University of British Columbia
CPSC 314 Computer Graphics
Jan-Apr 2007

Tamara Munzner

Viewing/Projections III

Week 4, Wed Jan 31

http://www.ugrad.cs.ubc.ca/~cs314/Vjan2007

News

- extra TA coverage in lab to answer questions
  - Wed 2-3:30
  - Thu 12:30-2
- my office hours reminder (in lab also)
  - Wed (today) 11-12
  - Fri 11-12

Reading for Today

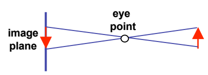
- FCG Chapter 7 Viewing
- FCG Section 6.3.1 Windowing Transforms
- RB rest of Chap Viewing
- RB rest of App Homogeneous Coords

Reading for Next Time

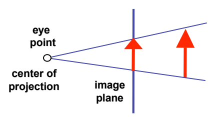
- RB Chap Color
- FCG Sections 3.2-3.3
- FCG Chap 20 Color
- FCG Chap 21 Visual Perception

Review: Graphics Cameras

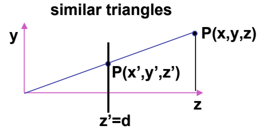
- real pinhole camera: image inverted



- computer graphics camera: convenient equivalent



Review: Basic Perspective Projection



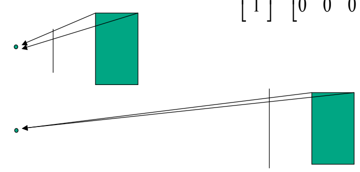
$$\frac{y'}{d} = \frac{y}{z} \rightarrow y' = \frac{y \cdot d}{z} \qquad x' = \frac{x \cdot d}{z} \qquad z' = d$$

$$\begin{bmatrix} \frac{x}{z/d} \\ \frac{y}{z/d} \\ d \end{bmatrix} \xrightarrow{\text{homogeneous coords}} \begin{bmatrix} x \\ y \\ z \\ z/d \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 1/d & 0 \end{bmatrix}$$

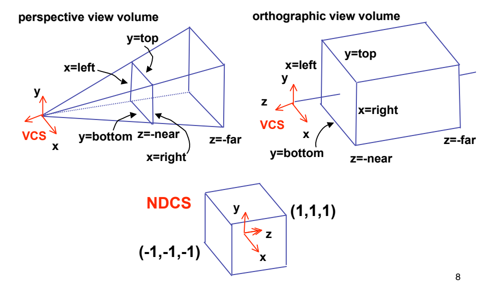
Review: Orthographic Cameras

- center of projection at infinity
- no perspective convergence
- just throw away z values

$$\begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

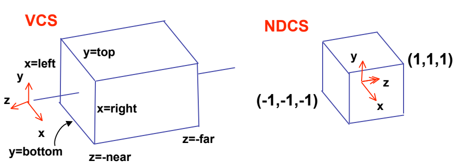


Review: Transforming View Volumes



Orthographic Derivation

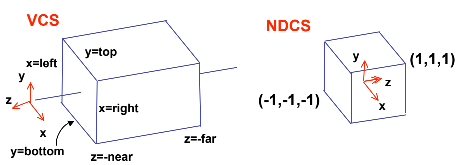
- scale, translate, reflect for new coord sys



Orthographic Derivation

- scale, translate, reflect for new coord sys

$$y' = a \cdot y + b \qquad \begin{aligned} y = top &\rightarrow y' = 1 \\ y = bot &\rightarrow y' = -1 \end{aligned}$$



Orthographic Derivation

- scale, translate, reflect for new coord sys

$$y' = a \cdot y + b \qquad \begin{aligned} y = top &\rightarrow y' = 1 \\ y = bot &\rightarrow y' = -1 \end{aligned} \qquad \begin{aligned} 1 &= a \cdot top + b \\ -1 &= a \cdot bot + b \end{aligned}$$

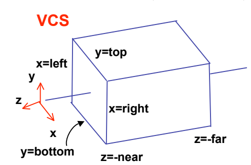
$$b = 1 - a \cdot top, b = -1 - a \cdot bot$$
$$1 - a \cdot top = -1 - a \cdot bot$$
$$1 - (-1) = -a \cdot bot - (-a \cdot top)$$
$$2 = a(-bot + top)$$
$$a = \frac{2}{top - bot}$$

$$1 = \frac{2}{top - bot} top + b$$
$$b = 1 - \frac{2 \cdot top}{top - bot}$$
$$b = \frac{(top - bot) - 2 \cdot top}{top - bot}$$
$$b = \frac{-top - bot}{top - bot}$$

