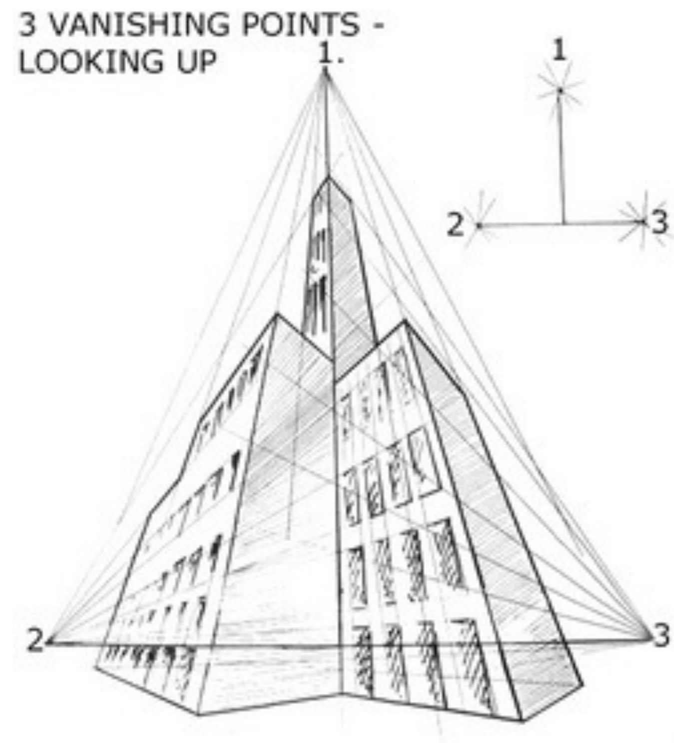


Planar Geometric Projections

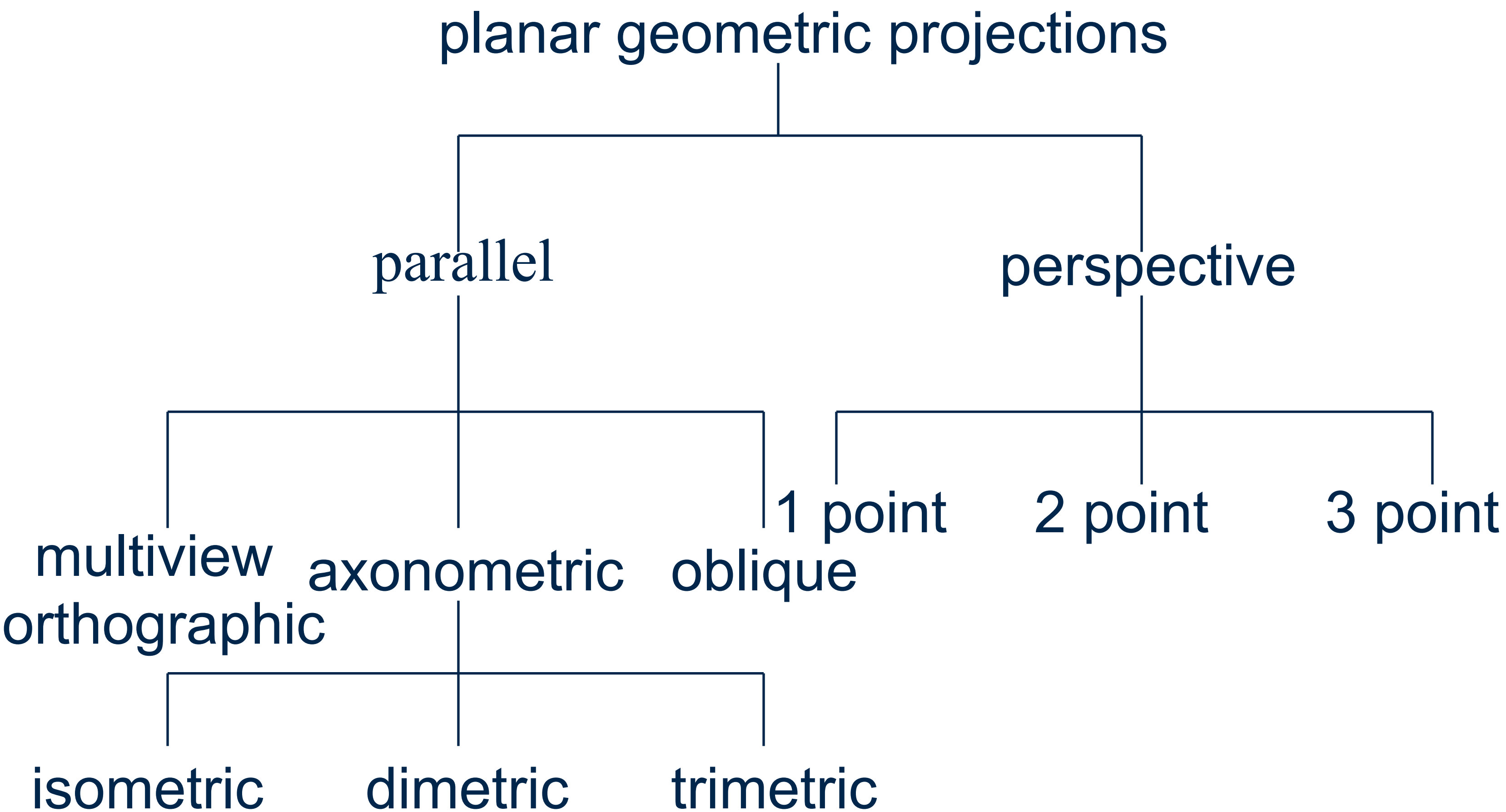
- Standard projections project onto a plane
- Projectors are lines that either:
  - Converge at center of projection
  - Are parallel
- Preserve lines but not angles

Remember Art Class?



