Overview

- Projection Mathematics
- Canonical View Volume
- Parallel Projection Pipeline
- Perspective Projection Pipeline

Lecture Credits: Most pictures are from Foley/VanDam; additional and extensive thanks also goes to those credited on individual slides.

Projection Mathematics

- What is the set of transformations needed to map 3D lines/planes onto a 2D screen positioned in 3D?
- Basic procedure
  - 4D homogeneous coordinates to
  - 3D homogeneous coordinates for
  - every primitive in
  - the 3D view volume

The Perspective Projection

Determined scale
- Consider
  - Point P
  - Projected onto projection plane as point \( P_p \)
- Idea: compute ratios via similar triangles

The Perspective Projection

- In the x direction ratio is
  \[
  \frac{z}{d} = \frac{x}{x_p} \quad x_p = \frac{x}{z/d}
  \]
The Perspective Projection

- In the y direction ratio is
  \[
  \frac{z}{d} = \frac{y}{y_p} = \frac{y}{z/d}
  \]

- Homogenous perspective projection matrix
  \[
  M_{\text{per}} = \begin{bmatrix}
  1 & 0 & 0 & 0 \\
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 1/d & 0
  \end{bmatrix}
  \]
  Assumes VPN is z axis.

The Perspective Projection

- Homogenous perspective projection
  \[
  \begin{bmatrix}
  x \\
  y \\
  z \\
  z/d
  \end{bmatrix} = \begin{bmatrix}
  1 & 0 & 0 & 0 \\
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 1/d & 0
  \end{bmatrix} \begin{bmatrix}
  x \\
  y \\
  z \\
  z/d
  \end{bmatrix}
  \]

The Orthographic Projection

- Homogenous perspective projection to 3D
  \[
  \begin{bmatrix}
  x \\
  y \\
  z \\
  z/d
  \end{bmatrix} = \begin{bmatrix}
  1 & 0 & 0 & 0 \\
  0 & 1 & 0 & 0 \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 1/d & 0
  \end{bmatrix} \begin{bmatrix}
  x \\
  y \\
  z \\
  z/d
  \end{bmatrix}
  \]

Implementing Projections (Foley et al.)

- A sequence of matrix operations and a clipping procedure...
Implementing Projections (Foley et al.)

1. Extend 3D coordinates to homogenous coords
2. Apply normalizing transformation, \( N_{\text{par}} \) or \( N_{\text{per}} \)
3. Divide by \( W \) to map back down to 3D
4. Clip in 3D against canonical view volume
   - parallel or perspective view volume
5. Extend 3D coordinates back to homogenous
6. Perform parallel projection using \( M_{\text{par}} \) or
   Perform perspective projection \( M_{\text{per}} \)
7. Divide by \( W \) to map from homogenous to 2D coordinates (division effects perspective projection)
8. Translate and scale (in 2D) to device coordinates

Canonical View Volume: Parallel Projection

- Defined by 6 planes:
  - \( x = 1 \)
  - \( x = -1 \)
  - \( y = 1 \)
  - \( y = -1 \)
  - \( z = 0 \)
  - \( z = -1 \)
- Easy to clip against

Canonical View Volume: Perspective Projection

- Defined by 6 planes:
  - \( x = z \)
  - \( x = -z \)
  - \( y = z \)
  - \( y = -z \)
  - \( z = z_{\text{min}} \)
  - \( z = -1 \)
- Easy to clip against

Parallel Projection Pipeline

Transforming an arbitrary view volume into the canonical one

1. Translate VRP to the origin
2. Rotate so VPN becomes \( z \), VUP becomes \( y \) and \( u \) becomes \( x \)
3. Shear to make direction of the projection become parallel to \( z \)
4. Translate and scale into a canonical view volume

1. Translate VRP to the origin
   - Simple translation \( T(-\text{VRP}) \)
   \[
   T = \begin{bmatrix} 1 & 0 & 0 & -\text{VRP}_x \\ 0 & 1 & 0 & -\text{VRP}_y \\ 0 & 0 & 1 & -\text{VRP}_z \\ 0 & 0 & 0 & 1 \end{bmatrix}
   \]

2. Rotate
   - VPN rotated to \( z \)
   - VUP rotated to \( y \)
   \[
   R = \begin{bmatrix} r_{1x} & r_{1y} & r_{1z} & 0 \\ r_{2x} & r_{2y} & r_{2z} & 0 \\ r_{3x} & r_{3y} & r_{3z} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
   \]

VRP, VPN & VUP in World Coordinates (x,y,z)
2. Rotate

3. Shear

3. Shear (Cont.)

3. Shear (Cont.)

3. Shear (Finally)

- For $\mathbf{Sh}_{xy}$: $z' = z, \quad x' = x + z \cdot sh_x, \quad y' = y + z \cdot sh_y$

- Computing direction of projection $\mathbf{DOP}$

- Find $\mathbf{Sh}_{xy}$ such that:

- Old Foley et al. is wrong!

