CMPE 466 COMPUTER GRAPHICS

Chapter 9

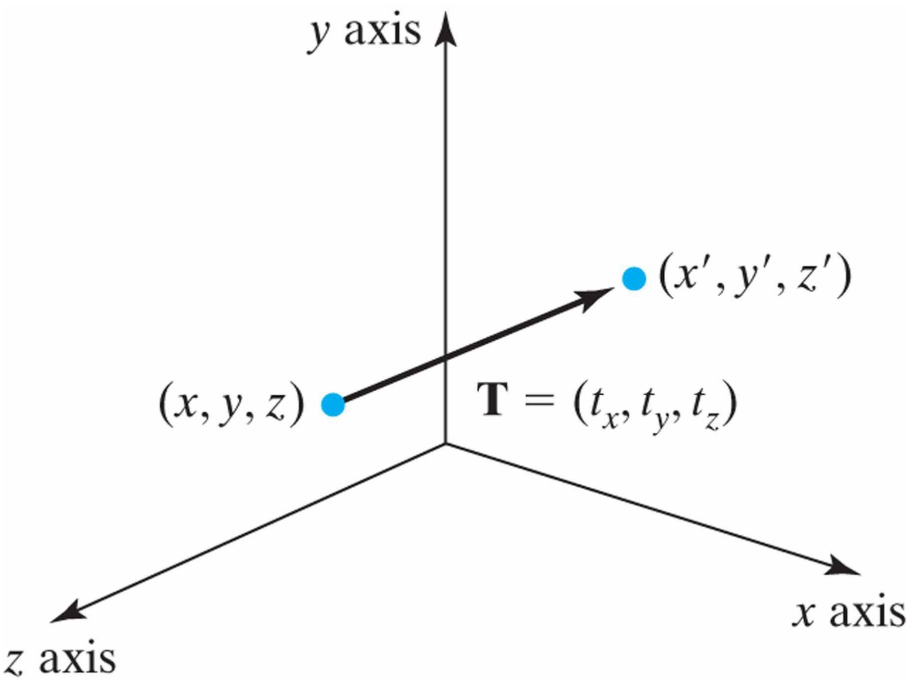
3D Geometric Transformations

Instructor: D. Arifler

Material based on
- *Computer Graphics with OpenGL®,* Fourth Edition by Donald Hearn, M. Pauline Baker, and Warren R. Carithers
- *Fundamentals of Computer Graphics*, Third Edition by by Peter Shirley and Steve Marschner
- *Computer Graphics* by F. S. Hill

3D translation

Figure 9-1 Moving a coordinate position with translation vector T = (t_x, t_y, t_z) .

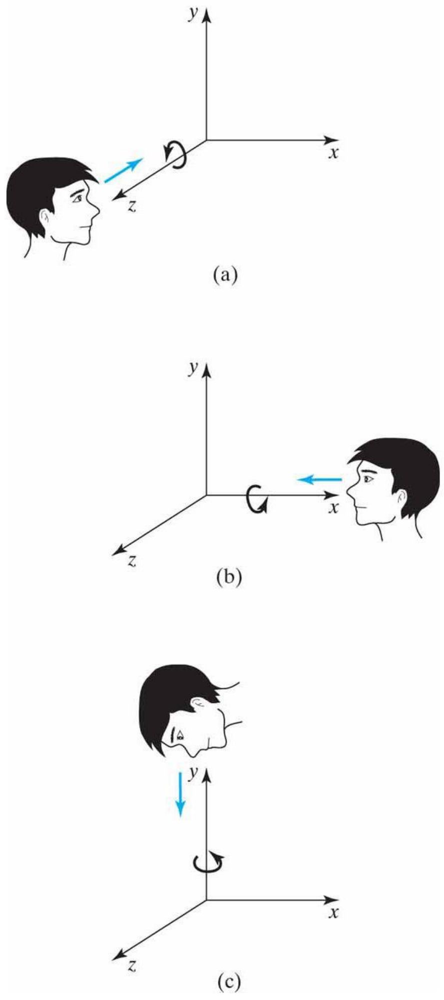


$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\mathbf{P}' = \mathbf{T} \cdot \mathbf{P}$$

