3D Polygon Rendering Pipeline

Michael Kazhdan

(601.457/657)

HB Ch. 12
FvDFH Ch. 6, 18.3
3D Polygon Rendering

• Many applications use (interactive) rendering of 3D polygons with direct illumination
3D Polygon Rendering

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

God of War
(Santa Monica Studio, 2018)
Ray Casting

• For each sample:
  - Construct ray from the camera into the scene
  - Find first surface intersected by ray through pixel
  - Compute color of sample based on surface radiance
  ↓
  - Send 2D pixels into the scene and get color
3D Polygon Rendering

- For each primitive:
  - Send 3D points to the camera and set the pixel color
3D Rendering Pipeline (for direct illumination)

3D Primitives
- Modeling Transformation
- Viewing Transformation
- Lighting
- Projection Transformation
- Clipping
- Viewport Transformation
- Scan Conversion

Image

3D Model

2D Image
3D Rendering Pipeline (for direct illumination)

- **3D Primitives**
- **Modeling Transformation**
- **Viewing Transformation**
- **Lighting**
- **Projection Transformation**
- **Clipping**
- **Viewport Transformation**
- **Scan Conversion**
- **Image**

Transform into 3D world coordinate system
3D Rendering Pipeline (for direct illumination)

- 3D Primitives
  - Modeling Transformation
  - Viewing Transformation
  - Lighting
  - Projection Transformation
  - Clipping
  - Viewport Transformation
  - Scan Conversion

Transform into 3D world coordinate system

Transform into 3D camera coordinate system
3D Rendering Pipeline (for direct illumination)

3D Primitives

Modeling Transformation

Viewing Transformation

Lighting

Transform into 3D world coordinate system

Transform into 3D camera coordinate system

Illuminate vertices using lighting and reflectance

Projection Transformation

Clipping

Viewport Transformation

Scan Conversion

Image
3D Rendering Pipeline (for direct illumination)

1. 3D Primitives
2. Modeling Transformation
3. Viewing Transformation
4. Lighting
5. Projection Transformation
6. Clipping
7. Viewport Transformation
8. Scan Conversion
9. Image

- Transform into 3D world coordinate system
- Transform into 3D camera coordinate system
- Illuminate vertices using lighting and reflectance
- Transform into 2D camera coordinate system
3D Rendering Pipeline (for direct illumination)

3D Primitives

- Modeling Transformation
- Viewing Transformation
- Lighting
- Projection Transformation
- Clipping
- Viewport Transformation
- Scan Conversion

Transform into 3D world coordinate system
Transform into 3D camera coordinate system
Illuminate vertices using lighting and reflectance
Transform into 2D camera coordinate system
Clip (parts of) primitives outside camera’s view

Image
3D Rendering Pipeline (for direct illumination)

3D Primitives

- Modeling Transformation
- Viewing Transformation
- Lighting
- Projection Transformation
- Clipping
- Viewport Transformation
- Scan Conversion

Transform into 3D world coordinate system
Transform into 3D camera coordinate system
Illuminate vertices using lighting and reflectance
Transform into 2D camera coordinate system
Clip (parts of) primitives outside camera’s view
Transform into 2D image coordinate system
3D Rendering Pipeline (for direct illumination)

- **3D Primitives**
- **Modeling Transformation**
- **Viewing Transformation**
- **Lighting**
- **Projection Transformation**
- **Clipping**
- **Viewport Transformation**
- **Scan Conversion**

Transform into 3D world coordinate system

Transform into 3D camera coordinate system

Illuminate vertices using lighting and reflectance

Transform into 2D camera coordinate system

Clip (parts of) primitives outside camera’s view

Transform into 2D image coordinate system

Draw pixels (includes texturing, hidden surface, etc.)
Transformations

3D Primitives

- **Modeling Transformation**
  - **Transform** into 3D world coordinate system

- **Viewing Transformation**
  - **Transform** into 3D camera coordinate system
  - Illuminate vertices using lighting and reflectance

- **Projection Transformation**
  - **Transform** into 2D camera coordinate system
  - Clip (parts of) primitives outside camera’s view

- **Viewport Transformation**
  - **Transform** into 2D image coordinate system
  - Draw pixels (includes texturing, hidden surface, etc.)