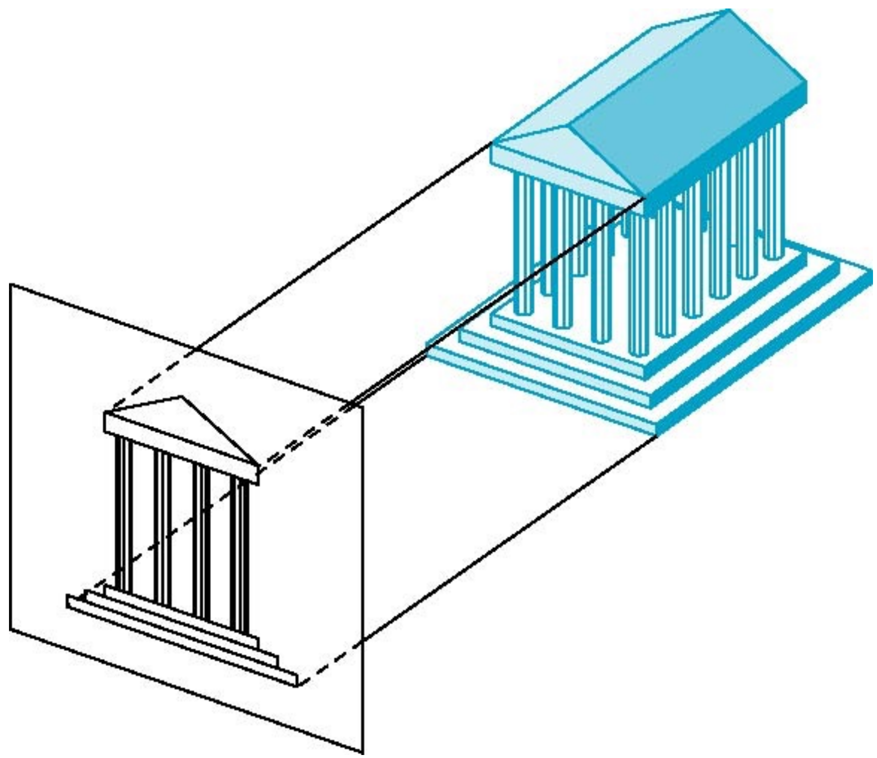


Projection Taxonomy



Orthographic Projection

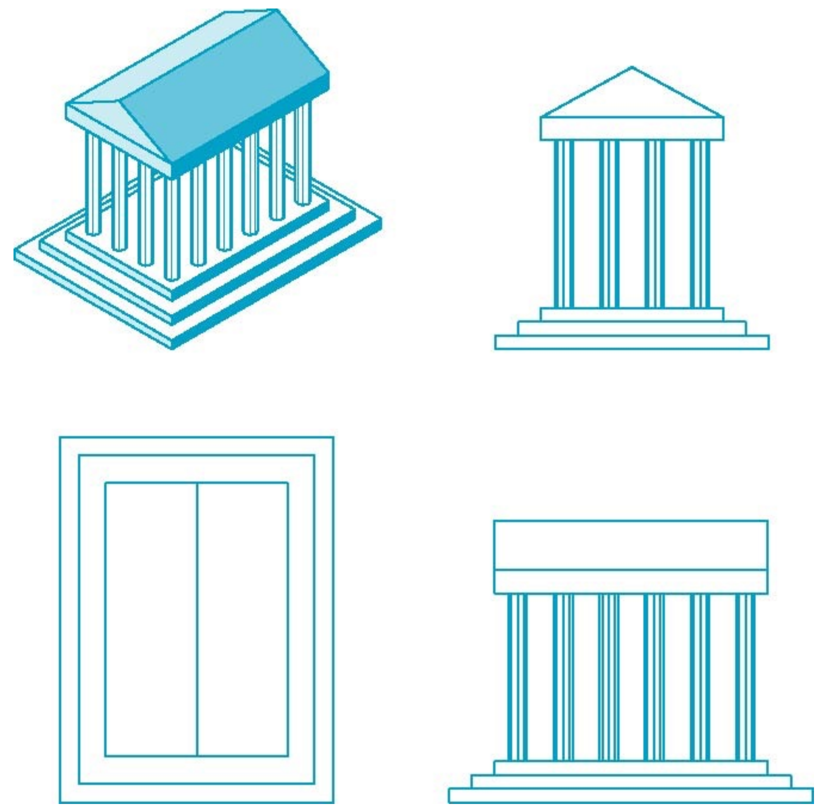
Projectors orthogonal to projection surface



Orthographic Uses

Preserves shape and measurements (great for CAD)

