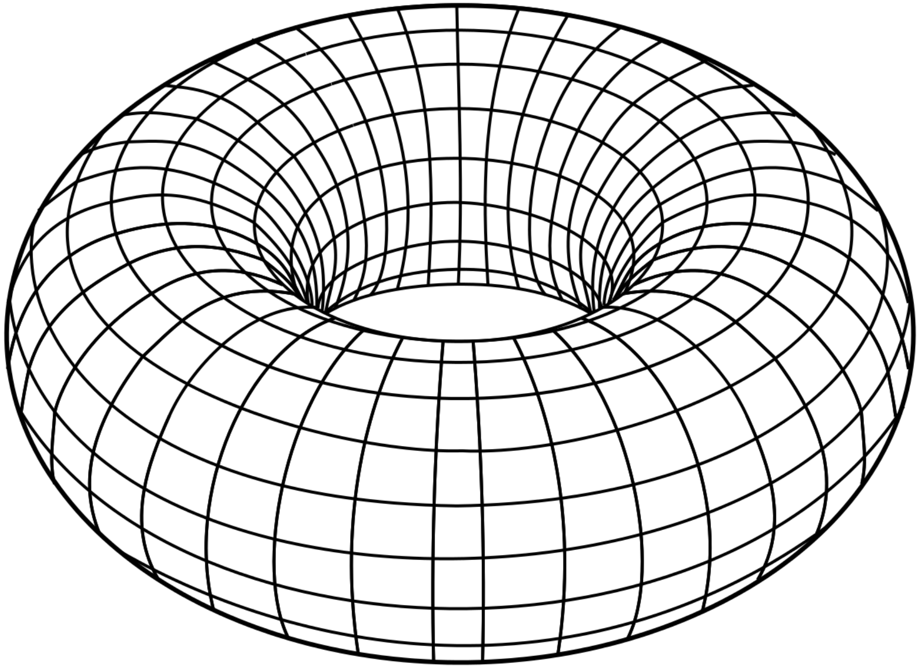
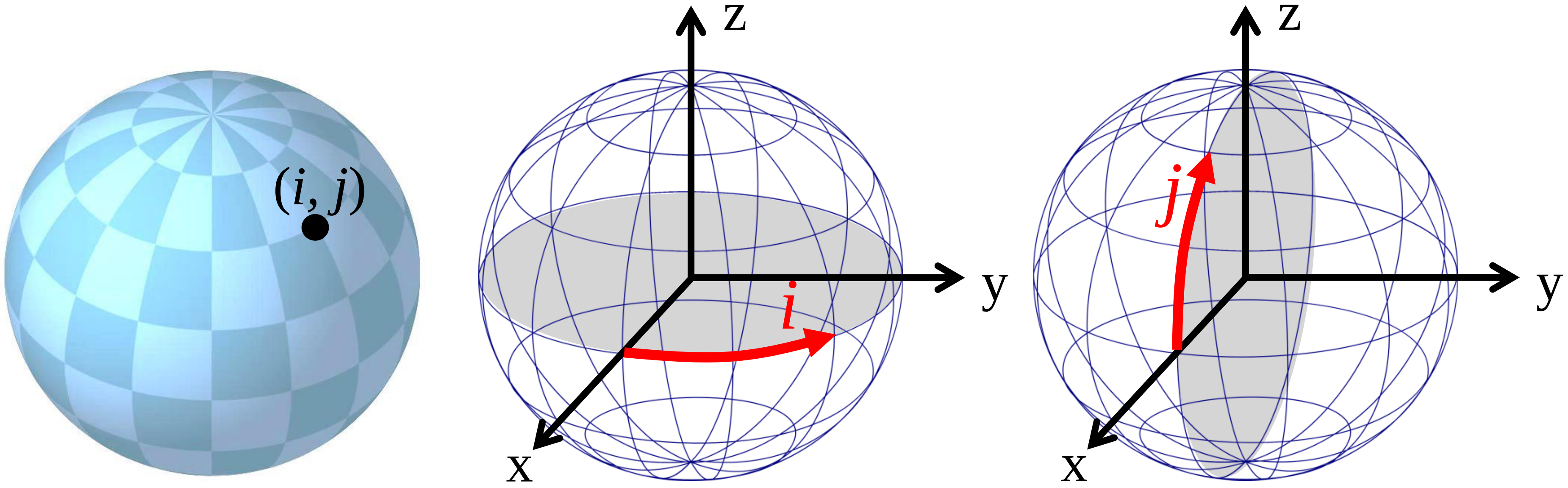
# Chapter 3:
# Other Geometric Objects

Sang Il Park
Dept. of Software

# Coding Practice : Sphere and Torus





(i, j)
z
y
x
i
z
y
x
j

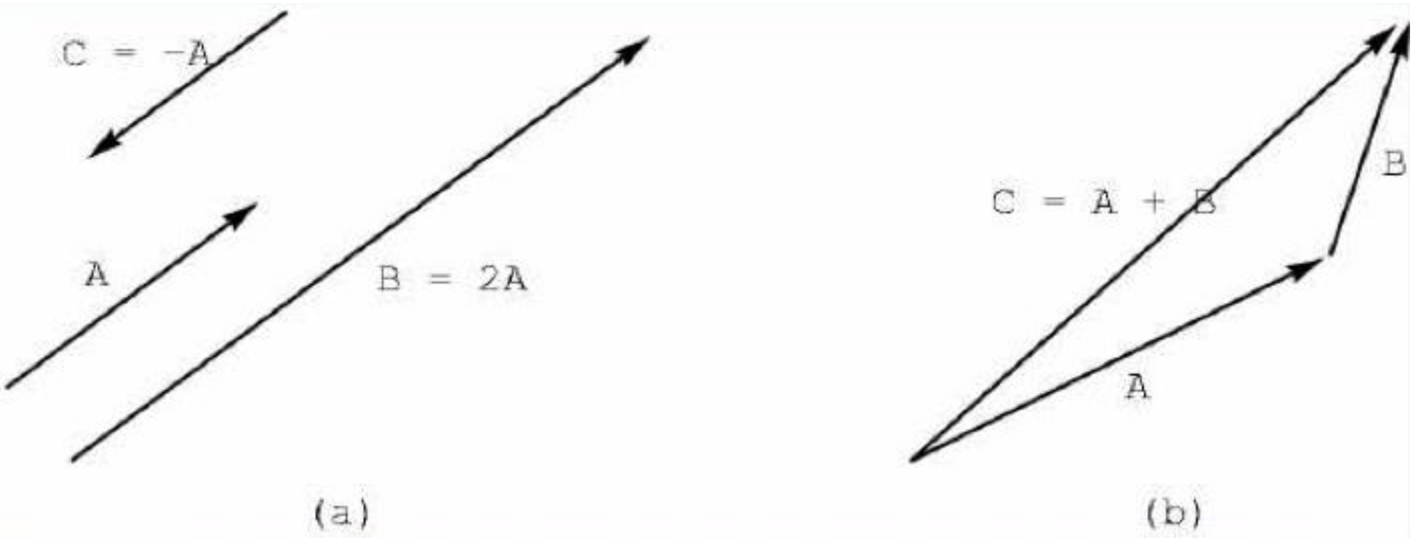
# Geometric objects and Its representations