- **Scan Conversion**
  - Image
Recall: Homogeneous Coordinates

- Add a 4\textsuperscript{th} coordinate to every 3D point
  - \((x, y, z, w)\) represents a point at location \(\left(\frac{x}{w}, \frac{y}{w}, \frac{z}{w}\right)\)
  - \((x, y, z, 0)\) represents a (directed) point at infinity
  - \((0, 0, 0, 0)\) is not allowed
Recall: 3D Transformations

- Using homogenous coordinates, we have two types of transformations:
  - Affine
    \[
    \begin{bmatrix}
    x' \\
y' \\
z' \\
1
    \end{bmatrix} = \begin{bmatrix}
a & b & c & d \\
e & f & g & h \\
i & j & k & l \\
0 & 0 & 0 & 1
    \end{bmatrix} \begin{bmatrix}
x \\
y \\
z \\
1
    \end{bmatrix}
    \]
  - Projective
    \[
    \begin{bmatrix}
x' \\
y' \\
z' \\
w'
    \end{bmatrix} = \begin{bmatrix}
a & b & c & d \\
e & f & g & h \\
i & j & k & l \\
m & n & o & p
    \end{bmatrix} \begin{bmatrix}
x \\
y \\
z \\
w
    \end{bmatrix}
    \]
Transformations

\[(x, y, z)\] 
- Modeling Transformation
  \[\rightarrow\] 3D Object Coordinates
- Viewing Transformation
  \[\rightarrow\] 3D World Coordinates
- Projection Transformation
  \[\rightarrow\] 3D Camera Coordinates
- Window-to-Viewport Transformation
  \[\rightarrow\] 2D Screen Coordinates
  \[\rightarrow\] 2D Image Coordinates
  \[(x', y')\]

Transformations map points from one coordinate system to another.
Transformations

\[(x, y, z)\]

- **Modeling Transformation**
  - 3D Object Coordinates

- **Viewing Transformation**
  - 3D World Coordinates
  - 3D Camera Coordinates

- **Projection Transformation**
  - 2D Screen Coordinates

- **Window-to-Viewport Transformation**
  - 2D Image Coordinates

\[(x', y')\]
Transformations

$\begin{pmatrix} x, y, z \end{pmatrix}$

- 3D Object Coordinates
  - Modeling Transformation
    - 3D World Coordinates
      - Viewing Transformation
        - 3D Camera Coordinates
          - Projection Transformation
            - 2D Screen Coordinates
              - Window-to-Viewport Transformation
                - 2D Image Coordinates
                  - $(x', y')$
Viewing Transformation

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

- The transformation, $T_{W \rightarrow C}$, taking us from world coordinates to camera coordinates should map:
  - The right vector to the $x$-axis:
    $$(R_x, R_y, R_z, 0) \rightarrow (1,0,0,0)$$
  - The up vector to the $y$-axis:
    $$(U_x, U_y, U_z, 0) \rightarrow (0,1,0,0)$$
  - The back vector to the $z$-axis:
    $$(B_x, B_y, B_z, 0) \rightarrow (0,0,1,0)$$
  - The eye position to the origin:
    $$(E_x, E_y, E_z, 1) \rightarrow (0,0,0,1)$$

How should we define this transformation/matrix?
Viewing Transformation

- Consider the inverse transformation, $T_{C\rightarrow W}$, taking us from camera coordinates to world coordinates:

  $\begin{align*}
  (R_x, R_y, R_z, 0) &\leftarrow (1,0,0,0) \\
  (U_x, U_y, U_z, 0) &\leftarrow (0,1,0,0) \\
  (B_x, B_y, B_z, 0) &\leftarrow (0,0,1,0) \\
  (E_x, E_y, E_z, 1) &\leftarrow (0,0,0,1)
  \end{align*}$

- This is described by the matrix:

  $$
  \begin{pmatrix}
  x^w \\
  y^w \\
  z^w \\
  1
  \end{pmatrix}
  =
  \begin{pmatrix}
  R_x & U_x & B_x & E_x \\
  R_y & U_y & B_y & E_y \\
  R_z & U_z & B_z & E_z \\
  0 & 0 & 0 & 1
  \end{pmatrix}
  \begin{pmatrix}
  x^c \\
  y^c \\
  z^c \\
  1
  \end{pmatrix}
  $$

