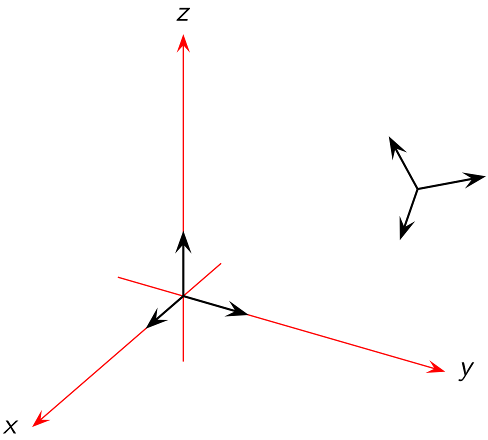
Transforms

Geoffrey Matthews

Department of Computer Science
Western Washington University

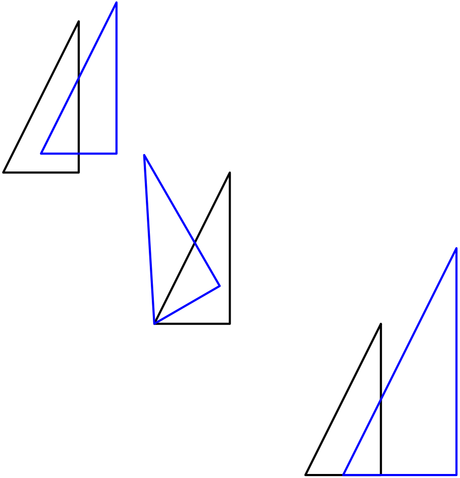
Fall 2011

Transforms



- Moving from one frame to another.
- Describe an object in its own frame, then describe all points in the object in the world frame.
- Describe the world in its natural frame, then describe everything in the world from the camera's frame.
- Describe everything in the 3D world, then move it to the 2D world of the screen.

Simple transformations

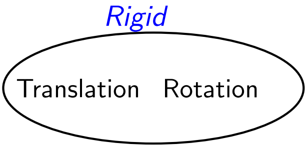


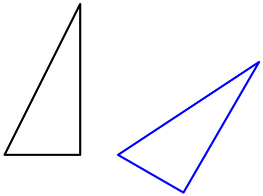
- Translation
- Rotation
- Uniform scaling

Transformations are used

- Position objects in a scene
- Change shape of objects
- Create multiple copies of objects
- Position camera
- Projection for virtual cameras
- Animations