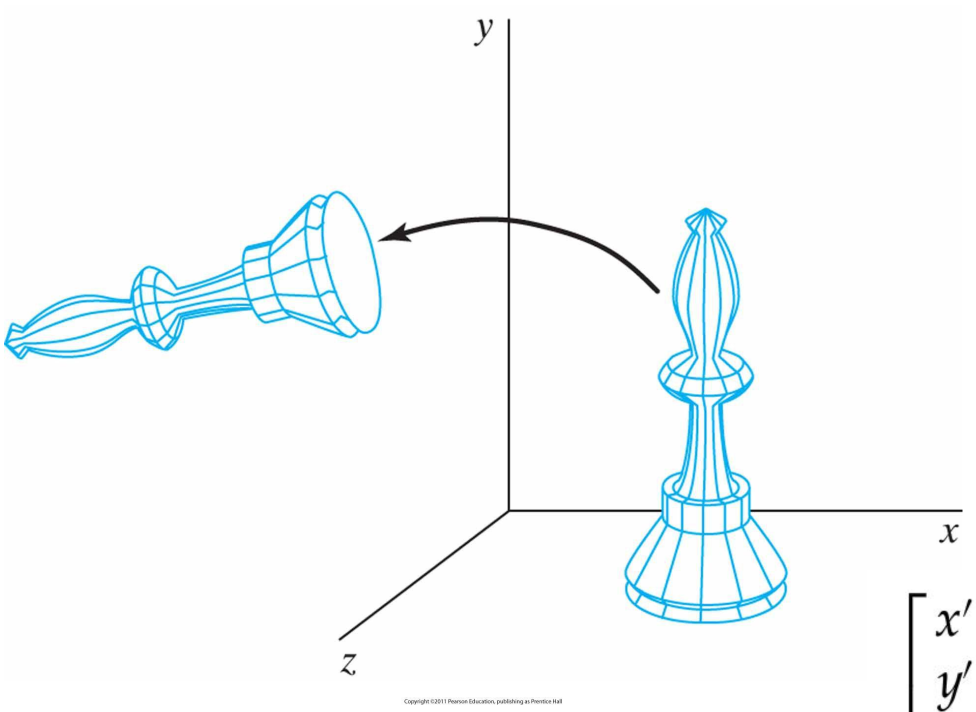
3D rotation

Figure 9-3 Positive rotations about a coordinate axis are counterclockwise, when looking along the positive half of the axis toward the origin.



3D z-axis rotation

Figure 9-4 Rotation of an object about the *z* axis.



$$x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta$$
$$z' = z$$

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

Rotations

- To obtain rotations about other two axes
  - x □ y □ z □ x
- E.g. x-axis rotation

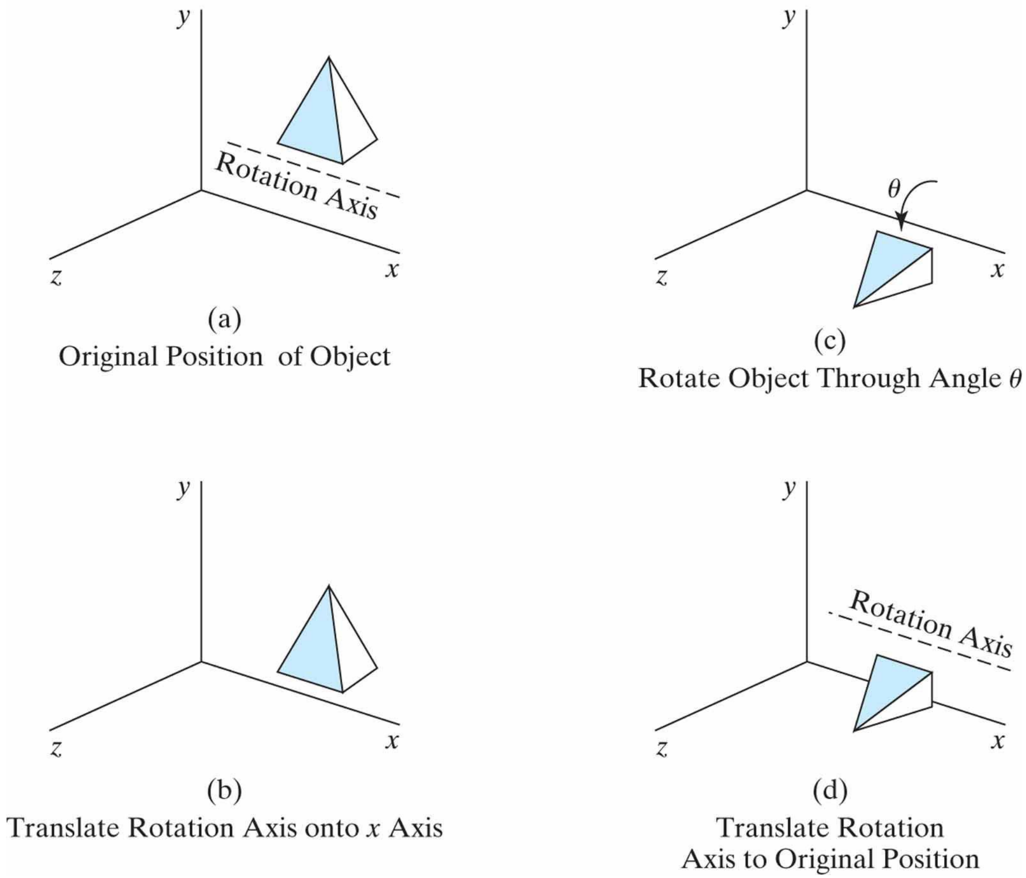
$$y' = y\cos\theta - z\sin\theta$$
$$z' = y\sin\theta + z\cos\theta$$
$$x' = x$$

