

Lecture 4

Space (axis) transformations

What have we seen so far?

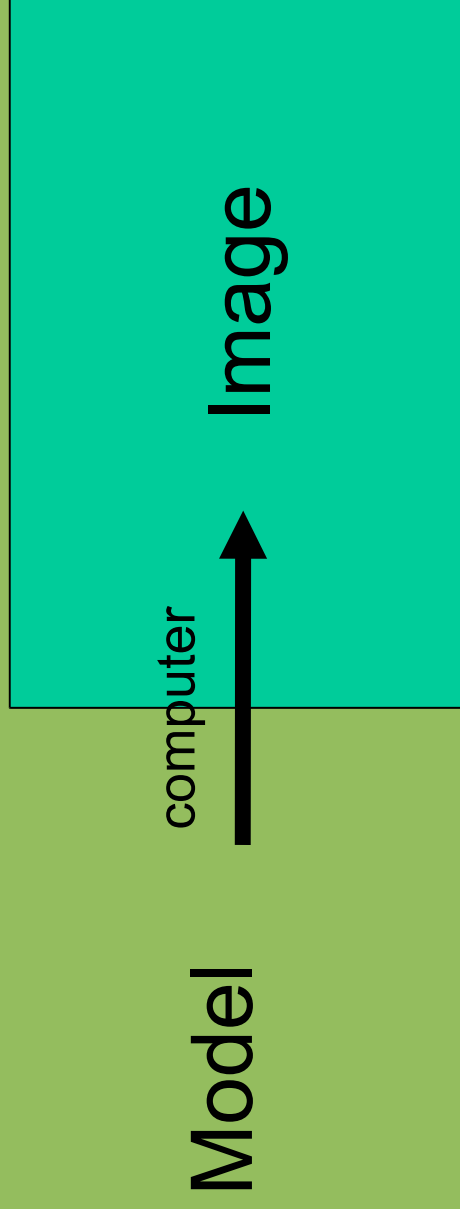
- Basic representations (point, vector)
- Basic operations on points and vectors (dot product, cross products, etc.)
- Transformation – manipulative operators on the basic representations (translate, rotate, deformations) – 4x4 matrices to “encode” all these.

Why do we need this?

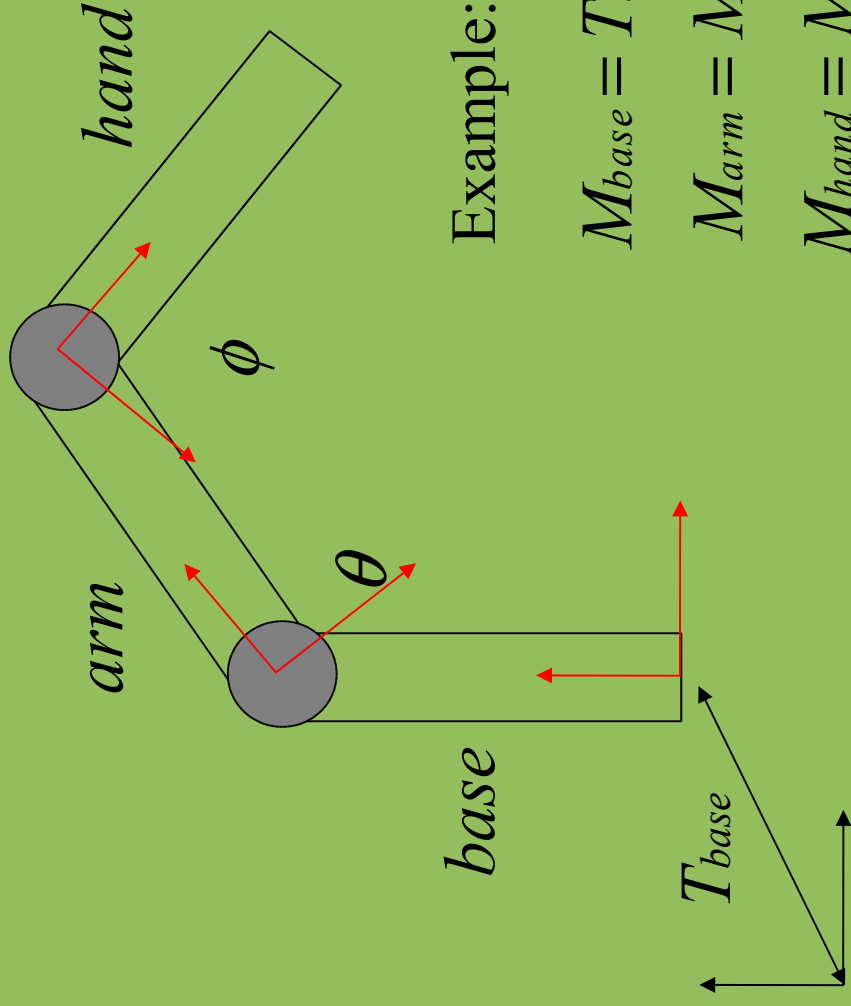
- In order to generate a picture from a model, we need to be able to not only specify a model (representation) but also manipulate the model in order to create more interesting images.

