Affine Transformations

**CPSC 314**

Modeling and Viewing Transformation

**Affine transformations**
- Linear transformations + translations
- Can be expressed as a 3x3 matrix + 3 vector

\[
x' = M \cdot x + t
\]

Scaling

**Scaling**
- a coordinate means multiplying each of its components by a scalar

**Uniform scaling**
- this scalar is the same for all components

Scaling (2D)

**scaling operation:**
\[
\begin{pmatrix}
x' \\
y'
\end{pmatrix} = \begin{pmatrix} ax \\ by
\end{pmatrix}
\]

**or, in matrix form:**
\[
\begin{pmatrix}
x' \\
y'
\end{pmatrix} = \begin{pmatrix} a & 0 \\ 0 & b
\end{pmatrix} \begin{pmatrix} x \\ y
\end{pmatrix}
\]

Scaling matrix

**how can we represent this in matrix form?**
Scaling (3D)

scaling operation:
\[
\begin{pmatrix}
  x' \\
  y' \\
  z'
\end{pmatrix} = \begin{pmatrix}
  ax \\
  by \\
  cz
\end{pmatrix}
\]

or, in matrix form:
\[
\begin{pmatrix}
  x' \\
  y' \\
  z'
\end{pmatrix} = \begin{pmatrix}
  a & 0 & 0 \\
  0 & b & 0 \\
  0 & 0 & c
\end{pmatrix}
\begin{pmatrix}
  x \\
  y \\
  z
\end{pmatrix}
\]

2D Rotation From Trig Identities

\[
x = r \cos(\phi) \\
y = r \sin(\phi) \\
x' = r \cos(\phi) \cos(\theta) - r \sin(\phi) \sin(\theta) \\
y' = r \sin(\phi) \cos(\theta) + r \cos(\phi) \sin(\theta)
\]

3D Rotation

- About x axis:
  \[
  \begin{pmatrix}
    x' \\
    y' \\
    z'
  \end{pmatrix} = \begin{pmatrix}
    1 & 0 & 0 \\
    0 & \cos(\theta) & -\sin(\theta) \\
    0 & \sin(\theta) & \cos(\theta)
  \end{pmatrix}
  \begin{pmatrix}
    x \\
    y \\
    z
  \end{pmatrix}
  \]

- About y axis:
  \[
  \begin{pmatrix}
    x' \\
    y' \\
    z'
  \end{pmatrix} = \begin{pmatrix}
    \cos(\theta) & 0 & \sin(\theta) \\
    0 & 1 & 0 \\
    -\sin(\theta) & 0 & \cos(\theta)
  \end{pmatrix}
  \begin{pmatrix}
    x \\
    y \\
    z
  \end{pmatrix}
  \]

- About z axis:
  \[
  \begin{pmatrix}
    x' \\
    y' \\
    z'
  \end{pmatrix} = \begin{pmatrix}
    \cos(\theta) & -\sin(\theta) & 0 \\
    \sin(\theta) & \cos(\theta) & 0 \\
    0 & 0 & 1
  \end{pmatrix}
  \begin{pmatrix}
    x \\
    y \\
    z
  \end{pmatrix}
  \]

Shear

Shear along x axis
- push points to right in proportion to height
\[
\begin{pmatrix}
  x' \\
  y'
\end{pmatrix} = \begin{pmatrix}
  x + shTy \\
  y
\end{pmatrix}
\]

Shear

Shear along x axis
- push points to right in proportion to height
\[
\begin{pmatrix}
  x' \\
  y'
\end{pmatrix} = \begin{pmatrix}
  x + shTy \\
  0 \\
  1
\end{pmatrix}
\begin{pmatrix}
  x \\
  y
\end{pmatrix}
\]
Reflection

Reflect across x axis
- Mirror

\[
\begin{pmatrix}
x' \\
y'
\end{pmatrix} = \begin{pmatrix}
? & ? \\
? & ?
\end{pmatrix} \begin{pmatrix}
x \\
y
\end{pmatrix}
\]

Affine Transformations

Translation:
- Add a constant (2D or 3D) vector:

\[
\begin{pmatrix}
x' \\
y'
\end{pmatrix} = \begin{pmatrix}
1 & 0 \\
0 & 1
\end{pmatrix} \begin{pmatrix}
x \\
y
\end{pmatrix} + \begin{pmatrix}
t_x \\
t_y
\end{pmatrix}
\]

