Computer Graphics
CS 543 – Lecture 3 (Part 3)
Introduction to Transformations

Prof Emmanuel Agu

Computer Science Dept.
Worcester Polytechnic Institute (WPI)

Introduction to Transformations

- Transformation changes an objects:
  - Position (translation)
  - Size (scaling)
  - Orientation (rotation)
  - Shapes (shear)
- Introduce first in 2D or *(x,y)*, build intuition
- Later, talk about 3D
- Transform object by applying sequence of matrix multiplications to object vertices

Why Matrices?

- All transformations can be performed using matrix/vector multiplication
- Allows pre-multiplication of all matrices
- Note: point (x,y) needs to be represented as (x,y,1), also called **Homogeneous coordinates**

Point Representation

- We use a column matrix (2x1 matrix) to represent a 2D point

$$\begin{pmatrix} x \\ y \end{pmatrix}$$

- General form of transformation of a point *(x,y)* to *(x',y')* can be written as:

$$x' = ax + by + c$$

$$y' = dx + ey + f$$

or

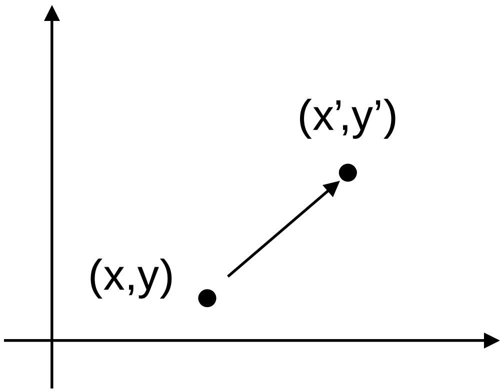
$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{pmatrix} \bullet \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Translation

- To reposition a point along a straight line
- Given point *(x,y)* and translation distance (t_x, t_y)
- The new point: *(x',y')*

$x' = x + t_x$

$y' = y + t_y$



or

$$P' = P + T \quad \text{where} \quad P' = \begin{pmatrix} x' \\ y' \end{pmatrix} \quad P = \begin{pmatrix} x \\ y \end{pmatrix} \quad T = \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

