

## Projections

- Planar Projections
- Parallel Orthographic Projection
- The Pinhole Camera Model
- Perspective Projection
- The Canonical View Volume
- The Projection Matrix
- Homogeneous versus Normalized Device Coordinates
- Clipping
- Mapping to Viewport

Copyright © Manuel M. Oliveira

## Projections

- Where are we in the pipeline?

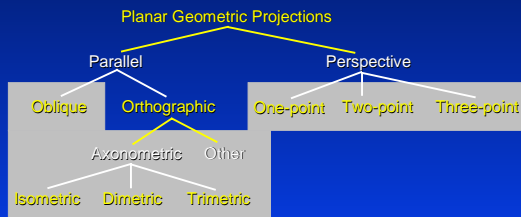


- Map from CCS to Screen CS (SCS)
- Clipping
- Perspective division

Copyright © Manuel M. Oliveira

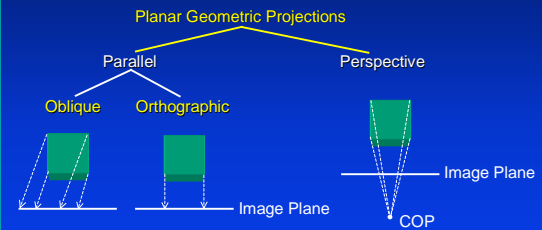
## Planar Projections

- Map points from the CCS into the image plane of the virtual camera



Copyright © Manuel M. Oliveira

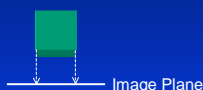
## Planar Projections



Copyright © Manuel M. Oliveira

## Parallel Orthographic Projection

- Preserves original X and Y coordinates
- Screen vs. Device Coordinate System



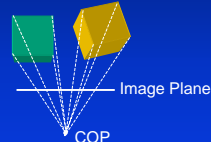
$$\begin{bmatrix} x_s \\ y_s \\ z_s \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

$$P_{scs} = M_{orth} P_{ccs}$$

Copyright © Manuel M. Oliveira

## Perspective Projection

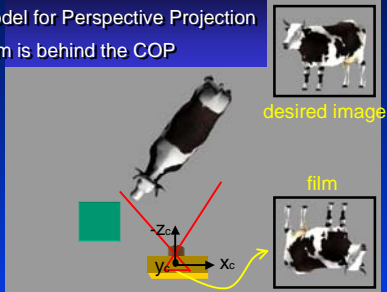
- Line segments are shortened with distance
- Only preserve parallel lines that are parallel to the image plane



Copyright © Manuel M. Oliveira

## Pinhole Camera Model

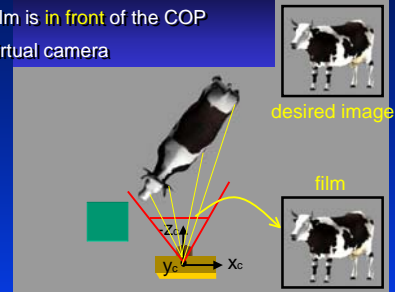
- Model for Perspective Projection
- Film is behind the COP



Copyright © Manuel M. Oliveira

## Pinhole Camera Model in CG

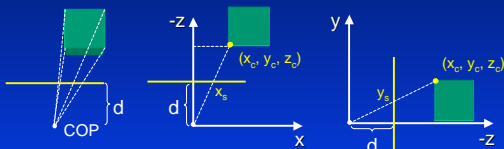
- Film is **in front** of the COP
- Virtual camera



Copyright © Manuel M. Oliveira

## Perspective Projection

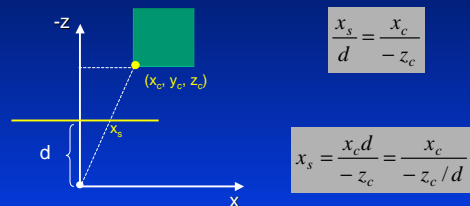
- Each Coordinate (x, y and z) can be projected separately
- Use of **similar triangles** to compute the projections



Copyright © Manuel M. Oliveira

## Perspective Projection (X)

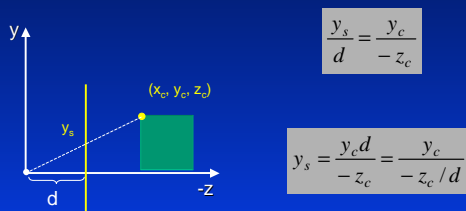
- Similar Triangles



Copyright © Manuel M. Oliveira

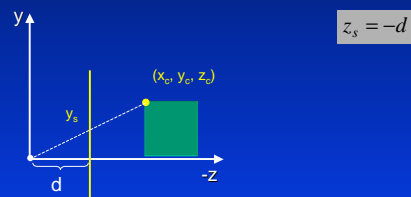
## Perspective Projection (Y)

- Similar Triangles



Copyright © Manuel M. Oliveira

## Perspective Projection (Z)



Copyright © Manuel M. Oliveira

## Perspective Projection Matrix

$$\begin{bmatrix} x'_s \\ y'_s \\ z'_s \\ w'_s \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1/d & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

$$P_{scs} = M_{perp} P_{ccs}$$

- No support to orthographic projection...