Overview

- The next set of slides will deal with the other half of the process (at least in a simplistic fashion)
- From a model, how do we generate an image



Accumulation of Transformations



Example: Robot arm

$$M_{base} = T_{base}$$

$$M_{arm} = M_{base} R(\theta) = T_{base} R(\theta)$$

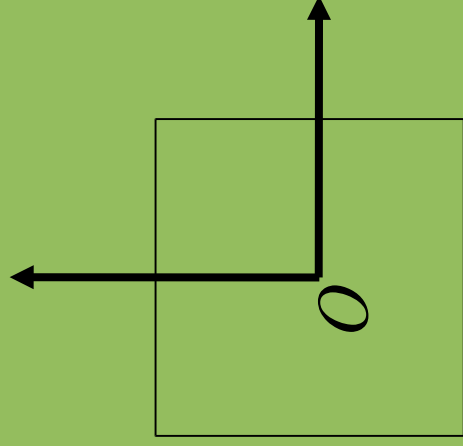
$$M_{hand} = M_{arm} R(\phi) = T_{base} R(\theta) R(\phi)$$

Coordinate Systems

- Object coordinates
- World coordinates
- Camera coordinates
- Normalized device coordinates
- Window coordinates

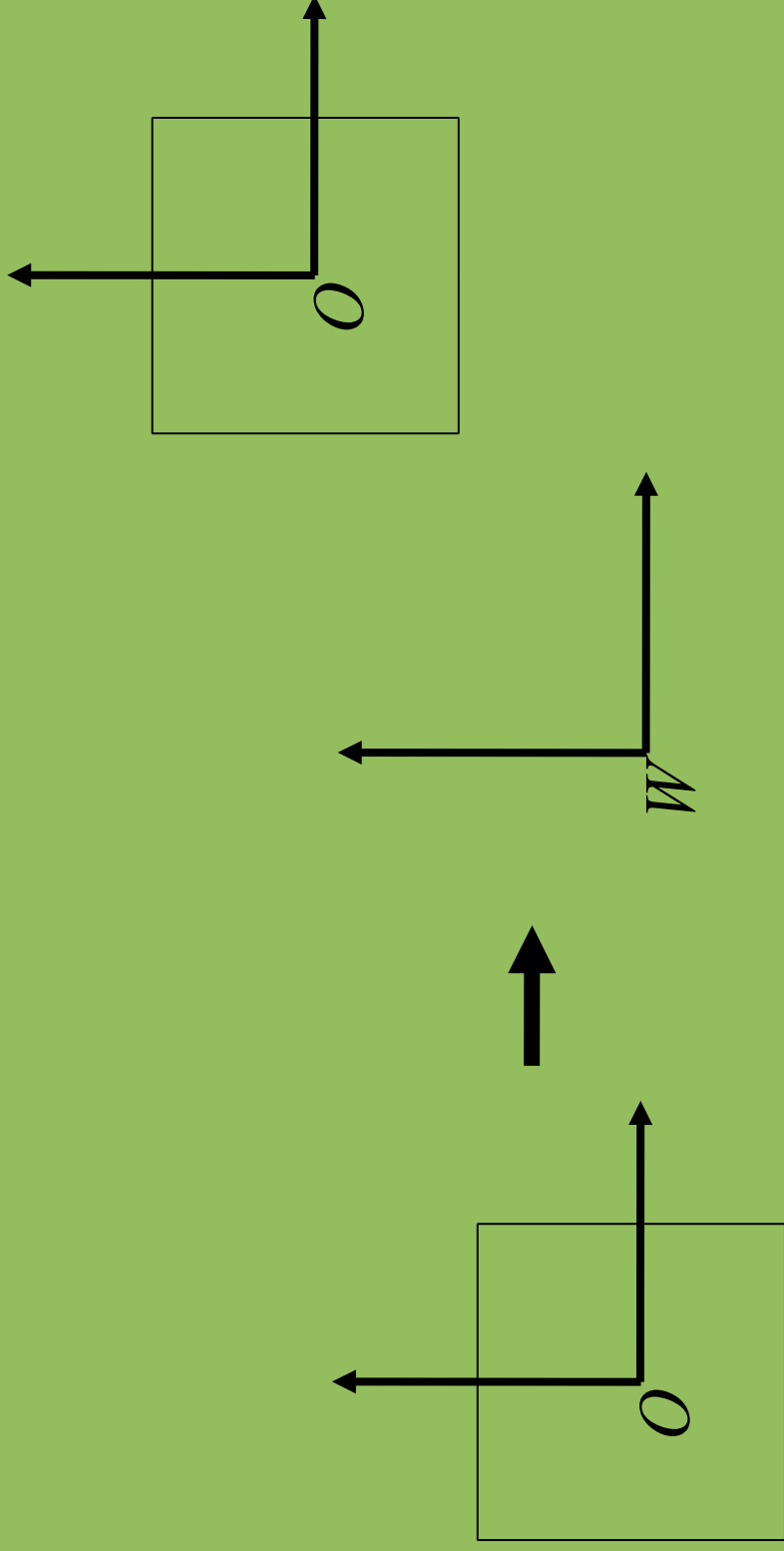
Object Coordinates

Convenient place to model the object



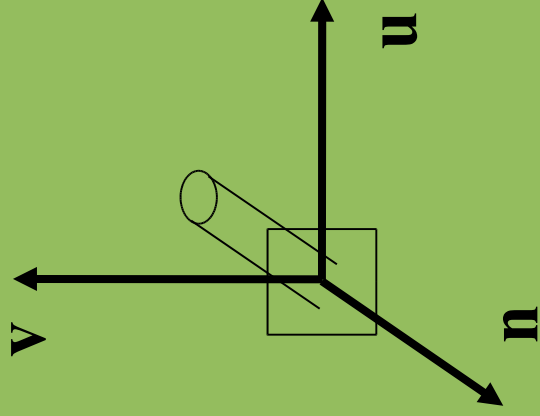
World Coordinates

Common coordinates for the scene



Camera Coordinates

Coordinate system with the camera in a convenient pose



$$M = \begin{bmatrix} u_x & u_y & u_z & -\mathbf{r} \cdot \mathbf{u} \\ v_x & v_y & v_z & -\mathbf{r} \cdot \mathbf{v} \\ n_x & n_y & n_z & -\mathbf{r} \cdot \mathbf{n} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Normalized Device Coordinates

Device independent coordinates

Visible coordinate usually range from:

$$-1 \leq x \leq 1$$

$$-1 \leq y \leq 1$$

$$-1 \leq z \leq 1$$

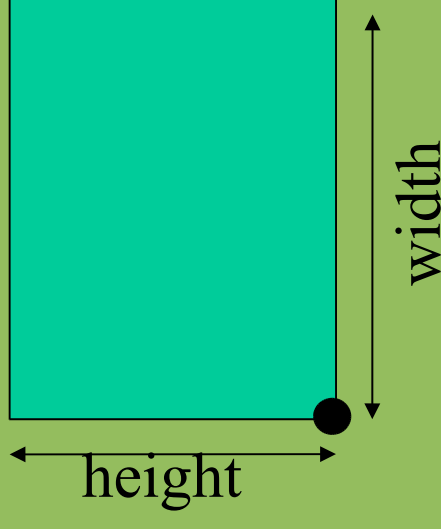
Window Coordinates

Adjusting the NDC to fit the window

(x_0, y_0) is the lower left of the window

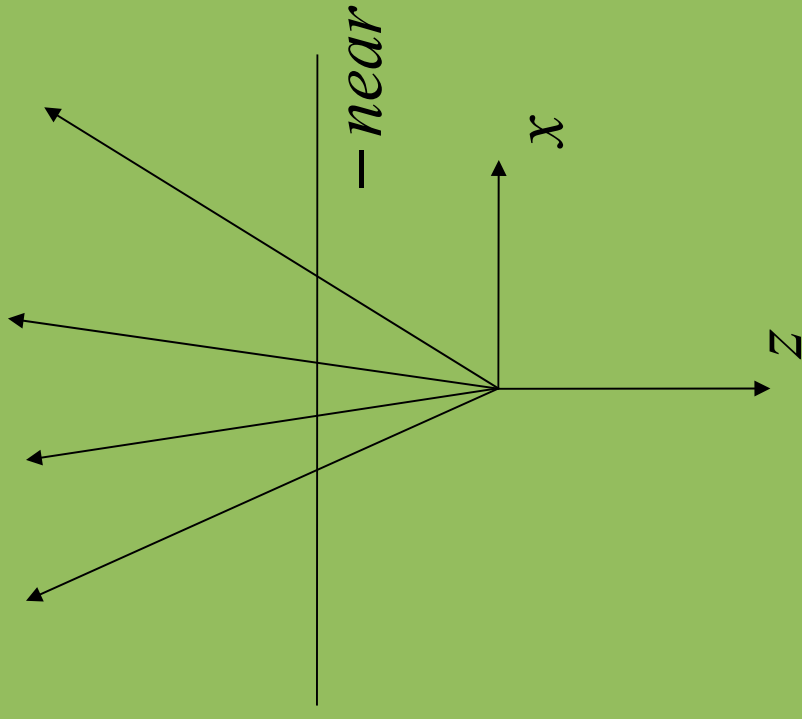
$$x_w = (x_{nd} + 1) \left(\frac{width}{2} \right) + x_0$$