3x3 2D Translation Matrix

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

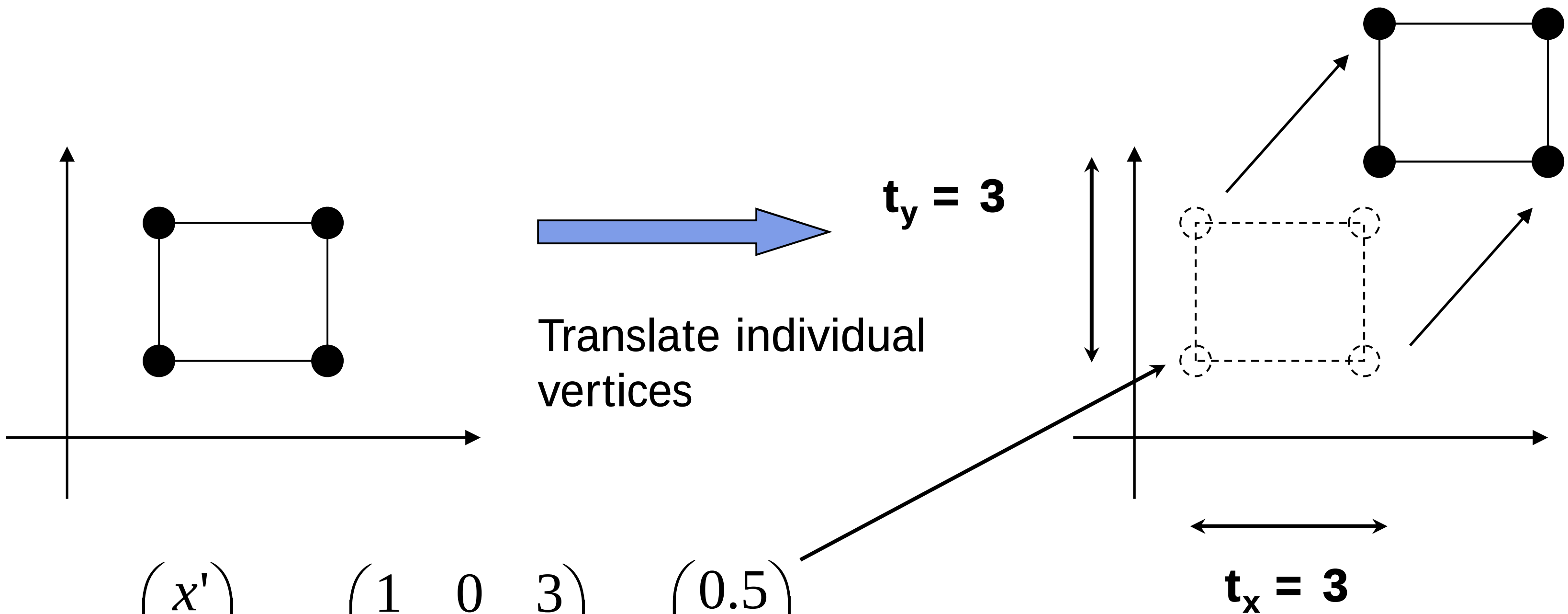
⬇ use 3x1 vector

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

- Note: it becomes a matrix-vector multiplication

Translation of Objects

- How to translate an object with multiple vertices?



$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 3 \\ 0 & 1 & 3 \\ 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} 0.5 \\ 0.5 \\ 1 \end{pmatrix}$$

Transforms in 3D

- 2D: 3x3 matrix multiplication
- 3D: 4x4 matrix multiplication: homogenous coordinates
- Again: transform object = transform each vertice
- General form:

$$M = \begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Xform of *P* →

$$\begin{pmatrix} Q_x \\ Q_y \\ Q_z \\ 1 \end{pmatrix} = M \begin{pmatrix} P_x \\ P_y \\ P_z \\ 1 \end{pmatrix}$$

3D Translation Matrix

- Now, 3D :

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \\ t_z \end{pmatrix}$$

translate(tx,ty,tz)

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

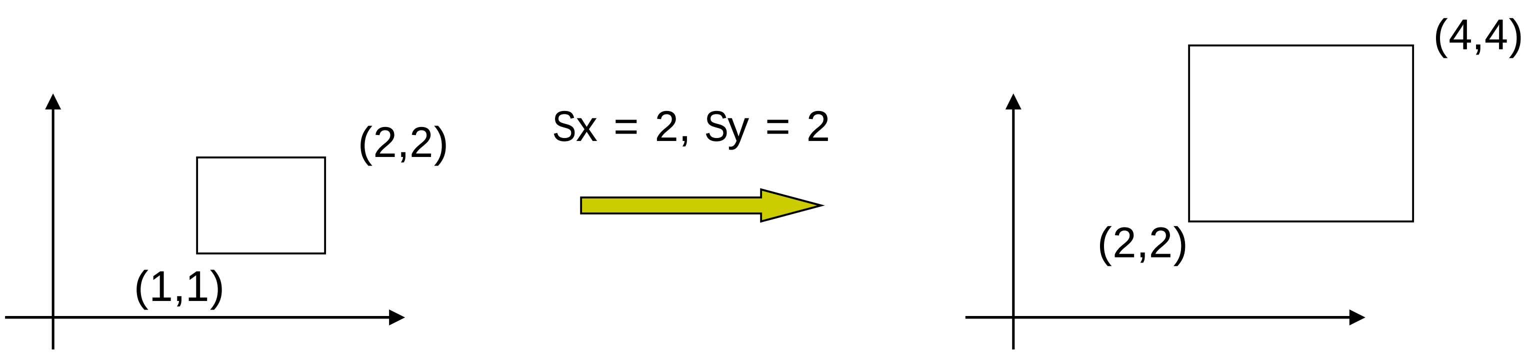
- Where: *x'= x.1 + y.0 + z.0 + tx.1 = x + tx*, … etc