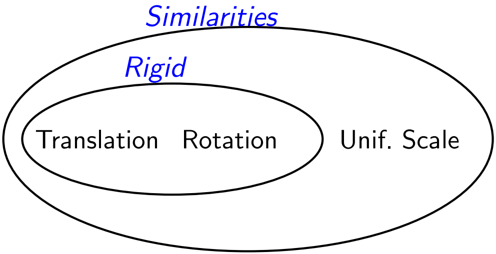
Rigid-body (Euclidean) Transforms

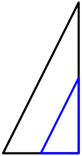




- Preserves distances and angles

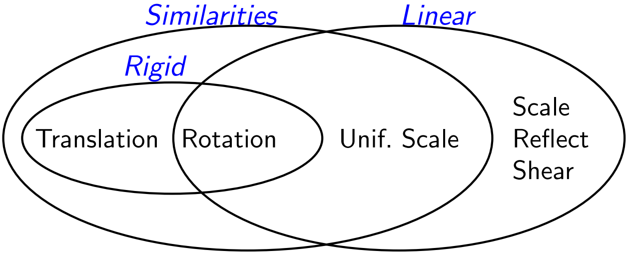
Similitudes / Similarity Transforms

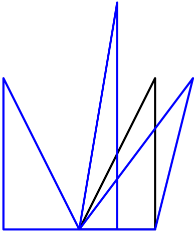




- Preserves angles

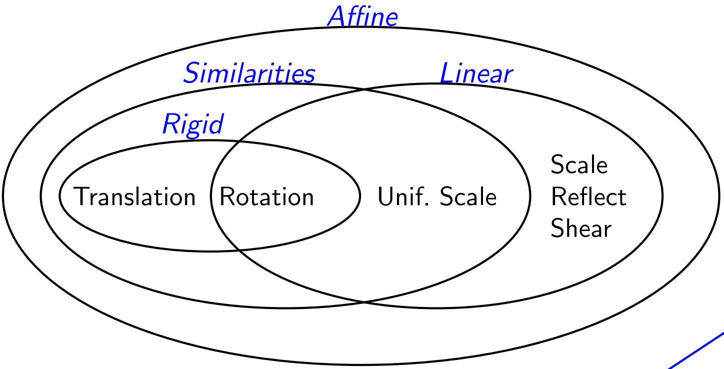
Linear transforms

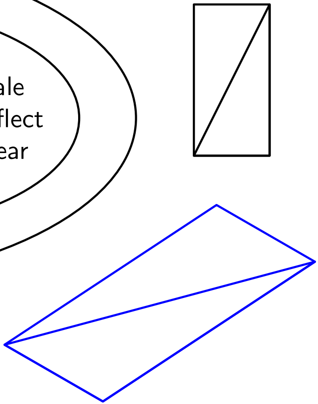




- $L(p + q) = L(p) + L(q)$
- $L(ap) = aL(p)$

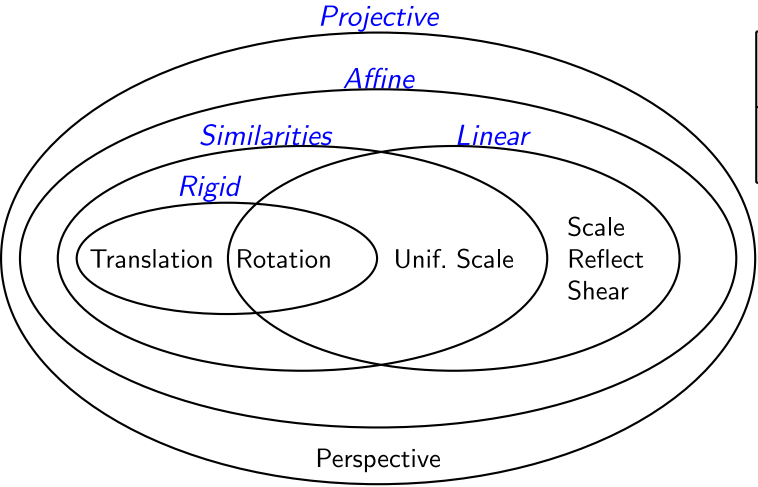
Affine Transforms

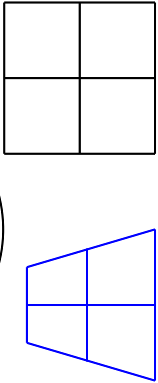




- Preserves parallel lines

Projective Transforms





- Preserves lines

Homogeneous coordinates

- Recall a 3D frame is three vectors and a point: x, y, z, p.
- We can represent a point q as *coordinates* in this frame as a 4-vector $(a, b, c, 1)$ because $q = [x, y, z, p] \cdot [a, b, c, 1]^T$
- Likewise we can represent a vector v as *coordinates* in this frame with a 4-vector $(a, b, c, 0)$ because $v = [x, y, z, p] \cdot [a, b, c, 0]^T$
- These are called *homogeneous coordinates*
- They help distinguish between points and vectors
- They simplify other calculations with points and vectors, in particular, transformations.