Need isometric to see what's hidden



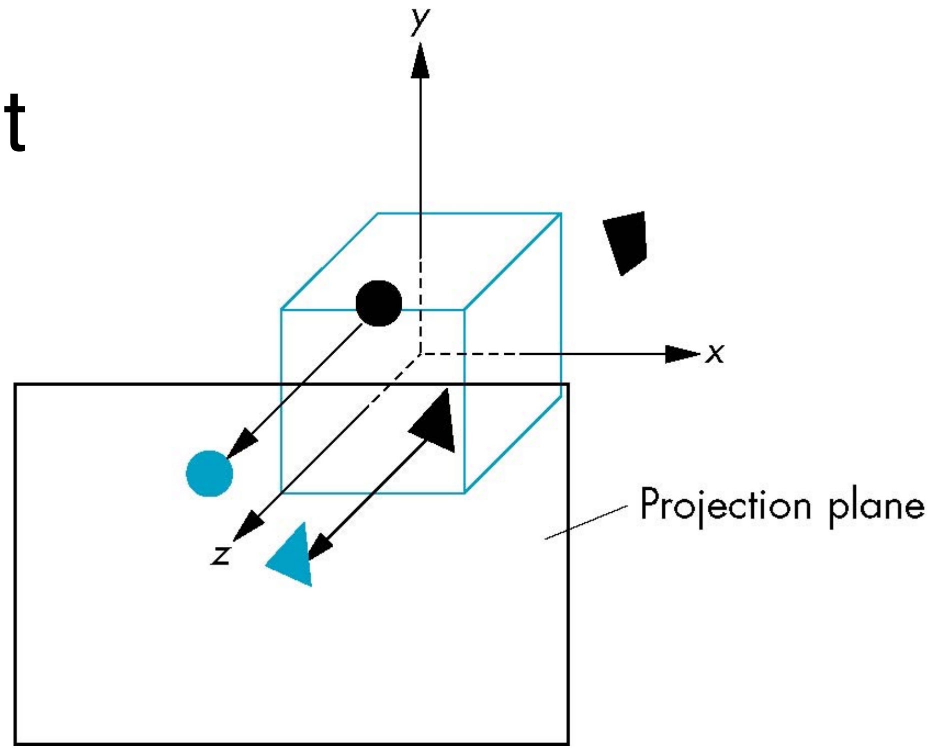
Default Camera Projection

Orthographic is default

$$\mathbf{p}_p = \mathbf{M}\mathbf{p}$$

$$x_p = x$$
$$y_p = y$$
$$z_p = 0$$
$$w_p = 1$$

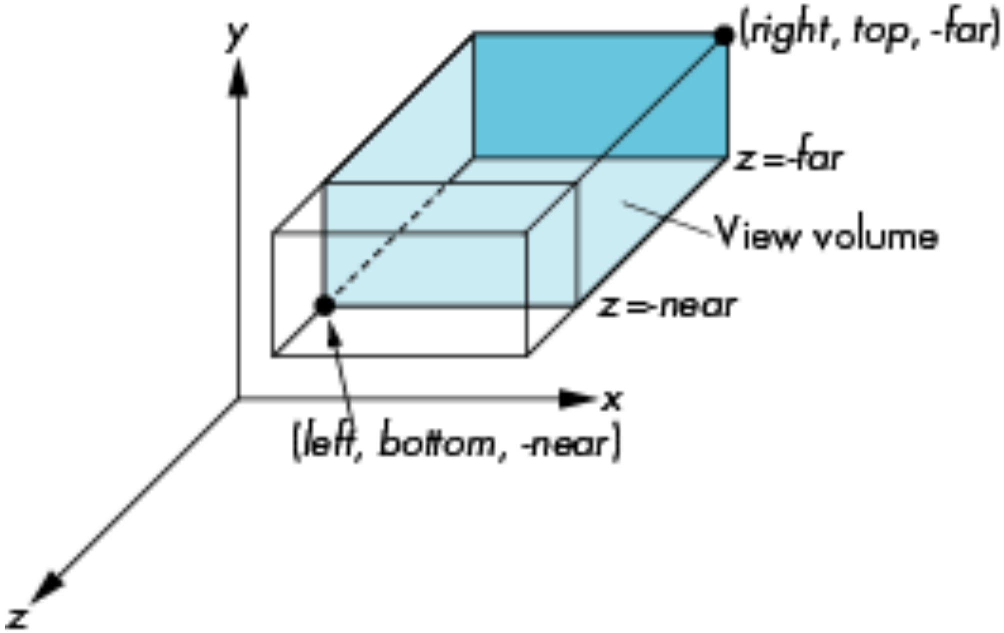
$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