# Scalars

- Scalars $\alpha, \beta, \gamma$ from a *scalar field*
- Operations $\alpha+\beta$, $\alpha\cdot\beta$, $0$, $1$, $-\alpha$, $(\ )^{-1}$
- "Expected" laws apply
- Examples: rationals or reals with addition and multiplication

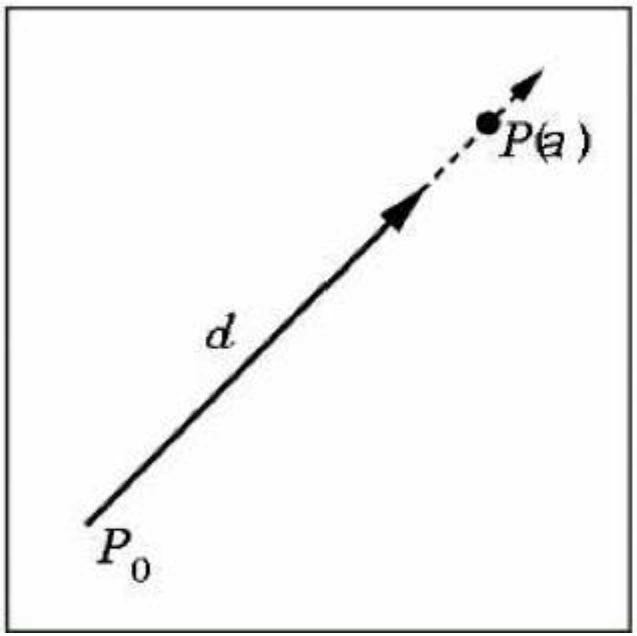
# Vectors

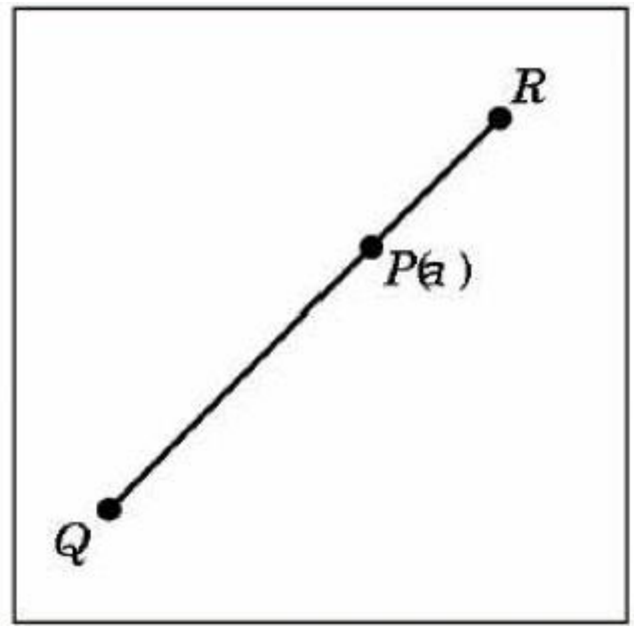
- Vectors *u, v, w* from a *vector space*
- Vector addition $u + v$ , subtraction $u - v$
- Zero vector **0**
- Scalar multiplication $\alpha\, v$



# Lines and line Segments

- Parametric form of line: $\mathbf{P}(\alpha) = \mathbf{P}_o + \alpha\,\mathbf{d}$




- Line segment between *Q* and *R*:

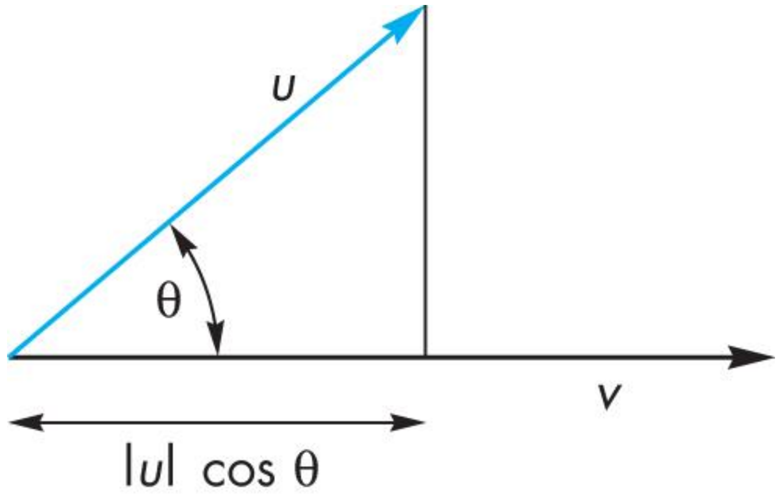
$$\mathbf{P}(\alpha) = (1-\alpha)\mathbf{Q} + \alpha\,\mathbf{R} \quad \mathit{for}\ 0 \le \alpha \le 1$$

# Dot Product (Projection)

- Dot product projects one vector onto another vector

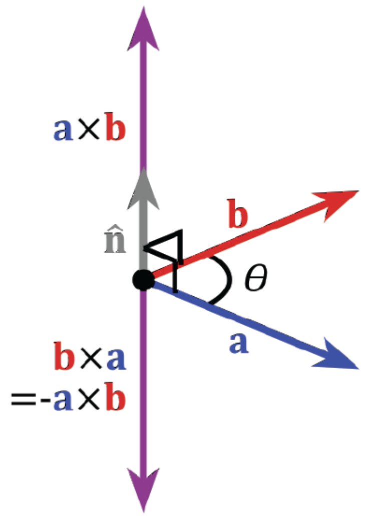
$$\mathbf{u} \cdot \mathbf{v} = \mathbf{u}_1\mathbf{v}_1 + \mathbf{u}_2\mathbf{v}_2 + \mathbf{u}_3\mathbf{v}_3 = |\mathbf{u}||\mathbf{v}|\cos(\theta)$$

$$pr_{\mathbf{v}}\mathbf{u} = (\mathbf{u} \cdot \mathbf{v})\,\mathbf{v} / |\mathbf{v}|^2$$

