To Do

- Start (thinking about) assignment 1
  - Much of information you need is in this lecture (slides)
  - Ask TA NOW if compilation problems, Visual C++ etc.
  - Not that much coding (solution is approx. 20 lines, but you may need more to implement basic matrix/vector math), but some thinking and debugging likely involved
- Specifics of HW 1
  - Axis-angle rotation and gltLookAt most useful (essential?). These are not covered in text (look at slides).
  - You probably only need final results, but try understanding derivations.
- Problems in text help understanding material. Usually, we have review sessions per unit, but this one before midterm

Course Outline

- 3D Graphics Pipeline
  - Modeling (Creating 3D Geometry)
  - Rendering (Creating, shading images from geometry, lighting, materials)

Motivation

- Many different coordinate systems in graphics
  - World, model, body, arms, …
- To relate them, we must transform between them
- Also, for modeling objects. I have a teapot, but
  - Want to place it at correct location in the world
  - Want to view it from different angles (HW 1)
  - Want to scale it to make it bigger or smaller

- Demo: HW 1, applet transformation_game.jar
General Idea

- Object in model coordinates
- Transform into world coordinates
- Represent points on object as vectors
- Multiply by matrices
- Demos with applet

Chapter 6 in text. We cover most of it essentially as in the book. Worthwhile (but not essential) to read whole chapter

Outline

- 2D transformations: rotation, scale, shear
- Composing transforms
- 3D rotations
- Translation: Homogeneous Coordinates (next time)
- Transforming Normals (next time)

(Nonuniform) Scale

\[
\text{Scale}(s_x, s_y) = \begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix}, \quad S^{-1} = \begin{pmatrix} s_x^{-1} & 0 \\ 0 & s_y^{-1} \end{pmatrix}
\]

\[
\begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} s_x x \\ s_y y \\ s_z z \end{pmatrix}
\]

transformation_game.jar

Shear

\[
\text{Shear} = \begin{pmatrix} 1 & a \\ 0 & 1 \end{pmatrix}, \quad S^{-1} = \begin{pmatrix} 1 & -a \\ 0 & 1 \end{pmatrix}
\]

Rotations

2D simple, 3D complicated. [Derivation? Examples?]

2D?

\[
\begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}
\]

- Linear \( R(X+Y) = R(X) + R(Y) \)
- Commutative

transformation_game.jar

Outline

- 2D transformations: rotation, scale, shear
- Composing transforms
- 3D rotations
- Translation: Homogeneous Coordinates
- Transforming Normals
Composing Transforms

- Often want to combine transforms
- E.g. first scale by 2, then rotate by 45 degrees
- Advantage of matrix formulation: All still a matrix
- Not commutative!! Order matters

E.g. Composing rotations, scales

\[ x_3 = Rx_2 \quad x_2 = Sx_1 \]
\[ x_3 = R(Sx_1) = (RS)x_1 \]
\[ x_3 \neq SRx_1 \]

Inverting Composite Transforms

- Say I want to invert a combination of 3 transforms
- Option 1: Find composite matrix, invert
- Option 2: Invert each transform and swap order
- Obvious from properties of matrices

\[
M = M_1M_2M_3 \\
M^{-1} = M_3^{-1}M_2^{-1}M_1^{-1} \\
M^{-1}M = M_3^{-1}(M_2^{-1}(M_1^{-1}M_1)M_2)M_3
\]

Outline

- 2D transformations: rotation, scale, shear
- Composing transforms
- 3D rotations
- Translation: Homogeneous Coordinates
- Transforming Normals

Rotations

Review of 2D case

\[
\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}
\]
- Orthogonal?, \( R^T R = I \)

Rotations in 3D

- Rotations about coordinate axes simple
- Always linear, orthogonal
- Rows/cols orthonormal
Geometric Interpretation 3D Rotations

- Rows of matrix are 3 unit vectors of new coord frame
- Can construct rotation matrix from 3 orthonormal vectors

\[ R_{uvw} = \begin{pmatrix} x_u & y_u & z_u \\ x_v & y_v & z_v \\ x_w & y_w & z_w \end{pmatrix} \]

\[ u = x_u X + y_u Y + z_u Z \]

\[ R_p = \begin{pmatrix} x_p & y_p & z_p \end{pmatrix} = u \cdot \begin{pmatrix} u \\ v \\ w \end{pmatrix} \]

- Rows of matrix are 3 unit vectors of new coord frame
- Can construct rotation matrix from 3 orthonormal vectors
- Effectively, projections of point into new coord frame
- New coord frame uvw taken to cartesian components xyz
- Inverse or transpose takes xyz cartesian to uvw

Non-Commutativity

- Not Commutative (unlike in 2D)!!
- Rotate by x, then y is not same as y then x
- Order of applying rotations does matter
- Follows from matrix multiplication not commutative
  - R1 * R2 is not the same as R2 * R1
- Demo: HW1, order of right or up will matter
  - simplestGlut.exe

Arbitrary rotation formula

- Rotate by an angle \( \theta \) about arbitrary axis a
- Not in book. Homework 1: must rotate eye, up direction
- Somewhat mathematical derivation (not covered here except relatively vaguely), but useful formula

Problem setup: Rotate vector \( b \) by \( \theta \) about \( a \)
- Helpful to relate \( b \) to X, \( a \) to Z, verify does right thing
- For HW1, you probably just need final formula
  - simplestGlut.exe

Axis-Angle formula

- Step 1: \( b \) has components parallel to \( a \), perpendicular
  - Parallel component unchanged (rotating about an axis leaves that axis unchanged after rotation, e.g. rot about z)
- Step 2: Define \( c \) orthogonal to both \( a \) and \( b \)
  - Analogous to defining Y axis
  - Use cross products and matrix formula for that
- Step 3: With respect to the perpendicular comp of \( b \)
  - Cos \( \theta \) of it remains unchanged
  - Sin \( \theta \) of it projects onto vector \( c \)
  - Verify this is correct for rotating X about Z
  - Verify this is correct for \( 0 \) as \( 0, 90 \) degrees

Axis-Angle: Putting it together

\[ (b \rightarrow a)_{\text{ROT}} = I_{3 \times 3} \cos \theta - aa^T \cos \theta \] + \( (A^{*} \sin \theta) b \)

\[ (b \rightarrow a)_{\text{ROT}} = (aa^T) b \]

\[ R(a, \theta) = I_{3 \times 3} \cos \theta + aa^T (1 - \cos \theta) + A^{*} \sin \theta \]

Unchanged

Component along \( a \) (hence unchanged)

Perpendicular

(rotated comp)
**Axis-Angle: Putting it together**

\[(b \cdot a)^{ROT} = (I_{3 \times 3} \cos \theta - a a^T \cos \theta) b + (A^\ast \sin \theta) b\]

\[(b \rightarrow a)^{ROT} = (a a^T) b\]

\[R(a, \theta) = I_{3 \times 3} \cos \theta + a a^T (1 - \cos \theta) + A^\ast \sin \theta\]

\[R(a, \theta) = \cos \theta \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} + (1 - \cos \theta) \begin{pmatrix} x & xy & xz \\ yx & 1 & yz \\ zy & yz & 1 \end{pmatrix} + \sin \theta \begin{pmatrix} 0 & -x & y \\ z & x & 0 \\ -y & 0 & x \end{pmatrix}\]

\((x, y, z)\) are cartesian components of \(a\)