Computer Graphics

CSE 167 [Win 19], Lecture 5: Viewing

Ravi Ramamoorthi

http://viscomp.ucsd.edu/classes/cse167/wi19

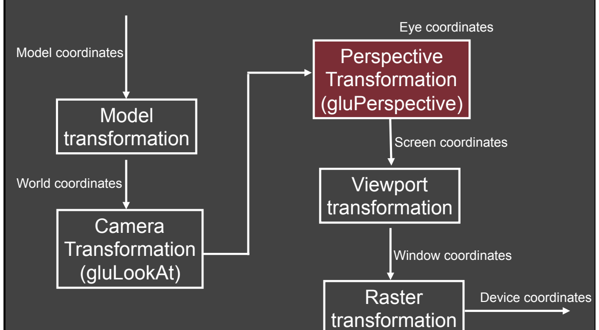
To Do

- Questions/concerns about assignment 1?
- Remember it is due tomorrow! (Jan 16).
- 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

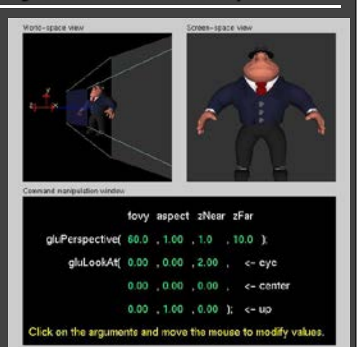
Summary: The Whole Viewing Pipeline



Slide courtesy Greg Humphreys

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)

Outline

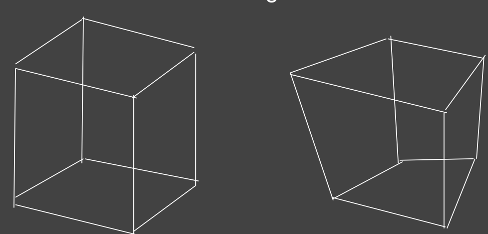
- *Orthographic projection (simpler)*
- Perspective projection, basic idea
- Derivation of gluPerspective (handout: glFrustum)
- Brief discussion of nonlinear mapping in z

Projections

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

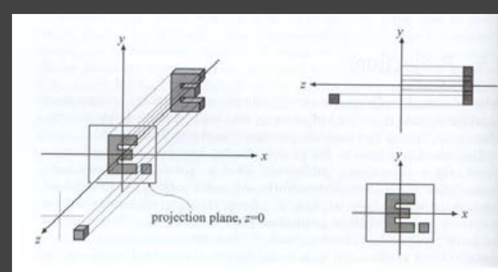
Orthographic Projection

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



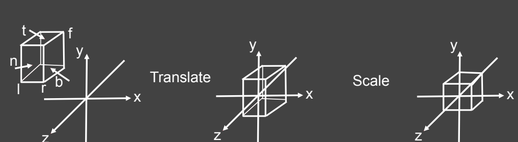
Example

- Simply project onto xy plane, drop z coordinate



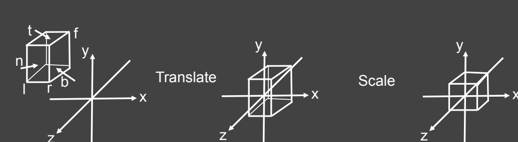
In general

- 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



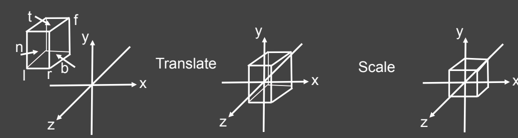
Transformation Matrix

Scale — Translation (centering)

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

Caveats

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



Final Result

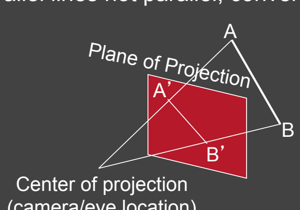
$$M = \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{2}{f-n} & -\frac{f+n}{f-n} \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad glOrtho = \begin{pmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\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