Copyright © Manuel M. Oliveira

$$x_s = \frac{x'_s}{w'_s} = \frac{x_c}{-z_c/d}$$

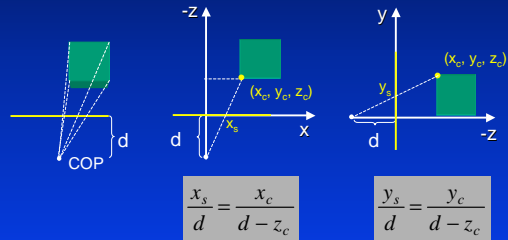
$$y_s = \frac{y'_s}{w'_s} = \frac{y_c}{-z_c/d}$$

$$z_s = \frac{z'_s}{w'_s} = -d$$

$$w'_s = -z_c/d$$

## Alternative Representation

- COP behind the origin along the Z-axis



Copyright © Manuel M. Oliveira

## Alternative Representation

$$\begin{bmatrix} x'_s \\ y'_s \\ z'_s \\ w'_s \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & -1/d & 1 \end{bmatrix} \begin{bmatrix} x_c \\ y_c \\ z_c \\ 1 \end{bmatrix}$$

$$P_{scs} = M_{perp} P_{ccs}$$

- As  $d$  goes to infinite, we have parallel orthographic projection!

Copyright © Manuel M. Oliveira

$$x_s = \frac{x'_s}{w'_s} = \frac{x_c}{1 - \frac{z_c}{d}}$$

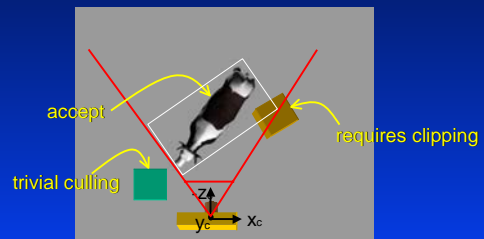
$$y_s = \frac{y'_s}{w'_s} = \frac{y_c}{1 - \frac{z_c}{d}}$$

$$z_s = 0$$

$$w'_s = 1 - \frac{z_c}{d}$$

## The View Frustum

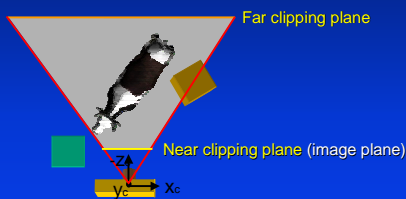
- Trivial accept/reject



Copyright © Manuel M. Oliveira

## The View Volume

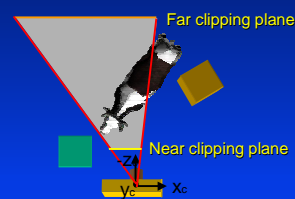
- Near and Far clipping planes



Copyright © Manuel M. Oliveira

## The View Volume in General

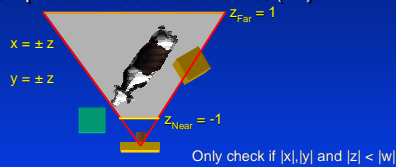
- It can be asymmetric!
- How can we clip efficiently?



Copyright © Manuel M. Oliveira

## The Canonical View Volume

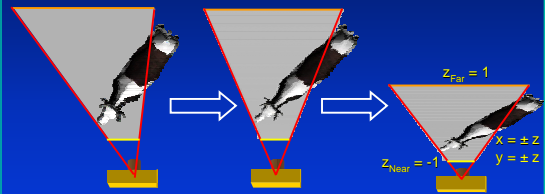
- Symmetric, 90° field of view,  $z_{\text{near}} = -1$ ;  $z_{\text{far}} = 1$
- Also called *Normalized Perspective View Volume*
- Simplify clipper arithmetic
- Clip to a simple built-in set of boundaries (hw)



Copyright © Manuel M. Oliveira

## Getting the Canonical Volume

- What does it take to accomplish this?