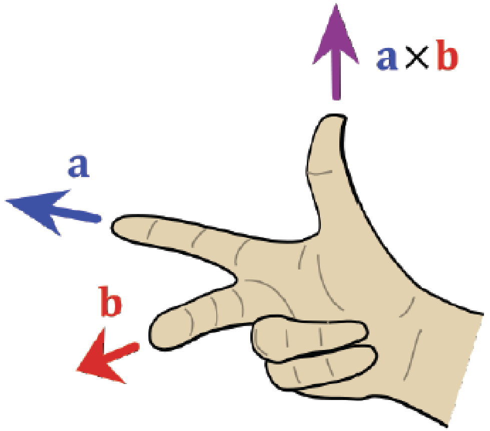


# Cross Product

$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{pmatrix}$$

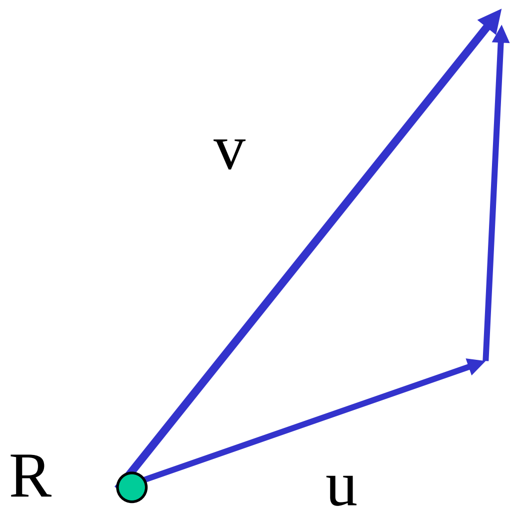


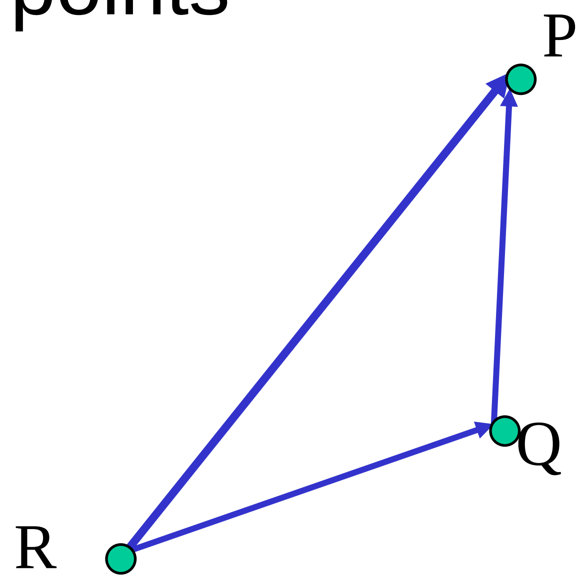

- $|\mathbf{a} \times \mathbf{b}| = |\mathbf{a}||\mathbf{b}||\sin(\theta)|$
- Cross product is perpendicular to both **a** and **b**
- Right-hand rule



# Planes

- A plane can be defined by a point and two vectors or by three points



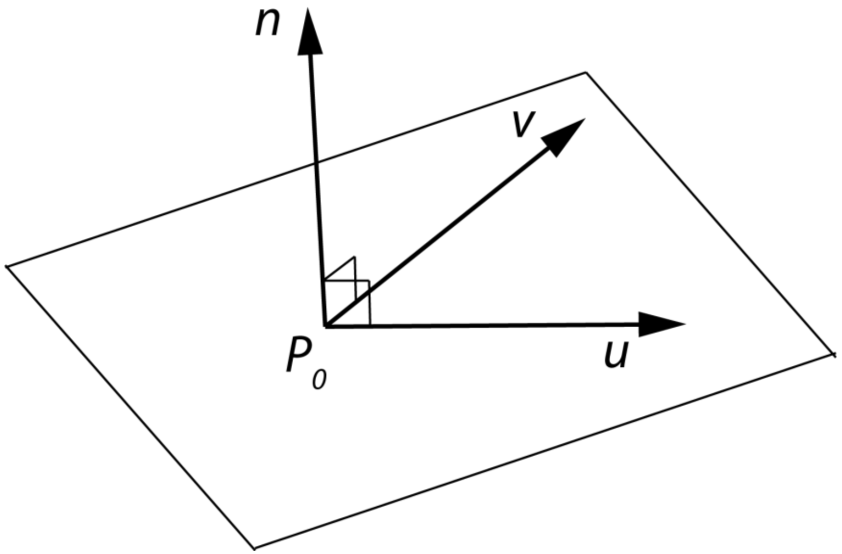




$P(\alpha,\beta)=R+\alpha u+\beta v$

$P(\alpha,\beta)=R+\alpha(Q-R)+\beta(P-Q)$

# Planes and normal

- Plane defined by point $P_0$ and vectors u and v
- u and v should not be parallel
- Parametric form:
  $T(\alpha, \beta) = P_0 + \alpha u + \beta v$
  ($\alpha$ and $\beta$ are scalars)

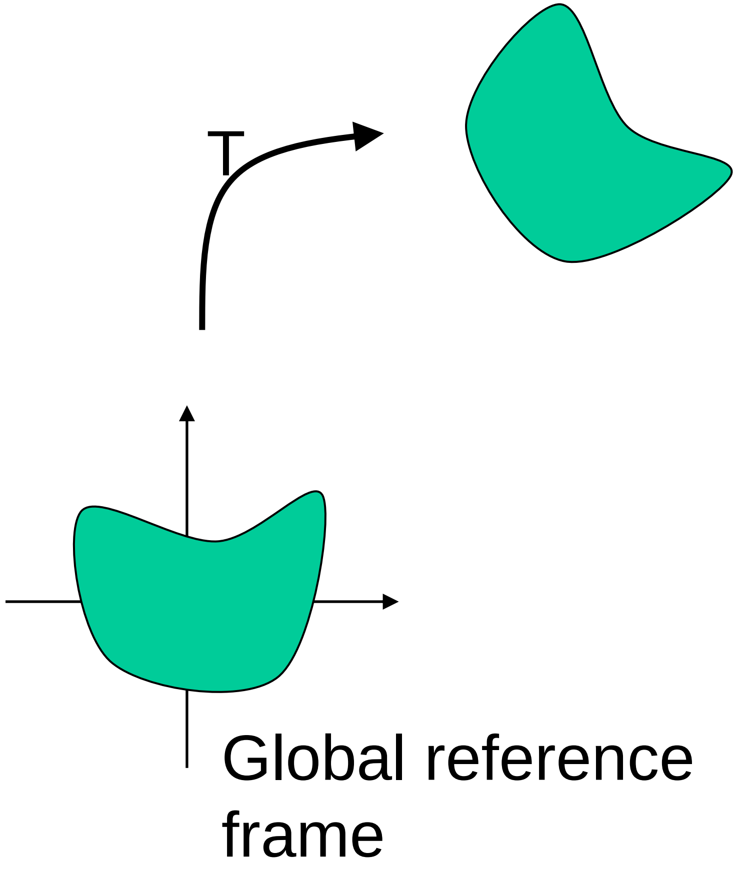



- $n = u \times v / |u \times v|$ is the normal
- $n \cdot (P - P_0) = 0$ if and only if P lies in plane

# Geometric Transformations

# Transformations

- Linear transformations
- Rigid transformations
- Affine transformations
- Projective transformations



# Homogeneous Coordinates

- Any affine transformation between 3D spaces can be represented by a 4x4 matrix

$$T(\mathbf{p}) = \begin{pmatrix} \mathbf{M}_{3\times 3} & \mathbf{T}_{3\times 1} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{p}_{3\times 1} \\ 1 \end{pmatrix}$$

