To Do

- Questions/concerns about assignment 1?
- Remember it is due Sep 12. Ask me or TAs re problems

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

Outline

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

Demo (Projection Tutorial)

- Nate Robbins OpenGL tutors
- Projection tutorial
- Download others

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)
**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

**In general**

- We have a cuboid that we want to map to the normalized or square cube from [-1, +1] in all axes
- We have parameters of cuboid (l,r; t,b; n,f)

**Orthographic Matrix**

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

**Transformation Matrix**

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

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

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

<table>
<thead>
<tr>
<th>x</th>
<th>y</th>
<th>z</th>
</tr>
</thead>
</table>

Translate
 Scale

Final Result

\[
M = \begin{bmatrix}
\frac{2}{f-l} & 0 & 0 & \frac{-l}{f-l} \\
0 & \frac{2}{f-b} & 0 & \frac{b}{f-b} \\
0 & 0 & \frac{2}{f-n} & \frac{n}{f-n} \\
0 & 0 & 0 & 1
\end{bmatrix} \quad \text{glOrtho} = \begin{bmatrix}
\frac{2}{f-l} & 0 & 0 & \frac{-l}{f-l} \\
0 & \frac{2}{f-b} & 0 & \frac{b}{f-b} \\
0 & 0 & \frac{2}{f-n} & \frac{n}{f-n} \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

Outline

- Orthographic projection (simpler)
- Perspective projection, basic idea
- Derivation of gluPerspective (handout: 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

\[
\begin{array}{c}
A \quad \text{Center of projection} \\
(\text{camera/eye location})
\end{array}
\]

Overhead View of Our Screen

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

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

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

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

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

- gluPerspective(fovy, aspect, zNear > 0, zFar > 0)
- Fovy, aspect control fov in x, y directions
- zNear, zFar control viewing frustum

Aspect ratio = width / height
Overhead View of Our Screen

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

\[ \theta = ? \quad d = ? \]

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

In Matrices

- Simplest form:
  \[
  P = \begin{pmatrix}
  \frac{1}{\text{aspect}} & 0 & 0 & 0 \\
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & -\frac{1}{d} & 0
  \end{pmatrix}
  \]

- Aspect ratio taken into account
- Homogeneous, simpler to multiply through by d
- Must map z vals based on near, far planes (not yet)

In Matrices

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

Z mapping derivation

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

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

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

\[
A = \frac{f + n}{f - n} \\
B = \frac{2fn}{f - n}
\]

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

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

- 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

Model coordinates

World coordinates

Eye coordinates

Screen coordinates

Window coordinates

Device coordinates

Slide courtesy Greg Humphreys