  $T_{C\rightarrow W}$

Finding the Viewing Transformation

- The camera-to-world matrix:
  \[
  \begin{pmatrix}
    x^w \\
    y^w \\
    z^w \\
    1
  \end{pmatrix} =
  \begin{pmatrix}
    R_x & U_x & B_x & E_x \\
    R_y & U_y & B_y & E_y \\
    R_z & U_z & B_z & E_z \\
    0 & 0 & 0 & 1
  \end{pmatrix}
  \begin{pmatrix}
    x^c \\
    y^c \\
    z^c \\
    1
  \end{pmatrix}
  \]

- The world-to-camera matrix is its inverse:
  \[
  \begin{pmatrix}
    x^c \\
    y^c \\
    z^c \\
    1
  \end{pmatrix} =
  \begin{pmatrix}
    R_x & U_x & B_x & E_x \\
    R_y & U_y & B_y & E_y \\
    R_z & U_z & B_z & E_z \\
    0 & 0 & 0 & 1
  \end{pmatrix}^{-1}
  \begin{pmatrix}
    x^w \\
    y^w \\
    z^w \\
    1
  \end{pmatrix}
  \]

\[T_{W \rightarrow C} = T_{C \rightarrow W}^{-1}\]
Transformations

$(x, y, z)$

- **Modeling Transformation**
  - 3D Object Coordinates

- **Viewing Transformation**
  - 3D World Coordinates

- **Projection Transformation**
  - 3D Camera Coordinates

- **Window-to-Viewport Transformation**
  - 2D Screen Coordinates

- 2D Image Coordinates
  - $(x', y')$
Projection

• General definition:
  ◦ A linear transformation of 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

Orthographic

Top (plan)

Front elevation

Side elevation

Axonometric

Oblique

Cabinet

Cavalier

One-point

Two-point

Three-point

Perspective

Other

Other
Projection

- Two general classes of projections, both of which shoot rays from the scene, through the view plane:
  - Parallel Projection:
    » Rays converge at a point at infinity and are parallel
  - Perspective “Projection”:
    » Rays converge at a finite point, giving rise to perspective distortion
Taxonomy of Projections

Planar geometric projections

Parallel

Orthographic
  - Top (plan)
  - Front elevation
  - Side elevation

Axonometric
  - Isometric

Oblique
  - Cabinet
    - Cavalier

One-point
  - Two-point
  - Three-point

Perspective
  - Other
Parallel Projection

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

✓ Parallel lines remain parallel
✓ Relative proportions of objects preserved
✗ Angles are not preserved
✗ Less realistic looking
Taxonomy of Projections

Planar geometric projections

Parallel

Orthographic

Top (plan)

Front elevation

Axonometric

Side elevation

Isometric

Oblique

Cabinet

Cavalier

One-point

Two-point

Three-point

Perspective

Other

Other
Orthographic Projections

- DoP perpendicular to view plane

Angel Figure 5.5
Orthographic Projections

- DoP perpendicular to view plane

- Lines perpendicular to the view plane vanish
- Faces parallel to the view plane are un-distorted.
Orthographic Projections

- DoP perpendicular to view plane
  - Maps a point in 3D space to the $(x, y)$-plane, through the origin, by projecting out the $z$-component:
    
    $$(x^c, y^c, z^c) \rightarrow (x^c, y^c, 0)$$

  - In terms of the matrix representation:
    
    $$\begin{bmatrix}
    x^s \\
    y^s \\
    0
    \end{bmatrix} = \begin{bmatrix}
    1 & 0 & 0 \\
    0 & 1 & 0 \\
    0 & 0 & 0
    \end{bmatrix} \begin{bmatrix}
    x^c \\
    y^c \\
    z^c
    \end{bmatrix}$$

    $$(x^c, y^c, z^c)$$

    $(0,0,0)$
Orthographic Projections

- DoP perpendicular to view plane
  - Maps a point in 3D space to the \((x, y)\)-plane, through the origin, by projecting out the \(z\)-component: 
    \((x^c, y^c, z^c) \rightarrow (x^c, y^c, 0)\)
  - In terms of the matrix representation:
    \[
    \begin{bmatrix} x^s \\ y^s \\ 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x^c \\ y^c \\ z^c \end{bmatrix}
    \]
  - Or, in homogenous coordinates:
    \[
    \begin{bmatrix} x^s \\ y^s \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x^c \\ y^c \\ z^c \\ 1 \end{bmatrix}
    \]
Taxonomy of Projections