- E.g. y-axis rotation

$$z' = z\cos\theta - x\sin\theta$$
$$x' = z\sin\theta + x\cos\theta$$
$$y' = y$$

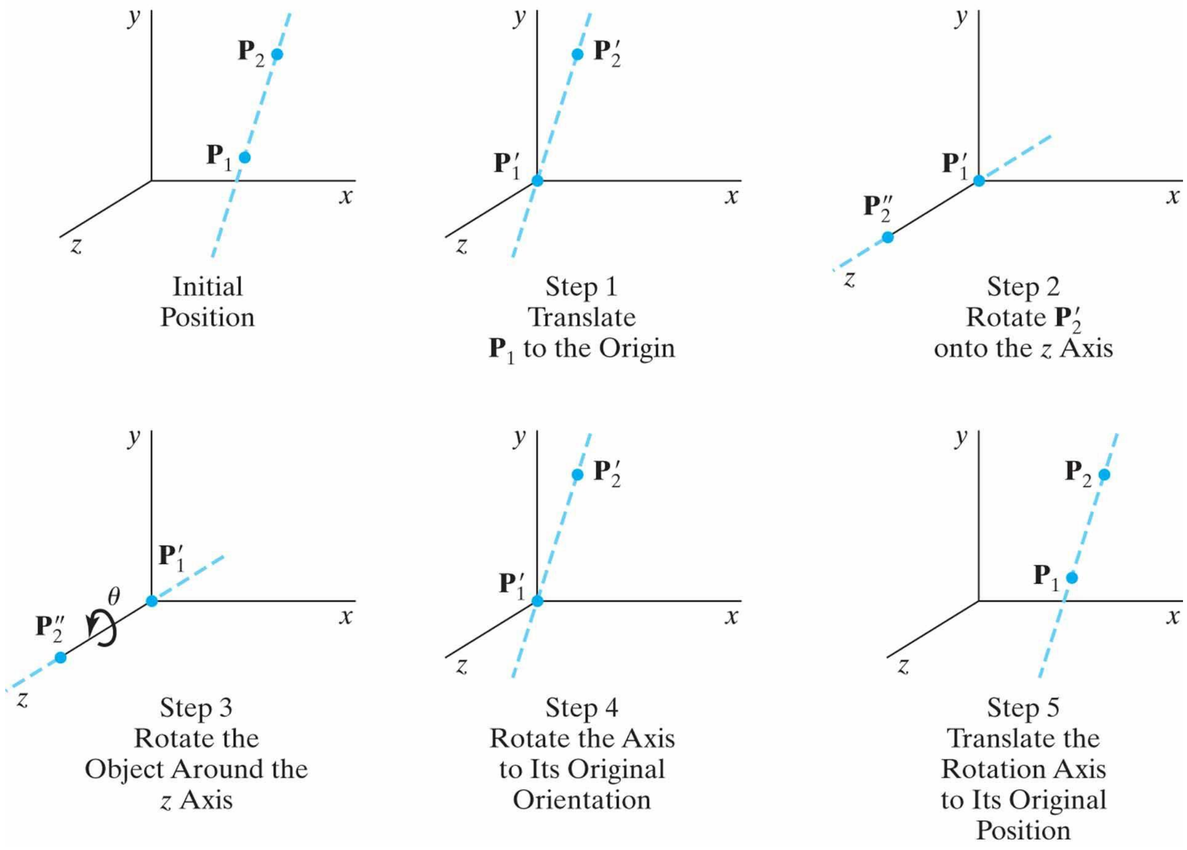
General 3D rotations

Figure 9-8 Sequence of transformations for rotating an object about an axis that is parallel to the *x* axis.



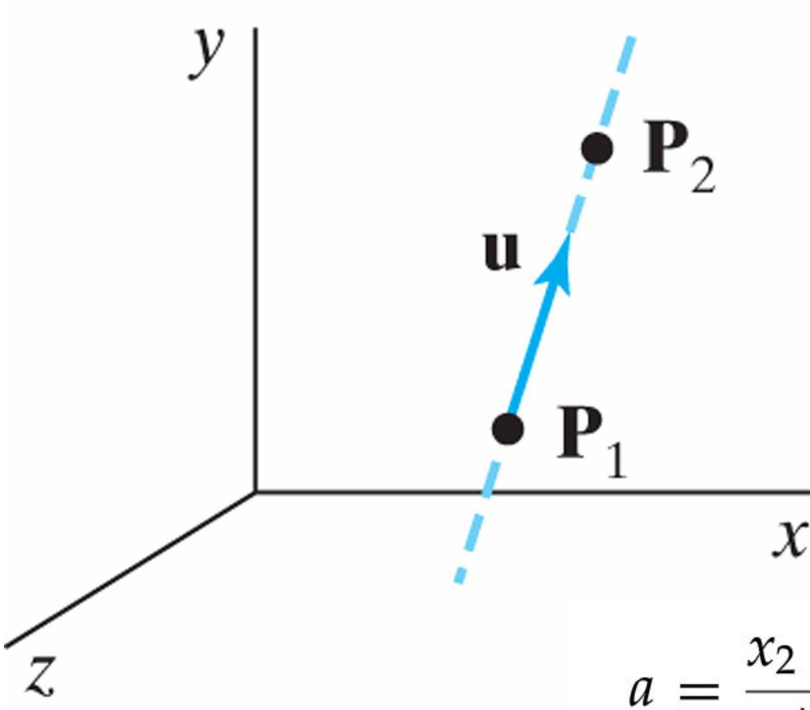
Arbitrary rotations

Figure 9-9 Five transformation steps for obtaining a composite matrix for rotation about an arbitrary axis, with the rotation axis projected onto the *z* axis.



Arbitrary rotations

Figure 9-10 An axis of rotation (dashed line) defined with points $\mathbf{P}_1$ and $\mathbf{P}_2$. The direction for the unit axis vector **u** is determined by the specified rotation direction.



