Composite transformations,
General rotation

Objectives

• Composite transformations
  - Rotations
  - Translation
  - Scaling
• Rotations in general - define more flexible rotation specifications
Review of Transformations

• Homogeneous coordinates (review)
  - 4X4 matrix used to represent translation, scaling, and rotation
  
  - a point in the space is represented as \( \mathbf{P} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \)
  
  - Treat all transformations the same so that they can be easily combined

Shear

• Helpful to add one more basic transformation
• Equivalent to pulling faces in opposite directions
Shear Matrix

Consider simple shear along $x$ axis

\[ x' = x + y \times \text{sh}_y \]
\[ y' = y \]
\[ z' = z \]

\[
\begin{bmatrix}
1 & \text{sh}_y & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

\[ H(\theta) = \]

Review of Transformations

- Homogeneous coordinates (review)
  - 4X4 matrix used to represent translation, scaling, and rotation
  - a point in the space is represented as $P = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$
  - Treat all transformations the same so that they can be easily combined
Rotation about an arbitrary axis

Rotating about an axis by theta degrees

- Rotate about x to bring axis to xz plane
- Rotate about y to align axis with z-axis
- Rotate theta degrees about z
- Unrotate about y, unrotate about x

\[ M = R_x^{-1} \ R_y^{-1} \ R_z(\theta) \ R_y \ R_x \]

- Can you determine the values of Rx and Ry?
Interpolation

• In order to “move things”, we need both translation and rotation
• Interpolating the translation is easy, but what about rotations?

Interpolation of orientation

• How about interpolating each entry of the rotation matrix?
• The interpolated matrix might no longer be orthonormal, leading to nonsense for the in-between rotations
Interpolation of orientation

Example: interpolate linearly from a positive 90 degree rotation about y axis to a negative 90 degree rotation about y

\[
\begin{bmatrix}
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 \\
-1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\rightarrow
\begin{bmatrix}
0 & 0 & -1 & 0 \\
0 & 1 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

Linearly interpolate each component and halfway between, you get this...

\[
\begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

Interpolating Rotation

- Rotation can take on many representations
  - Rotation Matrix (9 elements)
  - Euler angles (3 elements)
  - Axis-angle (3 elements)
  - Quaternions (4 elements)
General Rotation

A rotation by $\theta$ about an arbitrary axis can be decomposed into the concatenation of rotations about the $x$, $y$, and $z$ axes

$$ R = R_z(\theta_z) R_y(\theta_y) R_x(\theta_x) $$

$\theta_x, \theta_y, \theta_z$ are called the Euler angles

Note that rotations are not commutative, can use rotations in another order but need different angles

Euler Angles

- A general rotation is a combination of three elementary rotations: around the $x$-axis ($x$-roll), around the $y$-axis ($y$-pitch) and around the $z$-axis ($z$-yaw).
Euler angles interpolation

\[ R(0,0,0), \ldots, R(\pi t, 0, 0), \ldots, R(\pi, 0, 0) \]
\[ t \in [0,1] \]

Euler Angles Interpolation
\[ \Rightarrow \text{Unnatural movement!} \]
Gimbal Lock

- Phenomenon of two rotational axis of an object pointing in the same direction.
- Result: Lose a degree of freedom (DOF)

Summary

- Euler angles
  + intuitive
  + few (3) elements
  - not unique
  - gimbal lock
Interpolating Rotation

• Consider the two approaches
  - For a sequence of rotation matrices $R_0, R_1, \ldots, R_n$, find the Euler angles for each and use $R_i = R_{iz} R_{iy} R_{ix}$
    *Not unique!*
  
  - Use the extreme positions and determine the fixed axis and angle of rotation, then increment only the angle
    *Does this rotation make sense?*

• Quaternions often used over either

Quaternions

• Extension of imaginary numbers

• Requires one real and three imaginary components $i, j, k$
  \[ q = q_0 + q_1 i + q_2 j + q_3 k \]
Quaternions

• Extension of imaginary numbers

• Requires one real and three imaginary components $i, j, k$

\[ q = (w, x, y, z) = \cos \frac{\theta}{2} + \mathbf{v} \sin \frac{\theta}{2} \]

\[ \mathbf{v} = [x, y, z] \text{ of axis} \]

Quaternion interpolation

• Quaternions can express rotations on unit sphere smoothly and efficiently.

1-angle rotation can be represented by a unit circle

2-angle rotation can be represented by a unit sphere

• Interpolation means moving on n-D sphere
• Now imagine a 4-D sphere for 3-angle rotation
Quaternion interpolation

- Moving between two points on the 4D unit sphere
  - a unit quaternion at each step - another point on the 4D unit sphere
  - move along the great circle between the two points on the 4D unit sphere as an arc

Spherical linear interpolation (SLERP)

\[
slerp(q_1, q_2, u) = q_1 \frac{\sin((1-u)\theta)}{\sin \theta} + q_2 \frac{\sin(u\theta)}{\sin \theta}
\]

Process:
- Rotation matrix \(\rightarrow\) quaternion
- Carry out slerp/operations with quaternions
- Quaternion \(\rightarrow\) rotation matrix
Summary
Quaternions – points on a 4D unit hypersphere
+ better interpolation
+ almost unique
- less intuitive

Rotations in Reality
• It’s easiest to express rotations in Euler angles or Axis/angle

• We can convert to/from any of these representations

• Choose the best representation for the task
  - input: Euler angles
  - interpolation: quaternions
  - composing rotations: quaternions, orientation matrix
Introduction to Viewing
Objectives