- Affine transformation is *linear* in homogeneous coordinates

# Projective Spaces

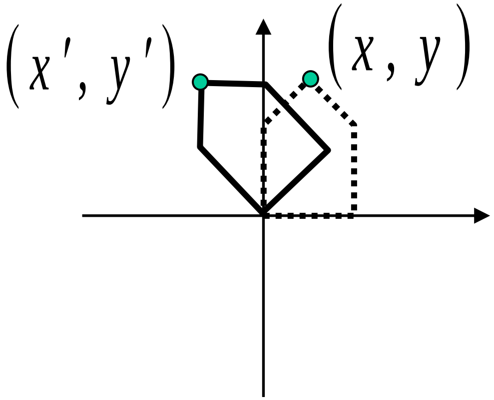
- Homogeneous coordinates
  - (x, y, z, w) = (x/w, y/w, z/w, 1)
  - Useful for handling perspective projection
- But, it is algebraically inconsistent !!

$$(1,0,0,1)+(1,1,0,1)=(2,1,0,2)=(1,\frac{1}{2},0,1)$$

$$(1,0,0,1)+(2,2,0,2)=(3,2,0,3)=(1,\frac{2}{3},0,1)$$

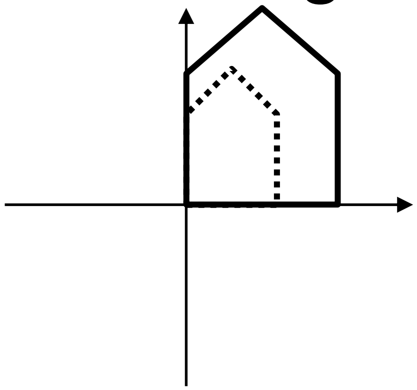
# Examples of Affine Transformations

- 2D rotation




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

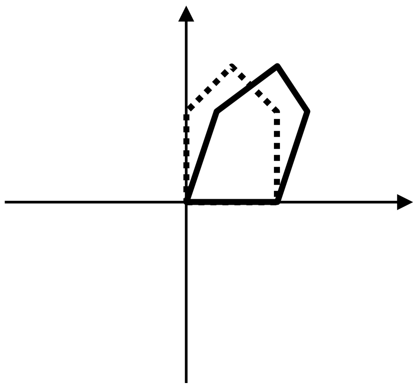
- 2D scaling



$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} s_x x \\ s_y y \\ 1 \end{pmatrix}$$

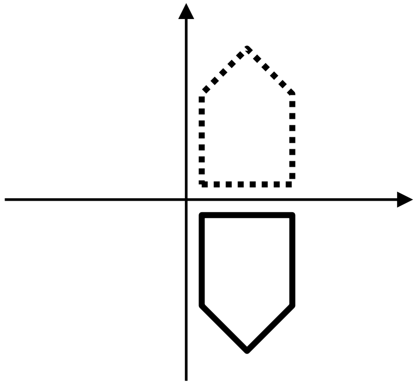
# Examples of Affine Transformations

- 2D shear



$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & d & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x + dy \\ y \\ 1 \end{pmatrix}$$

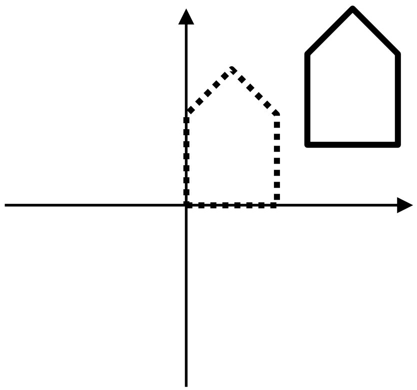
- 2D reflection



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

# Examples of Affine Transformations

- 2D translation

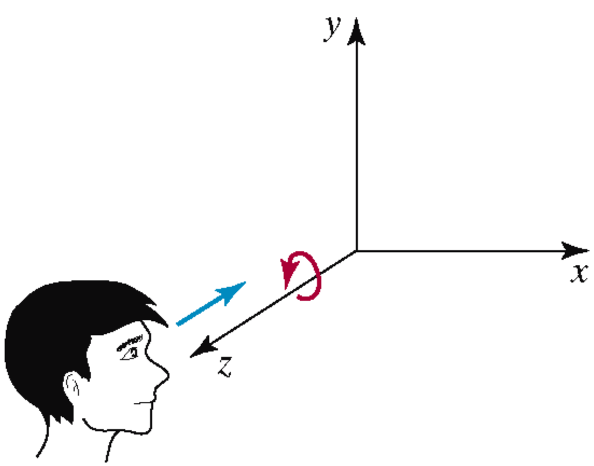


$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x + t_x \\ y + t_y \\ 1 \end{pmatrix}$$

# Examples of Affine Transformations

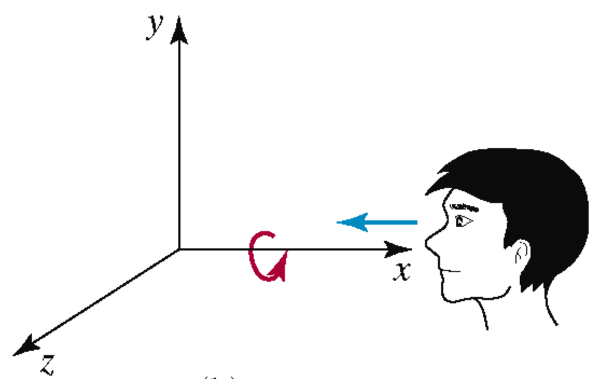
- 3D rotation

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



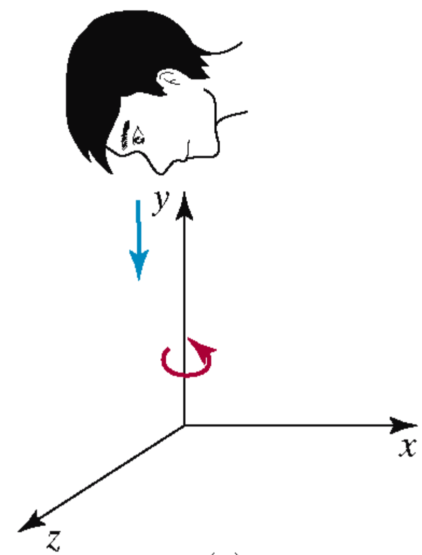

(a)

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




(b)