- We can compute the $\mathbf{SH}_{par}$ matrix
4. Translate and Scale

- Translate the center of the volume to the origin
- Scaling to 2x2x1
- Needed for 3D clipping

\[
T_{par} = \begin{bmatrix}
1 & 0 & -\frac{(u_{\text{max}} + u_{\text{min}})}{2} \\
0 & 1 & -\frac{(v_{\text{max}} + v_{\text{min}})}{2} \\
0 & 0 & 1
\end{bmatrix}
\]

\[
S_{par} = \begin{bmatrix}
2 & 0 & 0 \\
0 & 2 & 0 \\
0 & 0 & 1
\end{bmatrix}
\]

\[
R = \begin{bmatrix}
r_{11} & r_{12} & r_{13} & 0 \\
r_{21} & r_{22} & r_{23} & 0 \\
r_{31} & r_{32} & r_{33} & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
T_{VRP} = \begin{bmatrix}
1 & 0 & 0 & -\text{VRP}_x \\
0 & 1 & 0 & -\text{VRP}_y \\
0 & 0 & 1 & -\text{VRP}_z \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

Parallel Projection View

Volume Transformation

\[
N_{par} = (S_{par} \cdot (T_{par} \cdot (SH_{par} \cdot (R \cdot T_{-VRP}))))
\]

- Apply to all model vertices
- \( P' = N_{par} \cdot P \)

Parallel Projection Summary

Transforming an arbitrary view volume into the canonical one

1. Translate VRP to the origin
2. Rotate so VPN becomes \( z \), VUP becomes \( y \), and \( u \) becomes \( x \)
3. Translate COP to origin
4. Shear so volume centerline becomes \( z \) axis
5. Scale into a canonical view volume for clipping

1. Translate VRP to the origin

- Simple translation \( T(-\text{VRP}) \)

\[
T = \begin{bmatrix}
1 & 0 & 0 & -\text{VRP}_x \\
0 & 1 & 0 & -\text{VRP}_y \\
0 & 0 & 1 & -\text{VRP}_z \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

VRP, VPN & VUP in World Coordinates (x,y,z)

2. Rotate

- VPN rotated to \( z \)
- VUP rotated to \( y \)

\[
R = \begin{bmatrix}
r_{11} & r_{12} & r_{13} & 0 \\
r_{21} & r_{22} & r_{23} & 0 \\
r_{31} & r_{32} & r_{33} & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]
3. Translate

- Simple translation \( T(-PRP) \)
  
  \[
  T = \begin{bmatrix}
  1 & 0 & 0 & -pp_x \\
  0 & 1 & 0 & -pp_y \\
  0 & 0 & 1 & 0 \\
  0 & 0 & 0 & 1
  \end{bmatrix}
  \]
  
  PRP in VRC Coordinates (u,v,x,n)

4. Shear

- Goal is to transform the center line to the \( z \) axis
- Same as parallel projection
- Shear matrix is the same!

\[
SH_{xy} = \begin{bmatrix}
1 & 0 & 1/2(u_{max} + u_{min}) - pp_y \\
0 & 1 & pp_y \\
0 & 0 & 1
\end{bmatrix}
\]

5. Scaling

- We can define VRP after transformation as
  
  \[
  VRP = SH_{xy} \cdot T(-PRP) \cdot \begin{bmatrix} 0.0.0.1 \end{bmatrix}
  \]
- \( VRP \) z = -ppz, since shear does not affect z coordinates
- First, we scale differentially in x and y to set plane slopes to 1 and -1
  
  \[
  \begin{bmatrix}
  -2pp_x & -2pp_y \\
  -2pp_y & -2pp_y
  \end{bmatrix}
  \]
- Second, we scale uniformly by
  
  \[
  \frac{1}{pp_x} + \frac{1}{pp_y} = \frac{1}{pp_z}
  \]

5. Scaling (Cont)

- We can combine the transformations

\[
SH_{yz} = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

Important Projection Info

- Back clipping plane is \( z = -1 \)
- Front clipping plane is \( z = -\frac{vpp \cdot F - pp_x \cdot F}{pp_y \cdot B - pp_x} \)
- Projection plane is \( d = \frac{-vpp \cdot \text{proj}}{pp_y \cdot B - pp_x} \)
**Perspective Projection Pipeline**

\[ N_{\text{per}} = (S_{\text{par}} \cdot (S_{\text{par}} \cdot (T(-PRP) \cdot (R \cdot T(-VRP)))))) \]

- Apply to all model vertices
- \( P' = N_{\text{per}} P \)

**Perspective Projection Summary**

**Summary of 3D Transforms**

- We know how to take any projection and convert it into a canonical View Volume
- 3D edges can be clipped against it and projected onto screen

**Implementing Projections Without 3D Clipping**

1. Extend 3D coordinates to homogeneous coordinates
2. Apply normalizing transformation, \( N_{\text{par}} \) or \( N_{\text{per}} \)
3. Perform trivial reject test with view volume
4. Perform parallel projection or perform perspective projection
5. Clip against 2D “world” (view plane) window
6. Translate and scale (in 2D) to device coordinates (i.e. into viewport)
7. Draw/Fill polygons

**Transformed Window**

- Parallel Projection
  - \((-1,-1) \rightarrow (1,1)\)
- Perspective Projection
  - \((-|z_{\text{proj}}|, -|z_{\text{proj}}|) \rightarrow (|z_{\text{proj}}|, |z_{\text{proj}}|)\)

- Use these for world window parameters in viewport mapping and 2D clipping

**Programming assignment 4**

- Read SMF file
- Implement parallel projection
- Implement perspective projection
- Draw projected and clipped polygon edges