$$y_w = (y_{nd} + 1) \left(\frac{height}{2} \right) + y_0$$

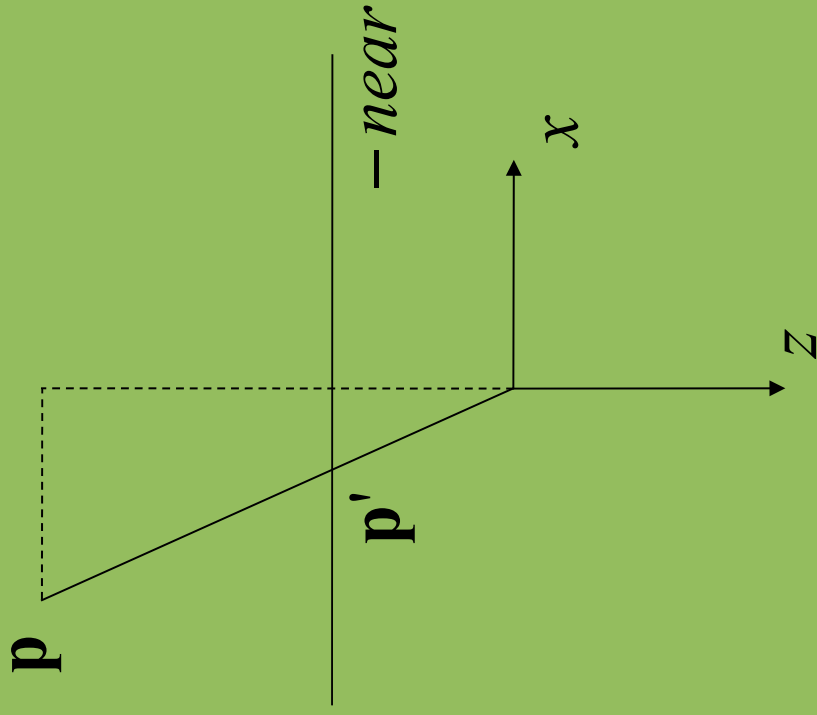


Perspective Projection

Taking the camera coordinates to NDC



Perspective Projection



$$\frac{x'}{-near} = \frac{x}{z}$$

$$x' = -near \frac{x}{z}$$

Perspective Projection

Map (left,right) to $(-1,1)$ when $z = -near$

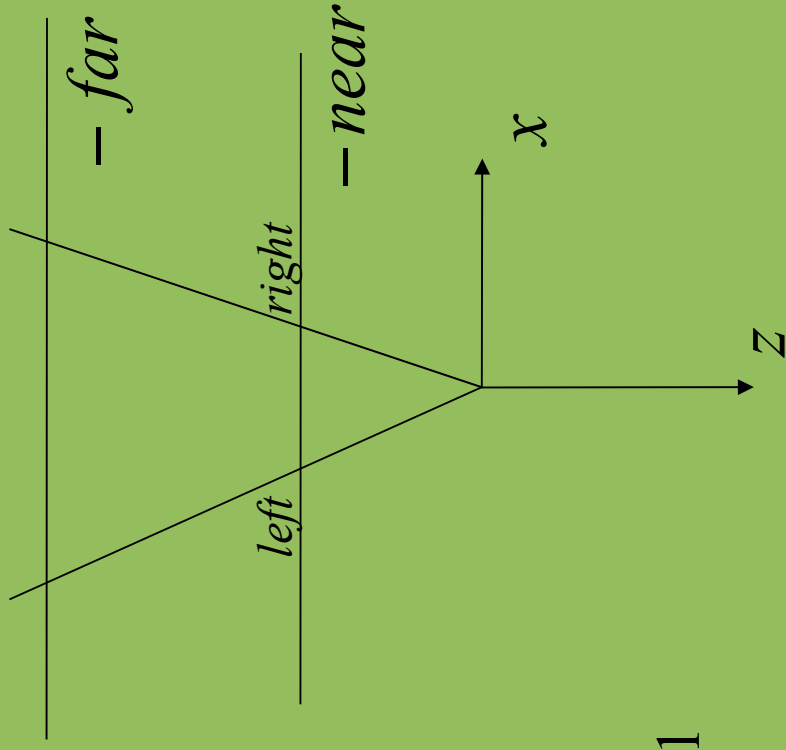
$$x' = -near \frac{x}{z}$$

$$-near \frac{x}{z} - left$$