2D Scaling

- Scale: Alter object size by scaling factor (s_x, s_y). i.e

$$x' = x \cdot Sx$$
$$y' = y \cdot Sy$$

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} Sx & 0 \\ 0 & Sy \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$



2D Scaling Matrix

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} Sx & 0 \\ 0 & Sy \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

4x4 3D Scaling Matrix

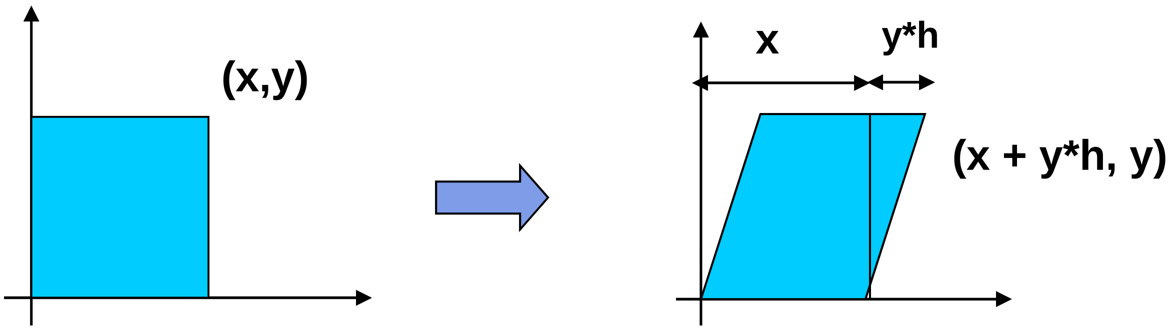
$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

- Example:
- If $Sx = Sy = Sz = 0.5$
- Can scale:
- big cube (sides = 1) to small cube (sides = 0.5)
- 2D: square, 3D cube

Scale(Sx,Sy,Sz)

Shearing



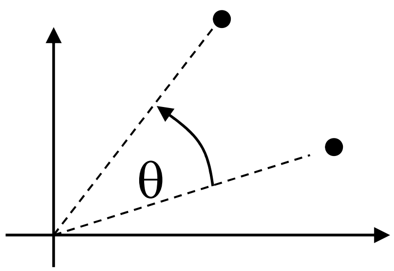
- Y coordinates are unaffected, but x cordinates are translated linearly with y
- That is:
  - **y' = y**
  - **x' = x + y * h**

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & h & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} * \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

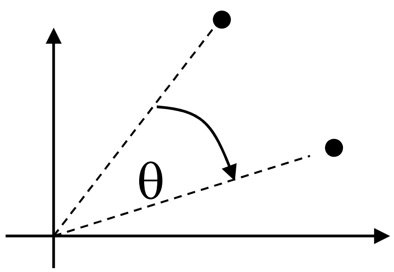
▪h is fraction of y to be added to x

2D Rotation

- Default rotation center is origin *(0,0)*



$\theta > 0$: Rotate counter clockwise



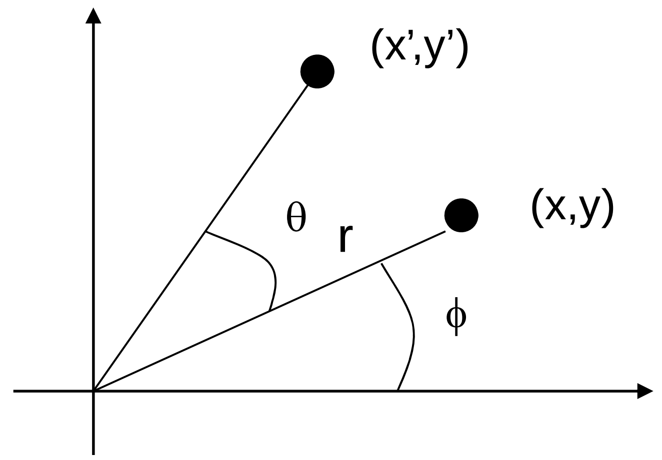
$\theta < 0$: Rotate clockwise

Rotation

(x,y) -> Rotate *about the origin* by θ

→ (x’, y’)

How to compute (x’, y’) ?