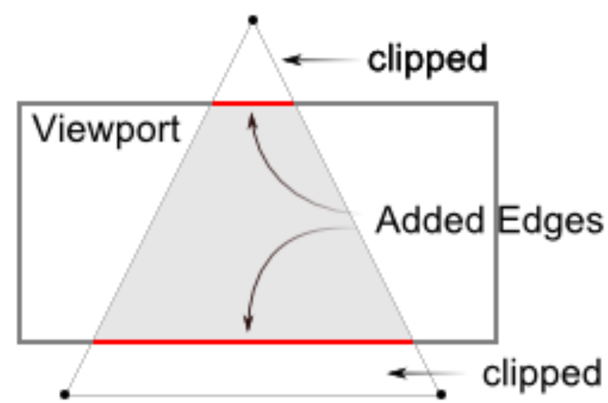


Projecting onto a Screen

Define area of screen and clip coordinates

`glOrtho(left,right,bottom,top,near,far)`

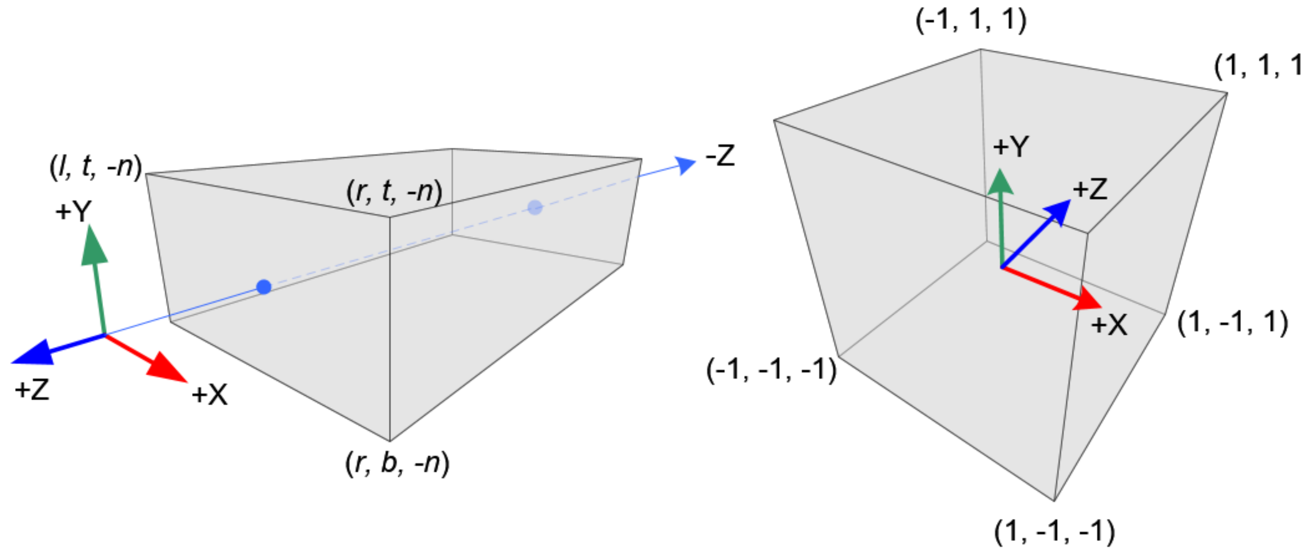




Normalized Device Coordinates

Transformed clipped coordinates to normalized device coordinates (NDC)

```
glOrtho(-1.0, 1.0, -1.0, 1.0, -1.0,
  1.0);
```



(coordinates outside NDC discarded)

Why Use NDC?

Provides a standard range for plotting onto a device/screen

“Screen space” coordinates that can then be transformed into device coordinates

Orthographic Eye to NDC

$$\begin{bmatrix} \frac{2}{right-left} & 0 & 0 & 0 \\ 0 & \frac{2}{top-bottom} & 0 & 0 \\ 0 & 0 & \frac{2}{far-near} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -\frac{left+right}{right-left} \\ 0 & 1 & 0 & -\frac{top+bottom}{top-bottom} \\ 0 & 0 & \boxed{-1} & -\frac{far+near}{far-near} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

NDC space flipped
(left-handed coordinate system)

P = ST

- Scale to have sides of length 2
- Move center to origin

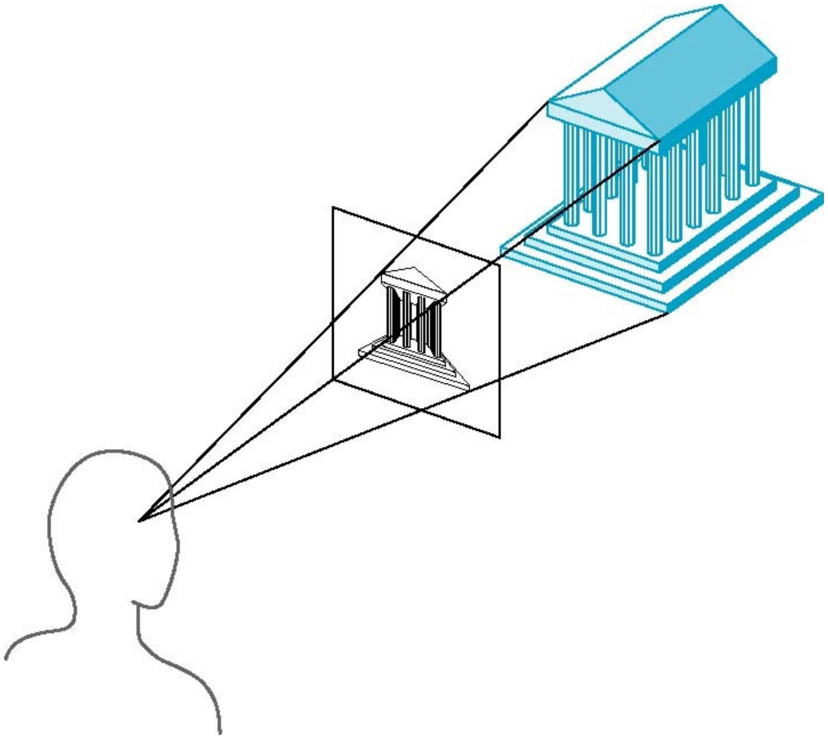
Orthographic Eye to NDC

- Scaled to have sides of length 2
- Centered at origin
- NDC looks down +Z axis

$$\begin{bmatrix} \frac{2}{right - left} & 0 & 0 & -\frac{right + left}{right - left} \\ 0 & \frac{2}{top - bottom} & 0 & -\frac{top + bottom}{top - bottom} \\ 0 & 0 & \frac{2}{near - far} & -\frac{far + near}{far - near} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Perspective Projection