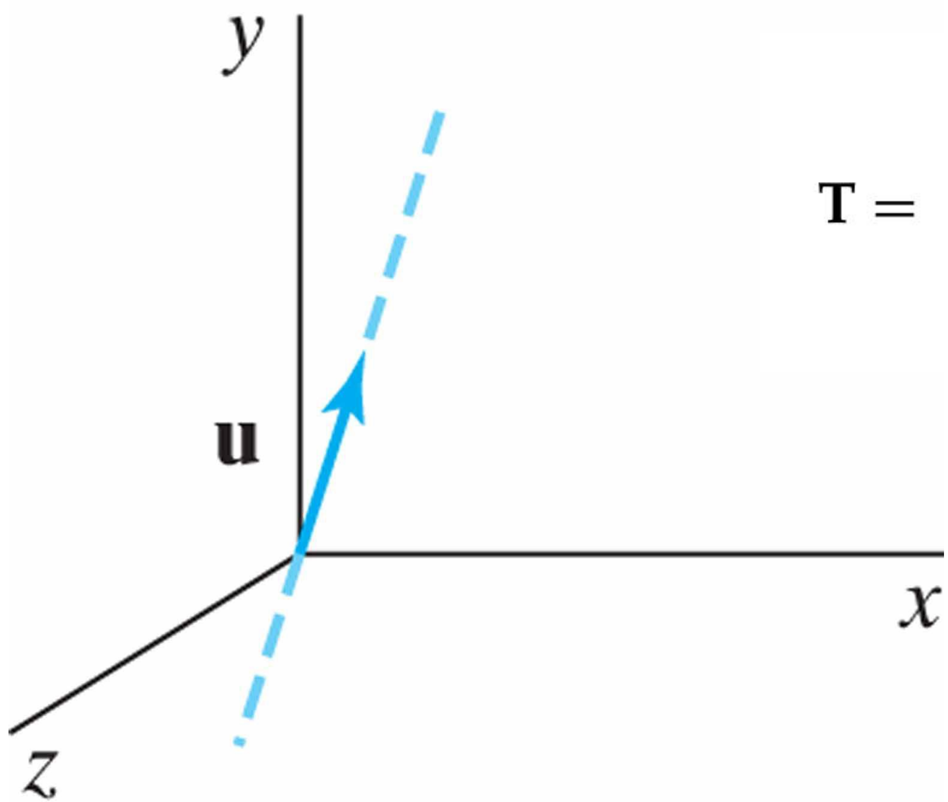
$$\mathbf{V} = \mathbf{P}_2 - \mathbf{P}_1 = (x_2 - x_1,\, y_2 - y_1,\, z_2 - z_1)$$

$$\mathbf{u} = \frac{\mathbf{V}}{|\mathbf{V}|} = (a, b, c)$$

$$a = \frac{x_2 - x_1}{|\mathbf{V}|}, \qquad b = \frac{y_2 - y_1}{|\mathbf{V}|}, \qquad c = \frac{z_2 - z_1}{|\mathbf{V}|}$$

Rotations

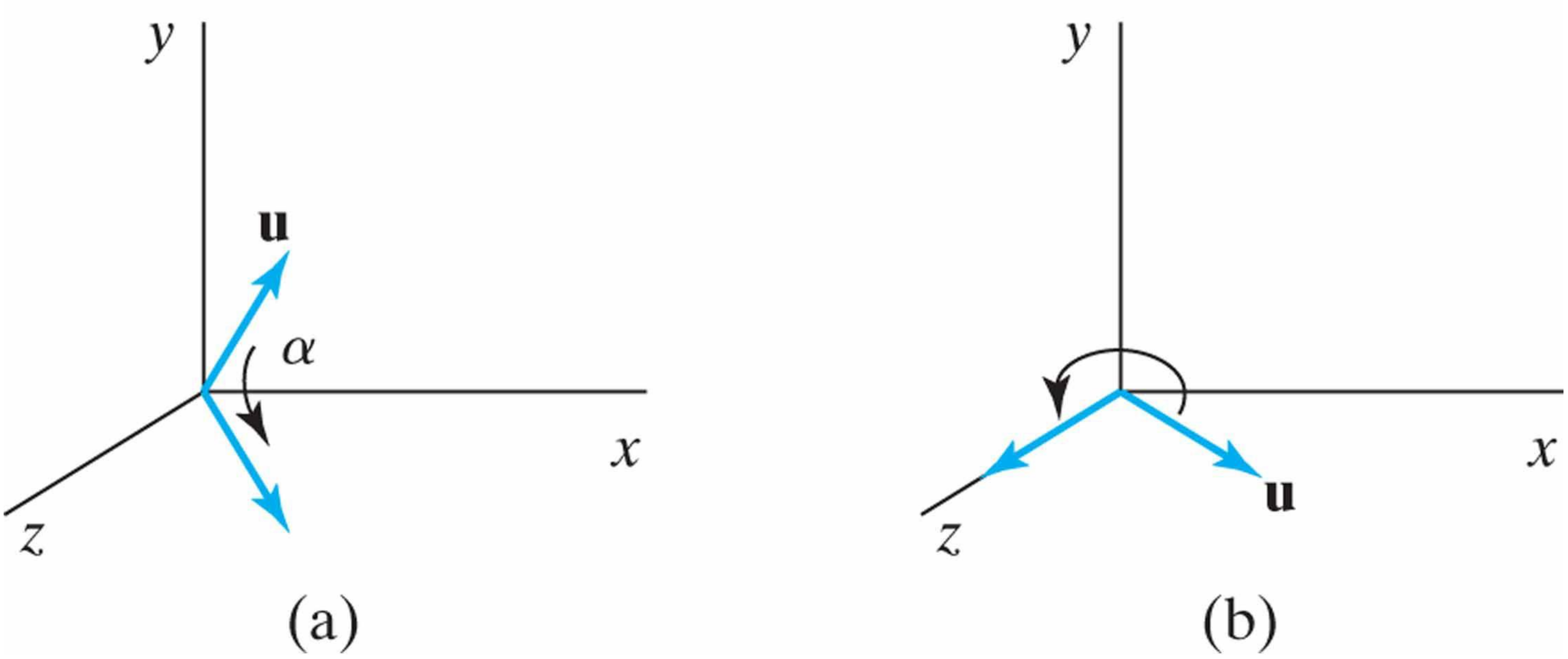
Figure 9-11 Translation of the rotation axis to the coordinate origin.