Orthographic Derivation

- scale, translate, reflect for new coord sys

$$y' = a \cdot y + b \qquad \begin{aligned} y = top &\rightarrow y' = 1 \\ y = bot &\rightarrow y' = -1 \end{aligned}$$



$$a = \frac{2}{top - bot}$$

$$b = -\frac{top + bot}{top - bot}$$

same idea for right/left, far/near

Orthographic Derivation

- scale, translate, reflect for new coord sys

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

Orthographic Derivation

- scale, translate, reflect for new coord sys

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

Orthographic Derivation

- scale, translate, reflect for new coord sys

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

Orthographic Derivation

- scale, translate, reflect for new coord sys

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

Orthographic OpenGL

```
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
glOrtho(left,right,bot,top,near,far);
```

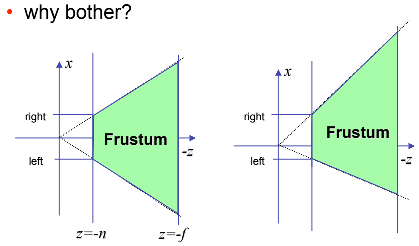

Demo

- Brown applets: viewing techniques
  - parallel/orthographic cameras
  - projection cameras
- http://www.cs.brown.edu/exploratories/freeSoftware/catalogs/viewing_techniques.html

Projections II

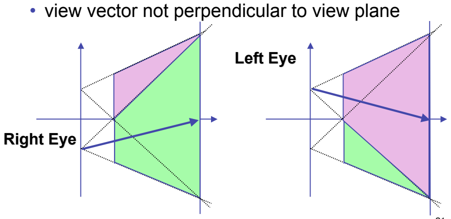
Asymmetric Frusta

- our formulation allows asymmetry
  - why bother?



Asymmetric Frusta

- our formulation allows asymmetry
  - why bother? binocular stereo
    - view vector not perpendicular to view plane

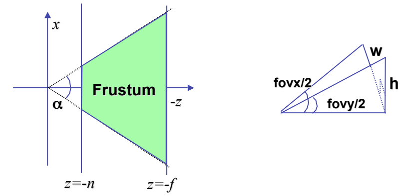


Simpler Formulation

- left, right, bottom, top, near, far
  - nonintuitive
  - often overkill
- look through window center
  - symmetric frustum
- constraints
  - left = -right, bottom = -top

Field-of-View Formulation

- FOV in one direction + aspect ratio (w/h)
  - determines FOV in other direction
  - also set near, far (reasonably intuitive)



Perspective OpenGL

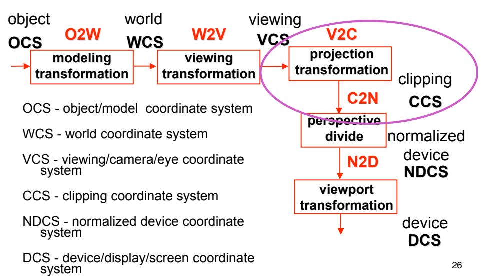
```
glMatrixMode(GL_PROJECTION);
glLoadIdentity();

glFrustum(left,right,bot,top,near,far);
  or
glPerspective(fovy,aspect,near,far);
```


Demo: Frustum vs. FOV

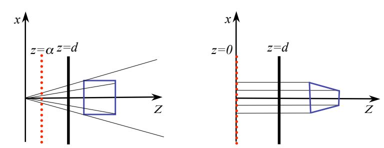
- Nate Robins tutorial (take 2):
  - http://www.xmission.com/~nate/tutors.html

Projective Rendering Pipeline



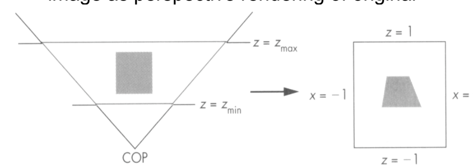
Projection Normalization

- warp perspective view volume to orthogonal view volume
  - render all scenes with orthographic projection!
  - aka perspective warp

