Review of Last Thursday

• 3D introduces perspective, occlusion, depth
• Camera parameters (position, fov, …) define how objects are mapped to 2D

• Vertex processor:
  • ModelView Transformation: From model space to camera space
  • Projection Transformation: From camera space to clip space
Review of Last Thursday

- Orthogonal, parallel, perspective projection

Near and far are positive values (default: distances along negative z)
Perspective Projection

- Points are projected along ray through camera origin
- Points along a ray are projected to the same location
Perspective Projection

\[
\frac{x}{z} = \frac{x_p}{z_p} \quad x_p = \frac{x}{z/z_p} \quad z_p = d
\]

\[
x_p = \frac{x}{z/d}
\]

\[
y_p = \frac{y}{z/d}
\]
Perspective Projection

- Projection matrix with camera at origin, plane at distance d

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

w component not equal to one!
Result is in clip coordinates
Perspective Division

- **Perspective division** gives normalized device coordinates
  - Divide projected x,y,z,w components by projected w.

\[
P \cdot \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} x_p \\ y_p \\ z_p \\ w_p \end{pmatrix}
\]

\[
\begin{pmatrix} x_n \\ y_n \\ z_n \\ w_n \end{pmatrix} = \begin{pmatrix} x_p/w_p \\ y_p/w_p \\ z_p/w_p \\ 1 \end{pmatrix}
\]
Perspective Projection

- We use the projection matrix to map normalized coordinates to the canonical clip volume $[-1, 1]^3$

$$z_n(-\text{near}) = -1 \quad \text{AND} \quad z_n(-\text{far}) = 1$$

$$\rightarrow \quad z_n = \left( -\frac{\text{far} + \text{near}}{\text{far} - \text{near}} \cdot z - \frac{2 \cdot \text{far} \cdot \text{near}}{\text{far} - \text{near}} \right) \cdot -\frac{1}{z}$$

$$x_n(x_p = \text{left}) = -1 \quad \text{AND} \quad x_n(x_p = \text{right}) = 1 \quad \text{AND} \quad x_p = \frac{\text{near} \cdot x}{-z}$$

$$\rightarrow \quad x_n = \left( \frac{2 \cdot \text{near}}{\text{right} - \text{left}} \cdot x + \frac{\text{right} + \text{left}}{\text{right} - \text{left}} \cdot z \right) \cdot -\frac{1}{z}$$
Perspective Projection

• Full perspective projection matrix:

\[
\begin{align*}
    z_n &= \left( -\frac{\text{far} + \text{near}}{\text{far} - \text{near}} \cdot z - \frac{2 \cdot \text{far} \cdot \text{near}}{\text{far} - \text{near}} \right) \cdot -\frac{1}{z} \\
    x_n &= \left( \frac{2 \cdot \text{near}}{\text{right} - \text{left}} \cdot x + \frac{\text{right} + \text{left}}{\text{right} - \text{left}} \cdot z \right) \cdot -\frac{1}{z}
\end{align*}
\]

\[
P = \begin{pmatrix}
    \frac{2 \cdot \text{near}}{\text{right} - \text{left}} & 0 & \frac{\text{right} + \text{left}}{\text{top} - \text{bottom}} & 0 \\
    0 & \frac{2 \cdot \text{near}}{\text{top} - \text{bottom}} & \frac{\text{right} - \text{left}}{\text{top} - \text{bottom}} & 0 \\
    0 & 0 & -\frac{\text{far} + \text{near}}{\text{far} - \text{near}} & -\frac{2 \cdot \text{far} \cdot \text{near}}{\text{far} - \text{near}} \\
    0 & 0 & -1 & 0
\end{pmatrix}
\]
Projection

- Projection transforms viewing volume into canonical volume
- Perspective projection requires perspective division

After perspective division:
- $x, y$ hold 2D coordinates in $[-1,1]^2$
- $z$ holds depth in $[-1,1]$
Vertex Processor - Summary

- Transforms vertices (points and vectors)
- Manipulates (and sometimes creates) vertices

- Performs ModelView transformation
  - Maps object coordinates to world space
  - Maps world coordinates to camera/eye space

- Performs Projection transformation
  - Transforms viewing frustum into canonical volume
Clipping and Assembling

Vertex Processor → Clipper and Assembler → Rasterizer → Fragment Processor

3D Triangles → Clipped Triangles → 2D Triangles → Fragments
Before perspective division, we clip primitives by testing in clip coordinates (this is how it is done in OpenGL):

\[-w_p \leq x_p \leq w_p\]

\[-w_p \leq y_p \leq w_p\]

\[-w_p \leq z_p \leq w_p\]
Line Clipping

- 2D

- Clipping requires information about connectivity (OpenGL: GL_LINES, …)
Cohen-Sutherland Clipping

- Compute ‘outcodes’ for endpoints to distinguish cases

Four cases:

\[ o_1 = o_2 = 0 \]
\[ o_1 \neq 0, o_2 = 0 \quad \text{OR} \quad o_1 = 0, o_2 \neq 0 \]
\[ (o_1 \& o_2) \neq 0 \]
\[ (o_1 \& o_2) = 0 \]
Liang-Barsky Clipping

- Clipping performed for parametric line representations

\[ f(u) = (1 - u) \cdot A + u \cdot B \]
Liang-Barsky Clipping

\[ f(u) = (1 - u) \cdot A + u \cdot B \]

For top boundary line of window:

\[ u = \frac{y_{max} - y_1}{y_2 - y_1} \]

\[ u \Delta y = y_{max} - y_1 \]

\[ u \Delta y \leq y_{max} - y_1 \]

Define inequality to test for ‘visible side’ of boundary line

\[ u p_i \leq q_i \]

One inequality for each edge
Liang-Barsky Clipping

- $p_i < 0$
  - $u \geq q_i/p_i$
    - Line crosses from invisible to visible side of boundary line
- $p_i > 0$
  - $u \leq q_i/p_i$
    - Line crosses from visible to invisible side of boundary line
- $p_i = 0$
  - $q_i < 0$
    - Trivial reject
  - Line is inclusive
Polygon Clipping

Shadows

Concave polygons
Polygon Clipping

- Clipping can be described as black-box clipper

\[ x_3 = x_1 + (y_{max} - y_1) \frac{\Delta x}{\Delta y} \]

\[ y_3 = y_{max} \]
Polygon Clipping

- Clipping as a pipeline
Polygon Clipping

- Bounding boxes, convex hulls

- How are curves, text, etc. clipped?
Clipping in 3D

- Line – plane intersections

\[ n \cdot (p - p_0) = 0 \]

Plane with normal \( n \), point \( p_0 \)

\[ p(u) = (1 - u) \cdot p_1 + u \cdot p_2 \]

Parametric line equation
Clipping in 3D

- Line – plane intersections

\[ n \cdot (p(u) - p_0) = 0 \]

Intersection equation, solve for \( u \)

\[ n \cdot ((1 - u) \cdot p_1 + u \cdot p_2 - p_0) = 0 \]

\[ u = \frac{n \cdot (p_0 - p_1)}{n \cdot (p_2 - p_1)} \]
A Note on Planes and Normals

- Planes can be fully specified by three points
- These points may be used to specify a triangle

- Use the cross product to find ‘third orthogonal vector’
- Normalized vector corresponds to triangle/plane normal