- Converge at point along projection (vanishing point)
- Multiple vanishing points in multi-point perspective

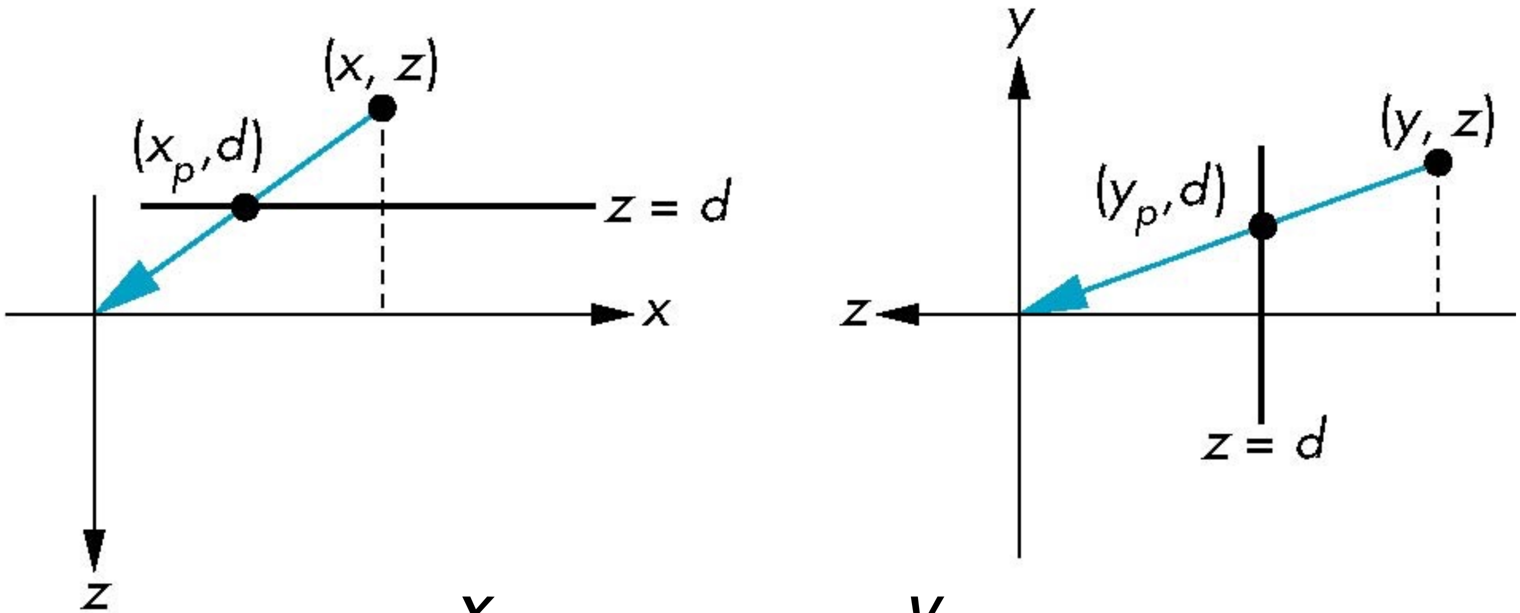


Projective Space

- **w** provides extra dimension to (x, y, z) coordinate space
- Acts as a scaling value to represent distance from projector
  - Larger w values correspond to more distance from viewer