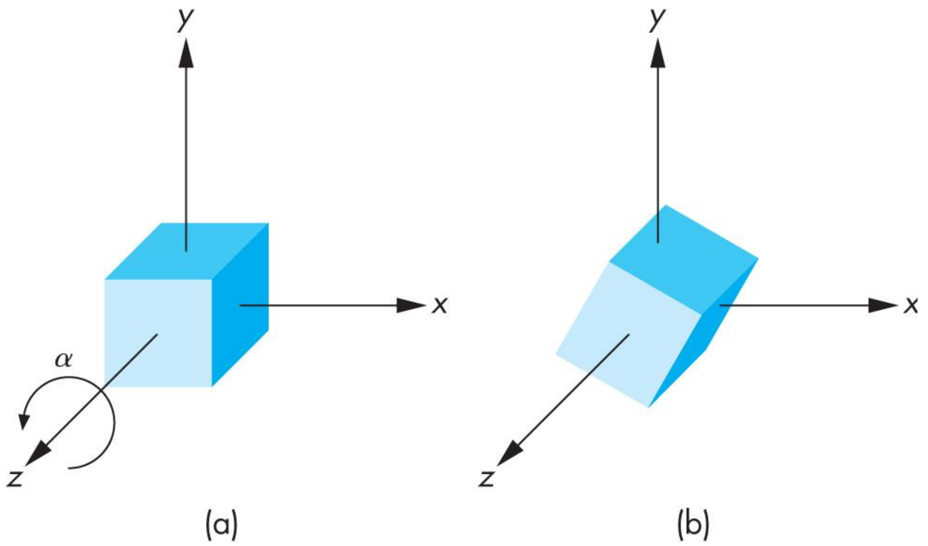
$$\begin{pmatrix} x' \\ y' \\ z' \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$




(c)

# 3D Rotation Matrix about Z Axis

$$\mathbf{R} = \mathbf{R}_Z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



# 3D Rotation about x and y axes

- Same argument as for rotation about *z* axis
  - For rotation about *x* axis, *x* is unchanged
  - For rotation about *y* axis, *y* is unchanged

$$\mathbf{R} = \mathbf{R}_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R} = \mathbf{R}_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

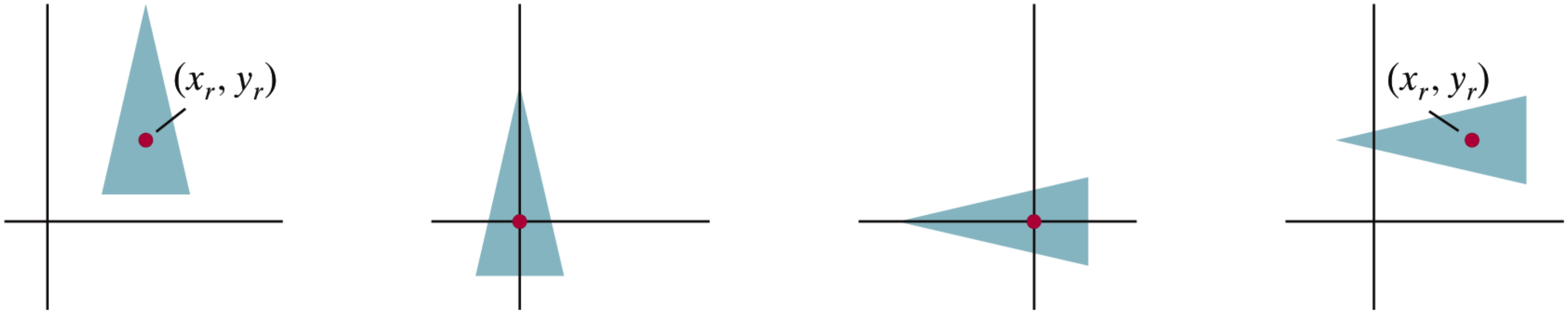
# 2D Pivot-Point Rotation

- Rotation with respect to a pivot point (x,y)

$$T(x,y)\cdot R(\theta)\cdot T(-x,-y)$$

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

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




(a) (b) (c) (d)

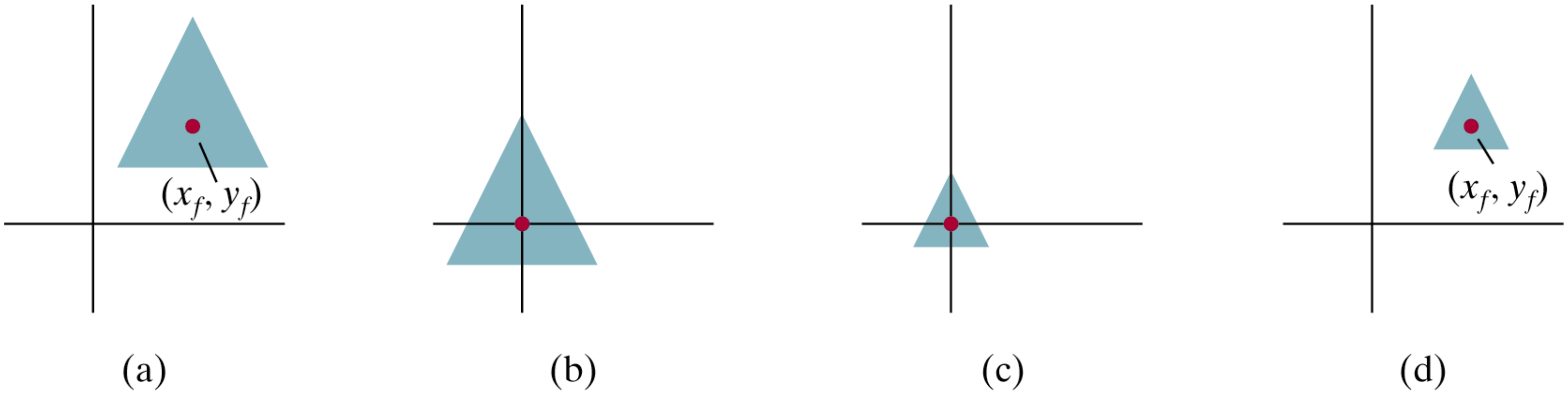
# 2D Fixed-Point Scaling

- Scaling with respect to a fixed point (x,y)

$$T(x,y)\cdot S(s_x,s_y)\cdot T(-x,-y)$$

$$=\begin{pmatrix}1 & 0 & x\\ 0 & 1 & y\\ 0 & 0 & 1\end{pmatrix}\cdot\begin{pmatrix}s_x & 0 & 0\\ 0 & s_y & 0\\ 0 & 0 & 1\end{pmatrix}\cdot\begin{pmatrix}1 & 0 & -x\\ 0 & 1 & -y\\ 0 & 0 & 1\end{pmatrix}$$

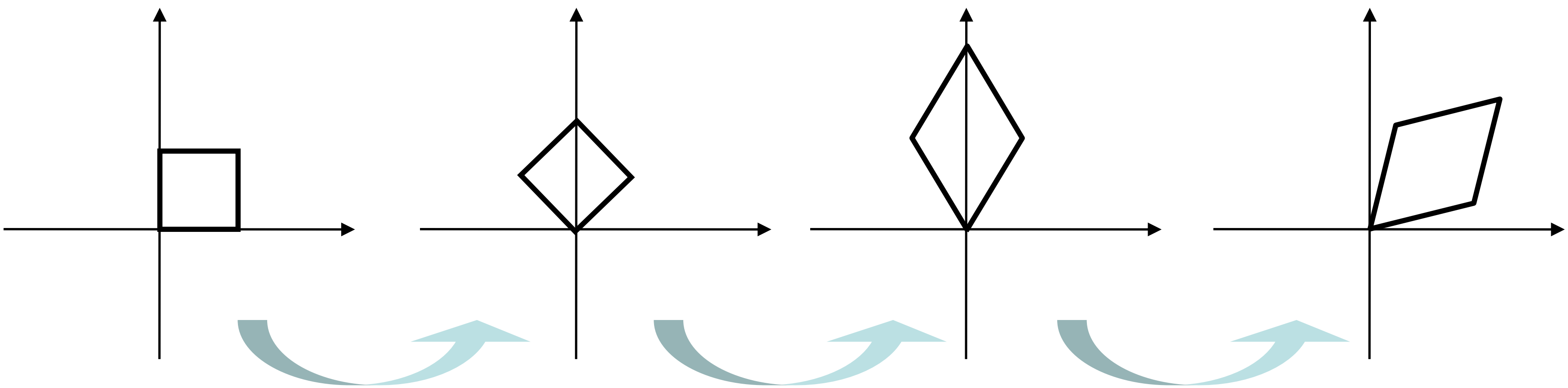
$$=\begin{pmatrix}s_x & 0 & (1-s_x)\cdot x\\ 0 & s_y & (1-s_y)\cdot y\\ 0 & 0 & 1\end{pmatrix}$$