$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & -x_1 \\ 0 & 1 & 0 & -y_1 \\ 0 & 0 & 1 & -z_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotations

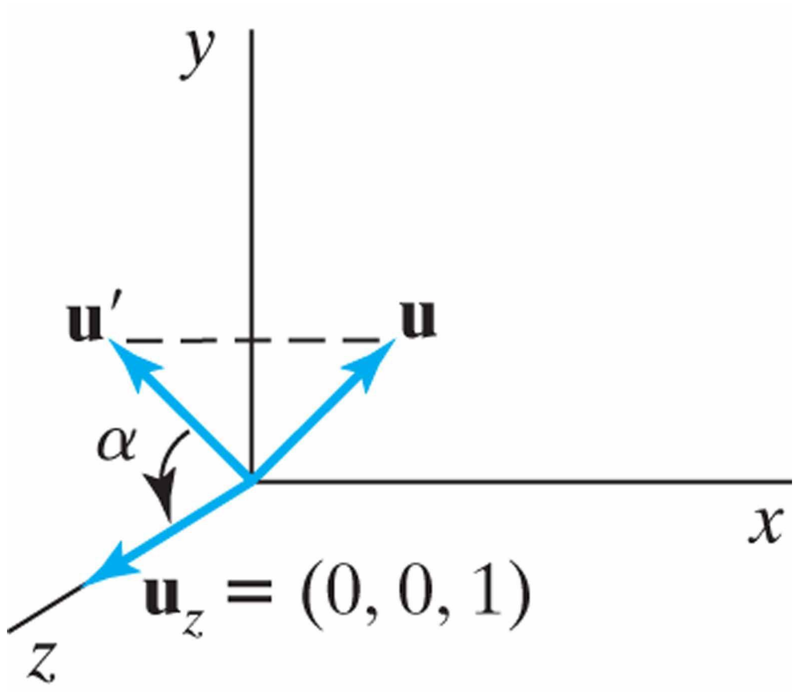
Figure 9-12 Unit vector u is rotated about the *x* axis to bring it into the *xz* plane (a), then it is rotated around the *y* axis to align it with the *z* axis (b).



Rotations

- Two steps for putting the rotation axis onto the z-axis
  - Rotate about the x-axis
  - Rotate about the y-axis

Figure 9-13 Rotation of u around the *x* axis into the *xz* plane is accomplished by rotating **u**′ (the projection of u in the *yz* plane) through angle α onto the *z* axis.



Rotations

- Projection of u in the yz plane

$$\mathbf{u}' = (0, b, c)$$

- Cosine of the rotation angle

$$\cos\alpha = \frac{\mathbf{u}' \cdot \mathbf{u}_z}{|\mathbf{u}'|\,|\mathbf{u}_z|} = \frac{c}{d}$$

where $d = \sqrt{b^2 + c^2}$

- Similarly, sine of rotation angle can be determined from the cross-product

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

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

Rotations

- Equating the right sides

$$d \sin \alpha = b \qquad \sin \alpha = \frac{b}{d}$$

where |u'|=d

- Then,

$$\mathbf{R}_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{c}{d} & -\frac{b}{d} & 0 \\ 0 & \frac{b}{d} & \frac{c}{d} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotations

- Next, swing the unit vector in the xz plane counter-clockwise around the y-axis onto the positive z-axis