Simple Perspective

- Center of projection at origin
- z is projection plane



$$x_p = \frac{x}{z/d} \qquad y_p = \frac{y}{z/d} \qquad z_p = d$$

Homogeneous Form

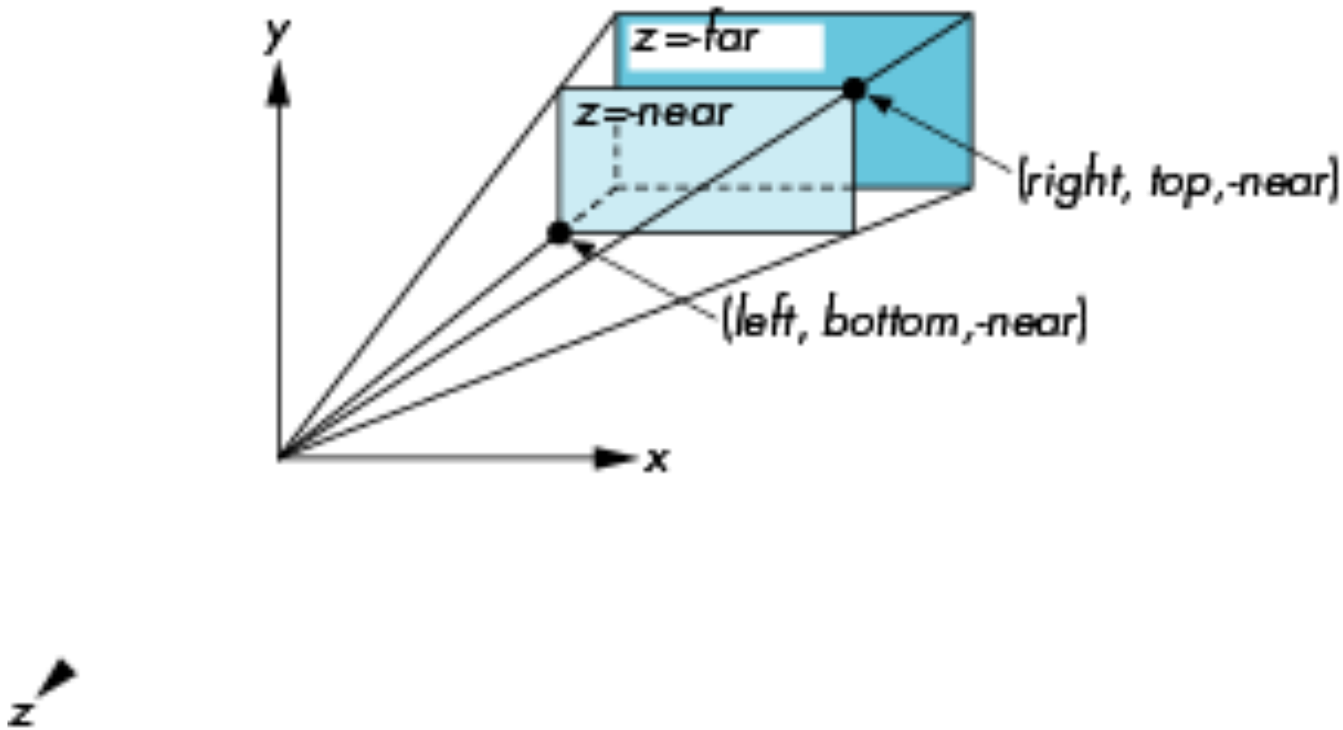
consider **Mp = p'** where:

$$\begin{bmatrix} x \\ y \\ z \\ z/d \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Apply perspective division (convert coordinate back to w=1) to be NDC
p' = (dx/z, dy/z, d, 1)

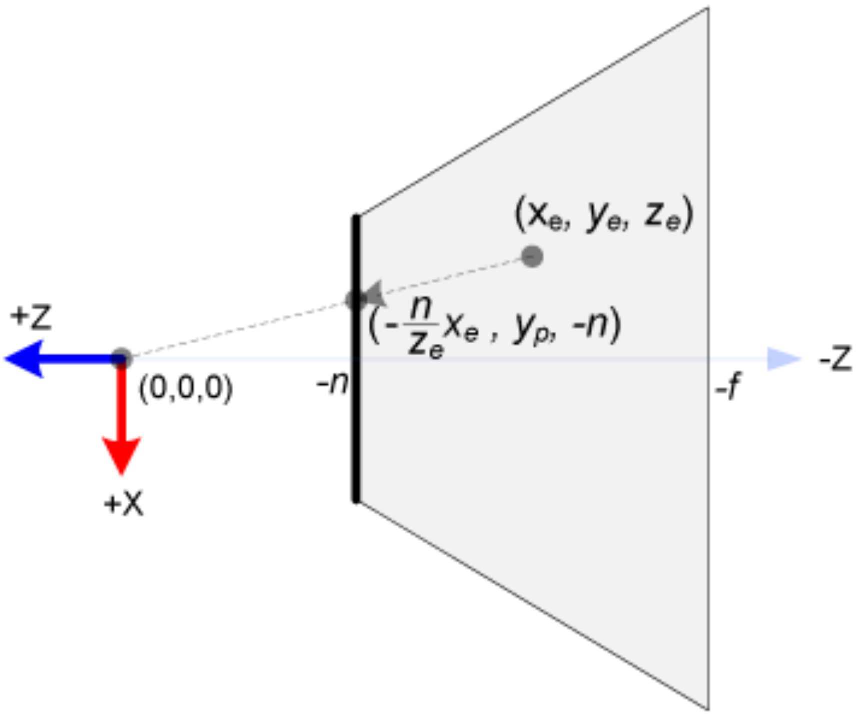
Perspective Projection

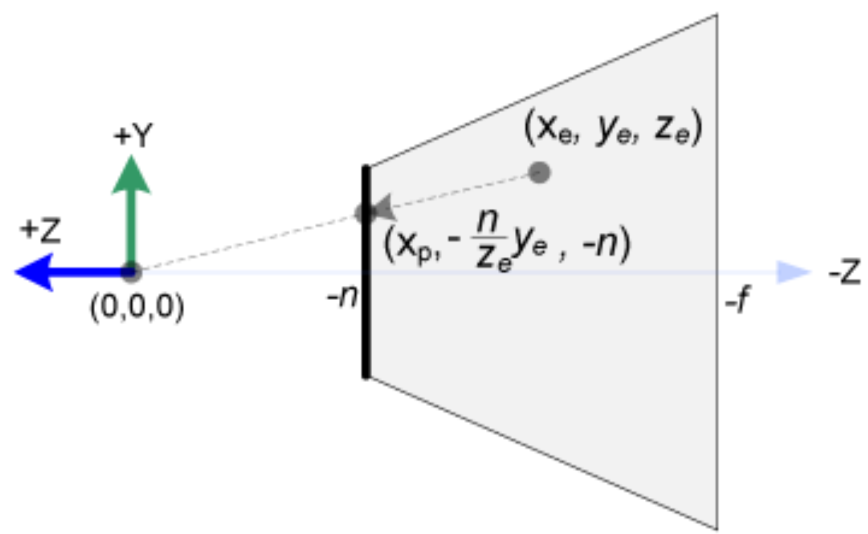
`glFrustum(left,right,bottom,top,near,far)`