Representing transforms with matrices

- A general linear transformation:

$$\begin{aligned} x' &= ax + by + c \\ y' &= dx + ey + f \end{aligned}$$

- Multiplication and addition:

$$\begin{aligned} \begin{bmatrix} x' \\ y' \end{bmatrix} &= \begin{bmatrix} a & b \\ d & e \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} c \\ f \end{bmatrix} \\ p' &= Mp + t \end{aligned}$$

Homogeneous coordinates

- If we add another dimension, we can get by with just multiplication.

$$
\begin{aligned}
x' &= ax + by + c \\
y' &= dx + ey + f \\
1 &= 1 \\
\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} &= \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \\
p' &= Mp
\end{aligned}
$$

- In 2D we use 3×3 matrices.

Homogeneous coordinates

- In 3D we use 4×4 matrices.
- Each point has an extra value, w, usually 1.

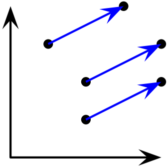
$$
\begin{aligned}
x' &= ax + by + cz + d \\
y' &= ex + fy + gz + h \\
z' &= ix + jy + kz + l \\
\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} &= \begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \\
p' &= Mp
\end{aligned}
$$

Homogeneous coordinates

- Each point has an extra value, w, usually 1.
- If M is an *affine* transformation, w will remain 1.
- We use $w \neq 1$ only in projections.
- If $w \neq 1$ for a point, we normalize by dividing by w before using.

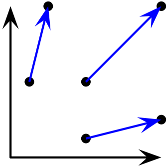
$$
\begin{aligned}
\begin{bmatrix} a & b & c & d \\ e & f & g & h \\ i & j & k & l \\ m & n & o & p \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} &= \begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} \\
&\Leftrightarrow \begin{bmatrix} x'/w' \\ y'/w' \\ z'/w' \\ 1 \end{bmatrix} \\
Mp &= p'
\end{aligned}
$$

Translate



$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

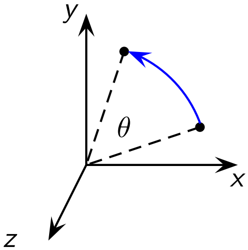
Scale



$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- Isotropic (uniform) scaling: $s_x = s_y = s_z$.
- Generally avoid scaling; creates difficulties with normals.

Rotation



- Righthand rotation about the z axis in a righthand frame.
- Lefthand rotation about the z axis in a lefthand frame.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Rotation

- Righthand rotation about the x axis.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) & 0 \\ 0 & \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- Righthand rotation about the y axis.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) & 0 \\ 0 & 1 & 0 & 0 \\ \sin(\theta) & 0 & \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

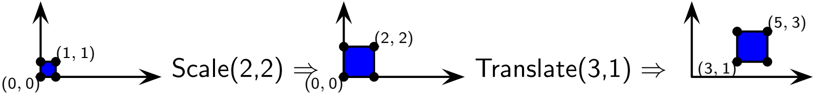
- Righthand rotation about the z axis.

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Rotation about an arbitrary axis

- Rodrigues rotation matrix
  `http://en.wikipedia.org/wiki/Rotation_matrix`
- Fairly easy derivation using vectors.
- Can also use quaternions.
- We will find other ways to deal with arbitrary rotations.

How are tranforms combined?

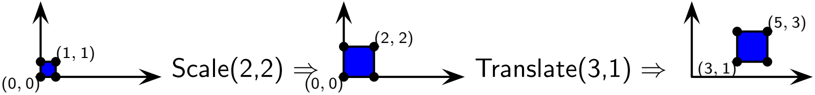


- Matrix multiplication is associative: $p' = T(Sp) = TSp$

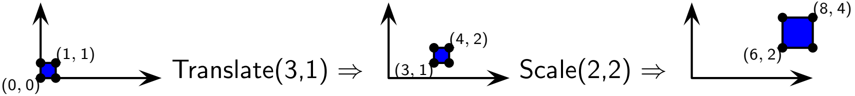
$$TS = \begin{bmatrix} 1 & 0 & 3 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 3 \\ 0 & 2 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

- Remember we multiply on the left, so in matrix TS scale is done first, translate second.