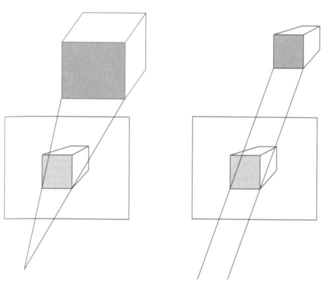


Perspective Normalization

- perspective viewing frustum transformed to cube
- orthographic rendering of cube produces same image as perspective rendering of original



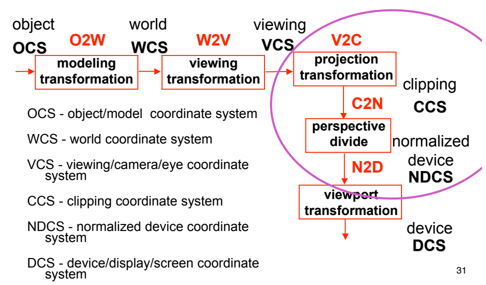
Predistortion



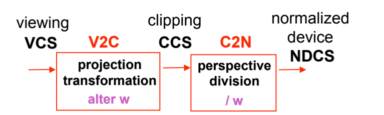
Demos

- Tuebingen applets from Frank Hanisch
  - http://www.gris.uni-tuebingen.de/projects/grdev/doc/html/etc/AppletIndex.html#Transformationen

Projective Rendering Pipeline



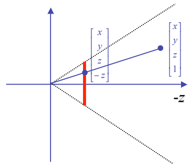
Separate Warp From Homogenization



- warp requires only standard matrix multiply
  - distort such that orthographic projection of distorted objects is desired persp projection
    - w is changed
  - clip after warp, before divide
  - division by w: homogenization

Perspective Divide Example

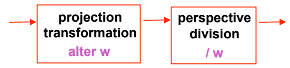
- specific example
  - assume image plane at $z = -1$
  - a point $[x,y,z,1]^T$ projects to $[-x/z,-y/z,-z/z,1]^T \equiv [x,y,z,-z]^T$



Perspective Divide Example

$$T\left(\begin{bmatrix}x\\y\\z\\1\end{bmatrix}\right)=\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&-1&0\end{bmatrix}\cdot\begin{bmatrix}x\\y\\z\\1\end{bmatrix}=\begin{bmatrix}x\\y\\z\\-z\end{bmatrix}\equiv\begin{bmatrix}-x/z\\-y/z\\-1\\1\end{bmatrix}$$

- after homogenizing, once again w=1



Perspective Normalization

- matrix formulation

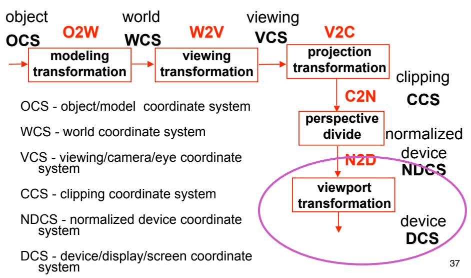
$$\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&\frac{d}{d-\alpha}&\frac{-\alpha\cdot d}{d-\alpha}\\0&0&\frac{1}{d}&0\end{bmatrix}\begin{bmatrix}x\\y\\z\\1\end{bmatrix}=\begin{bmatrix}x\\y\\\left(\frac{(z-\alpha)\cdot d}{d-\alpha}\right)\\\frac{z}{d}\end{bmatrix}\qquad\begin{bmatrix}x_p\\y_p\\z_p\end{bmatrix}=\begin{bmatrix}\frac{x}{z/d}\\\frac{y}{z/d}\\\frac{d^2}{d-\alpha}\left(1-\frac{\alpha}{z}\right)\end{bmatrix}$$

- warp and homogenization both preserve relative depth (z coordinate)

Demo

- Brown applets: viewing techniques
  - parallel/orthographic cameras
  - projection cameras
- http://www.cs.brown.edu/exploratories/freeSoftware/catalogs/viewing_techniques.html

