Computer Graphics (Fall 2008)
COMS 4160, Lecture 5: Viewing
http://www.cs.columbia.edu/~cs4160

Motivation
- We have seen transforms (between coord systems)
- But all that is in 3D
- We still need to make a 2D picture
  - Project 3D to 2D. How do we do this?
- This lecture is about viewing transformations

To Do
- Questions/concerns about assignment 1?
- Remember it is due Thu. Ask me or TA if any problems.

What we’ve seen so far
- Transforms (translation, rotation, scale) as 4x4 homogeneous matrices
  - Last row always 0 0 0 1. Last w component always 1

- For viewing (perspective), we will use that last row and w component no longer 1 (must divide by it)

Demo (Projection Tutorial)
- Nate Robbins OpenGL tutor
- Projection.exe
- Download others

Outline
- Orthographic projection (simpler)
- Perspective projection, basic idea
  - Derivation of gluPerspective (handout: glFrustum)
- Brief discussion of nonlinear mapping in z

Not well covered in textbook chapter 7. We follow section 3.5 of real-time rendering most closely. Handouts on this will be given out.
Projections

- To lower dimensional space (here 3D -> 2D)
- Preserve straight lines
- Trivial example: Drop one coordinate (Orthographic)

Orthographic Projection

- Characteristic: Parallel lines remain parallel
- Useful for technical drawings etc.

Example

- Simply project onto xy plane, drop z coordinate

Orthographic Matrix

- First center cuboid by translating
- Then scale into unit cube

Transformation Matrix

\[
\begin{pmatrix}
\frac{2}{r-l} & 0 & 0 & 0 \\
0 & \frac{2}{t-b} & 0 & 0 \\
0 & 0 & \frac{2}{f-n} & 0 \\
0 & 0 & 0 & 1 \\
\end{pmatrix}
\begin{pmatrix}
1 & 0 & -\frac{l+r}{2} & 0 \\
0 & 1 & -\frac{t+b}{2} & 0 \\
0 & 0 & 1 & -\frac{f+n}{2} \\
0 & 0 & 0 & 1 \\
\end{pmatrix}
\]
Caveats

- Looking down \(-z, f\) and \(n\) are negative (\(n > f\))
- OpenGL convention: positive \(n\), \(f\), negate internally

\[
M = \begin{pmatrix}
\frac{2}{r-l} & 0 & 0 & \frac{r+l}{r-l} \\
0 & \frac{2}{t-b} & 0 & \frac{r+l}{t-b} \\
0 & 0 & \frac{2}{f-n} & \frac{r+l}{f-n} \\
0 & 0 & 0 & 1
\end{pmatrix}
\]

\[
glOrtho = \begin{pmatrix}
\frac{2}{r-l} & 0 & 0 & \frac{-r+l}{r-l} \\
0 & \frac{2}{t-b} & 0 & \frac{-r+b}{t-b} \\
0 & 0 & \frac{2}{f-n} & \frac{-r+n}{f-n} \\
0 & 0 & 0 & 1
\end{pmatrix}
\]

Outline

- Orthographic projection (simpler)
- Perspective projection, basic idea
- Derivation of \text{gluPerspective} (handout: \text{glFrustum})
- Brief discussion of nonlinear mapping in \(z\)

Perspective Projection

- Most common computer graphics, art, visual system
- Further objects are smaller (size, inverse distance)
- Parallel lines not parallel; converge to single point

Overhead View of Our Screen

Looks like we’ve got some nice similar triangles here?

\[
\frac{x}{z} = \frac{x'}{d} \Rightarrow x' = \frac{d}{z} x \\
\frac{y}{z} = \frac{y'}{d} \Rightarrow y' = \frac{d}{z} y
\]

In Matrices

- Note negation of \(z\) coord (focal plane \(-d\))
- (Only) last row affected (no longer 0 0 0 1)
- \(w\) coord will no longer = 1. Must divide at end

\[
P = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & -\frac{1}{d} & 0
\end{pmatrix}
\]
Verify

\[
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & -\frac{1}{d} & 0
\end{pmatrix}
\begin{pmatrix}
x \\
y \\
z \\
1
\end{pmatrix}
= \begin{pmatrix}
x - d \times x \\
y - d \times y \\
-\frac{z}{d} \\
1
\end{pmatrix}
\]

Outline

- Orthographic projection (simpler)
- Perspective projection, basic idea
- Derivation of `gluPerspective` (handout: `glFrustum`)
- Brief discussion of nonlinear mapping in z

Remember projection tutorial

Viewing Frustum

Screen (Projection Plane)

`gluPerspective(fovy, aspect, zNear > 0, zFar > 0)`
- Fovy, aspect control fov in x, y directions
- zNear, zFar control viewing frustum
Overhead View of Our Screen

\[ (0,0,0) \rightarrow (x', y', d) \]

\[ \theta = \frac{\text{fovy}}{2} \quad d = \cot \theta \]

In Matrices

- Simplest form:
  \[ P = \begin{pmatrix} 1 & 0 & 0 & 0 \\ \text{aspect} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{1}{d} & 0 \end{pmatrix} \]
- Aspect ratio taken into account
- Homogeneous, simpler to multiply through by d
- Must map z values based on near, far planes (not yet)

In Matrices

\[
\begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & d
\end{pmatrix}
\begin{pmatrix}
x \\
y \\
z \\
a + b 
\end{pmatrix}
\]

A and B selected to map n and f to -1, +1 respectively

Z mapping derivation

\[
\begin{pmatrix}
z \\
A + B \\
-1 \\
0
\end{pmatrix}
= \begin{pmatrix}
A + B \\
-1
\end{pmatrix}
\]

Simultaneous equations:

\[
\begin{align*}
-A + \frac{B}{n} &= -1 \\
-A + \frac{B}{f} &= +1
\end{align*}
\]

Outline

- Orthographic projection (simpler)
- Perspective projection, basic idea
- Derivation of gluPerspective (handout: glFrustum)
- Brief discussion of nonlinear mapping in z

Mapping of Z is nonlinear

\[
\begin{pmatrix}
z \\
A + B \\
-z
\end{pmatrix}
= -A + \frac{B}{z}
\]

- Many mappings proposed: all have nonlinearities
- Advantage: handles range of depths (10cm – 100m)
- Disadvantage: depth resolution not uniform
- More close to near plane, less further away
- Common mistake: set near = 0, far = infty. Don’t do this. Can’t set near = 0; lose depth resolution.
- We discuss this more in review session
Summary: The Whole Viewing Pipeline

- Model transformation
- Camera Transformation (gluLookAt)
- Perspective Transformation (gluPerspective)
- Viewport transformation
- Raster transformation

Slide courtesy Greg Humphreys