$$\frac{2}{right - left} \left(-near \frac{x}{z} - left \right)$$

$$\frac{2}{right - left} \left(-near \frac{x}{z} - left \right) - 1$$

$$\frac{-2near}{right - left} \frac{x}{z} - \frac{right + left}{right - left}$$



Pseudodepth

Map (-near, -far) to (-1, 1)

$$z' = \frac{far + near}{far - near} + \frac{2 \cdot far \cdot near}{far - near} \cdot z$$

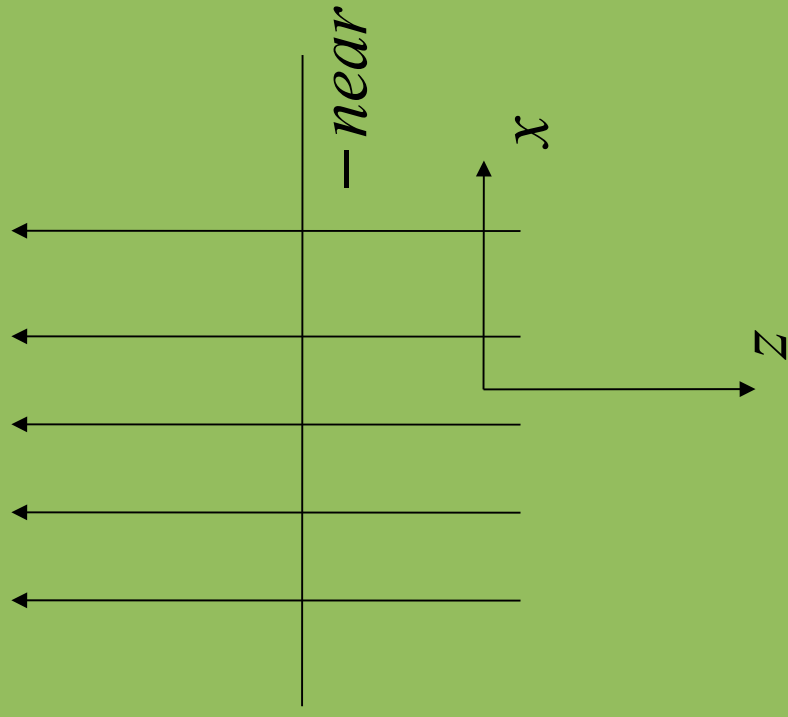
Lines are preserved through the transformation