$x = r\cos(\phi)$ $y = r\sin(\phi)$

$x' = r\cos(\phi+\theta)$ $y' = r\sin(\phi+\theta)$

Rotation

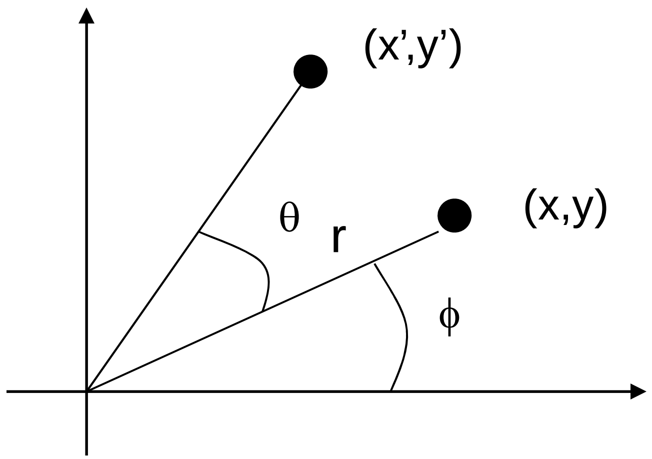
Using trig identities

$$\cos(\theta+\phi)=\cos\theta\cos\phi-\sin\theta\sin\phi$$

$$\sin(\theta+\phi)=\sin\theta\cos\phi+\cos\theta\sin\phi$$

x' = x cos(θ) – y sin(θ)

y' = y cos(θ) + x sin(θ)



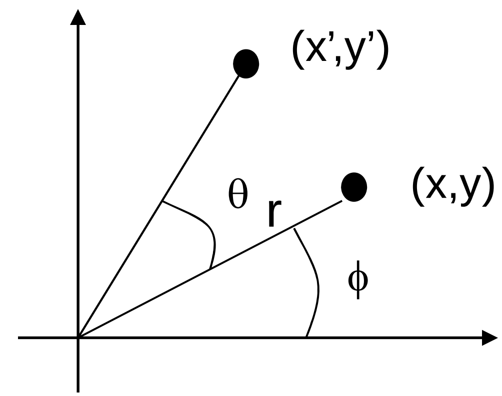
Matrix form?

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

3 x 3?

3x3 2D Rotation Matrix

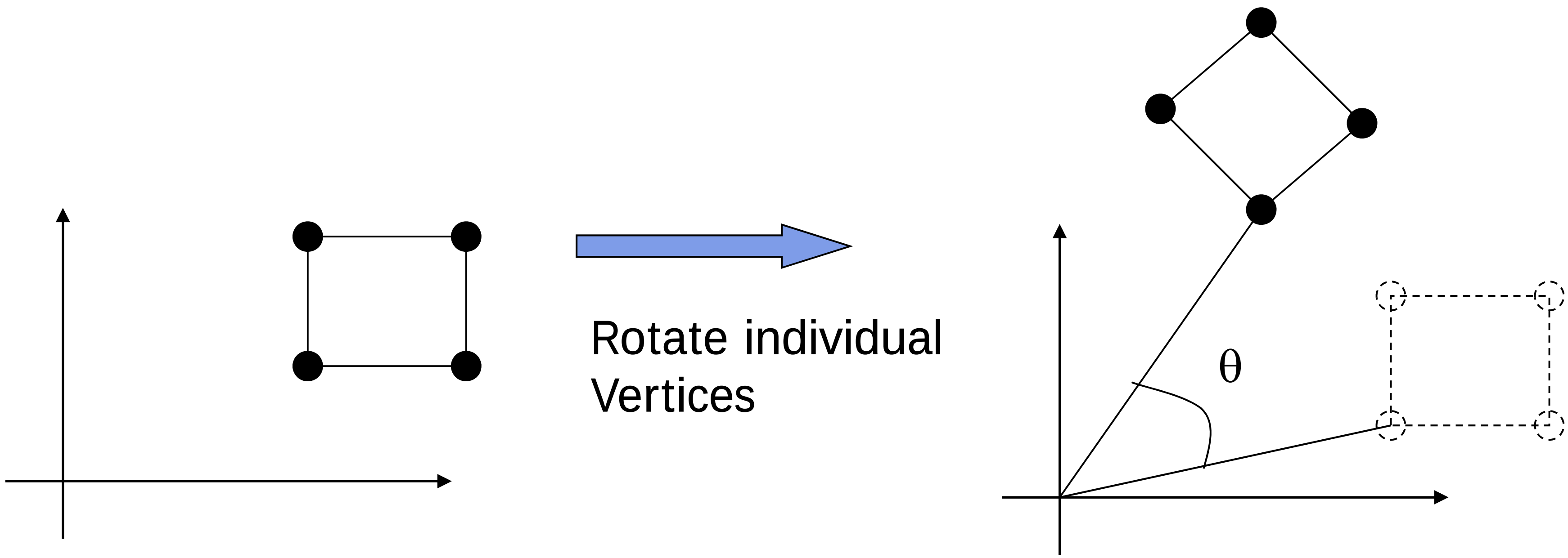
$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$