Matrix multiplication is not commutative: $TS \neq ST$



$$TS = \begin{bmatrix} 1 & 0 & 3 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 3 \\ 0 & 2 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$



$$ST = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 3 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 6 \\ 0 & 2 & 2 \\ 0 & 0 & 1 \end{bmatrix}$$

Easy inverses

- The inverse of a rotation matrix is its transpose.
- Easy to see with rotation about an axis:

$$\begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 & 0 \\ -\sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^T = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 & 0 \\ \sin(\theta) & \cos(\theta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Also holds true of any pure rotation.

Easy inverses

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} =$$

Easy inverses

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & 0 & -x \\ 0 & 1 & 0 & -y \\ 0 & 0 & 1 & -z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Easy inverses

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & 0 & -x \\ 0 & 1 & 0 & -y \\ 0 & 0 & 1 & -z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} x & 0 & 0 & 0 \\ 0 & y & 0 & 0 \\ 0 & 0 & z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} =$$

Easy inverses

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & 0 & -x \\ 0 & 1 & 0 & -y \\ 0 & 0 & 1 & -z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} x & 0 & 0 & 0 \\ 0 & y & 0 & 0 \\ 0 & 0 & z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1/x & 0 & 0 & 0 \\ 0 & 1/y & 0 & 0 \\ 0 & 0 & 1/z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Easy inverses

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & 0 & -x \\ 0 & 1 & 0 & -y \\ 0 & 0 & 1 & -z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} x & 0 & 0 & 0 \\ 0 & y & 0 & 0 \\ 0 & 0 & z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1/x & 0 & 0 & 0 \\ 0 & 1/y & 0 & 0 \\ 0 & 0 & 1/z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} =$$

Easy inverses

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & 0 & -x \\ 0 & 1 & 0 & -y \\ 0 & 0 & 1 & -z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} x & 0 & 0 & 0 \\ 0 & y & 0 & 0 \\ 0 & 0 & z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1/x & 0 & 0 & 0 \\ 0 & 1/y & 0 & 0 \\ 0 & 0 & 1/z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} c & s & 0 & 0 \\ -s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Decomposable Transforms

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

Decomposable Transforms

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c & -s & 0 & x \\ s & c & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Decomposable Transforms

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c & -s & 0 & x \\ s & c & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

Decomposable Transforms

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c & -s & 0 & x \\ s & c & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} c & -s & 0 & cx - sy \\ s & c & 0 & sx + cy \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Decomposable Transforms

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

Decomposable Transforms

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & x \\ 0 & b & 0 & y \\ 0 & 0 & c & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Decomposable Transforms

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & x \\ 0 & b & 0 & y \\ 0 & 0 & c & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} =$$

Decomposable Transforms

$$\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & x \\ 0 & b & 0 & y \\ 0 & 0 & c & z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} a & 0 & 0 & ax \\ 0 & b & 0 & by \\ 0 & 0 & c & cz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

More Easy Inverses

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

More Easy Inverses

$$\begin{bmatrix} c & -s & 0 & x \\ s & c & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \left(\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \right)^{-1}$$

$$=$$

More Easy Inverses

$$
\begin{aligned}
\begin{bmatrix} c & -s & 0 & x \\ s & c & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}
&= \left(\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \right)^{-1} \\
&= \begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \\
&=
\end{aligned}
$$

More Easy Inverses

$$
\begin{aligned}
\begin{bmatrix} c & -s & 0 & x \\ s & c & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}
&= \left(\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \right)^{-1} \\
&= \begin{bmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \\
&= \begin{bmatrix} c & s & 0 & 0 \\ -s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -x \\ 0 & 1 & 0 & -y \\ 0 & 0 & 1 & -z \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
$$

More Easy Inverses

$$\begin{bmatrix} a & 0 & 0 & x \\ 0 & b & 0 & y \\ 0 & 0 & c & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} =$$