Projecting onto the Near Plane

Map eye space point (x_e, y_e, z_e) to near plane point (x_p, y_p, z_p)





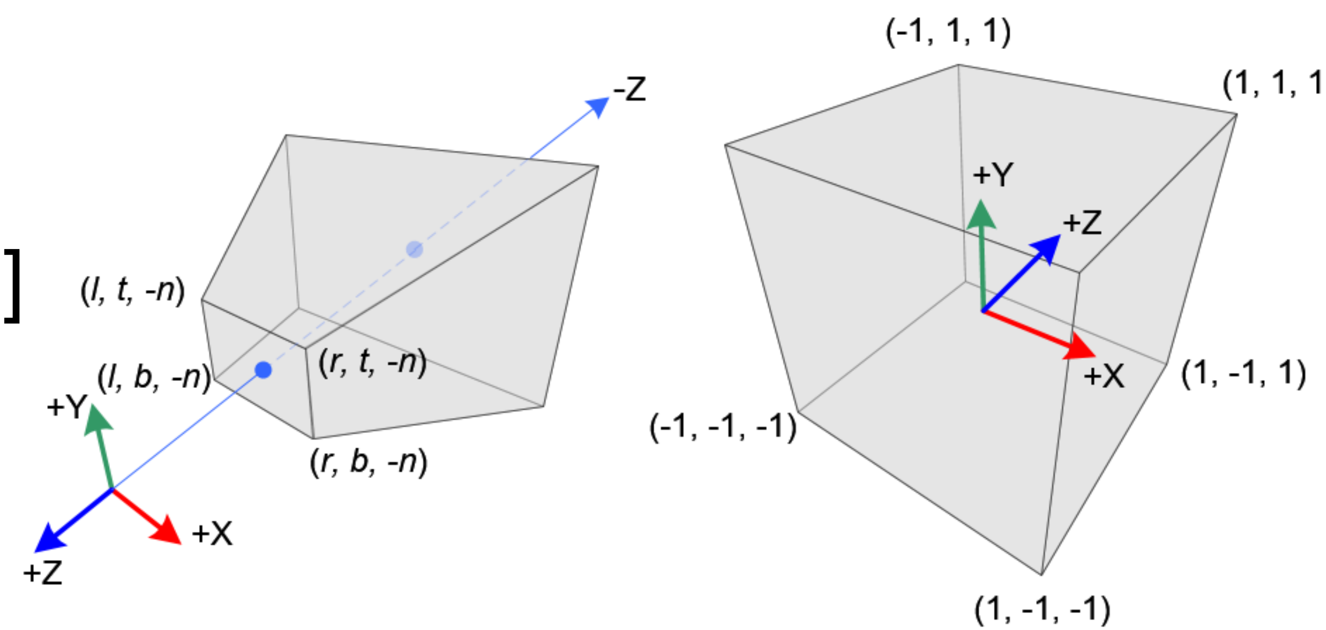
Perspective Normalization

Convert frustum into NDC coordinate system:

[l, r] = [-1, 1]

[b, t] = [-1, 1]

[-n, -f] = [-1, 1]



Frustum is in right-handed coordinate system; NDC is in left-handed coordinate system

Clipping

Only 4th column known
Use w to determine z in NDC space (3rd column)

$$\begin{bmatrix} x_c \\ y_c \\ z_c \\ w_c \end{bmatrix} = \begin{bmatrix} \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \alpha & \beta \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x_e \\ y_e \\ z_e \\ w_e \end{bmatrix} \qquad z_{ndc} = \frac{z_c}{w_c} = \frac{\alpha z_e + \beta w_e}{-z_e}$$

near plane is mapped to z = -1

far plane is mapped to z = 1

sides are mapped to $x = \pm 1$, $y = \pm 1$

Solving for Alpha and Beta

$$z_{ndc} = \frac{z_c}{w_c} = \frac{\alpha z_e + \beta w_e}{-z_e}$$

($w_e = 1$ in NDC)

Take ratio of near, far, and eye:

$$\frac{z_e}{z_{ndc}} = \frac{-n}{-1} = \frac{-f}{1}$$

$$\frac{-\alpha n + \beta}{n} = -1 \qquad \frac{-\alpha f + \beta}{f} = 1$$

Solving for Alpha and Beta

With a little algebra, we determine:

$$\alpha = \frac{-(f+n)}{f-n} \qquad \beta = \frac{-2nf}{f-n}$$

$$\begin{bmatrix} \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

General Frustum Transform

Mapping x and y into NDC using triangle ratios from earlier to determine 1st and 2nd columns...