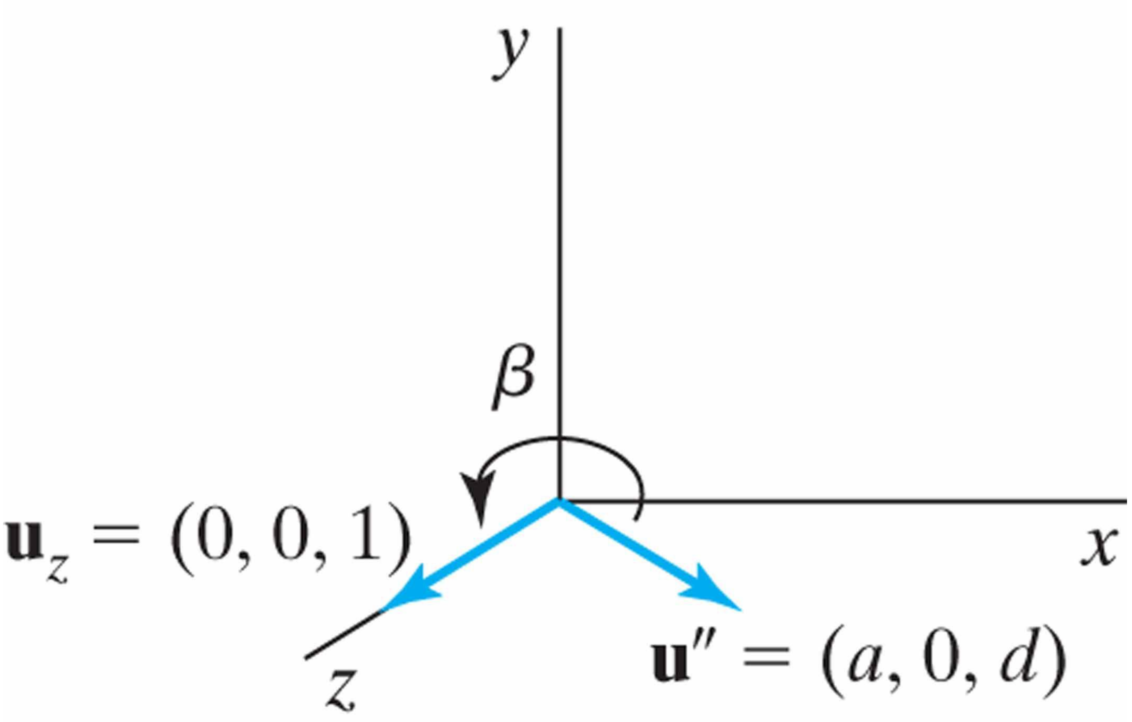


Figure 9-14
Rotation of unit vector **u**″ (vector u after rotation into the *xz* plane) about the *y* axis. Positive rotation angle β aligns **u**″ with vector $\mathbf{u}_z$.

Rotations

$$\cos\beta = \frac{\mathbf{u}'' \cdot \mathbf{u}_z}{|\mathbf{u}''|\,|\mathbf{u}_z|} = d \qquad \text{because } |\mathbf{u}_z| = |\mathbf{u}''| = 1$$

$$\mathbf{u}'' \times \mathbf{u}_z = \mathbf{u}_y\,|\mathbf{u}''|\,|\mathbf{u}_z|\sin\beta$$

and

$$\mathbf{u}'' \times \mathbf{u}_z = \mathbf{u}_y \cdot (-a)$$

so that $\sin\beta = -a$

Therefore

$$\mathbf{R}_y(\beta) = \begin{bmatrix} d & 0 & -a & 0 \\ 0 & 1 & 0 & 0 \\ a & 0 & d & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotations

Together with

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

$$\mathbf{R}(\theta) = \mathbf{T}^{-1} \cdot \mathbf{R}_x^{-1}(\alpha) \cdot \mathbf{R}_y^{-1}(\beta) \cdot \mathbf{R}_z(\theta) \cdot \mathbf{R}_y(\beta) \cdot \mathbf{R}_x(\alpha) \cdot \mathbf{T}$$

In general

$$\mathbf{u}'_z = \mathbf{u}$$

$$\mathbf{u}'_y = \frac{\mathbf{u} \times \mathbf{u}_x}{|\mathbf{u} \times \mathbf{u}_x|}$$

$$\mathbf{u}'_x = \mathbf{u}'_y \times \mathbf{u}'_z$$

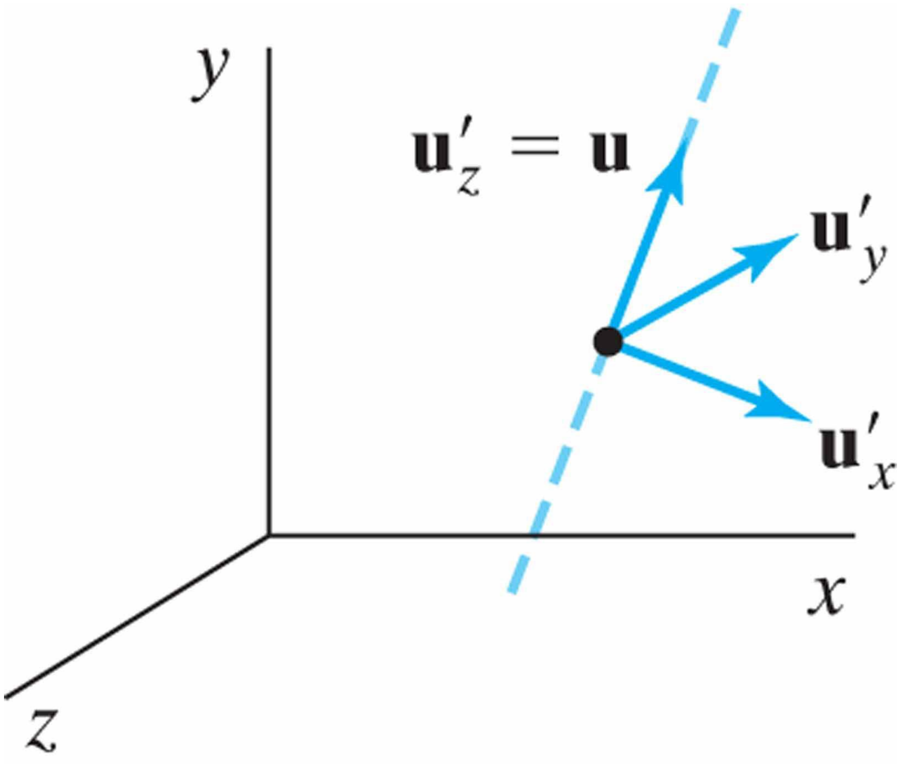
$$\mathbf{u}'_x = (u'_{x1}, u'_{x2}, u'_{x3})$$

$$\mathbf{u}'_y = (u'_{y1}, u'_{y2}, u'_{y3})$$

$$\mathbf{u}'_z = (u'_{z1}, u'_{z2}, u'_{z3})$$

$$\mathbf{R} = \begin{bmatrix} u'_{x1} & u'_{x2} & u'_{x3} & 0 \\ u'_{y1} & u'_{y2} & u'_{y3} & 0 \\ u'_{z1} & u'_{z2} & u'_{z3} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Figure 9-15 Local coordinate system for a rotation axis defined by unit vector **u**.