# Scaling Direction

- Scaling along an arbitrary axis

$$R^{-1}(\theta) \cdot S(s_x, s_y) \cdot R(\theta)$$



$R(\theta)$ $S(s_x, s_y)$ $R^{-1}(\theta)$

# Properties of Affine Transformations

- Any ***affine transformation*** between 3D spaces can be represented as a combination of a ***linear transformation*** followed by ***translation***
- An affine transf. maps ***lines*** to ***lines***
- An affine transf. maps ***parallel lines*** to ***parallel lines***
- An affine transf. preserves ***ratios of distance*** along a line
- An affine transf. does not preserve absolute distances and angles

# Rigid Transformations

- A ***rigid transformation*** $T$ is a mapping between affine spaces
  - $T$ maps vectors to vectors, and points to points
  - $T$ preserves distances between all points
  - $T$ preserves cross product for all vectors (to avoid reflection)
- In 3-spaces, T can be represented as

$$T(\mathbf{p}) = \mathbf{R}_{3\times 3}\mathbf{p}_{3\times 1} + \mathbf{T}_{3\times 1}, \quad \text{where}$$

$$\mathbf{R}\mathbf{R}^T = \mathbf{R}^T\mathbf{R} = \mathbf{I} \quad \text{and} \quad \det \mathbf{R} = 1$$

# Rigid Body Rotation

- Rigid body transformations allow only rotation and translation

- Rotation matrices form SO(3)
  - Special orthogonal group

$$RR^T = R^T R = I$$ (Distance preserving)

$$\det R = 1$$ (No reflection)

# Rigid Body Rotation

- R is normalized
  - The squares of the elements in any row or column sum to 1

$$R R^T = R^T R = I$$

- R is orthogonal
  - The dot product of any pair of rows or any pair columns is 0

- The rows (columns) of R correspond to the vectors of the principle axes of the rotated coordinate frame

# 3D Rotation About Arbitrary Axis

- How to rotate around **u** vector
  (**u** = given rotation axis)

➔ Rotate about x and y axes to make **u** align with the *z*-axis

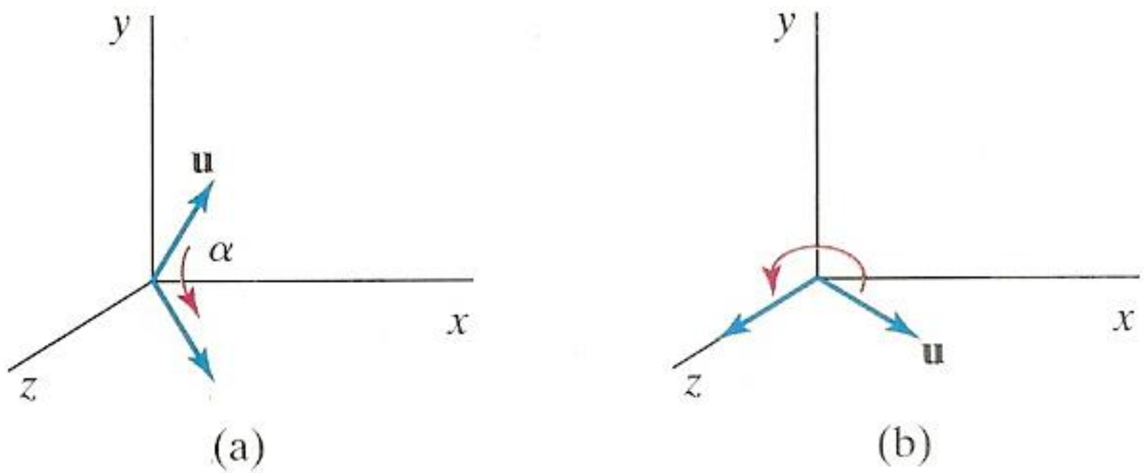



FIGURE 5-45 Unit vector **u** is rotated about the *x* axis to bring it into the *xz* plane (a), then it is rotated around the *y* axis to align it with the *z* axis (b).

# 3D Rotation About Arbitrary Axis

- Rotate **u** onto the z-axis
  - **u'**: Project **u** onto the yz-plane to compute angle $\alpha$
  - **u''**: Rotate **u** about the x-axis by angle $\alpha$
  - Rotate **u''** onto the z-asis

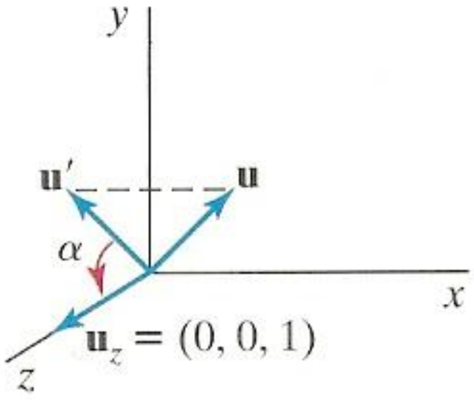



FIGURE 5-46 Rotation of $\mathbf{u}$ around the $x$ axis into the $xz$ plane is accomplished by rotating $\mathbf{u}'$ (the projection of $\mathbf{u}$ in the $yz$ plane) through angle $\alpha$ onto the $z$ axis.

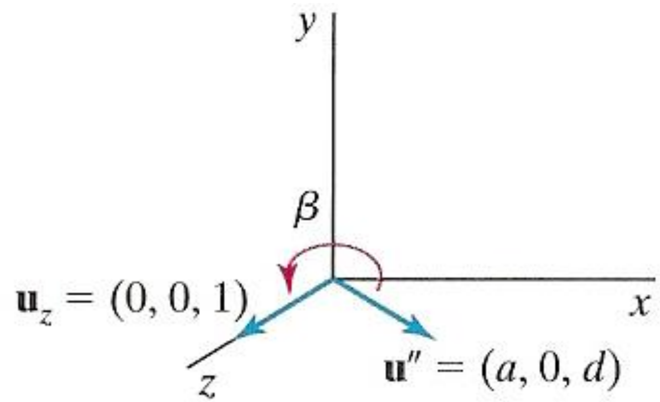



FIGURE 5-47 Rotation of unit vector $\mathbf{u}''$ (vector $\mathbf{u}$ after rotation into the $xz$ plane) about the $y$ axis. Positive rotation angle $\beta$ aligns $\mathbf{u}''$ with vector $\mathbf{u}_z$.