• Introduce viewing
• Compare and contrast image formation by computer with how images have been formed by architects, artists, and engineers

Classical Viewing

• Viewing requires three basic elements
  - Object(s) to be viewed
  - Projection surface (image plane)
  - Projectors: lines that go from the object(s) to the projection surface
• Classical views are based on the relationship among these elements
  - Must orient the object as it should be viewed
• Object assumed to be constructed from flat principal faces
  - Buildings, polyhedra, manufactured objects
Classical Viewing

- Projectors are lines that either
  - converge at a center of projection (COP)
  - or are parallel
- Standard projections project onto a plane
- Such projections preserve lines
  - but not necessarily angles
- Nonplanar projections are needed for applications such as map construction

Classical Projections

- Front elevation
- Elevation oblique
- Plan oblique
- Isometric
- One-point perspective
- Three-point perspective
Parallel vs Perspective

- Computer graphics treats all projections the same and implements them with a single pipeline
- Classical viewing developed different techniques for drawing different types of projections
- Fundamental distinction is between parallel and perspective viewing (even though *mathematically* parallel viewing is the limit of perspective viewing)

Parallel Projection
Perspective Projection

Object

Projector

Projection plane

COP

Taxonomy of Planar Geometric Projections

planar geometric projections

parallel perspective

multiview orthographic

axonometric oblique

isometric dimetric trimetric

1 point 2 point 3 point
Orthographic Projection

Projectors are orthogonal to projection surface

Advantages and Disadvantages

• Preserves both distance and angle
  - Shapes are preserved
  - Can be used for measurements
    - Building plans
    - Manufactured parts
• Cannot see what object really looks like because many surfaces hidden from view
  - Often we add the isometric
Multiview Orthographic Projection

- Projection plane parallel to principal face
- Usually form front, top, side views

isometric (not orthographic)

in CAD and architecture, we often display three multiviews plus isometric

Axonometric Projections

Move the projection plane relative to object

classify by how many angles of a corner of a projected cube are the equal to each other

three: isometric
two: dimetric
none: trimetric
Types of Axonometric Projections

- Isometric
- Dimetric
- Trimetric

Advantages and Disadvantages

- Used in CAD applications
- Parallel Lines preserved but angles are not
- Can see three principal faces of a box-like object
- Some optical illusions possible
  - Parallel lines appear to diverge
- Does not look real because far objects are scaled the same as near objects
**Oblique Projection**

Arbitrary relationship between projectors and projection plane

- Angles emphasize a particular face
  - Architecture: plan oblique, elevation oblique

- Angles in faces parallel to projection plane are preserved while we can still see “around” side

- No physical analogue, cannot create with real-world camera

**Advantages and Disadvantages**
Perspective Projection

Projectors converge at center of projection

Vanishing Points

- Parallel lines on the object not parallel to the projection plane converge at a single point in the projection (called the \textit{vanishing point})
- Can draw simple perspectives (by hand) using the vanishing point
One-Point Perspective

• One principal face parallel to projection plane
• One vanishing point for cube

Two-Point Perspective

• On principal direction parallel to projection plane
• Two vanishing points for cube
Three-Point Perspective

• No principal face parallel to projection plane
• Three vanishing points for cube

Advantages and Disadvantages

• Objects further from viewer are projected smaller than the same sized objects closer to the viewer
  - Looks realistic, gives sense of depth
• Equal distances along a line are not projected into equal distances (nonuniform foreshortening)
• Angles preserved only in planes parallel to the projection plane
• More difficult to construct by hand than parallel projections (but not more difficult by computer!)
Computer Viewing

• Need to build transformation that defines the projection plane based on the chosen projection

Orthogonal Projection
Orthogonal Projection

- Set $z = 0$
- Equivalent homogeneous coordinate transformation:

$$\begin{bmatrix}
px' \\
py' \\
0 \\
1
\end{bmatrix} =
\begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
px \\
py \\
pz \\
1
\end{bmatrix}$$

Oblique Projections
Oblique Projections

Direction of projection $\text{dop} = (\text{dop}_x, \text{dop}_y, \text{dop}_z)$

Let's say we project onto $z = 0$ plane

Oblique Projections

Direction of projection $\text{dop} = (\text{dop}_x, \text{dop}_y, \text{dop}_z)$

$$
\begin{bmatrix}
px' \\
py' \\
nz
\end{bmatrix} =
\begin{bmatrix}
1 & 0 & -\text{dop}_x/\text{dop}_z & 0 \\
0 & 1 & -\text{dop}_y/\text{dop}_z & 0 \\
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
px \\
pv \\
nz
\end{bmatrix}
$$

Note: $\tan \theta = \text{dop}_z/\text{dop}_x$
Shear

- Helpful to add one more basic transformation
- Equivalent to pulling faces in opposite directions

Shear Matrix

Consider simple shear along \( x \) axis

\[
x' = x + y \sinh_y \\
y' = y \\
z' = z
\]

\[
H(\theta) = \begin{bmatrix}
1 & \sinh_y & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{bmatrix}
\]
Oblique Projection

\[ H = \begin{bmatrix} 1 & 0 & -\text{dopx}/\text{dopz} & 0 \\ 0 & 1 & -\text{dopy}/\text{dopz} & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \]

Projection matrix

\[ P = M_{\text{ortho}} H \]

Computer Viewing

There are three aspects of the viewing process:

- Selecting a lens
  Setting the projection matrix

- Positioning the camera
  Setting the view-orientation matrix

- Clipping
  Setting the view volume