Quaternions

- Scalar part and vector part $q = (s, \mathbf{v})$
  - Think of it as a higher-order complex number
- Rotation about any axis passing through the coordinate origin is accomplished by first setting up a unit quaternion

$$s = \cos\frac{\theta}{2}, \qquad \mathbf{v} = \mathbf{u}\sin\frac{\theta}{2}$$

where **u** is a unit vector along the selected rotation axis and θ is the specified rotation angle

- Any point P in quaternion notation is P=(0, **p**) where **p**=(x, y, z)

Quaternions

- The rotation of the point P is carried out with quaternion operation $\mathbf{P}' = q\mathbf{P}q^{-1}$ ere $q^{-1} = (s, -\mathbf{v})$
  - This produces P'=(0, **p'**) where

$$\mathbf{p}' = s^2\mathbf{p} + \mathbf{v}(\mathbf{p} \cdot \mathbf{v}) + 2s(\mathbf{v} \times \mathbf{p}) + \mathbf{v} \times (\mathbf{v} \times \mathbf{p})$$

  - Many computer graphics systems use efficient hardware implementations of these vector calculations to perform rapid three-dimensional object rotations.
- Noting that v=(a, b, c), we obtain the elements for the composite rotation matrix. We then have

$$\mathbf{M}_R(\theta) = \begin{bmatrix} 1 - 2b^2 - 2c^2 & 2ab - 2sc & 2ac + 2sb \\ 2ab + 2sc & 1 - 2a^2 - 2c^2 & 2bc - 2sa \\ 2ac - 2sb & 2bc + 2sa & 1 - 2a^2 - 2b^2 \end{bmatrix}$$

Quaternions

- Using $\cos^2\frac{\theta}{2} - \sin^2\frac{\theta}{2} = 1 - 2\sin^2\frac{\theta}{2} = \cos\theta, \qquad 2\cos\frac{\theta}{2}\sin\frac{\theta}{2} = \sin\theta$

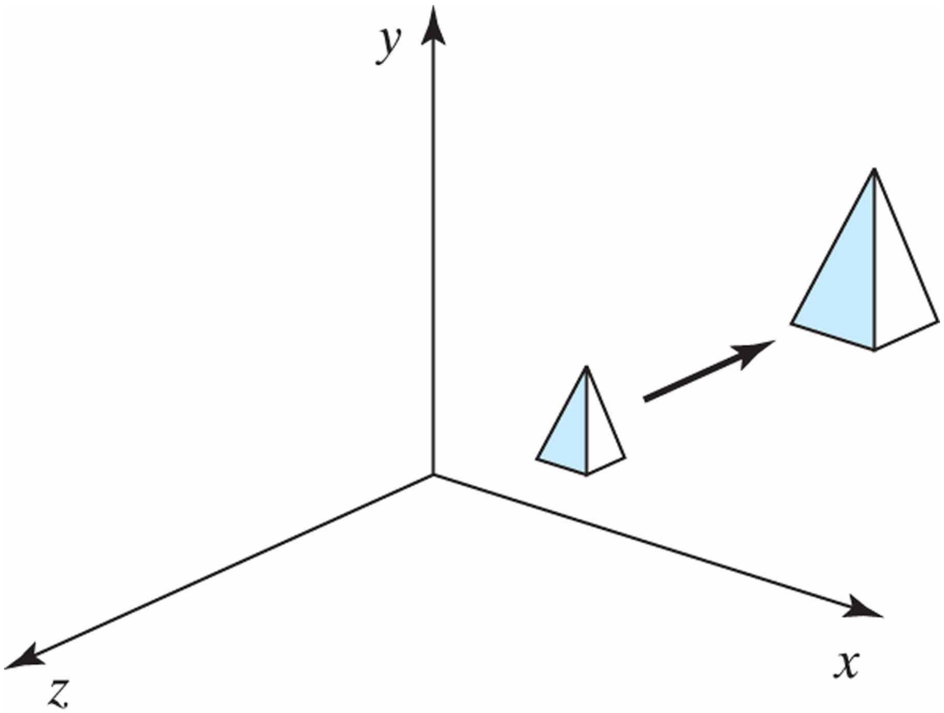
$$\mathbf{M}_R(\theta) =$$

$$\begin{bmatrix} u_x^2(1-\cos\theta)+\cos\theta & u_xu_y(1-\cos\theta)-u_z\sin\theta & u_xu_z(1-\cos\theta)+u_y\sin\theta \\ u_yu_x(1-\cos\theta)+u_z\sin\theta & u_y^2(1-\cos\theta)+\cos\theta & u_yu_z(1-\cos\theta)-u_x\sin\theta \\ u_zu_x(1-\cos\theta)-u_y\sin\theta & u_zu_y(1-\cos\theta)+u_x\sin\theta & u_z^2(1-\cos\theta)+\cos\theta \end{bmatrix}$$

- About an arbitrarily placed rotation axis: $\mathbf{R}(\theta) = \mathbf{T}^{-1}\cdot\mathbf{M}_R\cdot\mathbf{T}$
- Quaternions require less storage space than 4 × 4 matrices, and it is simpler to write quaternion procedures for transformation sequences.
- This is particularly important in animations, which often require complicated motion sequences and motion interpolations between two given positions of an object.