Planar geometric projections

Parallel
- Orthographic
  - Top (plan)
  - Front elevation
  - Side elevation
- Axonometric
- Isometric

Oblique
- Cabinet
- Cavalier
- Other

Perspective
- One-point
- Two-point
- Three-point
Oblique Projections

- DoP not perpendicular to view plane

- $\phi$ is the angle of the projection of the view plane’s normal
- $L$ is the scale factor applied to the view plane’s normal

H&B Figure 12.21
Parallel Projection Matrix

- General parallel projection transformation:

\[
\begin{bmatrix}
  x^s \\
y^s \\
0 \\
1
\end{bmatrix} =
\begin{bmatrix}
  1 & 0 & L \cos \phi & 0 \\
  0 & 1 & L \sin \phi & 0 \\
  0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x^c \\
y^c \\
z^c \\
1
\end{bmatrix}
\]

Cavalier
(DoP \(\alpha = 45^\circ\))

\[
\phi = 45^\circ
\quad L = 1
\]

Cabinet
(DoP \(\alpha = 63.4^\circ\))

\[
\phi = 45^\circ
\quad L = 1/2
\]

H&B Figure 12.21
Parallel Projection Matrix

• General parallel projection transformation:

\[
\begin{bmatrix}
    x^s \\
    y^s \\
    0 \\
    1
\end{bmatrix} =
\begin{bmatrix}
    1 & 0 & L \cos \phi & 0 \\
    0 & 1 & L \sin \phi & 0 \\
    0 & 0 & 0 & 0 \\
    0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
    x^c \\
    y^c \\
    z^c \\
    1
\end{bmatrix}
\]

Note:
This matrix represents an affine transformation.

H&B Figure 12.21
Parallel Projection View Volume

H&B Figure 12.30
Taxonomy of Projections

Planar geometric projections

Parallel

Orthographic
- Top (plan)
- Front elevation
- Side elevation

Axonometric
- Isometric

Oblique
- Cabinet
- Cavalier

One-point

Two-point

Three-point

Perspective

Other
Perspective “Projection”

- Map points onto “view plane” along “projectors” emanating from “center of projection” (CoP)
Perspective Projection

- How many vanishing points?

Number of vanishing points determined by number of axes parallel to the view plane

Angel Figure 5.10
Perspective Projection

- Not all parallel lines remain parallel!
Perspective Projection

- What are the coordinates of the point resulting from projection of \((x^c, y^c, z^c)\) onto the view plane a unit distance along the \(z\)-axis?
Perspective Projection

• For any point \((x^c, y^c, z^c)\) and any scalar \(\alpha\), the points \((x^c, y^c, z^c)\) and \((\alpha x^c, \alpha y^c, \alpha z^c)\) map to the same location.
Perspective Projection

• For any point \((x^c, y^c, z^c)\) and any scalar \(\alpha\), the points \((x^c, y^c, z^c)\) and \((\alpha x^c, \alpha y^c, \alpha z^c)\) map to the same location.

• Since we want the position on the view plane that intersect the line from \((x^c, y^c, z^c)\) to the origin:

\[
(x^c, y^c, z^c) \rightarrow \left(\frac{x^c}{z^c}, \frac{y^c}{z^c}, 1\right)
\]
Perspective Projection Matrix

\[(x^c, y^c, z^c) \rightarrow \left(\frac{x^c}{z^c}, \frac{y^c}{z^c}, 1\right)\]

We can’t represent this with a $3 \times 3$ matrix!