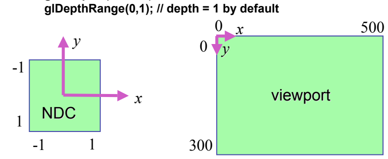
Projective Rendering Pipeline



NDC to Device Transformation

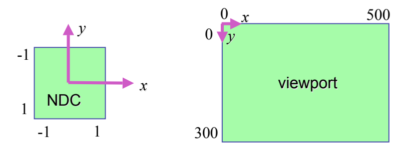
- map from NDC to pixel coordinates on display
  - NDC range is x = -1...1, y = -1...1, z = -1...1
  - typical display range: x = 0...500, y = 0...300
    - maximum is size of actual screen
    - z range max and default is (0, 1), use later for visibility

```
glViewport(0,0,w,h);
glDepthRange(0,1); // depth = 1 by default
```



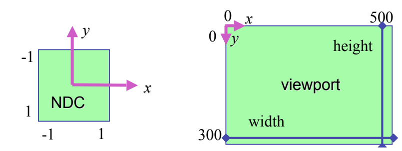
Origin Location

- yet more (possibly confusing) conventions
  - OpenGL origin: lower left
  - most window systems origin: upper left
- then must reflect in y
- when interpreting mouse position, have to flip your y coordinates



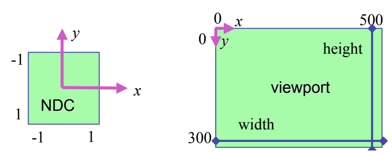
N2D Transformation

- general formulation
  - reflect in y for upper vs. lower left origin
  - scale by width, height, depth
  - translate by width/2, height/2, depth/2
    - FCG includes additional translation for pixel centers at (.5, .5) instead of (0,0)



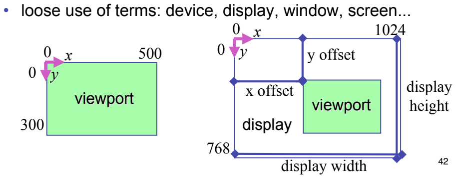
N2D Transformation

$$\begin{bmatrix}x_D\\y_D\\z_D\\1\end{bmatrix}=\begin{bmatrix}1&0&0&\frac{width}{2}-\frac{1}{2}\\0&1&0&\frac{height}{2}-\frac{1}{2}\\0&0&1&\frac{depth}{2}\\0&0&0&1\end{bmatrix}\begin{bmatrix}\frac{width}{2}&0&0&0\\0&\frac{height}{2}&0&0\\0&0&\frac{depth}{2}&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&-1&0&0\\0&0&1&0\\0&0&0&1\end{bmatrix}\begin{bmatrix}x_N\\y_N\\z_N\\1\end{bmatrix}=\begin{bmatrix}\frac{width(x_N+1)-1}{2}\\\frac{height(-y_N+1)-1}{2}\\\frac{depth(z_N+1)}{2}\\1\end{bmatrix}$$



Device vs. Screen Coordinates

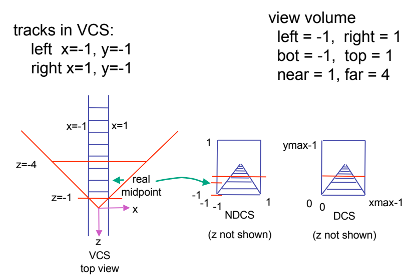
- viewport/window location wrt actual display not available within OpenGL
  - usually don't care
    - use relative information when handling mouse events, not absolute coordinates
  - could get actual display height/width, window offsets from OS
- loose use of terms: device, display, window, screen...



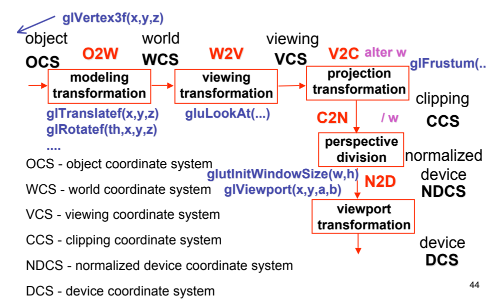
Perspective Example

tracks in VCS:
left x=-1, y=-1
right x=1, y=-1

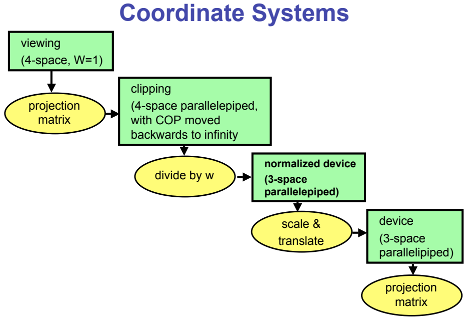
view volume
left = -1, right = 1
bot = -1, top = 1
near = 1, far = 4