Final matrix:

$$\begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

Symmetric Viewing Volume

When right = -left and top = -bottom:

r + l = 0

r - l = 2r

t + b = 0

t - b = 2t

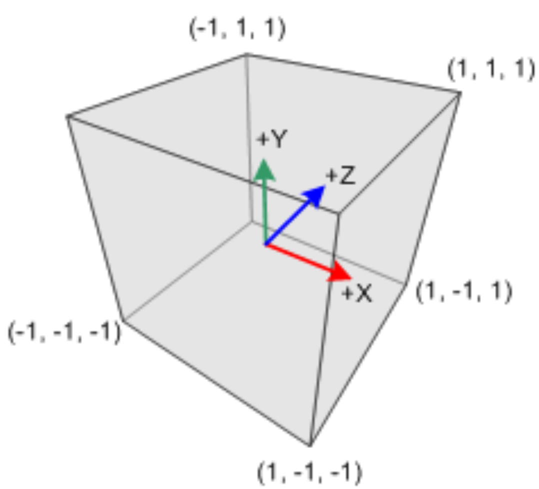
$$\begin{pmatrix} \frac{n}{r} & 0 & 0 & 0 \\ 0 & \frac{n}{t} & 0 & 0 \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

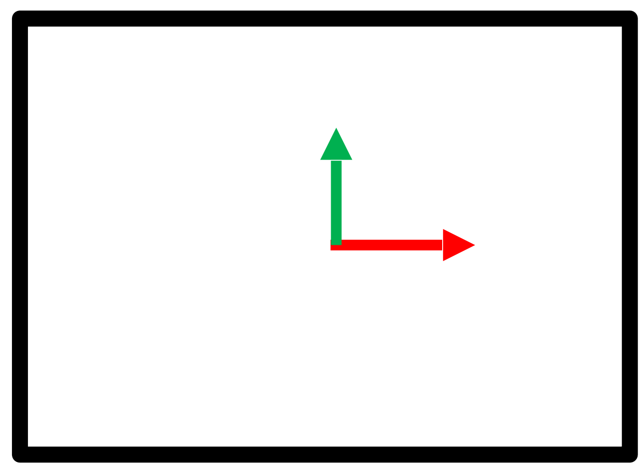
Normalized Device Coordinates

Note:

X and Y map to screen width and height

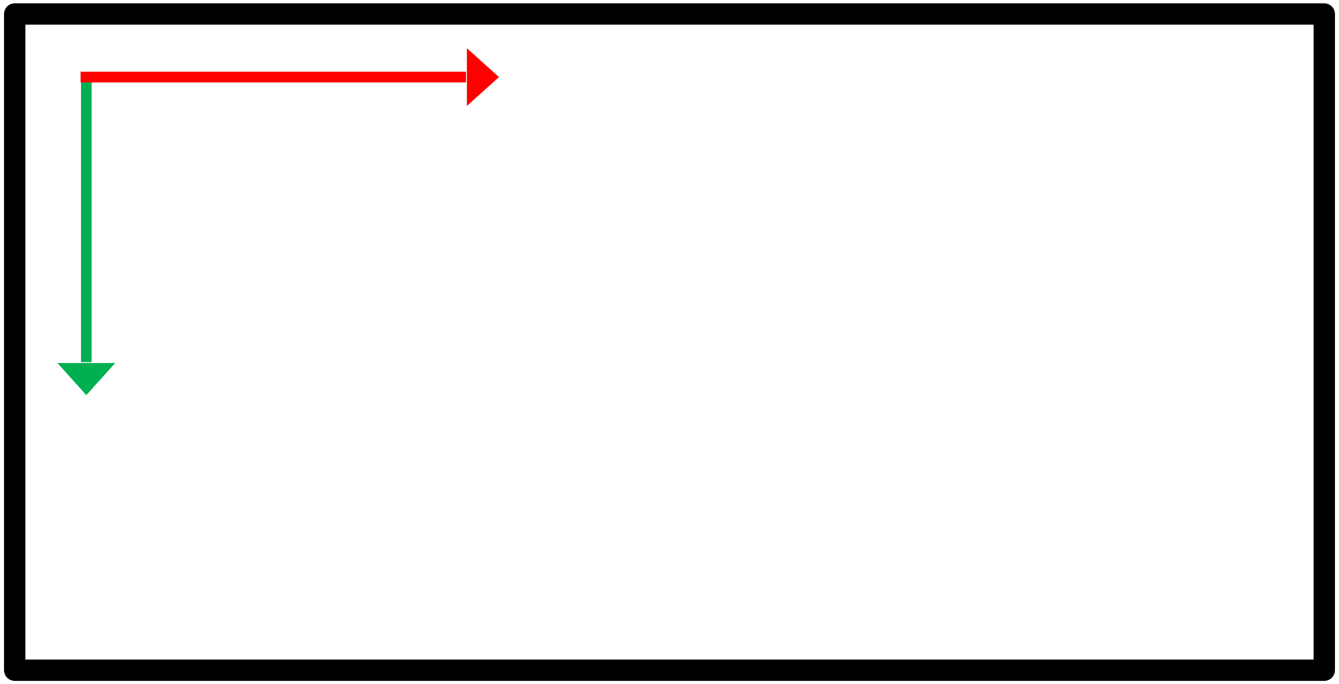
Z used for depth (deeper points are higher)





Screen Coordinates

Screen coordinates use different system!



Handling Aspect Ratio

`glViewPort(x, y, width, height)` transforms NDC to window coordinates

Allows for an aspect ratio in final display to screen after being normalized

Incidentally (x, y) specifies the *lower* left corner of the viewport

Note about Deprecation

glOrtho and glFrustum are deprecated as of OpenGL 3.0

Replacements:

glm::glOrtho

glm::glFrustum

Additional Reading

- http://www.songho.ca/opengl/gl_projectionmatrix.html
- https://www.scratchapixel.com/lessons/3d-basic-rendering/perspective-and-orthographic-projection-matrix/opengl-perspective-projection-matrix