Ref

Composition of Transformations

reminder:

translate(a,b,c)

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

scale(a,b,c)

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

$Rotate(z,\theta)$

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

rotation & scale

translation

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} i & j & k & Org. \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

4×4 "affine" transformation

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} 3\times3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$ "linear transformation"

Simple Compositions

translate(a,b,c) translate(d,e,f) = translate(a+d, b+e, c+f)

$$\begin{bmatrix} 1 & & & a \\ & 1 & & b \\ & & 1 & c \\ & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & d \\ & 1 & & e \\ & & 1 & f \\ & & & 1 \end{bmatrix} = \begin{bmatrix} 1 & & & a+d \\ & 1 & & b+e \\ & & 1 & c+f \\ & & & 1 \end{bmatrix}$$

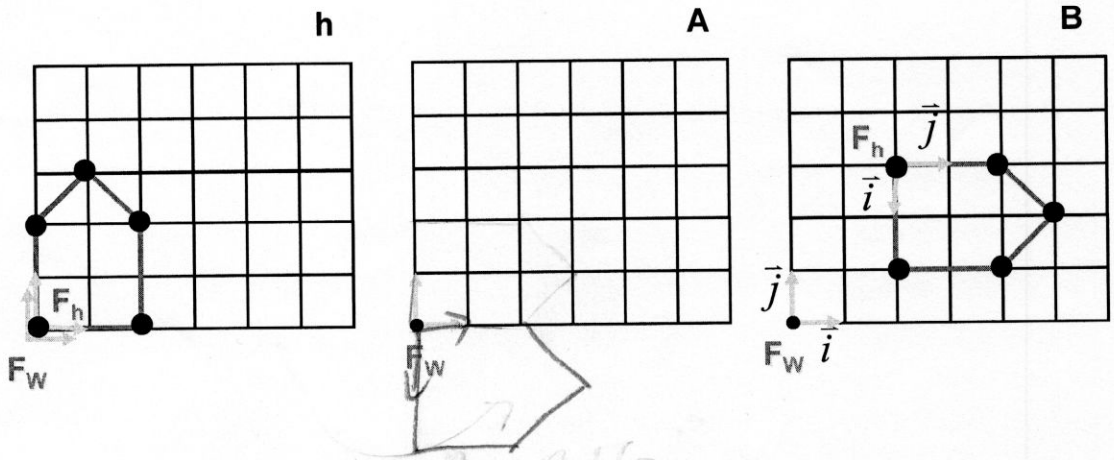
scale(a,b,c) scale(d,e,f) = scale(ad, be, cf)

$$\begin{bmatrix} a & & & \\ & b & & \\ & & c & \\ & & & 1 \end{bmatrix} \begin{bmatrix} d & & & \\ & e & & \\ & & f & \\ & & & 1 \end{bmatrix} = \begin{bmatrix} ad & & & \\ & be & & \\ & & cf & \\ & & & 1 \end{bmatrix}$$

$Rotate(z,\theta_2)\ Rotate(z,\theta_2)$ = $Rot(z, \theta_1+\theta_2)$

$$\begin{bmatrix} C_1 & -S_1 & & \\ S_1 & C_1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} \begin{bmatrix} C_2 & -S_2 & & \\ S_2 & C_2 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} = \begin{bmatrix} C_1C_2 - S_1S_2 & -S_1C_2 - C_1S_2 & & \\ S_1C_2 + C_1S_2 & C_1C_2 - S_1S_2 & & \\ & & 1 & \\ & & & 1 \end{bmatrix} = \begin{bmatrix} C_{12} & -S_{12} & & \\ S_{12} & C_{12} & & \\ & & 1 & \\ & & & 1 \end{bmatrix}$$

Composing Transformations
(thinking in world coords)