See Newman and Sproull '81 for the full derivation

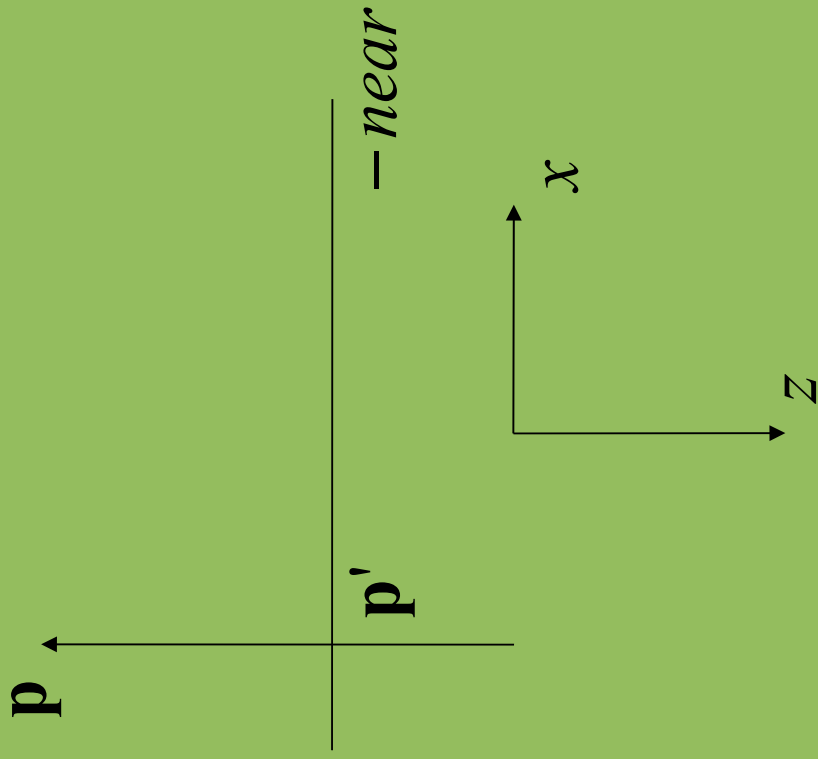
Perspective Projection

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

Orthographic Projection



Orthographic Projection



$$x' = x$$

Orthographic Projection

Map (left,right) to $(-1,1)$

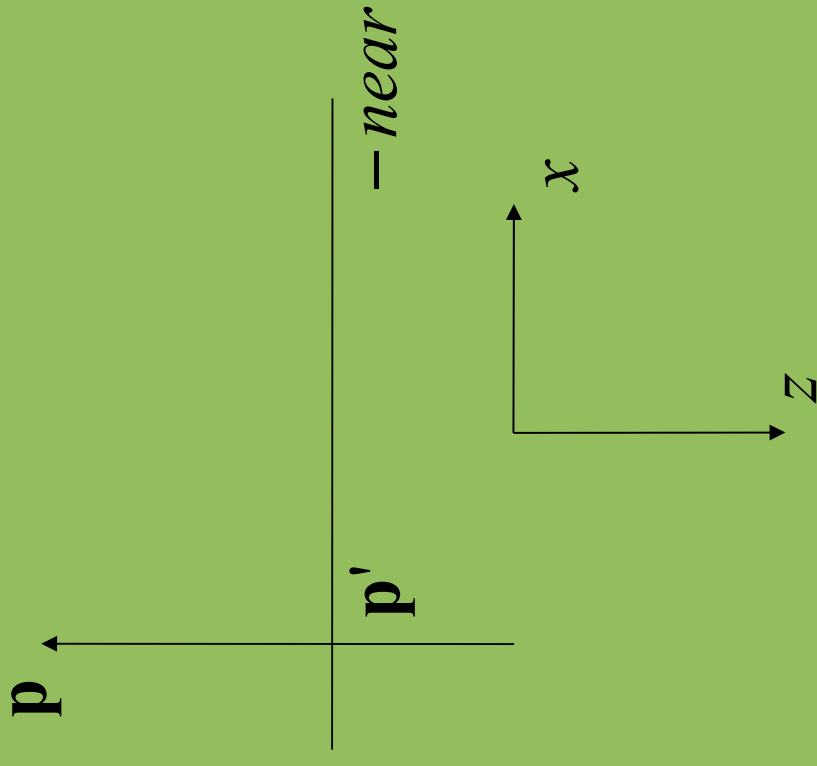
$$x' = x$$

$$x - left$$

$$\frac{2}{right - left}(x - left)$$

$$\frac{2}{right - left}(x - left) - 1$$

$$\frac{2}{right - left}x - \frac{right + left}{right - left}$$



Orthographic Projection

Map (near, far) to (-1, 1)

z

$z - near$

$$\frac{2}{far - near} (z - near)$$

$$\frac{2}{far - near} (z - near) - 1$$

$$\frac{2}{far - near} z - \frac{far + near}{far - near}$$

Orthographic Projection

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

