3D Polygon Rendering

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

Ray Casting Revisited

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

More efficient algorithms utilize spatial coherence!
This is a pipelined sequence of operations to draw a 3D primitive into a 2D image.

OpenGL executes steps of 3D rendering pipeline for each polygon.
3D Rendering Pipeline (for direct illumination)

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

Transformation
- Transform into 3D camera coordinate system
- Transform into 2D camera coordinate system

Viewing
- Clip primitives outside camera’s view
- Draw pixels (includes texturing, hidden surface, ...)

Lighting
- Illuminate according to lighting and reflectance

Clipping
- Clip primitives outside camera’s view

Scan Conversion
- Clip primitives outside camera’s view

Image
- Draw pixels (includes texturing, hidden surface, ...)

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.)

Transformations map points from one coordinate system to another

3D Object Coordinates

3D World Coordinates

3D Camera Coordinates

2D Screen Coordinates

2D Image Coordinates

Viewing Transformations

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

p(x,y,z)

p'(x',y')

3D Object Coordinates

3D World Coordinates

3D Camera Coordinates

2D Screen Coordinates

2D Image Coordinates

p(x,y,z)

p'(x',y')
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

- 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

- 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

- Planar geometric projections
  - Parallel
  - Perspective
- Orthographic
- Oblique
- One-point
- Two-point
- Three-point
- Cabinet
- Isometric
**Taxonomy of Projections**

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

- **Orthographic Projections**
  - DOP perpendicular to view plane

- **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
- How many vanishing points?

Perspective Projection View Volume

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?

\[
\begin{bmatrix}
    x & y & z & w
\end{bmatrix}
= \begin{bmatrix}
    x' & y' & z' & 1
\end{bmatrix}
\]

\[
\begin{bmatrix}
    x' \overline{x} & y' \overline{y} & z' \overline{z} & 1
\end{bmatrix}
= \begin{bmatrix}
    x & y & z & w
\end{bmatrix}
\]

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

Taxonomy of Projections

Classical Projections

- 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

Modeling Transformation → Lighting
Viewing Transformation → Projection Transformation → Clipping
Scan Conversion → Image

Transform into 3D world coordinate system
 Illuminate according to lighting and reflectance
 Transform into 3D camera coordinate system
 Transform into 2D camera coordinate system
 Clipping primitives outside camera’s view
 Draw pixels (includes texturing, hidden surface, etc.)