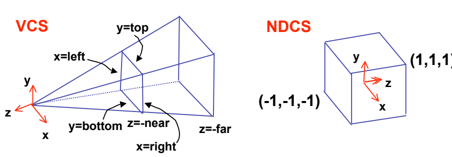
Projective Rendering Pipeline



Coordinate Systems



Perspective To NDCS Derivation



Perspective Derivation

simple example earlier:

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

complete: shear, scale, projection-normalization

$$\begin{bmatrix}x'\\y'\\z'\\w'\end{bmatrix}=\begin{bmatrix}E&0&A&0\\0&F&B&0\\0&0&C&D\\0&0&-1&0\end{bmatrix}\begin{bmatrix}x\\y\\z\\1\end{bmatrix}$$

Perspective Derivation

earlier:

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

complete: shear, scale, projection-normalization

$$\begin{bmatrix}x'\\y'\\z'\\w'\end{bmatrix}=\begin{bmatrix}E&0&A&0\\0&F&B&0\\0&0&C&D\\0&0&-1&0\end{bmatrix}\begin{bmatrix}x\\y\\z\\1\end{bmatrix}$$

Perspective Derivation

earlier:

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

complete: shear, scale, projection-normalization

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \begin{bmatrix} E & 0 & A & 0 \\ 0 & F & B & 0 \\ 0 & 0 & C & D \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Perspective Derivation

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \begin{bmatrix} E & 0 & A & 0 \\ 0 & F & B & 0 \\ 0 & 0 & C & D \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$x' = Ex + Az$

$y' = Fy + Bz$

$z' = Cz + D$

$w' = -z$

$x = left \rightarrow x'/w' = 1$

$x = right \rightarrow x'/w' = -1$

$y = top \rightarrow y'/w' = 1$

$y = bottom \rightarrow y'/w' = -1$

$z = -near \rightarrow z'/w' = 1$

$z = -far \rightarrow z'/w' = -1$

$$y' = Fy + Bz, \quad \frac{y'}{w'} = \frac{Fy + Bz}{w'}, \quad 1 = \frac{Fy + Bz}{w'}, \quad 1 = \frac{Fy + Bz}{-z},$$

$$1 = F\frac{y}{-z} + B\frac{z}{-z}, \quad 1 = F\frac{y}{-z} - B, \quad 1 = F\frac{top}{-(-near)} - B,$$

$$1 = F\frac{top}{near} - B$$

Perspective Derivation

- similarly for other 5 planes
- 6 planes, 6 unknowns

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

Perspective Example

view volume

- left = -1, right = 1
- bot = -1, top = 1
- near = 1, far = 4

$$\begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -5/3 & -8/3 \\ 0 & 0 & -1 & 0 \end{bmatrix}$$

Perspective Example

$$\begin{bmatrix} 1 \\ -1 \\ -5z_{VCS}/3 - 8/3 \\ -z_{VCS} \end{bmatrix} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & -5/3 & -8/3 \\ & & -1 & \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ z_{VCS} \\ 1 \end{bmatrix}$$

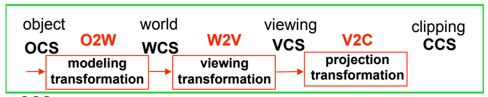
/ w

$$x_{NDCS} = -1/z_{VCS}$$

$$y_{NDCS} = 1/z_{VCS}$$

$$z_{NDCS} = \frac{5}{3} + \frac{8}{3z_{VCS}}$$

OpenGL Example




```
CCS    glMatrixMode( GL_PROJECTION );
       glLoadIdentity();
       gluPerspective( 45, 1.0, 0.1, 200.0 );
VCS    glMatrixMode( GL_MODELVIEW );
       glLoadIdentity();
       glTranslatef( 0.0, 0.0, -5.0 );
WCS    glPushMatrix()
       glTranslate( 4, 4, 0 );   W2O
OCS1   glutSolidTeapot(1);
       glPopMatrix();
       glTranslate( 2, 2, 0);    W2O
OCS2   glutSolidTeapot(1);
```

- transformations that are applied first are specified last