More Easy Inverses

$$\begin{bmatrix} a & 0 & 0 & x \\ 0 & b & 0 & y \\ 0 & 0 & c & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \left(\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \right)^{-1}$$

$$=$$

More Easy Inverses

$$\begin{aligned}
\begin{bmatrix} a & 0 & 0 & x \\ 0 & b & 0 & y \\ 0 & 0 & c & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} &= \left(\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \right)^{-1} \\
&= \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \\
&=
\end{aligned}$$

More Easy Inverses

$$
\begin{aligned}
\begin{bmatrix} a & 0 & 0 & x \\ 0 & b & 0 & y \\ 0 & 0 & c & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1}
&= \left(\begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \right)^{-1} \\
&= \begin{bmatrix} a & 0 & 0 & 0 \\ 0 & b & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} \\
&= \begin{bmatrix} 1/a & 0 & 0 & 0 \\ 0 & 1/b & 0 & 0 \\ 0 & 0 & 1/c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -x \\ 0 & 1 & 0 & -y \\ 0 & 0 & 1 & -z \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned}
$$

The Modelview Matrix

- Most worlds are modelled by
  1. positioning the model in world coordinates
  2. position the camera in world coordinates
- To put all objects in camera coordinates
  1. multiply each object by model transform
  2. multiply by inverse camera transform
- The product of these two matrices is called the **modelview** matrix:

$$C^{-1}M$$

- Each point in an object will be multiplied by this matrix to put it into camera coordinates.

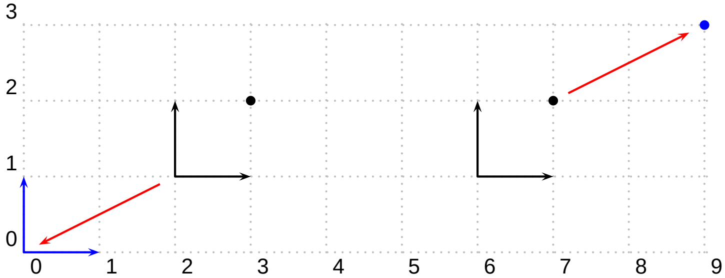
Finding a frame

- When positioning a model or camera in the world it is generally easy to find a *forward* vector $\mathbf{v_1}$ and an *up* $\mathbf{v_2}$ vector in world coordinates.
- These two vectors need not be orthonormal, just not parallel.
- A third vector, pointing *right*, can be defined as $\mathbf{v_3} = \mathbf{v_1} \times \mathbf{v_2}$.
- Using the **Gram Schmidt** process you can create an orthonormal frame $\mathbf{e_1}, \mathbf{e_2}, \mathbf{e_3}$ from $\mathbf{v_1}, \mathbf{v_2}, \mathbf{v_3}$

Finding a frame

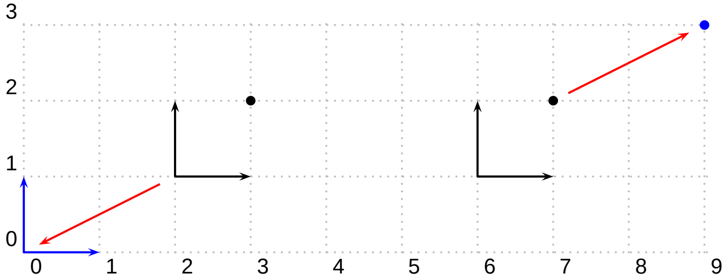
- When positioning a model or camera in the world it is generally easy to find a *forward* vector $\mathbf{v_1}$ and an *up* $\mathbf{v_2}$ vector in world coordinates.
- These two vectors need not be orthonormal, just not parallel.
- A third vector, pointing *right*, can be defined as $\mathbf{v_3} = \mathbf{v_1} \times \mathbf{v_2}$.
- Using the **Gram Schmidt** process you can create an orthonormal frame $\mathbf{e_1}, \mathbf{e_2}, \mathbf{e_3}$ from $\mathbf{v_1}, \mathbf{v_2}, \mathbf{v_3}$
- Slightly faster to Gram-Schmidt-ize the *forward* and *up* vectors, then the cross product is automatically orthonormal.
- It is usually easy to find a *forward* vector for an object—what is it "looking at"?
- An *up* vector is also usually easy, can almost always start with $(0, 1, 0)$ and then Gram-Schmidt it.
- Use *forward*$\times$*up* to get *right*.