$Rot(z, -90°)$ $\quad$ $Trans(2,3,0)$

① $P_A = Rot(z, -90°)\, P_h$

② $P_B = Trans(2,3,0)\, P_A$

$P_B = Trans(2,3,0)\, Rot(z, -90°)\, P_h$

Composing Transformations

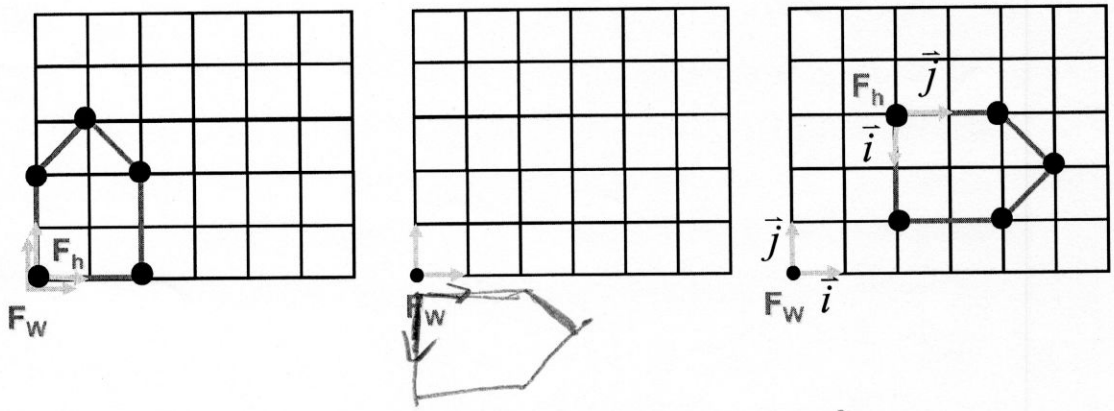
$P_w = Trans(2,3,0)\, Rot(z, -90°)\, P_h$

- R-to-L: left multiply → world coords
  - *interpret operations wrt fixed coords*
- L-to-R: right multiply
  - *interpret operations wrt local coords*
  - default for graphics APIs

code: `translate(2,3,0)` $\quad M \leftarrow M \cdot Trans(2,3,0)$

`rotate(z, -90°)` $\quad M \leftarrow M \cdot Rot(z, -90°)$

Composing Transformations
(thinking in local coords)



$\Rightarrow$ Rot(z, -90°) $\Rightarrow$ Trans(-3, 2, 0)

$P_w = Rot(z, -90°)\ Trans(-3, 2, 0)$

Composing Transformations

Compare ← world.

matrix mult.

$P_W = Trans(2,3,0)\,Rot(z,-90)\,P_h$ } equivalent $AB \neq BA$

$P_W = Rot(z,-90)\,Trans(-3,2,0)\,P_h$

local →

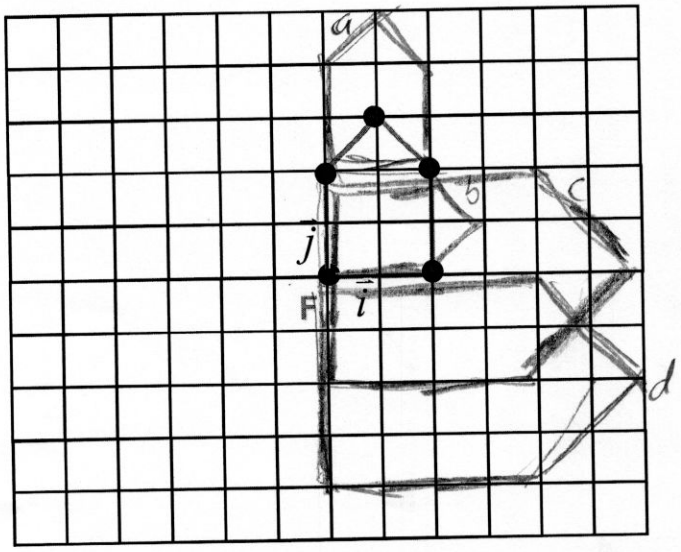
Undoing Transformations

$Trans(a,b,c) \cdot Trans(-a,-b,-c) = I$

$Rot(z,\theta) \cdot Rot(z,-\theta) = I$

$Scale(a,b,c) \cdot Scale(-a,-b,-c) = I$

Test yourself ...

