Perspective Projection (cont.)

Wolfgang Heidrich

Course News

Assignment 1

- Due February 2

Homework 2

- Exercise problems for perspective
- Discussed in labs next week
- Solutions online (as prep for quiz)

Quiz 1

- Next Wednesday (Jan 26)

Course News (cont.)

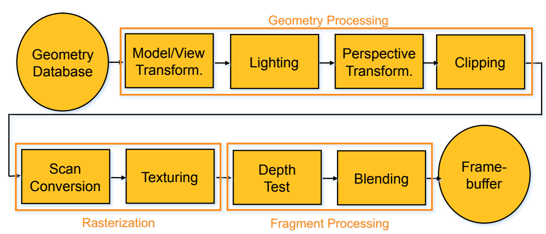
Reading list

- Previously published chapters numbers were from an old book version…

Reading for Quiz (new book version):

- Math prereq: Chapter 2.1-2.4, 4
- Intro: Chapter 1
- Affine transformations: Ch. 6 (was: Ch. 5, old book)
- Perspective: Ch 7 (was: Ch. 6, old book)
  - *Also reading for this week…*

The Rendering Pipeline



Projective Rendering Pipeline

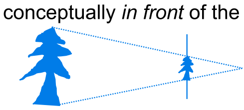
object **OCS** → O2W **modeling transformation** → world **WCS** → W2V **viewing transformation** → viewing **VCS** → V2C **projection transformation** → clipping **CCS** → C2N **perspective divide** → normalized device **NDCS** → N2D **viewport transformation** → device **DCS**

- OCS - object/model coordinate system
- WCS - world coordinate system
- VCS - viewing/camera/eye coordinate system
- CCS - clipping coordinate system
- NDCS - normalized device coordinate system
- DCS - device/display/screen coordinate system

Perspective Transformation

In computer graphics:

- Image plane is conceptually *in front* of the center of projection

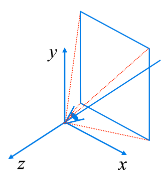


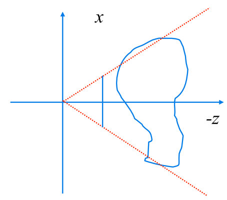
- Perspective transformations belong to a class of operations that are called *projective transformations*
- Linear and affine transformations also belong to this class
- *All* projective transformations can be expressed as *4x4* matrix operations

Perspective Projection

Synopsis:

- Project all geometry through a common *center of projection* (*eye point*) onto an *image plane*

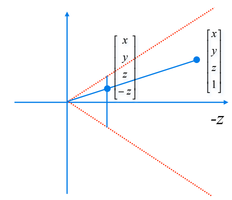




Perspective Projection

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 Projection

Analysis:

- This is a special case of a general family of transformations called *projective transformations*
- These can be expressed as 4x4 homogeneous matrices!
  - *E.g. in the example:*

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

Projective Transformation

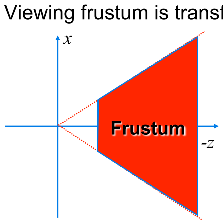
Note:

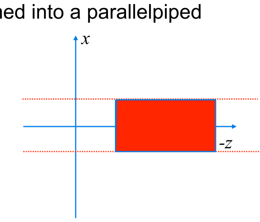
- This version of the perspective transformation removes all information about the original object depth
  - *The matrix is singular, so the information is irrevocably lost*
- Later it will be important to have information about the original object depth for visibility computations
  - *We can achieve this by modifying the third row of the matrix, as we'll see later*

Projective Transformations

Transformation of space:

- Center of projection moves to infinity
- Viewing frustum is transformed into a parallelpiped





Demos

Tuebingen applets from Frank Hanisch

- http://www.gris.uni-tuebingen.de/edu/projects/grdev/doc/html/
- (this is the English version)

Projective Transformations

Convention:

- Viewing frustum is mapped to a specific parallelpiped
  - *Normalized Device Coordinates (NDC)*
- Only objects inside the parallelpiped get rendered
- Which parallelpied is used depends on the rendering system

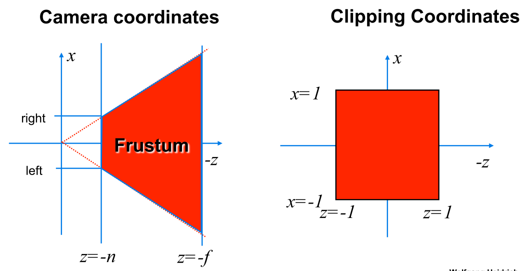
OpenGL:

- Left and right image boundary are mapped to x=-1 and x=+1
- Top and bottom are mapped to y=-1 and y=+1
- Near and far plane are mapped to -1 and 1

Wolfgang Heidrich

Projective Transformations

OpenGL Convention



Wolfgang Heidrich

Projective Transformations

Why near and far plane?

