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

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)

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

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

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

Final Result

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

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

```
P = \begin{pmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & \frac{-1}{d} & 0
\end{pmatrix}
```

Overhead View of Our Screen

```
\( \frac{x}{z} = \frac{x'}{d} \Rightarrow x' = dx \\
\frac{y}{z} = \frac{y'}{d} \Rightarrow y' = dy \\
\) Looks like we’ve got some nice similar triangles here?
```

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
Verify

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

Outline

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

Remember projection tutorial

Screen (Projection Plane)

Viewing Frustum

\[
\text{gluPerspective}(\text{fovy}, \text{aspect}, \text{zNear} > 0, \text{zFar} > 0)
\]
- Fovy, aspect control fov in x, y directions
- zNear, zFar control viewing frustum
Overhead View of Our Screen

\[
\begin{align*}
(0,0,0) & \rightarrow (x', y', d) \\
(x', y', d) & \rightarrow (x, y, z) \\
\theta = ? & \quad d = ? \\
\theta = \frac{\text{fovy}}{2} & \quad d = \cot \theta
\end{align*}
\]

In Matrices

- Simplest form:

\[
\begin{pmatrix}
\frac{1}{\text{aspect}} & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & \frac{-1}{d} & 0
\end{pmatrix}
\]

- 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

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

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

Z mapping derivation

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

- Simultaneous equations?

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

\[
\begin{align*}
A & = \frac{f + n}{f - n} \\
B & = \frac{2fn}{f - n}
\end{align*}
\]

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

\[
\begin{pmatrix}
Az + B/z \\
-1
\end{pmatrix} = -A \frac{B}{z}
\]

- 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