# 3D Rotation About Arbitrary Axis

- Rotate **u'** about the x-axis onto the *z*-axis
  - Let **u**=(a,b,c) and thus **u'**=(0,b,c)
  - Let $\mathbf{u}_z$=(0,0,1)

$$\cos\alpha = \frac{\mathbf{u}' \cdot \mathbf{u}_z}{\|\mathbf{u}'\| \|\mathbf{u}_z\|} = \frac{c}{\sqrt{b^2 + c^2}}$$

$$\mathbf{u}' \times \mathbf{u}_z = \mathbf{u}_x \|\mathbf{u}'\| \|\mathbf{u}_z\| \sin\alpha$$
$$= \mathbf{u}_x \cdot b$$

$$\Rightarrow \quad \sin\alpha = \frac{b}{\|\mathbf{u}'\| \|\mathbf{u}_z\|} = \frac{b}{\sqrt{b^2 + c^2}}$$

# 3D Rotation About Arbitrary Axis

- Rotate **u'** about the x-axis onto the *z*-axis
  - Since we know both cos α and sin α, the rotation matrix can be obtained

$$\mathbf{R}_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{c}{\sqrt{b^2+c^2}} & \frac{-b}{\sqrt{b^2+c^2}} & 0 \\ 0 & \frac{b}{\sqrt{b^2+c^2}} & \frac{c}{\sqrt{b^2+c^2}} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

  - Or, we can compute the signed angle α

$$\mathrm{atan2}\left(\frac{c}{\sqrt{b^2+c^2}}, \frac{b}{\sqrt{b^2+c^2}}\right)$$

  - Do not use acos() since its domain is limited to [-1,1]

# Gimble

- Hardware implementation of Euler angles
- Aircraft, Camera

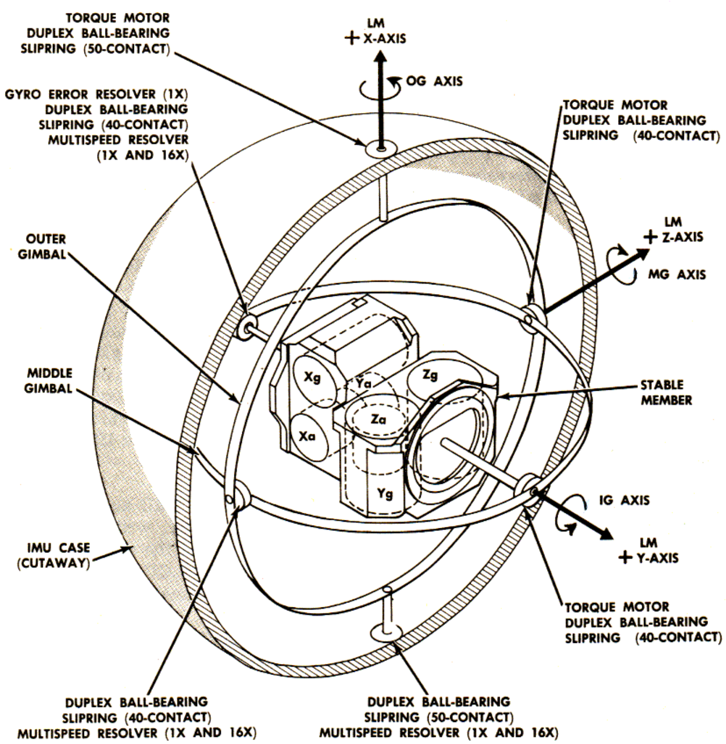
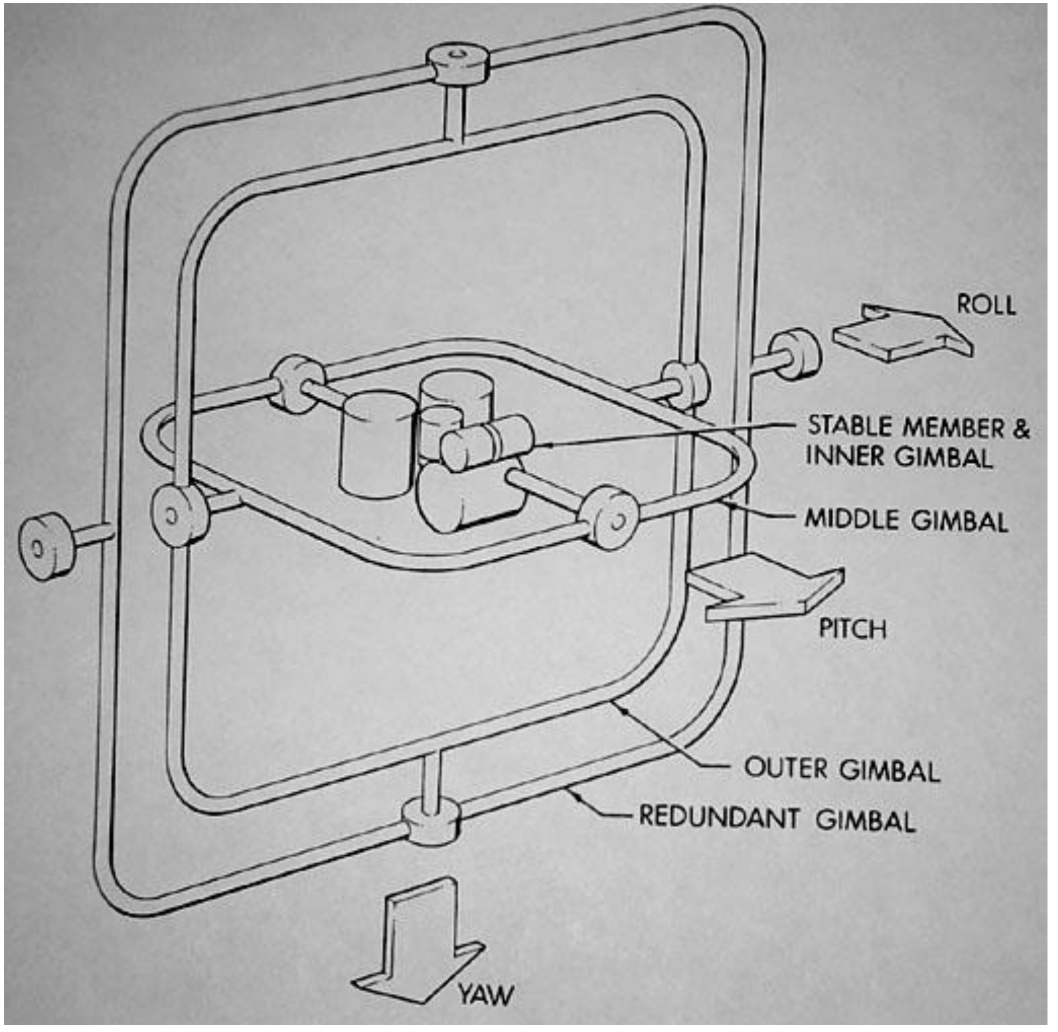