3D scaling

Figure 9-17 Doubling the size of an object with transformation 9-41 also moves the object farther from the origin.

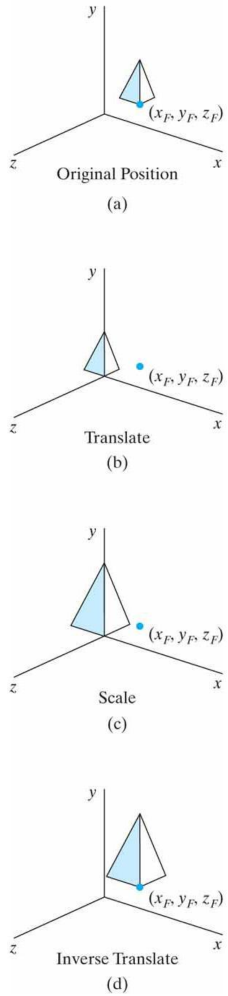


$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$\mathbf{P}' = \mathbf{S} \cdot \mathbf{P}$$

3D scaling

Figure 9-18 A sequence of transformations for scaling an object relative to a selected fixed point, using Equation 9-41.



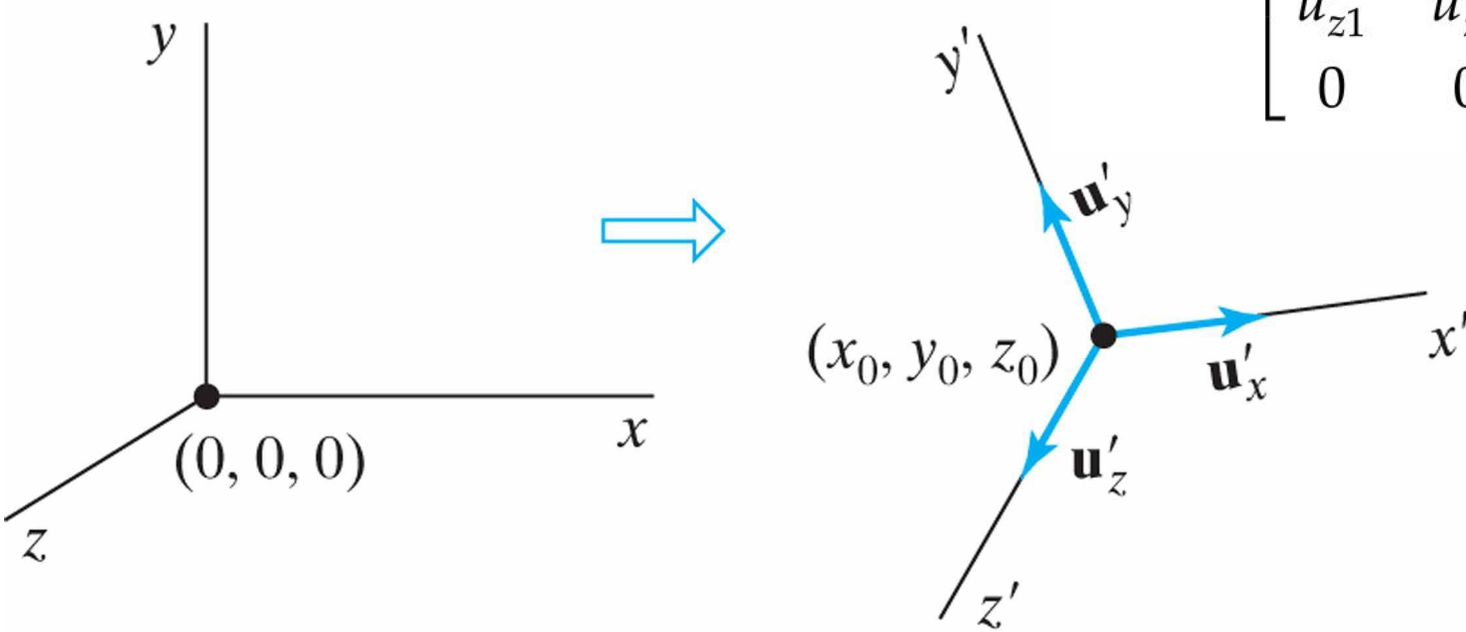
$$\mathbf{T}(x_f, y_f, z_f) \cdot \mathbf{S}(s_x, s_y, s_z) \cdot \mathbf{T}(-x_f, -y_f, -z_f) = \begin{bmatrix} s_x & 0 & 0 & (1-s_x)x_f \\ 0 & s_y & 0 & (1-s_y)y_f \\ 0 & 0 & s_z & (1-s_z)z_f \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Composite 3D transformation example

Transformations between 3D coordinate systems

Figure 9-21 An *x′y′z′* coordinate system defined within an *x y z* system. A scene description is transferred to the new coordinate reference using a transformation sequence that superimposes the *x′y′z′* frame on the *xyz* axes.

$$\mathbf{R} = \begin{bmatrix} u'_{x1} & u'_{x2} & u'_{x3} & 0 \\ u'_{y1} & u'_{y2} & u'_{y3} & 0 \\ u'_{z1} & u'_{z2} & u'_{z3} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



$\mathbf{R} \cdot \mathbf{T}$