Orthographic Projections Preserve Straight Lines

$$\begin{bmatrix} a_x + b_{xt} \\ a_y + b_{yt} \\ a_z + b_{zt} \\ 1 \end{bmatrix}$$

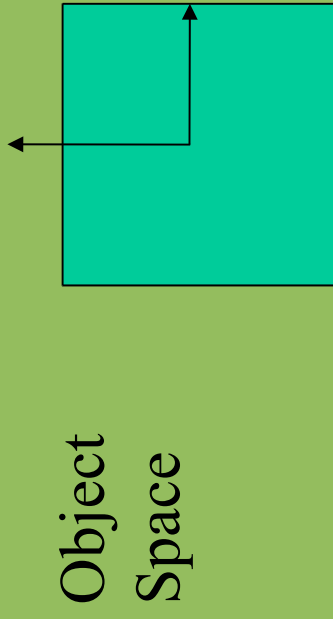
maps to

$$\begin{bmatrix} A(a_x + b_{xt}) + B \\ C(a_y + b_{yt}) + D \\ E(a_z + b_{zt}) + F \\ 1 \end{bmatrix}$$

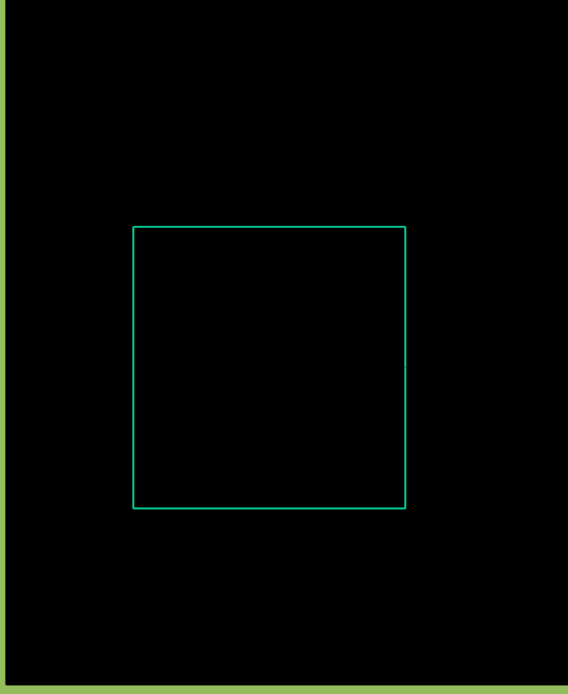
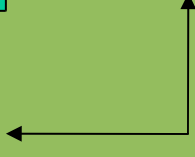
Putting it all together!!

- Take your representation (points) and transform it from Object Space to World Space
- Take your World Space point and transform it to Camera Space
- Perform the remapping and projection onto the image plane in Normalized Device Coordinates
- Perform this set of transformations on each point of the polygonal object
- “Connect the dots” through line rasterization

Intuitively



World Space



Rasterization



Camera Space