- Near plane:
  - *Avoid singularity (division by zero, or very small numbers)*
- Far plane:
  - *Store depth in fixed-point representation (integer), thus have to have fixed range of values (0…1)*
  - *Avoid/reduce numerical precision artifacts for distant objects*

Wolfgang Heidrich

Projective Transformations

Determining the matrix representation

- Need to observe 5 points in general position, e.g.
  - $[left,0,0,1]^T \rightarrow [1,0,0,1]^T$
  - $[0,top,0,1]^T \rightarrow [0,1,0,1]^T$
  - $[0,0,-f,1]^T \rightarrow [0,0,1,1]^T$
  - $[0,0,-n,1]^T \rightarrow [0,0,0,1]^T$
  - $[left*f/n,top*f/n,-f,1]^T \rightarrow [1,1,1,1]^T$
- Solve resulting equation system to obtain matrix

Wolfgang Heidrich

Perspective Derivation

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

$$x' = Ex + Az \quad y' = Fy + Bz \quad z' = Cz + D \quad w' = -z$$

$$x = left \rightarrow x'/w' = 1$$
$$x = right \rightarrow x'/w' = -1$$
$$y = top \rightarrow y'/w' = 1$$
$$y = bottom \rightarrow y'/w' = -1$$
$$z = -near \rightarrow z'/w' = 1$$
$$z = -far \rightarrow z'/w' = -1$$

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

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

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

Wolfgang Heidrich

Perspective Derivation

similarly for other 5 planes
6 planes, 6 unknowns

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

Wolfgang Heidrich

Perspective Example

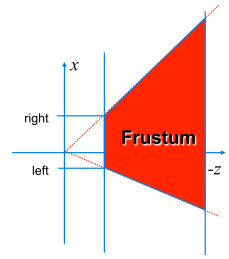
view volume
left = -1, right = 1
bot = -1, top = 1
near = 1, far = 4

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

Wolfgang Heidrich

Projective Transformations

Asymmetric Viewing Frusta



Wolfgang Heidrich

Sheared Perspective

©2007 www.northlight-images.co.uk

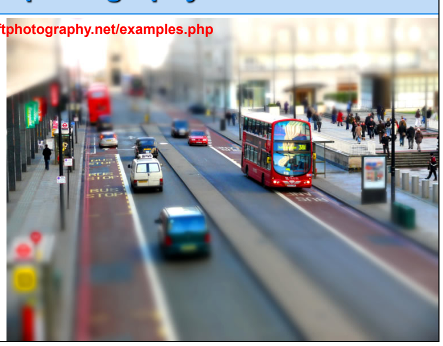
Wolfgang Heidrich

Sheared Perspective

Architectural Photography

Aside: Shift/Tilt photography

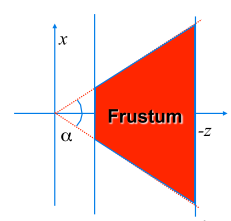
http://www.tiltshiftphotography.net/examples.php



Projective Transformations

Alternative specification of symmetric frusta

- Field-of-view (fov) α
- Fov/2
- Field-of-view in y-direction (fovy) + aspect ratio



Wolfgang Heidrich

Perspective Matrices in OpenGL

Perspective Matrices:

- glFrustum(left, right, bottom, top, near, far)
  - *Specifies perspective transform (near, far are always positive)*

Convenience Function:

- gluPerspective(fovy, aspect, near, far)
  - *Another way to do perspective*

Wolfgang Heidrich

Projective Transformations

Properties:

- All transformations that can be expressed as homogeneous 4x4 matrices (in 3D)
- 16 matrix entries, but multiples of the same matrix all describe the same transformation
  - *15 degrees of freedom*
  - *The mapping of 5 points uniquely determines the transformation*