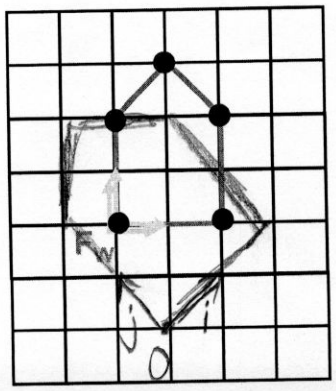


(a) **translate(0,2,0);**
(b) **rotate(-90,0,0,1);**
(c) **scale(2,2,2);**
(d) **translate(1,0,0);**
e **drawHouse();**

Test yourself

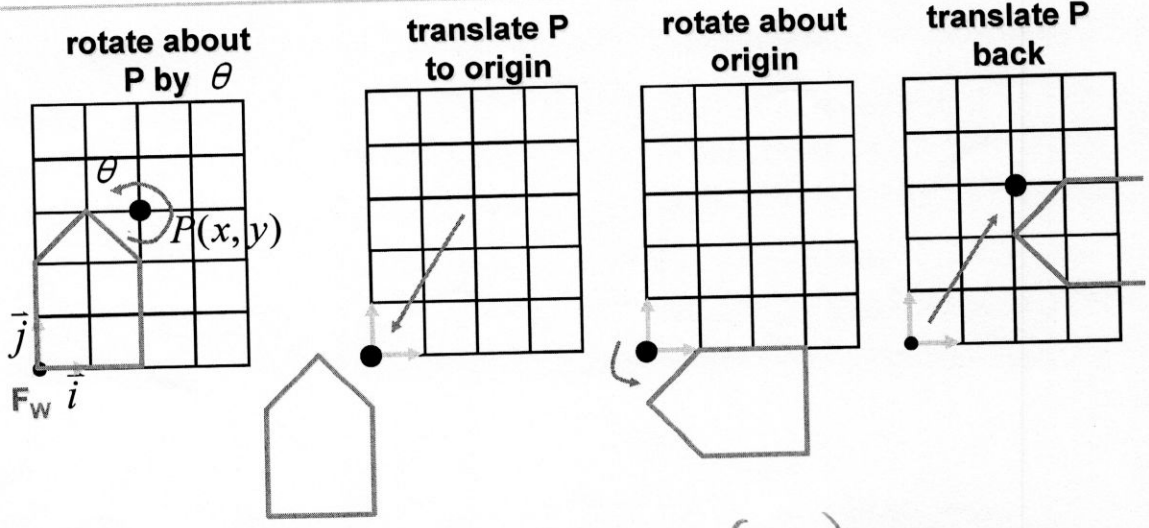
$$M = \begin{bmatrix} 1 & -1 & 0 & 1 \\ 1 & 1 & 0 & -2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

(a) Sketch the origin and basis vectors of the transformed house
(b) Draw the transformed house
(c) Give a sequence of translate(), rotate(), and scale() that implements this



Trans(1, -2, 0)
Rot(z, -45°)
Scale($\sqrt{2}$, $\sqrt{2}$, $\sqrt{2}$)

Rotation about a point



Lets think about this in world coords (R to L)

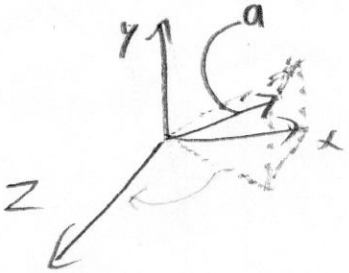
$$M = \text{Trans}(x,y,0)\,\text{Rot}(z,\theta)\,\text{Trans}(-x,-y,0)$$