With homogenous coordinates, we can write this as:

\[(x^c, y^c, z^c, 1) \rightarrow \left(\frac{x^c}{z^c}, \frac{y^c}{z^c}, 1,1\right) \equiv (x^c, y^c, z^c, z^c)\]

In matrix form, this gives:

\[
\begin{bmatrix}
    x^s \\
    y^s \\
    1
\end{bmatrix}
\equiv
\begin{bmatrix}
    1 & 0 & 0 & 0 \\
    0 & 1 & 0 & 0 \\
    0 & 0 & 1 & 0 \\
    0 & 0 & 1 & 1
\end{bmatrix}
\begin{bmatrix}
    x^c \\
    y^c \\
    z^c \\
    1
\end{bmatrix}
\]
Perspective Projection Matrix

\[(x^c, y^c, z^c) \rightarrow \left( \frac{x^c}{z^c}, \frac{y^c}{z^c}, 1 \right)\]

We can’t represent this with a 3 × 3 matrix!

With homogenous coordinates, we can write this as:

\[(x^c, y^c, z^c, 1) \rightarrow \left( \frac{x^c}{z^c}, \frac{y^c}{z^c}, 1,1 \right) \equiv (x^c, y^c, z^c, z^c)\]

In matrix form, this gives:

\[
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
x^s \\
y^s \\
1 \\
1
\end{bmatrix}
=\begin{bmatrix}
x^c \\
y^c \\
z^c \\
1
\end{bmatrix}
\]

Note:
This matrix represents a projective transformation

\[
\begin{bmatrix}
x^c \\
y^c \\
z^c \\
1
\end{bmatrix}
= \begin{bmatrix}
x^s \\
y^s \\
1 \\
1
\end{bmatrix}
\]
Perspective Projection View Volume

H&B Figure 12.30
Taxonomy of Projections

Planar geometric projections

Parallel

Orthographic
- Top (plan)
- Front elevation
- Axonometric
- Side elevation
- Isometric

Oblique
- Cabinet
- Cavalier

One-point

Perspective

Two-point

Three-point

Other
Classical Projections

- Front elevation
- Elevation oblique
- Plan oblique
- Isometric
- One-point perspective
- Three-point perspective

Angel Figure 5.3
Perspective vs. Parallel

• Perspective projection
  ✓ Size varies inversely with distance - looks realistic
  ✓ Angles are preserved on faces parallel to the view plane
  ✗ Distance are not preserved

• Parallel projection
  ✓ Good for exact measurements
  ✓ Parallel lines remain parallel
  ✓ Angles and distance are preserved on faces parallel to the view plane
  ✗ Less realistic looking
Transformations

\((x, y, z)\)

3D Object Coordinates

Modeling Transformation

3D World Coordinates

Viewing Transformation

3D Camera Coordinates

Projection Transformation

2D Screen Coordinates

Window-to-Viewport Transformation

2D Image Coordinates

\((x', y')\)

Window

\((w_x, w_y)\)

Viewport

\((v_x, v_y)\)

Screen Coordinates

Image Coordinates

\[ V = \text{viewport transform} \]

\[
V = \begin{bmatrix}
1 & 0 & v_x^1 \\
0 & 1 & v_y^1 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
v_x^2 - v_x^1 \\
v_y^2 - v_y^1 \\
0
\end{bmatrix}
\begin{bmatrix}
1 & 0 & -w_x^1 \\
0 & 1 & -w_y^1 \\
0 & 0 & 1
\end{bmatrix}
\]

Note that this may scale non-uniformly.
3D Rendering Pipeline (for direct illumination)

\((x, y, z)\)

- 3D Object Coordinates
  - Modeling Transformation
  - 3D World Coordinates
    - Viewing Transformation
      - 3D Camera Coordinates
        - Projection Transformation
          - 2D Screen Coordinates
            - Window-to-Viewport Transformation
              - 2D Image Coordinates
                - \((x', y')\)

3D Model

2D Screen
Transformations

\[ I = I_E + \sum L \left[ K_A \cdot I_L^A + \left( K_D \cdot \langle \vec{N}, \vec{L} \rangle + K_S \cdot \langle \vec{V}, \vec{R^n} \rangle \right) \cdot I_L \right] \]
Transformations

- 3D Primitives
- Modeling Transformation
- Viewing Transformation
- Lighting
- Projection Transformation
- Clipping
- Viewport Transformation
- Scan Conversion
- Image
- 3D Model
- 2D Screen

Vertex processing

- Originally, vertex processing was fixed
- On modern cards this can be programmed in the vertex shader