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

Rotation

- How to rotate an object with multiple vertices?

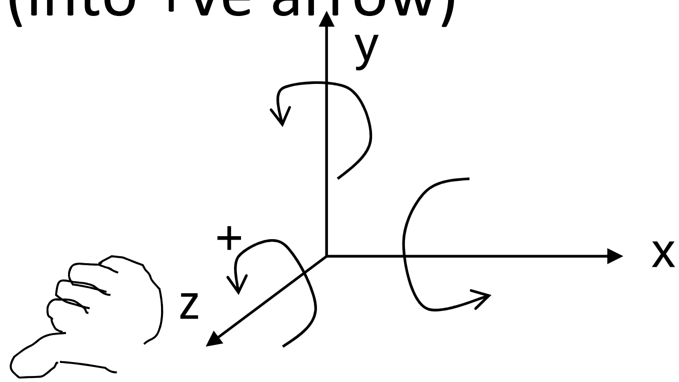


Rotating in 3D

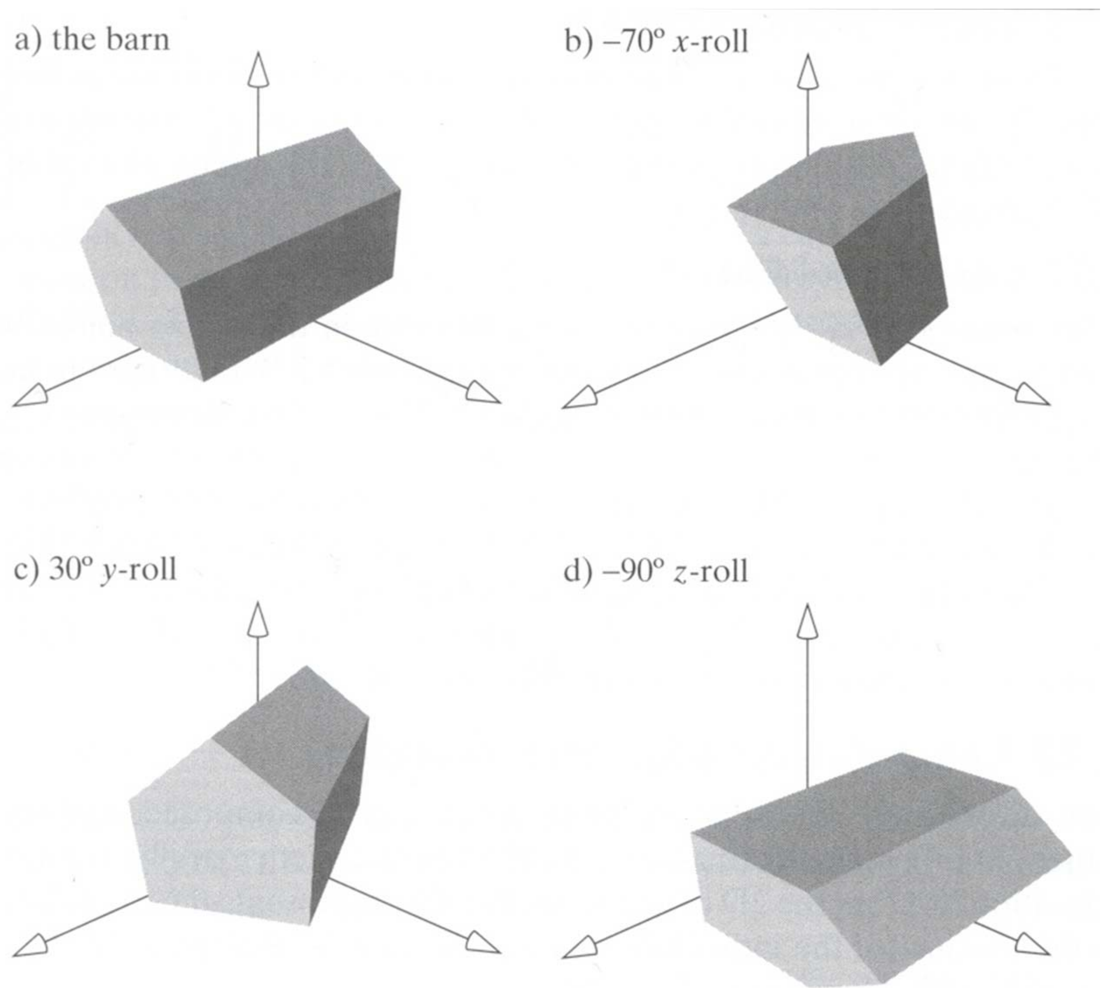
- Cannot do mindless conversion like before. Why?
  - Rotate about what axis?
  - 3D rotation: about a defined axis
  - Different Xform matrix for:
    - Rotation about x-axis
    - Rotation about y-axis
    - Rotation about z-axis
- New terminology
  - X-roll: rotation about x-axis
  - Y-roll: rotation about y-axis
  - Z-roll: rotation about z-axis

Rotating in 3D

- New terminology
  - X-roll: rotation about x-axis
  - Y-roll: rotation about y-axis
  - Z-roll: rotation about z-axis
- Which way is +ve rotation
  - Look in –ve direction (into +ve arrow)
  - CCW is +ve rotation



Rotating in 3D



Rotating in 3D

- For a rotation angle, β about an axis
- Define:

$$c = \cos(\beta) \qquad s = \sin(\beta)$$

A x-roll:

$$R_x(\beta) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & c & -s & 0 \\ 0 & s & c & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rotating in 3D

A y-roll:

$$R_y(\beta) = \begin{pmatrix} c & 0 & s & 0 \\ 0 & 1 & 0 & 0 \\ -s & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Rules:

- Rotate row, column int. is 1
- Rest of row/col is 0
- c,s in rect pattern

A z-roll:

$$R_z(\beta) = \begin{pmatrix} c & -s & 0 & 0 \\ s & c & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Example: Rotating in 3D

Q: Using y-roll equation, rotate *P* = (3,1,4) by 30 degrees:

A: c = cos(30) = 0.866, s = sin(30) = 0.5, and

$$Q = \begin{pmatrix} c & 0 & s & 0 \\ 0 & 1 & 0 & 0 \\ -s & 0 & c & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 3 \\ 1 \\ 4 \\ 1 \end{pmatrix} = \begin{pmatrix} 4.6 \\ 1 \\ 1.964 \\ 1 \end{pmatrix}$$

E.g. first line: 3.c + 1.0 + 4.s + 1.0 = 4.6

References

- Angel and Shreiner
- Hill and Kelley, appendix 4