Figure 2.1-24. IMU Gimbal Assembly

# Euler Angles

- Rotation about three orthogonal axes
  - 12 combinations
    - XYZ, XYX, XZY, XZX
    - YZX, YZY, YXZ, YXY
    - ZXY, ZXZ, ZYX, ZYZ

- ***Gimble lock***
  - Coincidence of inner most and outmost gimbles' rotation axes
  - Loss of degree of freedom



# Euler angles

- Arbitrary rotation can be represented by three rotation along x,y,z axis

$$R_{XYZ}(\gamma,\beta,\alpha) = R_z(\alpha)R_y(\beta)R_x(\gamma)$$

$$= \begin{bmatrix} C\alpha C\beta & C\alpha S\beta S\gamma - S\alpha C\gamma & C\alpha S\beta C\gamma + S\alpha S\gamma & 0 \\ S\alpha C\beta & S\alpha S\beta S\gamma + C\alpha C\gamma & S\alpha S\beta C\gamma - C\alpha S\gamma & 0 \\ -S\beta & C\beta S\gamma & C\beta C\gamma & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

# Euler Angles

- Euler angles are ambiguous
  - Two different Euler angles can represent the same orientation

$$R_1 = (r_x, r_y, r_z) = (\theta, \frac{\pi}{2}, 0) \quad \text{and} \quad R_2 = (0, \frac{\pi}{2}, -\theta)$$

  - This ambiguity brings unexpected results of animation where frames are generated by interpolation.

# Smooth Rotation

- Create transformations from $\mathbf{M_0}$ to $\mathbf{M_n}$ ***smoothly***
  - Problem: find a sequence of model-view matrices $\mathbf{M_0, M_1, \ldots\ldots, M_n}$ for each frame to see a smooth transition

- One solution for rotation (using Euler angles):
  - Find $\mathbf{R_0 = R_{0z} R_{0y} R_{0x}}$ and $\mathbf{R_n = R_{nz} R_{ny} R_{nx}}$
  - Then, Create a sequence of rotation $\mathbf{R_0, R_1, \ldots\ldots, R_n}$ : $\mathbf{R_i = R_{iz} R_{iy} R_{ix}}$ (where, ix, iy, iz is the interpolated angles from the beginning and the end)
  - Not very effective!
  - Quaternions can do it better!

# Quaternions

- Extension of imaginary numbers from two to three dimensions
- Requires one real and three imaginary components $\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$

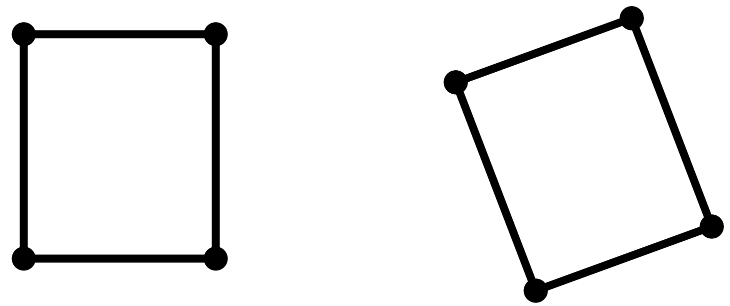
$$q = q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k}$$

- Quaternions can express rotations on sphere smoothly and efficiently. Process:
  - Model-view matrix $\rightarrow$ quaternion
  - Carry out operations with quaternions
  - Quaternion $\rightarrow$ Model-view matrix

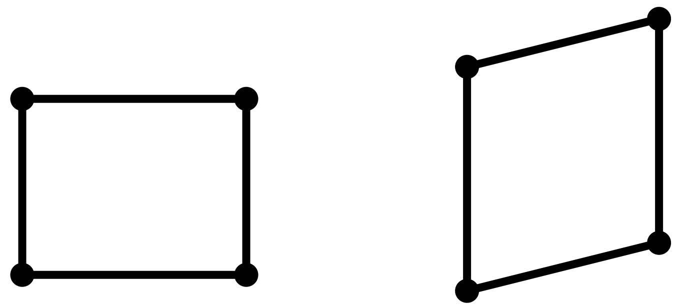
Computer Animation 수업에서 다룹니다.

# Taxonomy of Transformations

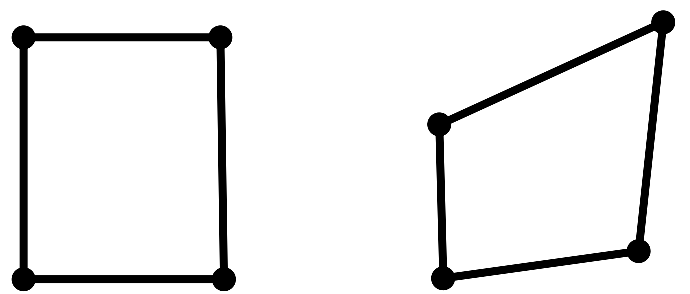
Rigid



Affine



Projective



# Composite Transformations

- Composite 2D Translation

$$T = \mathbf{T}(t_{x1}, t_{y1}) \cdot \mathbf{T}(t_{x2}, t_{y2})$$
$$= \mathbf{T}(t_{x1} + t_{x2}, t_{y1} + t_{y2})$$

$$\begin{pmatrix} 1 & 0 & t_{x2} \\ 0 & 1 & t_{y2} \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 0 & t_{x1} \\ 0 & 1 & t_{y1} \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & t_{x1} + t_{x2} \\ 0 & 1 & t_{y1} + t_{y2} \\ 0 & 0 & 1 \end{pmatrix}$$

# Composite Transformations

- Composite 2D Scaling

$$T = \mathbf{S}(s_{x1}, s_{y1}) \cdot \mathbf{S}(s_{x2}, s_{y2})$$
$$= \mathbf{S}(s_{x1}s_{x2}, s_{y1}s_{y2})$$

$$\begin{pmatrix} s_{x2} & 0 & 0 \\ 0 & s_{y2} & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} s_{x1} & 0 & 0 \\ 0 & s_{y1} & 0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} s_{x1} \cdot s_{x2} & 0 & 0 \\ 0 & s_{y1} \cdot s_{y2} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

# Composite Transformations

- Composite 2D Rotation

$$T = \mathbf{R}(\theta_2)\cdot\mathbf{R}(\theta_1)$$
$$= \mathbf{R}(\theta_2+\theta_1)$$

$$\begin{pmatrix} \cos\theta_2 & -\sin\theta_2 & 0 \\ \sin\theta_2 & \cos\theta_2 & 0 \\ 0 & 0 & 1 \end{pmatrix}\cdot\begin{pmatrix} \cos\theta_1 & -\sin\theta_1 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{pmatrix}=\begin{pmatrix} \cos(\theta_2+\theta_1) & -\sin(\theta_2+\theta_1) & 0 \\ \sin(\theta_2+\theta_1) & \cos(\theta_2+\theta_1) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$