Compositing of Linear and Affine Transformations

Example: 3D rotation around arbitrary axis
- Rotate axis to z-axis
- Rotate by \( \phi \) around z-axis
- Rotate z-axis back to original axis
- Composite transformation:

\[
R(v, \phi) = R_z^{-1}(\phi) \cdot R_z(\beta) \cdot R_x(\theta) \cdot R_z(\gamma)
\]

In general:
- Transformation of geometry into coordinate system where operation becomes simpler
- Perform operation
- Transform geometry back to original coordinate system

Compositing of Affine Transformations

Example: Rotation around arbitrary center
Compositing of Affine Transformations

Example: Rotation around arbitrary center

- Step 1: translate coordinate system to rotation center

- Step 2: perform rotation

- Step 3: back to original coordinate system

Composite transformation: Is again an affine transformation

\[ x' = \text{Id} \cdot (R(\phi) \cdot (\text{Id} \cdot x + t)) - t \]
\[ = \text{Id} \cdot (R(\phi) \cdot x + R(\phi) \cdot t) - t \]
\[ = R(\phi) \cdot x + (R(\phi) \cdot t - t) \]
\[ = R(\phi) \cdot (x + t') \]

Properties of Affine Transformations

Definition:

- A linear combination of points or vectors is given as
  \[ \mathbf{x} = \sum_{i=1}^{n} a_i \cdot \mathbf{x}_i \quad \text{for} \quad a_i \in \mathbb{R} \]

- An affine combination of points or vectors is given as
  \[ \mathbf{x} = \sum_{i=1}^{n} a_i \cdot \mathbf{x}_i \quad \text{with} \quad \sum_{i=1}^{n} a_i = 1 \]

Example:

- Affine combination of 2 points

  \[ \mathbf{x} = a_1 \cdot \mathbf{x}_1 + a_2 \cdot \mathbf{x}_2 \quad \text{with} \quad a_1 + a_2 = 1 \]
  \[ = (1 - a_2) \cdot \mathbf{x}_1 + a_2 \cdot \mathbf{x}_2 \]
  \[ = \mathbf{x}_1 + a_2 \cdot (\mathbf{x}_2 - \mathbf{x}_1) \]
**Properties of Affine Transformations**

*Definition:*
- A convex combination is an affine combination where all the weights \( a_i \) are positive
- Note: this implies \( 0 \leq a_i \leq 1, \forall i \neq n \)

*Example:*
- Convex combination of 3 points
  \[ x = \alpha \cdot x_1 + \beta \cdot x_2 + \gamma \cdot x_3 \]
  with \( \alpha + \beta + \gamma = 1, \ 0 \leq \alpha, \beta, \gamma \leq 1 \)
- \( \alpha, \beta, \gamma \) are called barycentric coordinates

**Properties of Affine Transformations**

*Theorem:*
- The following statements are synonymous
  - A transformation \( T(x) \) is affine, i.e.: \( x' = T(x) := M \cdot x + t \) for some matrix \( M \) and vector \( t \)
  - \( T(x) \) preserves affine combinations, i.e.
    \[ T(\sum a_i \cdot x_i) = \sum a_i \cdot T(x_i) \]
  - \( T(x) \) maps parallel lines to parallel lines

**Homogeneous Coordinates**

*Homogeneous representation of points:*
- Add an additional component \( w = 1 \) to all points
- All multiples of this vector are considered to represent the same 3D point
- Use square brackets (rather than round ones) to denote homogeneous coordinates (different from text book)

\[
\begin{pmatrix}
  x' \\
  y' \\
  z' \\
  w'
\end{pmatrix} =
\begin{pmatrix}
  x \\
  y \\
  z \\
  1
\end{pmatrix} \begin{pmatrix}
  x \cdot w \\
  y \cdot w \\
  z \cdot w \\
  w
\end{pmatrix} =
\begin{pmatrix}
  x' \\
  y' \\
  z' \\
  w'
\end{pmatrix} \quad \forall w \neq 0
\]
Geometrically In 2D
Homogeneous Coordinates:

\[
\begin{bmatrix}
  x' \\
  y' \\
  z \\
  1
\end{bmatrix} =
\begin{bmatrix}
  m_{11} & m_{12} & m_{13} & t_x \\
  m_{21} & m_{22} & m_{23} & t_y \\
  m_{31} & m_{32} & m_{33} & t_z \\
  0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  z \\
  1
\end{bmatrix}
\]

