News

- make sure you’ve signed up for demo slot

Review: Display Lists

- reuse block of OpenGL code
  - efficiency
  - multiple instances of same object
  - static objects redrawn often
  - exploit hierarchical structure when possible
- set up list once with glNewList/glEndList
- call multiple times

Review: Camera Motion

- rotate/translate/scale difficult to control
- arbitrary viewing position
  - eye point, gaze/lookat direction, up vector

Review: World to View Coordinates

- translate eye to origin
- rotate view vector (lookat – eye) to w axis
- rotate around w to bring up into vw-plane

$$M_{wv} = \begin{bmatrix} u & u & u & -u \cdot e \\ v & v & v & -v \cdot e \\ w & w & w & -w \cdot e \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
Pinhole Camera

- Ingredients
- Box
- Film
- Hole punch
- Results
- Pictures!

www.kodak.com

Pinhole Camera

- Theory: perfect pinhole

Pinhole Camera

- Non-zero sized hole

Pinhole Camera

- Field of view and focal length

Pinhole Camera

- Actual: moving film plane

Real Cameras

- Pinhole camera has small aperture (lens opening)
  - Hard to get enough light to expose the film

- Lens permits larger apertures
  - Lens permits changing distance to film plane without actually moving the film plane

Price to pay: Limited depth of field
Graphics Cameras

- real pinhole camera: image inverted
- computer graphics camera: convenient equivalent

General Projection

- image plane need not be perpendicular to view plane

Perspective Projection

- our camera must model perspective

Perspective Projections

- classified by vanishing points

Page 3
Projective Transformations

- Planar geometric projections
  - Planar: onto a plane
  - Geometric: using straight lines
  - Projections: 3D -> 2D
  - Aka projective mappings
  - Counterexamples?

Properties

- Lines mapped to lines and triangles to triangles
- Parallel lines do NOT remain parallel
  - E.g. rails vanishing at infinity
- Affine combinations are NOT preserved
  - E.g. center of a line does not map to center of projected line (perspective foreshortening)

Perspective Projection

- Project all geometry
  - Through common center of projection (eye point)
  - Onto an image plane

Basic Perspective Projection

- Desired result for a point $[x, y, z, 1]^T$ projected onto the view plane:
  \[
  \frac{x'}{d} = \frac{x}{z}, \quad \frac{y'}{d} = \frac{y}{z}
  \]
  \[
  x' = \frac{x \cdot d}{z}, \quad y' = \frac{y \cdot d}{z/d}, \quad z = d
  \]
  - Nonuniform foreshortening
  - Not affine

What could a matrix look like to do this?
Perspective Projection Matrix
\[
\begin{bmatrix}
\frac{x}{z/d} \\
\frac{y}{z/d} \\
\frac{z}{d}
\end{bmatrix}
\] is homogenized version of
\[
\begin{bmatrix}
x \\
y \\
z \\
z/d
\end{bmatrix}
\] where \( w = z/d \)

Perspective Projection Matrix
\[
\begin{bmatrix}
x \\
y \\
z \\
z/d
\end{bmatrix}
\]

Moving COP to Infinity
- as COP moves away, lines approach parallel
- when COP at infinity, orthographic view

Orthographic Camera Projection
\[
\begin{bmatrix}
x_p \\
y_p \\
z_p \\
1
\end{bmatrix} =
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z \\
1
\end{bmatrix}
\]

Orthographic Camera Projection
- camera's back plane parallel to lens
- infinite focal length
- no perspective convergence
- just throw away z values
**Perspective to Orthographic**

- transformation of space
- center of projection moves to infinity
- view volume transformed
  - from frustum (truncated pyramid) to parallelepiped (box)

**View Volumes**

- specifies field-of-view, used for clipping
- restricts domain of $z$ stored for visibility test

**View Volume**

- convention
- viewing frustum mapped to specific parallelepiped
  - Normalized Device Coordinates (NDC)
  - same as clipping coords
- only objects inside the parallelepiped get rendered
- which parallelepiped?
  - depends on rendering system

**Normalized Device Coordinates**

left/right $x = +/- 1$, top/bottom $y = +/- 1$, near/far $z = +/- 1$

**Understanding Z**

- $z$ axis flip changes coord system handedness
- RHS before projection (eye/view coords)
- LHS after projection (clip, norm device coords)

**Understanding Z**

near, far always positive in OpenGL calls

```c
glOrtho(left,right,bot,top,near,far);
glFrustum(left,right,bot,top,near,far);
glPerspective(fovy,aspect,near,far);
```
Understanding Z

- why near and far plane?
  - near plane:
    - avoid singularity (division by zero, or very small numbers)
  - far plane:
    - store depth in fixed-point representation (integer), thus have to have fixed range of values (0…1)
    - avoid/reduce numerical precision artifacts for distant objects

Orthographic Derivation

- scale, translate, reflect for new coord sys

\[ y' = a \cdot y + b \quad y = \text{top} \rightarrow y' = 1 \]
\[ y' = a \cdot y + b \quad y = \text{bot} \rightarrow y' = -1 \]
Orthographic Derivation

- scale, translate, reflect for new coord sys

\[
P' = \begin{bmatrix}
\frac{2}{\text{right-left}} & 0 & 0 & \frac{\text{right-left}}{\text{right-left}} \\
0 & \frac{2}{\text{top-bot}} & 0 & \frac{\text{top-bot}}{\text{top-bot}} \\
0 & 0 & -2 & \frac{\text{far-near}}{\text{far-near}} \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

Orthographic OpenGL

\begin{verbatim}
glMatrixMode(GL_PROJECTION);
glLoadIdentity();
glOrtho(left,right,bot,top,near,far);
\end{verbatim}