Transformations: Viewing & Perspective

Chapter 5

Viewing/Perspective Transformations

Rendering Pipeline

- Specify view point (change of coordinate system)
- Project from 3D to 2D (introduce perspective)

Scene graph
Object geometry
- Modelling Transforms
- Viewing Transform
- Projection Transform

Result
- all vertices of scene in shared 3D world coordinate system

Scene graph
Object geometry
- Modelling Transforms
- Viewing Transform
- Projection Transform

Result
- 2D screen coordinates of clipped vertices

Copyright 2013, A. Sheffer, UBC
Transformations:
Viewing & Perspective

Camera Description/Motion
- arbitrary viewing position
  - eye point, gaze/lookat point, up vector

Deriving W2V Transformation
- \( M = RT \)
- \( t = u \)
- \( g = \text{view} \)
- \( e = \text{eye} \)

OpenGL Viewing Transformation
- \( \text{gluLookAt}(ex, ey, ez, lx, ly, lz, ux, uy, uz) \)
  - postmultiplies current matrix, so to be safe:
    - \( \text{glMatrixMode}(\text{GL_MODELVIEW}); \)
    - \( \text{glLoadIdentity}(); \)
    - \( \text{gluLookAt}(ex, ey, ez, lx, ly, lz, ux, uy, uz) \)
      // now ok to do model transformations
Computer Graphics

Transformations: Viewing & Perspective

Projective Rendering Pipeline

- OCS - object coordinate system
- WCS - world coordinate system
- VCS - viewing coordinate system
- NDCS - normalized device coordinate system
- DCS - device coordinate system

Projection Transformations

- glutInitWindowSize(x,y)
- glVertex3f(x,y,z)
- glColor3f(r,g,b)
- glOrtho(xmin,xmax,ymin,ymax,near,far)
- glFrustum(xmin,xmax,ymin,ymax,near,far)

Clipping: View Volumes

- viewport transformation
- NDCS

Understanding Z

- why near and far plane?
  - near plane:
    - avoid singularity for perspective projection (division by zero, or very small numbers)
  - far plane:
    - store depth in fixed-point representation (integer), thus have to have fixed range of values (0...1)
    - avoid/reduce numerical precision artifacts for distant objects

Orthographic Derivation

- scale, translate, reflect for new coord sys
  - near plane:
    - avoid singularity for perspective projection (division by zero, or very small numbers)
  - far plane:
    - store depth in fixed-point representation (integer), thus have to have fixed range of values (0...1)
    - avoid/reduce numerical precision artifacts for distant objects

Copyright 2013, A. Sheffer, UBC
Orthographic Derivation

- scale, translate, reflect for new coord sys

\[
P' = \begin{bmatrix}
\frac{2}{\text{right} - \text{left}} & 0 & 0 & \frac{\text{right} + \text{left}}{\text{right} - \text{left}} \\
0 & \frac{2}{\text{top} - \text{bot}} & 0 & \frac{\text{top} + \text{bot}}{\text{top} - \text{bot}} \\
0 & 0 & \frac{-2}{\text{far} - \text{near}} & \frac{\text{far} + \text{near}}{\text{far} - \text{near}} \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

Orthographic OpenGL

- \text{glMatrixMode}(\text{GL_PROJECTION});
- \text{glLoadIdentity();}
- \text{glOrtho}(\text{left}, \text{right}, \text{bot}, \text{top}, \text{near}, \text{far});

NDC to Viewport Transformation

- generate pixel coordinates
- map x, y from range -1...1 (NDC) to pixel coordinates on the display
- involves 2D scaling and translation

\[
\begin{bmatrix}
y' \\
x'
\end{bmatrix} = \begin{bmatrix}
y - (y_{\text{bot}}) \\
x - (x_{\text{left}})
\end{bmatrix}
\]

OpenGL
- \text{glViewport}(x, y, a, b):

Origin Location

- yet more possibly confusing conventions
  - OpenGL: lower left
  - most window systems: upper left
  - often have to flip your y coordinates
    - when interpreting mouse position

Perspective Projection

- Viewing is from point at finite distance
- Without loss of generality:
  - Viewpoint at origin
  - Viewing (near) plane is \(z=n\)
- Given \(P=(x,y,z)\) triangle similarity gives:

\[
\frac{x}{z} = \frac{x'}{n} \quad \text{and} \quad \frac{y}{z} = \frac{y'}{n} \implies x' = \frac{x}{z/n} \quad \text{and} \quad y' = \frac{y}{z/n}
\]

Perspective Projection (cont’d)

- In matrix notation with homogeneous coordinates:

\[
P(x,y,z) = \begin{bmatrix}
1 & 0 & 0 & x \\
0 & 1 & 0 & y \\
0 & 0 & 1 & z \\
0 & 0 & 1/n & 1
\end{bmatrix}
\]

- In Euclidean coordinates:

\[
\begin{bmatrix}
x \\
y \\
z \\
z/n
\end{bmatrix} = \begin{bmatrix}
x/z/n \\
y/z/n \\
z/n \\
z/n
\end{bmatrix}
\]

- \(P\) singular: \(det(P)=0\)
Computer Graphics

Transformations:
Viewing & Perspective

Perspective Projection (cont’d)
- What is (if any) is the difference between:
  - Moving projection plane
  - Moving viewpoint (center of projection)?

Perspective Projection
- Have both near and far planes
- Transformation well defined in-between
- Conversion to device coordinates
  - Warp view frustum to box

Alternative Formulation
- Before
- After (cancel division by n)

How to make non-degenerate?
- z’ monotonically increasing function of z

Introduce Far Plane
- Matrix formulation
- Preserves relative depth (third coordinate)

Perspective Derivation

Copyright 2013, A. Sheffer, UBC
Perspective Derivation (full)

- Solve linear system to get A-F
  - 6 planes, 6 unknowns

\[
\begin{bmatrix}
x' \\
y' \\
z' \\
w'
\end{bmatrix}
= \begin{bmatrix}
E & 0 & A & 0 \\
0 & F & B & 0 \\
0 & 0 & C & D \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
x \\
y \\
z \\
w
\end{bmatrix}
\]

\[
\begin{bmatrix}
2n & 0 & r + l & 0 \\
r - l & 0 & r - l & 0 \\
0 & 2n & t + b & 0 \\
0 & 0 & - (f + n) & - 2fn \\
0 & 0 & f - n & f - n \\
0 & 0 & - 1 & 0
\end{bmatrix}
\]

Perspective OpenGL

- glMatrixMode(GL_PROJECTION);
- glLoadIdentity();
- glFrustum(left, right, bot, top, near, far);
- or
- glPerspective(fov, aspect, near, far);

- Symmetric version using field-of-view angles
  - In x-direction (fov)
  - In y-direction (fovy) given by aspect ratio

Another Transformations Quiz

- What does each transformation preserve?

<table>
<thead>
<tr>
<th></th>
<th>straight lines</th>
<th>parallel lines</th>
<th>distance</th>
<th>angles</th>
<th>normals</th>
</tr>
</thead>
<tbody>
<tr>
<td>scaling</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>rotation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>translation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>perspective</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Copyright 2013, A. Sheffer, UBC