## 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

## Projections

- To lower dimensional space (here 3D -> 2D)
- Preserve straight lines
- Trivial example: Drop one coordinate (Orthographic)
Orthographic Projection
- Characteristic: Parallel lines remain parallel
- Simplest form: project onto x-y plane, drop z coordinate
- Useful for technical drawings etc.

Orthographic Projection

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 Perspective

In general

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

Orthographic Matrix

Transformation Matrix
- Scale
- Translation (centering)

Transformation Matrix

Transformation Matrix
- Looking down –z, f and n are negative (n > f)
- OpenGL convention: positive n, f, negate internally
Final Result

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

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

Foundations of Computer Graphics

Online Lecture 5: Viewing

Perspective Projection

Ravi Ramamoorthi

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?

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

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

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

\[ P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{1}{d} & 0 \end{bmatrix} \]
Foundations of Computer Graphics

Online Lecture 5: Viewing

Derivation of glvPerspective

Ravi Ramamoorthi

Outline

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

Remember projection tutorial

Viewing Frustum

Verify

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

Verify

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

Near plane

Far plane
**Screen (Projection Plane)**

Field of view (fovy)

Aspect ratio = width / height

---

**gluPerspective**

- `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**

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

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

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

---

**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**

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

- A and B selected to map n and f to -1, +1 respectively
### Z mapping derivation

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

- Simultaneous equations?

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

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

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

- 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

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

- 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.
Summary: The Whole Viewing Pipeline

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

Model coordinates → World coordinates

Eye coordinates

Screen coordinates

Window coordinates

Device coordinates

Slide courtesy Greg Humphreys