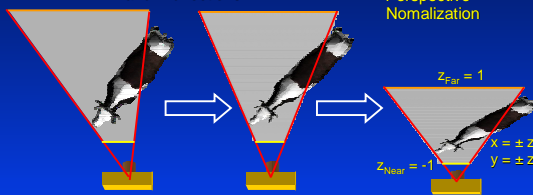


Copyright © Manuel M. Oliveira

## Getting the Canonical Volume

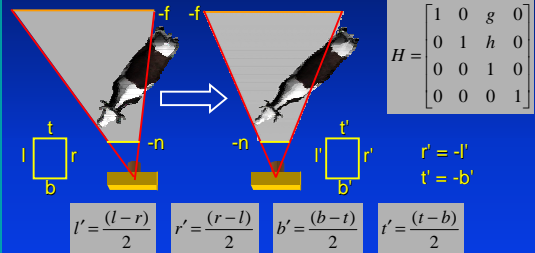
Shear the volume in X and Y directions

- Scale in X and Y and
- Perspective Normalization



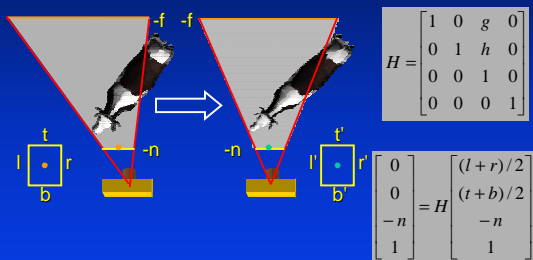
Copyright © Manuel M. Oliveira

## Shear in X and Y



Copyright © Manuel M. Oliveira

## Shear in X and Y



Copyright © Manuel M. Oliveira

## The Shear Parameters

$$\begin{bmatrix} 0 \\ 0 \\ -n \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & g & 0 \\ 0 & 1 & h & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} (l+r)/2 \\ (t+b)/2 \\ -n \\ 1 \end{bmatrix}$$

$$g = \frac{(l+r)}{2n} \quad h = \frac{(t+b)}{2n}$$

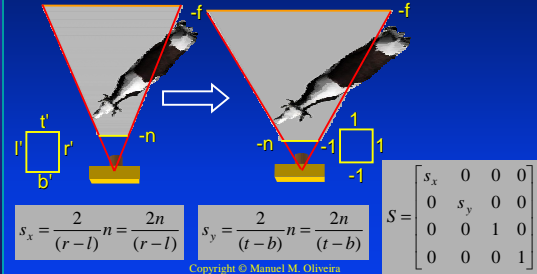
Copyright © Manuel M. Oliveira

## The Shear Matrix

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

Copyright © Manuel M. Oliveira

## Scale in X and Y



$$s_x = \frac{2}{(r-l)} n = \frac{2n}{(r-l)} \quad s_y = \frac{2}{(t-b)} n = \frac{2n}{(t-b)}$$

$$S = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

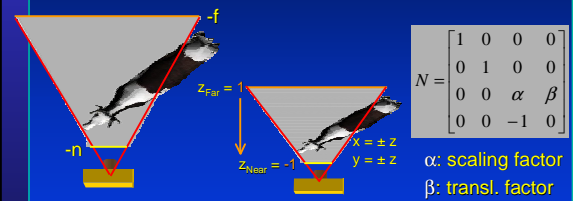
Copyright © Manuel M. Oliveira

## The Scale Matrix

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

Copyright © Manuel M. Oliveira

## Perspective Normalization



$$N = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \alpha & \beta \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

$\alpha$ : scaling factor  
 $\beta$ : transl. factor

Copyright © Manuel M. Oliveira

## The Parameters of N

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \\ -1 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & \alpha & \beta \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ -n & -f \\ 1 & 1 \end{bmatrix} \begin{bmatrix} -\alpha n + \beta \\ -\alpha f + \beta \end{bmatrix} = \begin{bmatrix} -1 \\ 1 \end{bmatrix}$$

$$\alpha = 1 + \frac{\beta}{n} \quad (I)$$

$$\beta = f(1 + \alpha) \quad (II)$$

(II) into (I)

$$\alpha = -\frac{(f+n)}{(f-n)} \quad \text{and} \quad \beta = -\frac{2fn}{(f-n)}$$

Copyright © Manuel M. Oliveira

## Perspective Normalization Matrix

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

Copyright © Manuel M. Oliveira

## Projection Matrix

- Putting all together

$$P = NSH$$

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

Copyright © Manuel M. Oliveira

## Another Way to Derive P

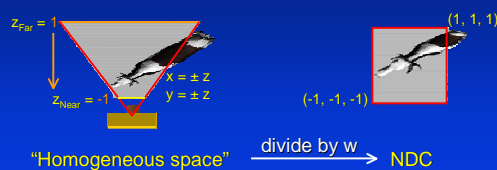
- We know the shape of the matrix: just apply some known mappings and solve for the unknowns
- Not as elegant!