Wolfgang Heidrich

Projective Transformations

Properties

- Lines are mapped to lines and triangles to triangles
- Parallel lines do NOT remain parallel
  - *E.g. rails vanishing at infinity*
- Affine combinations are NOT preserved
  - *E.g. center of a line does not map to center of projected line (perspective foreshortening)*

Wolfgang Heidrich

Orthographic Camera Projection

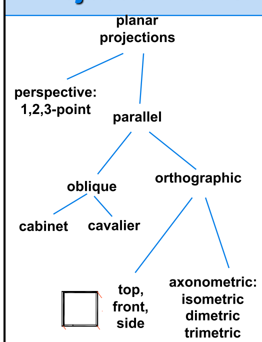
- Camera's back plane parallel to lens
- Infinite focal length
- No perspective convergence

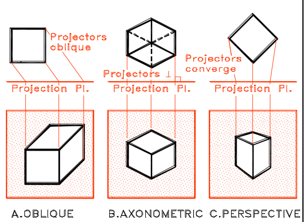
- Just throw away z values

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

Wolfgang Heidrich

Projection Taxonomy

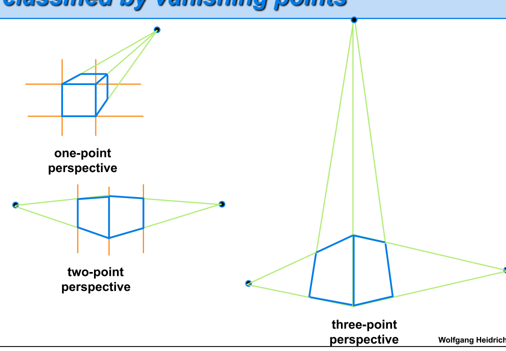




http://ceprofs.tamu.edu/tkramer/ENGR%20111/5.1/20

Wolfgang Heidrich

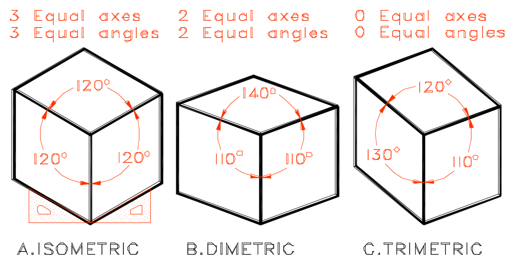
Perspective Projections
classified by vanishing points



Wolfgang Heidrich

Axonometric Projections

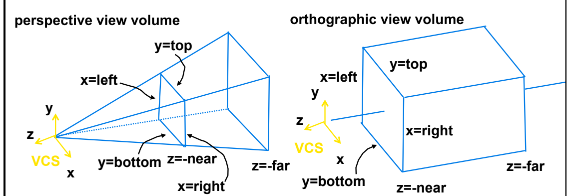
- projectors perpendicular to image plane



http://ceprofs.tamu.edu/tkramer/ENGR%20111/5.1/20

View Volumes

- specifies field-of-view, used for clipping
- restricts domain of ***z*** stored for visibility test

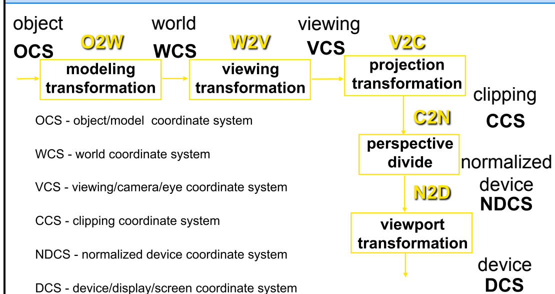


View Volume

Convention

- Viewing frustum mapped to specific parallelepiped
  - *Normalized Device Coordinates (NDC)*
  - *Same as clipping coords*
- Only objects inside the parallelepiped get rendered
- Which parallelepiped?
  - *Depends on rendering system*

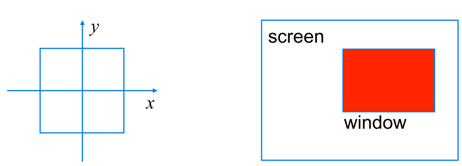
Projective Rendering Pipeline



Window-To-Viewport Transformation

Generate pixel coordinates

- Map x, y from range –1…1 (*normalized device coordinates*) to pixel coordinates on the screen
- Map z from -1…1 to 0…1 (used later for visibility)
- Involves 2D scaling and translation



Coming Up:

Monday:

- Transformations of planes and normals

Wednesday

- Quiz…

Friday

- Lighting/shading