Change of frame



$$\begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

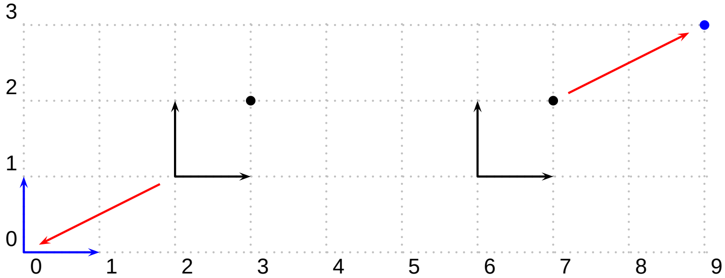
Change of frame



- Does a transform move the object or the frame?

$$\begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Change of frame



$$\begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Does a transform move the object or the frame?
- If you move the *object*, just use the transform.
- However, if you want to *move the frame* by a given transform, you need the inverse.

Effect of a transform on frame vectors and origin

$$\begin{bmatrix} a & d & g & j \\ b & e & h & k \\ c & f & i & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} a & d & g & j \\ b & e & h & k \\ c & f & i & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} d \\ e \\ f \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} a & d & g & j \\ b & e & h & k \\ c & f & i & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} g \\ h \\ i \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} a & d & g & j \\ b & e & h & k \\ c & f & i & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} j \\ k \\ l \\ 1 \end{bmatrix}$$

Effect of a transform on frame vectors and origin

$$\begin{bmatrix} a & d & g & j \\ b & e & h & k \\ c & f & i & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} a \\ b \\ c \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} a & d & g & j \\ b & e & h & k \\ c & f & i & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} d \\ e \\ f \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} a & d & g & j \\ b & e & h & k \\ c & f & i & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} g \\ h \\ i \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} a & d & g & j \\ b & e & h & k \\ c & f & i & l \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} j \\ k \\ l \\ 1 \end{bmatrix}$$

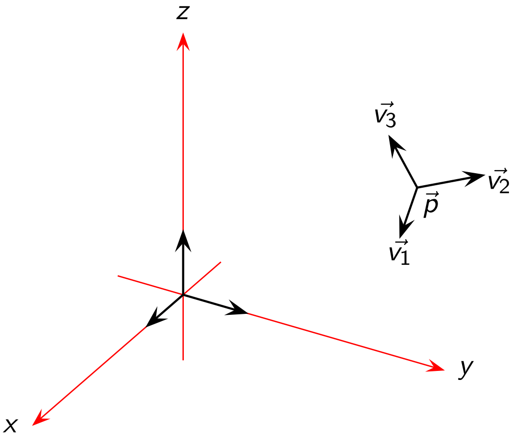
$$(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_1} = \vec{v_1}$$

$$(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_2} = \vec{v_2}$$

$$(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_3} = \vec{v_3}$$

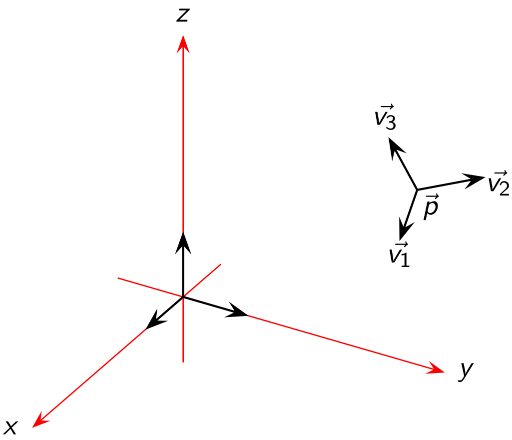
$$(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot o = \vec{p}$$