$$\begin{bmatrix} -1 & 1 & -1 & -1 \\ 1 & 1 & -1 & 1 \\ -1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} A & 0 & B & 0 \\ 0 & C & D & 0 \\ 0 & 0 & E & F \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} l & r & l & (f/n)l \\ t & t & b & (f/n)t \\ -n & -n & -n & -f \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Copyright © Manuel M. Oliveira

## Homogeneous versus ND

- Normalized Device Coordinates
- Easy to clip in both canonical viewing volumes



Copyright © Manuel M. Oliveira

## PM

- The product  $PM$ , where  $M$  is the **Model/View** Matrix and  $P$  is the **Projection** Matrix, maps coordinates defined with respect to the World Coordinate System into coordinates defined with respect to the Normalized Perspective View Volume.

Copyright © Manuel M. Oliveira

## Clipping. What for?

- Avoids conditional tests during rasterization



Copyright © Manuel M. Oliveira

## But Why Clip in Z?

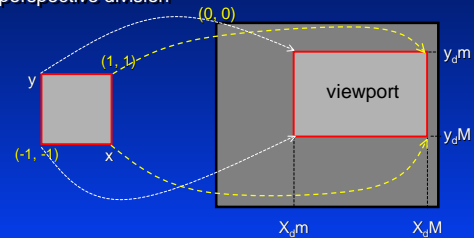
- Avoids division by zero
- This implies clipping before perspective division!!!

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

Copyright © Manuel M. Oliveira

## Mapping to the Viewport

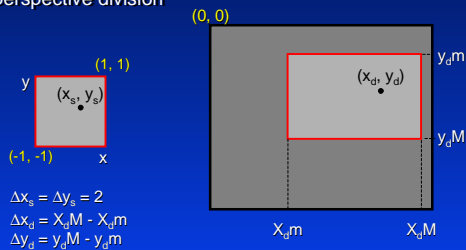
- Essentially a scaling and a translation done after the perspective division



Copyright © Manuel M. Oliveira

## Mapping to the Viewport

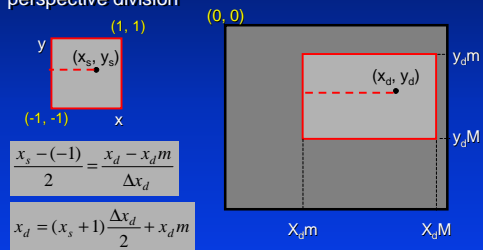
- Essentially a scaling and a translation done after the perspective division



Copyright © Manuel M. Oliveira

## Mapping to the Viewport

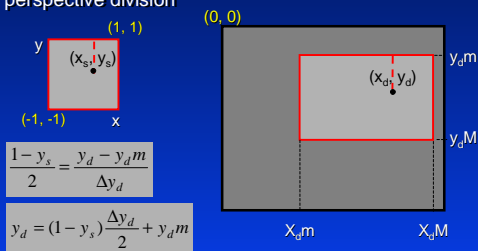
- Essentially a scaling and a translation done after the perspective division



Copyright © Manuel M. Oliveira

## Mapping to the Viewport

- Essentially a scaling and a translation done after the perspective division



Copyright © Manuel M. Oliveira

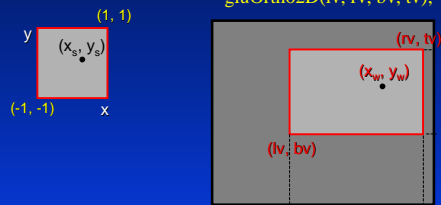
## Viewport Transformation Matrix

$$V = \begin{bmatrix} \frac{\Delta x_d}{2} & 0 & 0 & x_{dM} + \frac{\Delta x_d}{2} \\ 0 & -\frac{\Delta y_d}{2} & 0 & y_{dM} + \frac{\Delta y_d}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Copyright © Manuel M. Oliveira

## Viewport in GLUT

- Local coordinates system wrt the window (viewport)



Copyright © Manuel M. Oliveira

## Viewport Transformation (GLUT)

$$V_{GLUT} = \begin{bmatrix} \frac{rv - lv}{2} & 0 & 0 & \frac{rv + lv}{2} \\ 0 & \frac{tv - bv}{2} & 0 & \frac{tv + bv}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Copyright © Manuel M. Oliveira