Outline

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

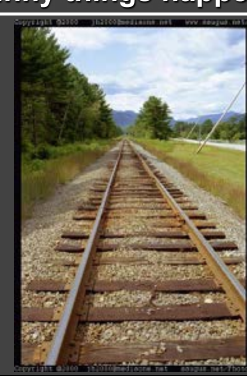
Perspective Projection

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

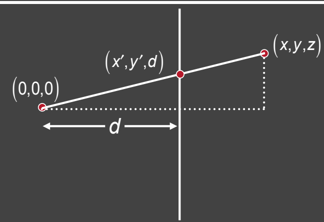


Slides inspired by Greg Humphreys

Funny things happen…



Overhead View of Our Screen



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

$$\frac{x}{z} = \frac{x'}{d} \Rightarrow x' = \frac{d * x}{z} \qquad \frac{y}{z} = \frac{y'}{d} \Rightarrow y' = \frac{d * y}{z}$$

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

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

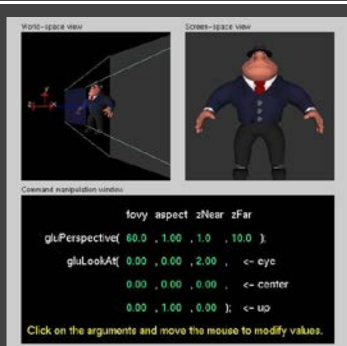
Verify

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -\frac{1}{d} & 0 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = ? \quad \begin{pmatrix} x \\ y \\ z \\ -\frac{z}{d} \end{pmatrix} = \begin{pmatrix} -\frac{d * x}{z} \\ -\frac{d * y}{z} \\ -d \\ 1 \end{pmatrix}$$

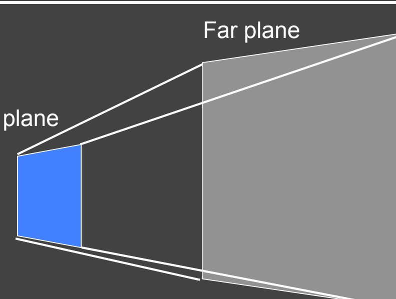
Outline

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

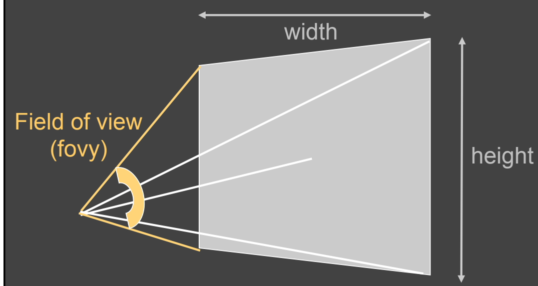
Remember projection tutorial



Viewing Frustum



Screen (Projection Plane)

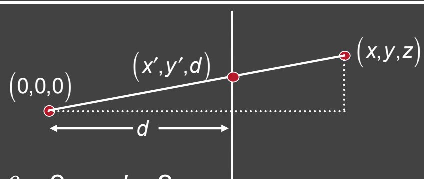


Aspect ratio = width / height

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



$\theta = ?$ $\quad d = ?$

$\theta = \frac{fovy}{2}$ $\quad d = \cot\theta$

In Matrices

- Simplest form:

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

- 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

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

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

Z mapping derivation

$$\begin{pmatrix} A & B \\ -1 & 0 \end{pmatrix}\begin{pmatrix} z \\ 1 \end{pmatrix} = ? \qquad \begin{pmatrix} Az+B \\ -z \end{pmatrix} = -A - \frac{B}{z}$$

- Simultaneous equations?

$$-A + \frac{B}{n} = -1 \qquad A = -\frac{f+n}{f-n}$$

$$-A + \frac{B}{f} = +1 \qquad B = -\frac{2fn}{f-n}$$

Outline

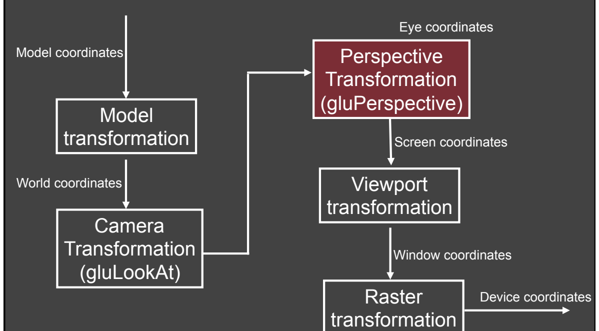
- Orthographic projection (simpler)
- Perspective projection, basic idea
- Derivation of gluPerspective (handout: glFrustum)
- *Brief discussion of nonlinear mapping in z*

Mapping of Z is nonlinear

$$\left(\frac{Az + B}{-z} \right) = -A - \frac{B}{z}$$

- 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



Slide courtesy Greg Humphreys