Change of Frame



$$
\begin{aligned}
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_1} &= \vec{v_1} \\
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_2} &= \vec{v_2} \\
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_3} &= \vec{v_3} \\
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot o &= p
\end{aligned}
$$

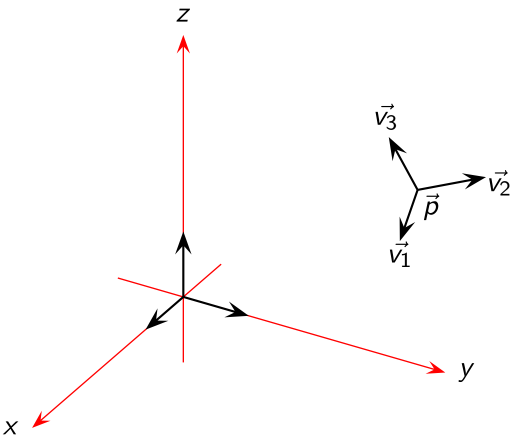
Change of Frame



$$
\begin{aligned}
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_1} &= \vec{v_1} \\
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_2} &= \vec{v_2} \\
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_3} &= \vec{v_3} \\
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot o &= p
\end{aligned}
$$

- Use to put model points into the world.

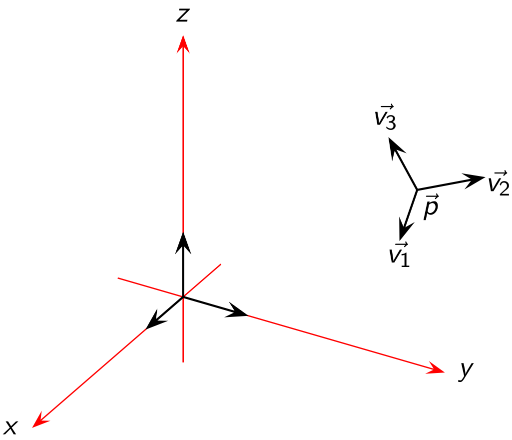
Change of Frame



$$
\begin{aligned}
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_1} &= \vec{v_1} \\
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_2} &= \vec{v_2} \\
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_3} &= \vec{v_3} \\
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot o &= p
\end{aligned}
$$

- Use to put model points into the world.
- Use inverse to put the world in camera coords.
- Multiply to get modelview matrix.

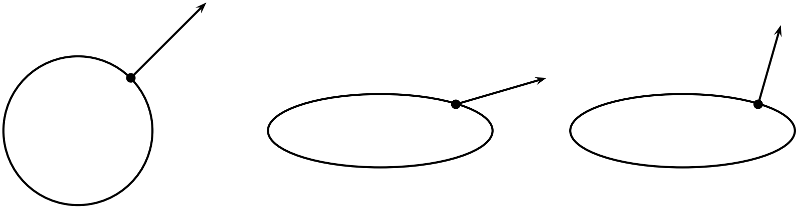
Camera transforms



$$
\begin{aligned}
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_1} &= \vec{v_1} \\
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_2} &= \vec{v_2} \\
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot \vec{e_3} &= \vec{v_3} \\
(\vec{v_1}, \vec{v_2}, \vec{v_3}, p) \cdot o &= p
\end{aligned}
$$

- If both frames are orthonormal, it is a rigid transformation.
- Can use "easy inverse" for cameras.
- Only one matrix multiply needed.

Transforming normals



- Normals do not stay normalized after scale transforms.
- Must use the inverse transpose

$$\left(M^{-1}\right)^T$$

- Might be good to maintain inverses.
- Rigid transforms OK.

Online Resources

Readings

- http://en.wikipedia.org/wiki/Transformation_matrix
- http://xkcd.com/184/
- http://ocw.mit.edu/courses/electrical-engineering-and-computer-science/, Computer graphics
- http://www.songho.ca/opengl/index.html