OpenGL Projection Tutorial

Parameterized by: \( \text{glFrustum} \)
- left, right, top, bottom (generally symmetric)
- near, far
Or, when symmetric, by: \( \text{gluPerspective} \)
- Field of view (FOV), aspect ratio
- near, far
- Aspect ratio is the x/y ratio of the final displayed image. Common values:
  - 4/3 for TV & old movies; 1.66 for cartoons & European movies; 16/9 for American movies & HDTV; 2.35 for epic movies

View Frustum

OpenGL

- \( \text{gluPerspective}(...) \)
  - Field of view in the \( y \) direction, FOV, (vertical field-of-view)
  - Aspect ratio, \( a \), should match window aspect ratio
  - Near and far clipping planes, \( n \) and \( f \)
  - Defines a symmetric view volume
- \( \text{glFrustum}(...) \)
  - Give the near and far clip plane, and places where the other clip planes cross the near plane
  - Defines the general case
  - Used for stereo viewing, mostly

 gluPerspective to glFrustum

- As noted previously, \( \text{glu} \) functions don’t add basic functionality, they are just more convenient
- So how does \( \text{gluPerspective} \) convert to \( \text{glFrustum} \)?
- Symmetric, so only need \( t \) and \( n \)

Demo Projection Tutor

Viewing System PDFs
3D Projection

\[ \text{Proj} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{d} & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ \frac{z}{d} \\ \frac{z}{d} \end{bmatrix} \]

How many vanishing points?

PDF of OpenGL projection

Canonical View Volume
Canonical to Window

- Canonical Viewing Volume (what is it?)
- To Window

\[ M_{\text{view}} = \begin{bmatrix}
\frac{n_o}{2} & 0 & 0 & \frac{n_o - 1}{2} \\
0 & \frac{n_o}{2} & 0 & \frac{n_o - 1}{2} \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix} \]

\[ M_{\text{sys}} = M_{\text{window}} M_{\text{persp}} M_{\text{view}} \]

Complete Perspective Projection

- After applying the perspective matrix, we map the orthographic view volume to the canonical view volume:

\[ M_{\text{persp}} = \begin{bmatrix}
2 & 0 & 0 & \frac{(r+1)}{r-f} \\
0 & 2 & 0 & \frac{(s+1)}{s-f} \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1 \\
\end{bmatrix} \]

\[ M_{\text{sys}} = M_{\text{window}} M_{\text{persp}} M_{\text{view}} \]

Full OpenGL Ortho Projection

GL Matrix Example

```c
// Clear screen
glClear(GL_COLOR_BUFFER_BIT, GL_DEPTH_BUFFER_BIT);

// Set up projection
glMatrixMode(GL_PROJECTION);
gluPerspective(fov, aspect, nearclip, farclip);

// Set up camera view
glMatrixMode(GL_MODELVIEW);
gluLookAt(eye.x, eye.y, eye.z, target.x, target.y, target.z, 0, 1, 0);

// Draw all objects
for (each object) {
    glPushMatrix();
    glTranslatef(pos[i].x, pos[i].y, pos[i].z);
    glRotatef(axis[i].x, axis[i].y, axis[i].z, angle[i]);
    Model[i]->Draw();
    glPopMatrix();
}

// Finish
glFlush();
glSwapBuffers();
```