3D Polygon Rendering Pipeline

Adam Finkelstein
Princeton University
COS 426, Fall 2001

• Many applications use rendering of 3D polygons with direct illumination

Quake II
(3D Realms)

Ray Casting Revisited

• For each sample …
  o Construct ray from eye position through view plane
  o Find first surface intersected by ray through pixel
  o Compute color of sample based on surface radiance

More efficient algorithms utilize spatial coherence!

3D Polygon Rendering

• What steps are necessary to utilize spatial coherence while drawing these polygons into a 2D image?
This is a pipelined sequence of operations to draw a 3D primitive into a 2D image.

3D Rendering Pipeline (for direct illumination)

- Transform into 3D world coordinate system
- Illuminate according to lighting and reflectance
- Transform into 3D camera coordinate system

Example: OpenGL

OpenGL executes steps of 3D rendering pipeline for each polygon.

```
glBegin(GL_POLYGON);
glVertex3f(0.0, 0.0, 0.0);
glVertex3f(1.0, 0.0, 0.0);
glVertex3f(1.0, 1.0, 1.0);
glVertex3f(0.0, 1.0, 1.0);
glEnd();
```
### 3D Rendering Pipeline (for direct illumination)

- **Modeling**
  - Transform into 3D world coordinate system
- **Lighting**
  - Illuminate according to lighting and reflectance
- **Viewing Transformation**
  - Transform into 3D camera coordinate system
- **Projection Transformation**
  - Transform into 2D camera coordinate system
- **Clipping**
  - Clip primitives outside camera’s view
- **Scan Conversion**
  - Draw pixels (includes texturing, hidden surface, etc.)

### Transformations

- **Modeling Transformation**
  - Transform into 3D world coordinate system
- **Lighting**
  - Illuminate according to lighting and reflectance
- **Viewing Transformation**
  - Transform into 3D camera coordinate system
- **Projection Transformation**
  - Transform into 2D camera coordinate system
- **Clipping**
  - Clip primitives outside camera’s view
- **Scan Conversion**
  - Draw pixels (includes texturing, hidden surface, etc.)

### Viewing Transformations

- **Modeling Transformation**
  - Transform into 3D world coordinate system
- **Lighting**
  - Illuminate according to lighting and reflectance
- **Viewing Transformation**
  - Transform into 3D camera coordinate system
- **Projection Transformation**
  - Transform into 2D camera coordinate system
- **Clipping**
  - Clip primitives outside camera’s view
- **Scan Conversion**
  - Draw pixels (includes texturing, hidden surface, etc.)

### Camera Coordinates

- **Canonical coordinate system**
  - Convention is right-handed (looking down -z axis)
  - Convenient for projection, clipping, etc.

- **Camera up vector** maps to Y axis
- **Camera back vector** maps to Z axis (pointing out of page)
- **Camera right vector** maps to X axis
Viewing Transformation

- Mapping from world to camera coordinates
  - Eye position maps to origin
  - Right vector maps to X axis
  - Up vector maps to Y axis
  - Back vector maps to Z axis

Finding the viewing transformation

- We have the camera (in world coordinates)
- We want $T$ taking objects from world to camera
- Trick: find $T^{-1}$ taking objects in camera to world

$$p'' = T^{-1} p'$$

Finding the Viewing Transformation

- Trick: map from camera coordinates to world
  - Origin maps to eye position
  - Z axis maps to Back vector
  - Y axis maps to Up vector
  - X axis maps to Right vector

$$\begin{bmatrix}
  x' \\
  y' \\
  z'
\end{bmatrix} = \begin{bmatrix}
  R_x & U_x & B_x & E_x \\
  R_y & U_y & B_y & E_y \\
  R_z & U_z & B_z & E_z
\end{bmatrix} \begin{bmatrix}
  x \\
  y \\
  z
\end{bmatrix}$$

- This matrix is $T^{-1}$ so we invert it to get $T$ … easy!

Projection

- General definition:
  - Transform points in $n$-space to $m$-space ($m<n$)
- In computer graphics:
  - Map 3D camera coordinates to 2D screen coordinates

Taxonomy of Projections

- Plane geometric projections
  - Parallel
  - Oblique
  - Perspective
  - Orthographic
  - Axonometric
  - Front elevation
  - Top (plan)
  - Isometric
  - Cabinet
  - Cavalier
  - One-point
  - Two-point
  - Three-point
  - Isometric
  - Other
  - Other
Taxonomy of Projections

Parallel Projection
• Center of projection is at infinity
  o Direction of projection (DOP) same for all points

Orthographic Projections
• DOP perpendicular to view plane
  - Top
  - Front
  - Side

Oblique Projections
• DOP not perpendicular to view plane
  - Cavalier (DOP $\alpha = 45^\circ$)
  - Cabinet (DOP $\alpha = 63.4^\circ$)

Parallel Projection View Volume

Parallel Projection Matrix
• General parallel projection transformation:
Taxonomy of Projections

- Perspective Projection: Map points onto "view plane" along "projectors" emanating from "center of projection" (COP)

Perspective Projection View Volume

- How many vanishing points?

Perspective Projection

- Compute 2D coordinates from 3D coordinates with similar triangles

What are the coordinates of the point resulting from projection of (x,y,z) onto the view plane?
### Perspective Projection Matrix

- **4x4 matrix representation?**

\[
\begin{bmatrix}
    x_s \\
    y_s \\
    z_s \\
    w_s
\end{bmatrix} =
\begin{bmatrix}
    1 & 0 & 0 & 0 \\
    0 & 1 & 0 & 0 \\
    0 & 0 & 1 & D \\
    0 & 0 & 1/D & 1
\end{bmatrix}
\begin{bmatrix}
    x_r \\
    y_r \\
    z_r \\
    w_r
\end{bmatrix}
\]

\[
x_s = x_r D / z_r
\]

\[
y_s = y_r D / z_r
\]

\[
z_s = D
\]

\[
w_s = 1
\]

### Taxonomy of Projections

- **Perspective**
  - Size varies inversely with distance - looks realistic
  - Distance and angles are not (in general) preserved
  - Parallel lines do not (in general) remain parallel

- **Parallel**
  - Good for exact measurements
  - Parallel lines remain parallel
  - Angles are not (in general) preserved
  - Less realistic looking

### Classical Projections

- **Front elevation**
- **Elevation oblique**
- **Rise oblique**

- **Isometric**
- **One-point perspective**
- **Three-point perspective**

### Perspective vs. Parallel

- **Perspective projection**
  - Size varies inversely with distance - looks realistic
  - Distance and angles are not (in general) preserved
  - Parallel lines do not (in general) remain parallel

- **Parallel projection**
  - Good for exact measurements
  - Parallel lines remain parallel
  - Angles are not (in general) preserved
  - Less realistic looking
Summary

- Camera transformation
  - Map 3D world coordinates to 3D camera coordinates
  - Matrix has camera vectors as rows

- Projection transformation
  - Map 3D camera coordinates to 2D screen coordinates
  - Two types of projections:
    - Parallel
    - Perspective

What’s next?

3D Geometric Primitives

- Transform into 3D world coordinate system
- Transform into 3D camera coordinate system
- Transform into 2D camera coordinate system
- Clip primitives outside camera’s view
- Draw pixels (includes texturing, hidden surface, etc.)