Homogeneous Matrices

Combining the two matrices into one:

\[
\begin{bmatrix}
  x' \\
  y' \\
  z \\
  1
\end{bmatrix} =
\begin{bmatrix}
  m_{11} & m_{12} & m_{13} & t_x \\
  m_{21} & m_{22} & m_{23} & t_y \\
  m_{31} & m_{32} & m_{33} & t_z \\
  0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  z \\
  1
\end{bmatrix}
\]

Homogeneous Matrices

Affine Transformations

\[
\begin{bmatrix}
  x' \\
  y' \\
  z \\
  1
\end{bmatrix} =
\begin{bmatrix}
  m_{11} & m_{12} & m_{13} & t_x \\
  m_{21} & m_{22} & m_{23} & t_y \\
  m_{31} & m_{32} & m_{33} & t_z \\
  0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  z \\
  1
\end{bmatrix}
\]

Homogeneous Matrices

Note:
- Multiplication of the matrix with a constant does not change the transformation!

\[
\begin{bmatrix}
  x' \\
  y' \\
  z' \\
  1
\end{bmatrix} =
\begin{bmatrix}
  m_{11} & m_{12} & m_{13} & t_x \\
  m_{21} & m_{22} & m_{23} & t_y \\
  m_{31} & m_{32} & m_{33} & t_z \\
  0 & 0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  x \\
  y \\
  z \\
  1
\end{bmatrix}
\]

Homogeneous Vectors

Earlier discussion describes points only
- What about vectors (directions)?
- What is the affine transformation of a vector?
  - Rotation
  - Scaling
  - Translation

Vectors are invariant under translation!

Homogeneous Vectors

Representing vectors in homogeneous coordinates
- Need representation that is only affected by linear transformations, but not by translations
- This is achieved by setting \( w = 0 \)
Homogeneous Coordinates

Properties
- Unified representation as 4-vector (in 3D) for
  - Points
  - Vectors / directions
- Affine transformations become 4x4 matrices
  - Composing multiple affine transformations involves simply multiplying the matrices
  - 3D affine transformations have 12 degrees of freedom
- Need mapping of 4 points to uniquely define transformation

Modeling Transformation

Purpose:
- Map geometry from local object coordinate system into a global world coordinate system
- Same as placing objects

Transformations:
- Arbitrary affine transformations are possible
  - Even more complex transformations may be desirable, but are not available in hardware
  - Freeform deformations

Viewing Transformation

Purpose:
- Map geometry from world coordinate system into camera coordinate system
- Camera coordinate system is right-handed, viewing direction is negative z-axis
- Same as placing camera

Transformations:
- Usually only rigid body transformations
  - Rotations and translations
- Objects have same size and shape in camera and world coordinates

Model/View Transformation

Combine modeling and viewing transform.
- Combine both into a single matrix
- Saves computation time if many points are to be transformed
- Possible because the viewing transformation directly follows the modeling transformation without intermediate operations

Homogeneous Planes And Normals

Planes in Cartesian Coordinates:
\[ \{(x, y, z) \mid n_x x + n_y y + n_z z + d = 0\} \]
- \( n_x, n_y, n_z \), and \( d \) are the parameters of the plane (normal and distance from origin)

Planes in Homogeneous Coordinates:
\[ \{(x, y, z, w) \mid n_x x + n_y y + n_z z + dw = 0\} \]
**Homogeneous Planes And Normals**

*Planes in homogeneous coordinates are represented as row vectors*

- $E = [a, b, c, d]$ 
- Condition that a point $[x, y, z, w]^T$ is located in $E$:
  \[
  \begin{bmatrix}
  x \\
  y \\
  z \\
  w
  \end{bmatrix} \in E = [n_x, n_y, n_z, d] \iff [n_x, n_y, n_z, d] \cdot \begin{bmatrix}
  x \\
  y \\
  z \\
  w
  \end{bmatrix} = 0
  \]

- Works for $T([n_x, n_y, n_z, d]) - [n_x, n_y, n_z, d]A^{-1}$
- Thus: planes have to be transformed by the inverse of the affine transformation (multiplied from left as a row vector)!

**Transformations of planes**

- $[a_x, a_y, a_z, a_d] = 0 \iff T([n_x, n_y, n_z, d]) \cdot (A^{-1})^T = 0$

**Homogeneous Normals**

- The plane definition also contains its normal
- Normal written as a vector $[n_x, n_y, n_z, 0]^T$
- Thus: the normal to any surface has to be transformed by the inverse transpose of the affine transformation (multiplied from the right as a column vector)!

**Coming Up...**

**Tuesday, Sep 18:**
- OpenGL transformations, matrix stacks

**Thursday, Sep 20:**
- Cameras & perspective transformations