Rotation about an arbitrary axis

θ axis $\vec{A}$

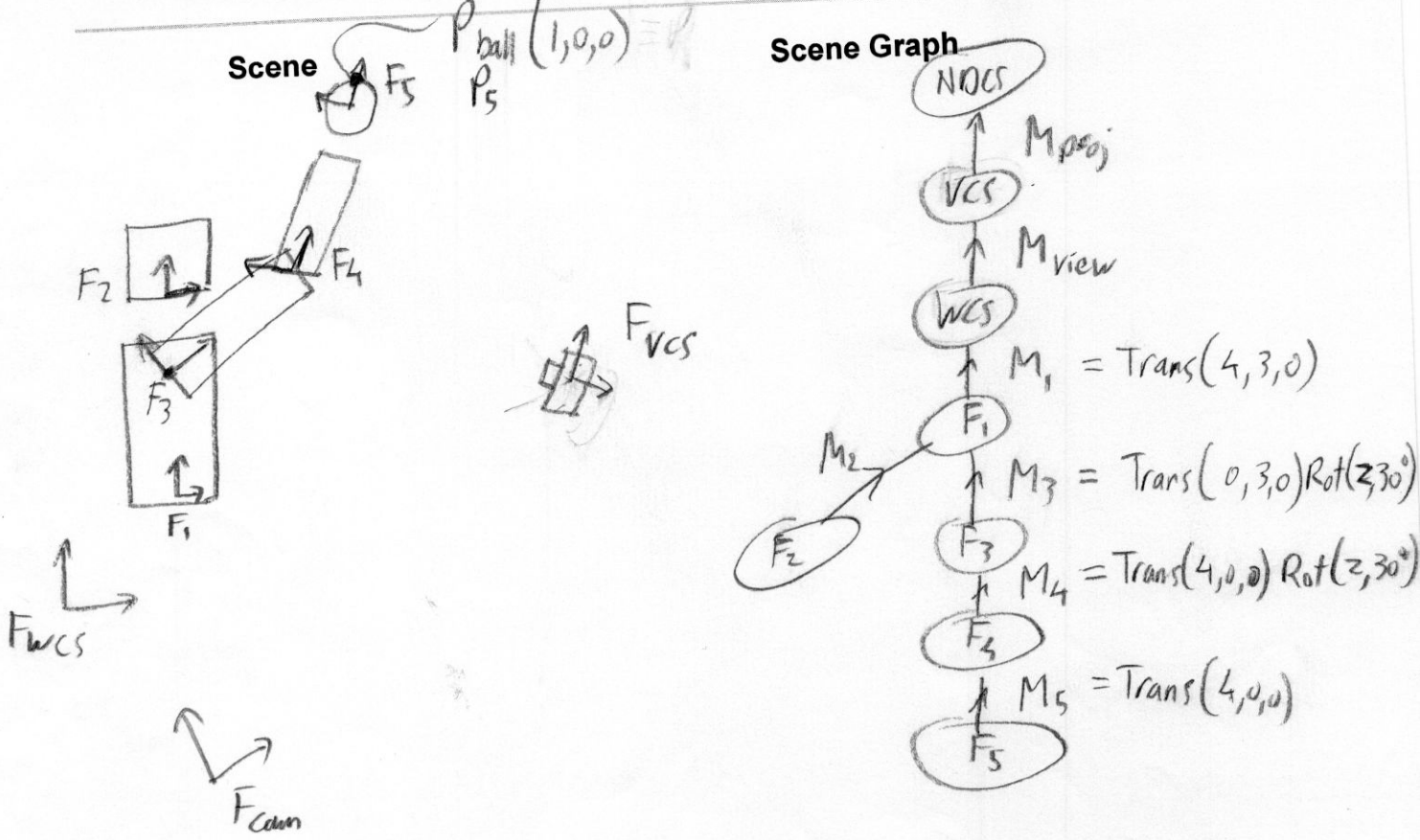
glRotatef(angle, x, y, z);

$$\vec{a} = \frac{\vec{A}}{\|\vec{A}\|}$$



(skip the derivation)

Transformations in Scene Graphs (1)



Transformations in Scene Graphs (2)

Transforming Vertices

Math

$$P_{NDCS} = M_{proj}\, M_{view}\, M_1\, M_3\, M_4\, M_5\, P_5 \qquad (M_{model} = M_1 M_3 M_4 M_5)$$

Code

```
M.identity()
M.setPerspective()
M.lookAt()
M.translate(4,3,0)
...
drawBall()
```

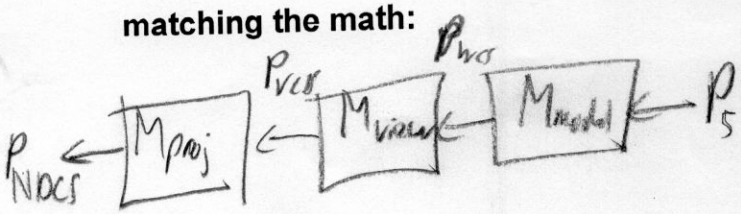
$M = M_{proj}$

$M = M_{proj} \cdot M_{view}$

$M = M_{proj} \cdot M_{view} \cdot M_1$

Graphics Pipeline diagram

matching the math:



how we'll usually draw it: