Three Dimensional Modeling Transformations

Dr. S.M. Malaek

Assistant: M. Younesi

Three Dimensional Modeling Transformations

- Methods for object modeling transformation in three dimensions are extended from two dimensional methods by including consideration for the z coordinate.

Three Dimensional Modeling Transformations

- Generalize from 2D by including **z** coordinate
- Straightforward for translation and scale, rotation more difficult
- Homogeneous coordinates: 4 components
- Transformation matrices: 4×4 elements

3D Point

- We will consider points as column vectors. Thus, a typical point with coordinates (x, y, z) is represented as:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

3D Point Homogenous Coordinate

- A 3D point **P** is represented in homogeneous coordinates by a 4-dim. Vect:

$$\mathbf{P} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

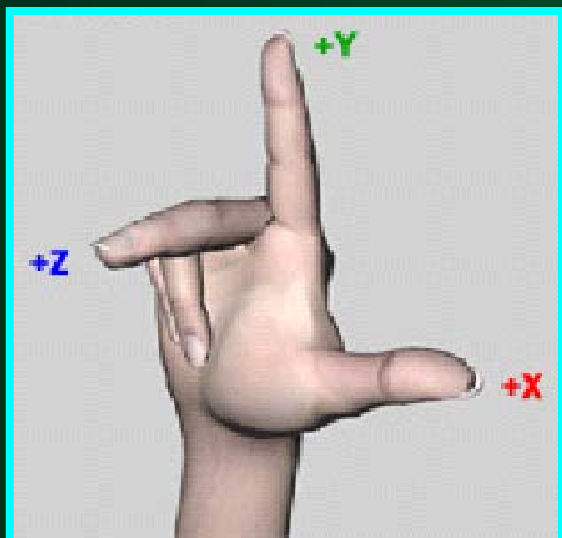
3D Point Homogenous Coordinate

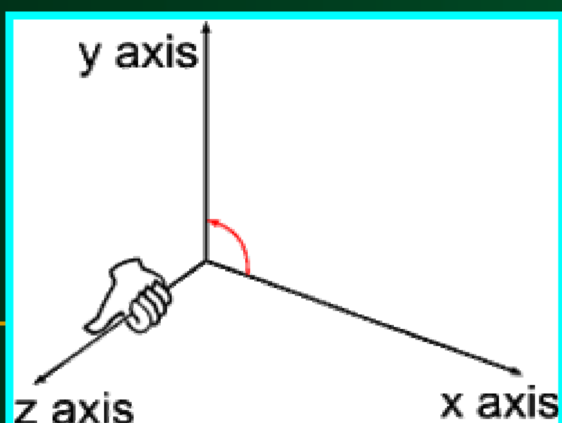
- We don't lose anything
- The main advantage: it is easier to compose translation and rotation
- Everything is matrix multiplication

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

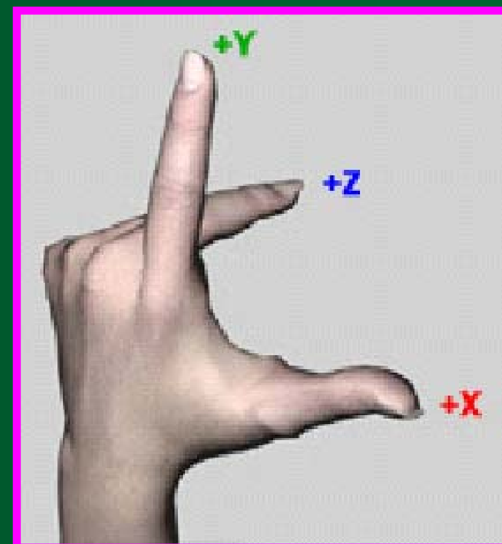
3D Coordinate Systems

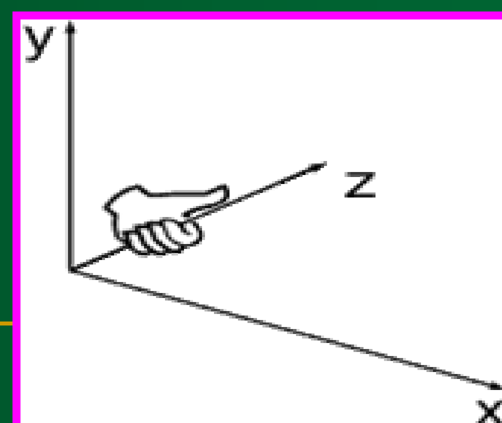
- ***Right Hand*** **coordinate system:**





- ***Left Hand*** **coordinate system:**





3D Transformation

- In homogeneous coordinates, 3D transformations are represented by 4×4 matrixes:

$$\boxed{\begin{bmatrix} a & b & c & t_x \\ d & e & f & t_y \\ g & h & i & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}}$$

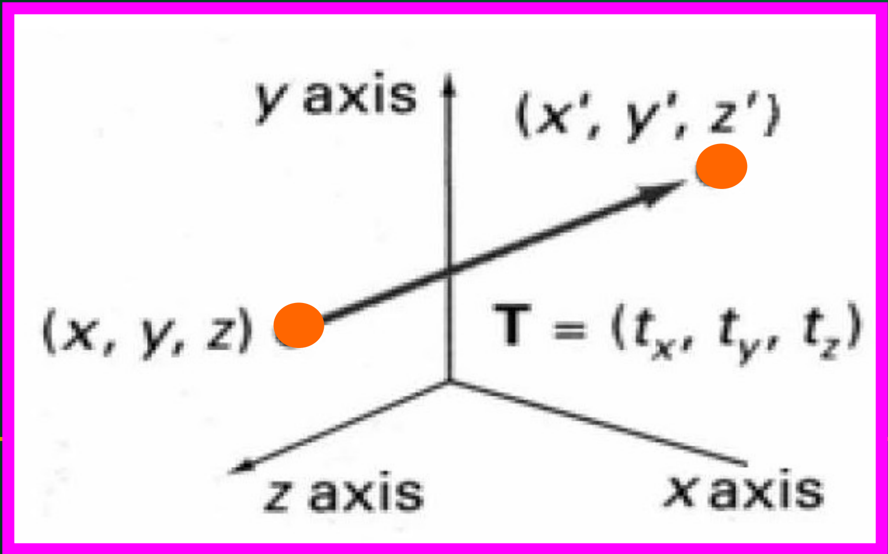
3D Translation

3D Translation

- **P** is **translated** to **P'** by:

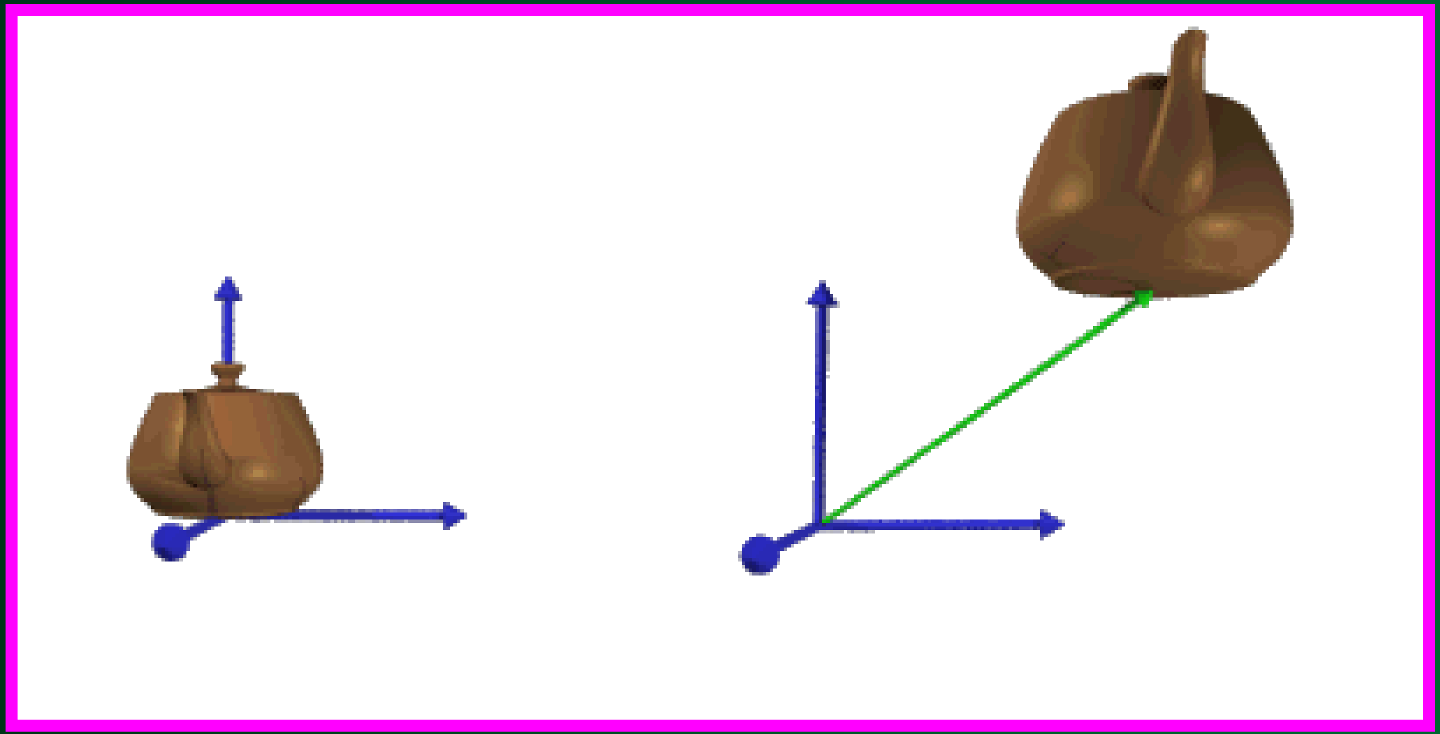
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

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



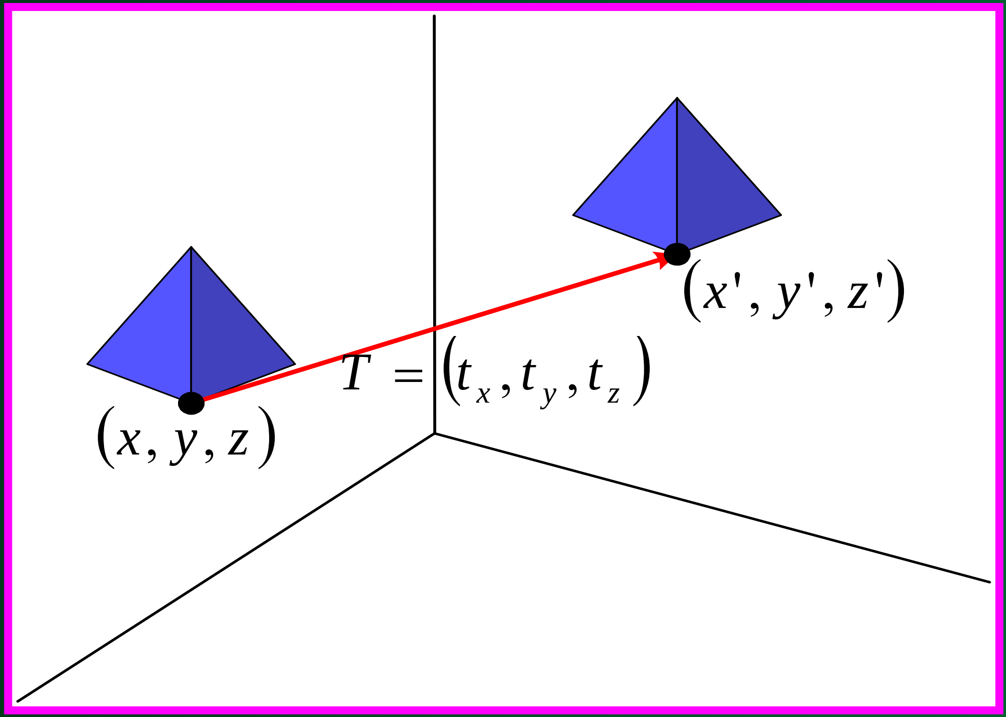
3D Translation

- An object is translated in 3D dimensional by transforming each of the defining points of the objects .



3D Translation

- An Object represented as a set of polygon surfaces, is translated by translate each vertex of each surface and redraw the polygon facets in the new position.



- **Inverse Translation:**

$$T^{-1}(t_x, t_y, t_z) = T(-t_x, -t_y, -t_z)$$

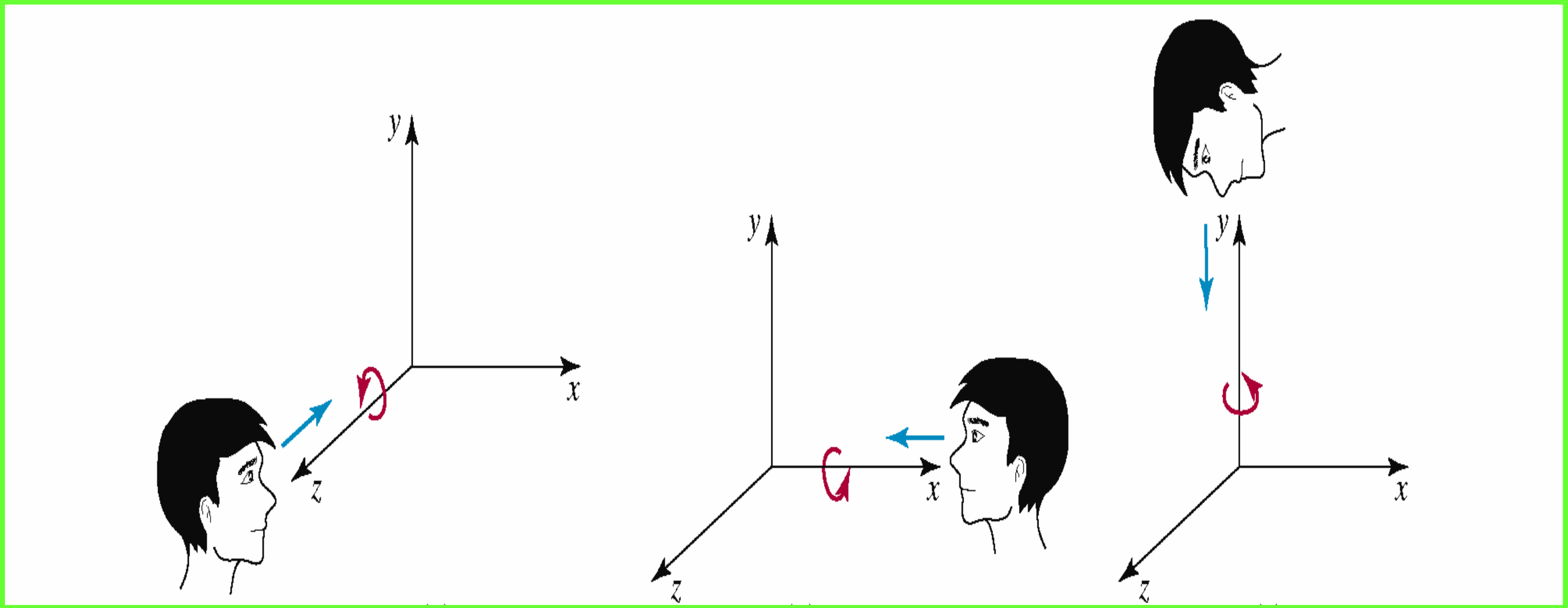
3D Rotation

3D Rotation

- In general, rotations are specified by a ***rotation axis*** and an ***angle***. In two-dimensions there is only one choice of a rotation axis that leaves points in the plane.

3D Rotation

- The easiest **rotation axes** are those that parallel to the coordinate axis.
- Positive rotation **angles** produce counterclockwise rotations about a coordinate axix, if we are looking along the positive half of the axis toward the coordinate origin.



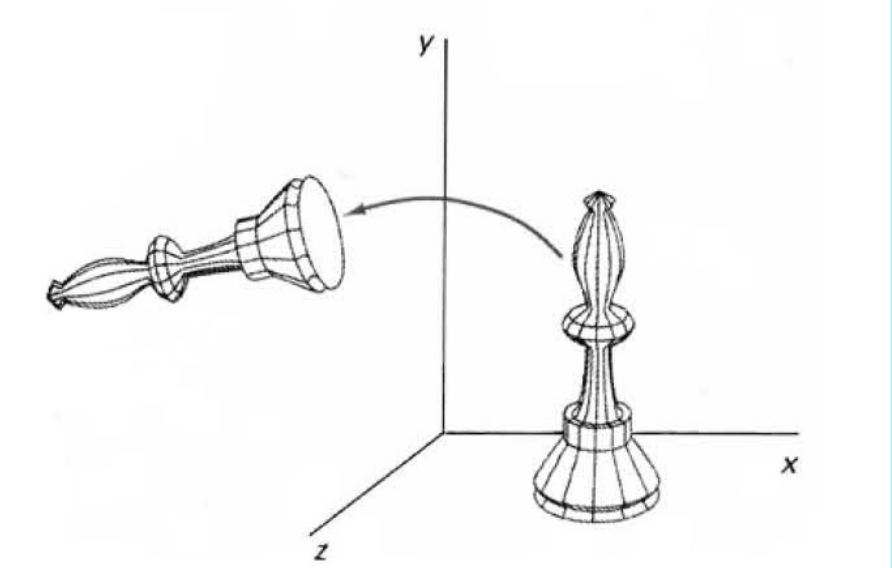
Coordinate Axis Rotations

Coordinate Axis Rotations

- **Z-axis rotation:** For z axis same as 2D rotation:

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

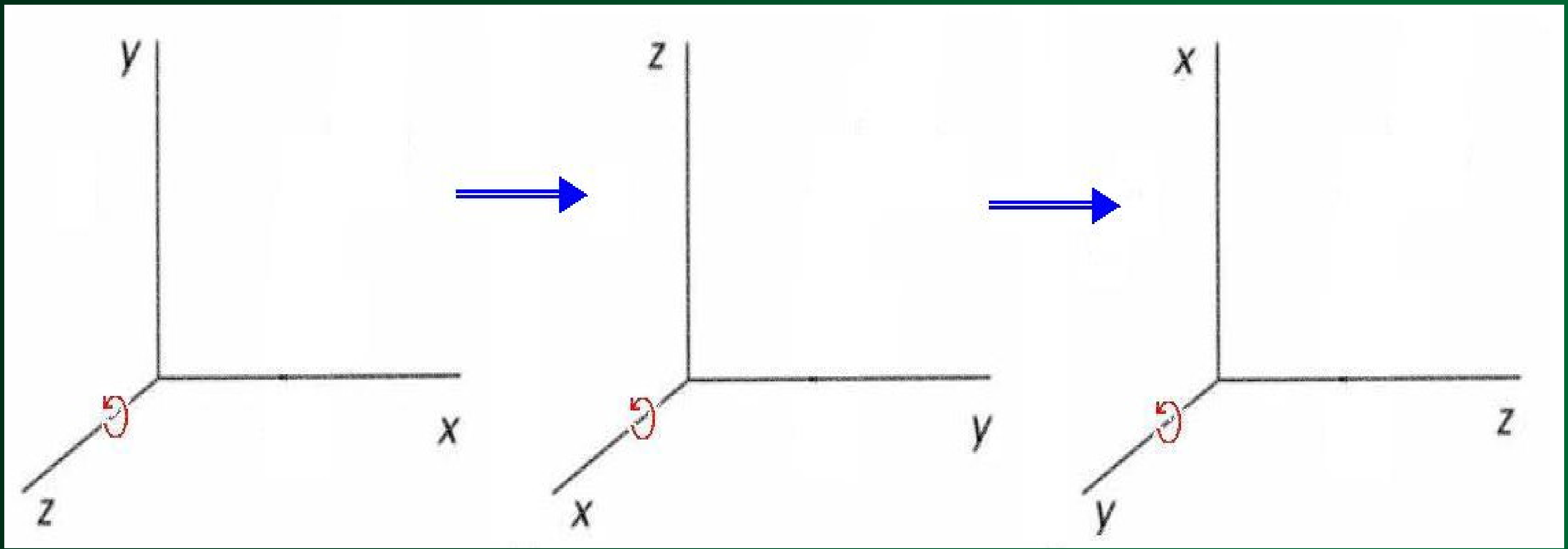
$$\mathbf{P}' = \mathbf{R}_z(\theta) \cdot \mathbf{P}$$



Coordinate Axis Rotations

- Obtain rotations around other axes through cyclic permutation of coordinate parameters:

$$x \rightarrow y \rightarrow z \rightarrow x$$

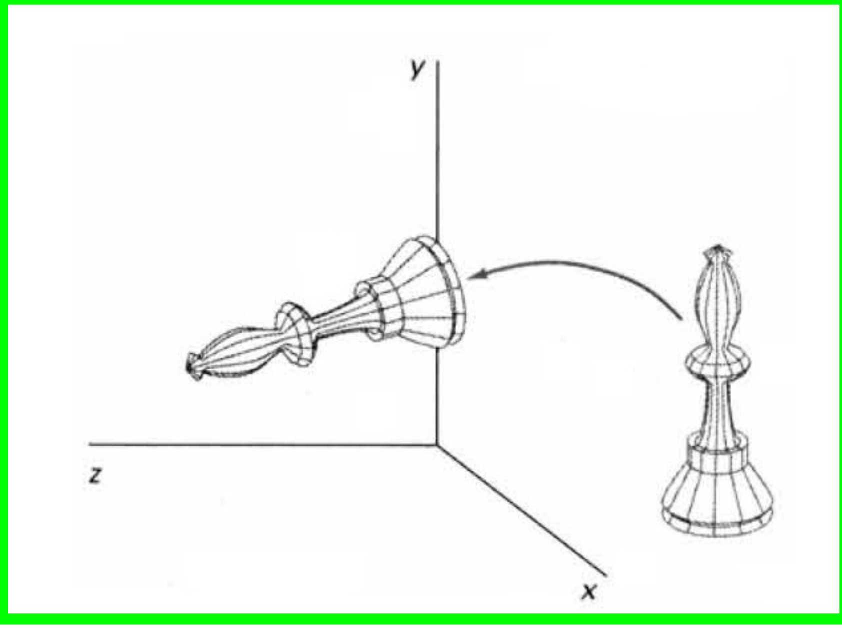


Coordinate Axis Rotations

- **X-axis rotation:**

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

$$\mathbf{P}' = \mathbf{R}_x(\theta) \cdot \mathbf{P}$$

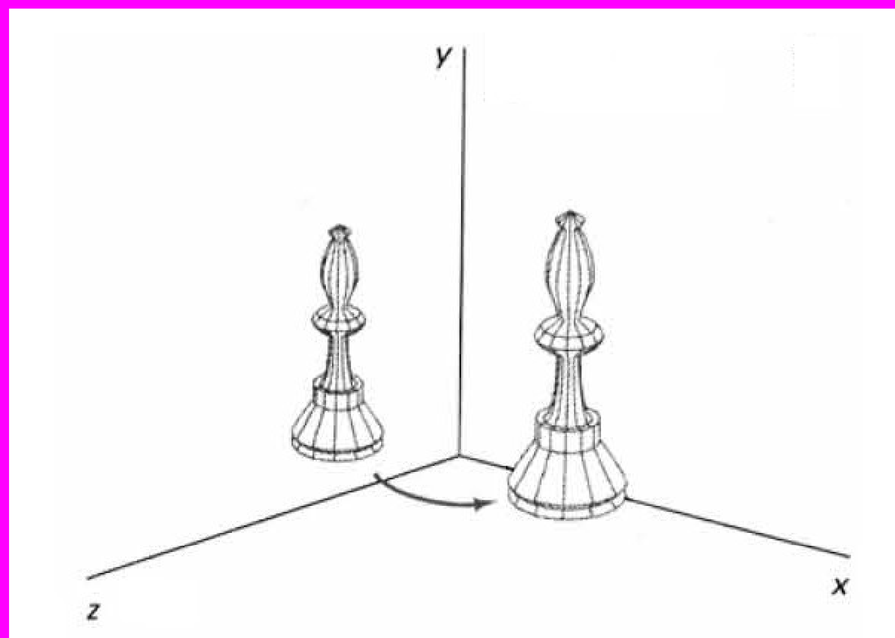


Coordinate Axis Rotations

- **Y-axis rotation:**

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

$$\mathbf{P}' = \mathbf{R}_y(\theta) \cdot \mathbf{P}$$

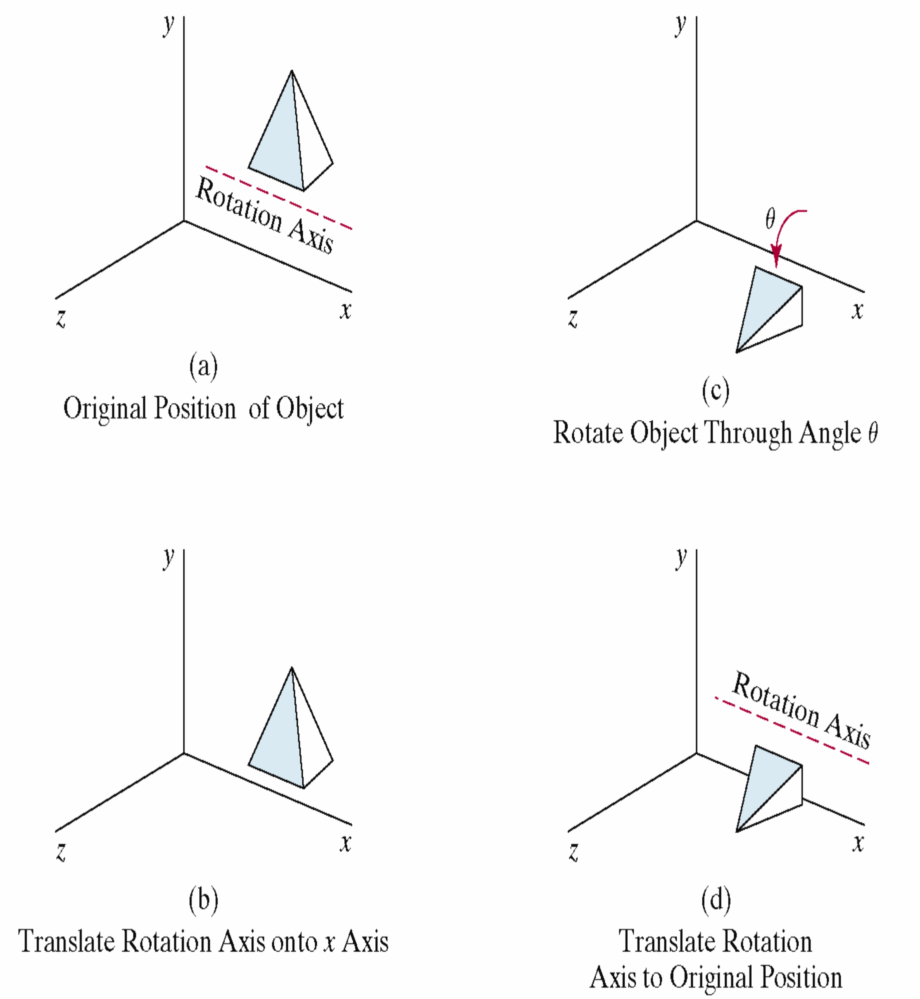


General Three Dimensional

Rotations

General Three Dimensional Rotations

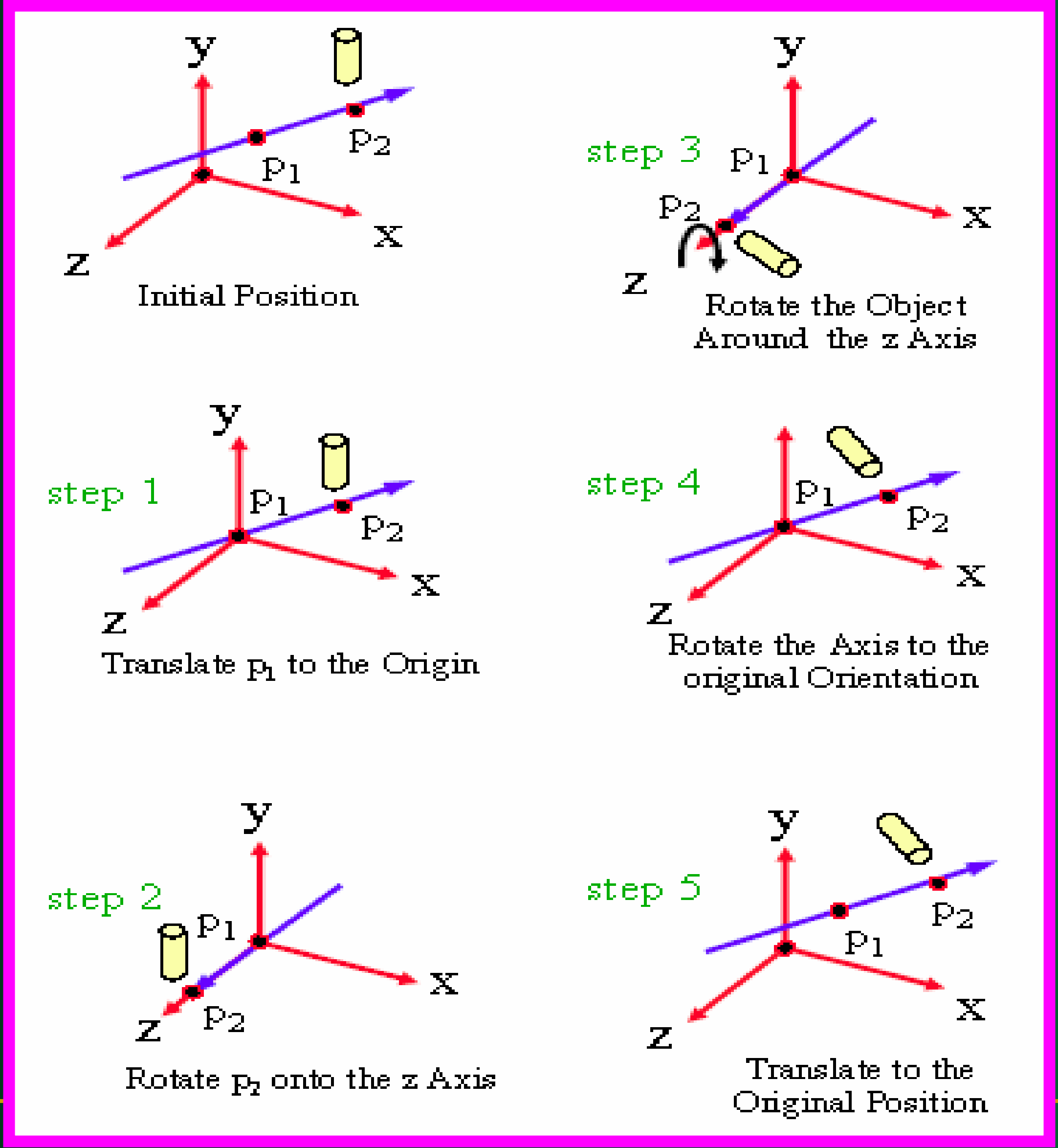
Rotation axis parallel with coordinate axis (Example x axis):



$$\mathbf{P}' = \mathbf{T}^{-1} \cdot \mathbf{R}_x(\theta) \cdot \mathbf{T} \cdot \mathbf{P}$$

General Three Dimensional Rotations

An arbitrary axis (with the rotation axis projected onto the **Z** axis):



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

General Three Dimensional Rotations

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

- A rotation matrix for any axis that does not coincide with a coordinate axis can be set up as a composite transformation involving combination of translations and the coordinate-axes rotations:

1. Translate the object so that the rotation axis passes through the coordinate origin
2. Rotate the object so that the axis rotation coincides with one of the coordinate axes
3. Perform the specified rotation about that coordinate axis
4. Apply inverse rotation axis back to its original orientation
5. Apply the inverse translation to bring the rotation axis back to its original position

Other way to look at rotation

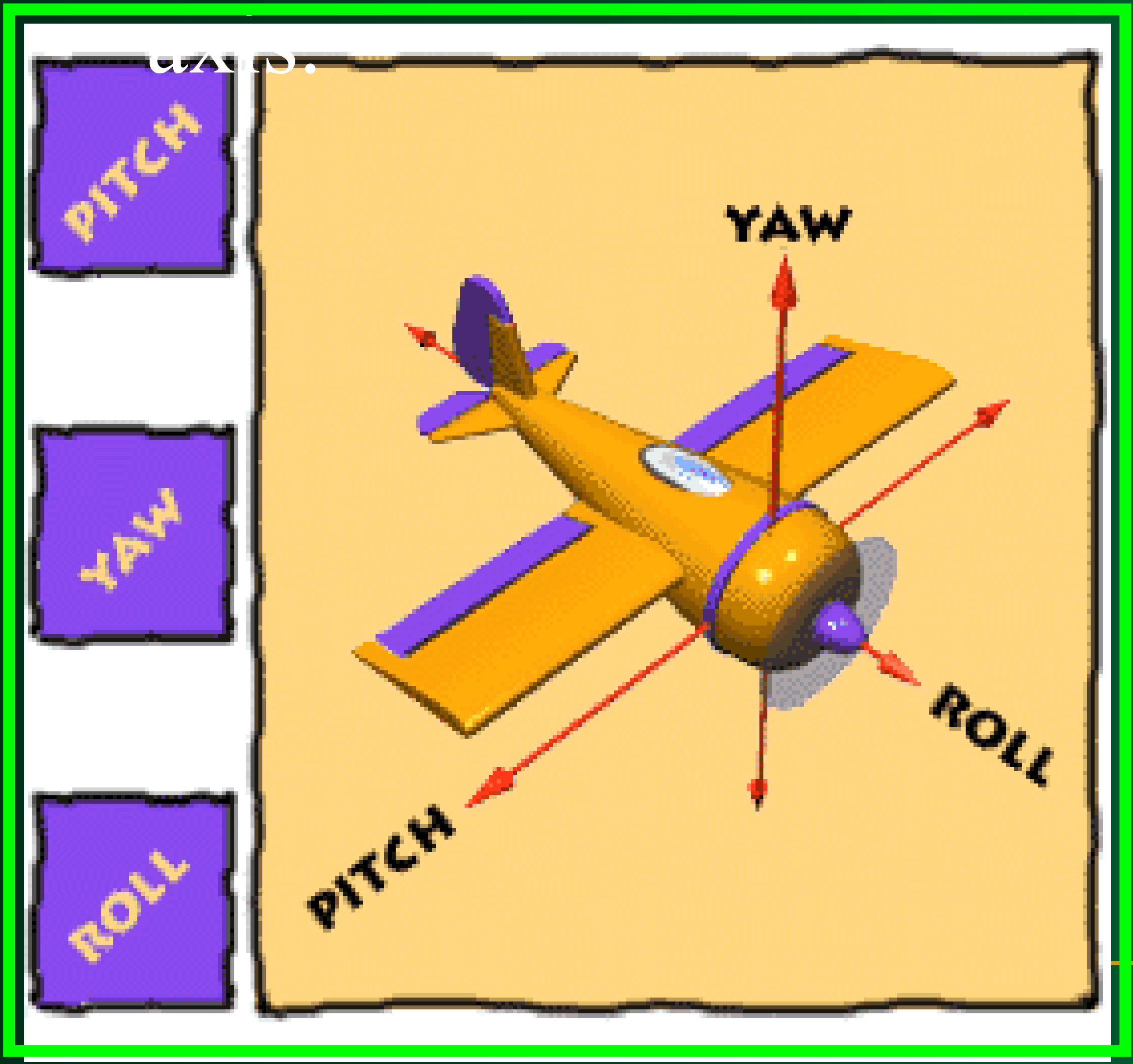
Roll, Pitch, Yaw

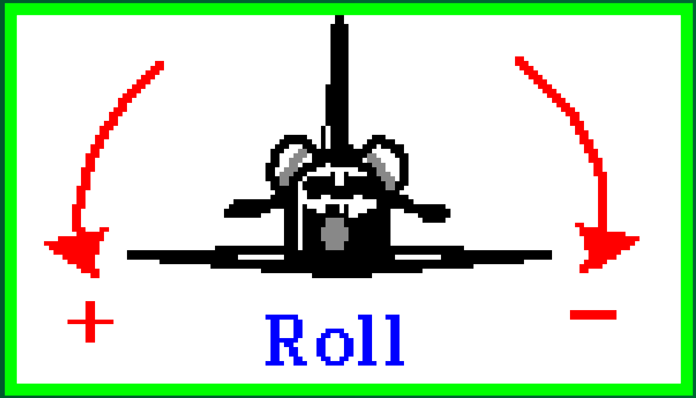
Roll, Pitch, Yaw

- Imagine three **lines** running through an airplane and intersecting at right angles at the airplane's center of gravity.

Roll

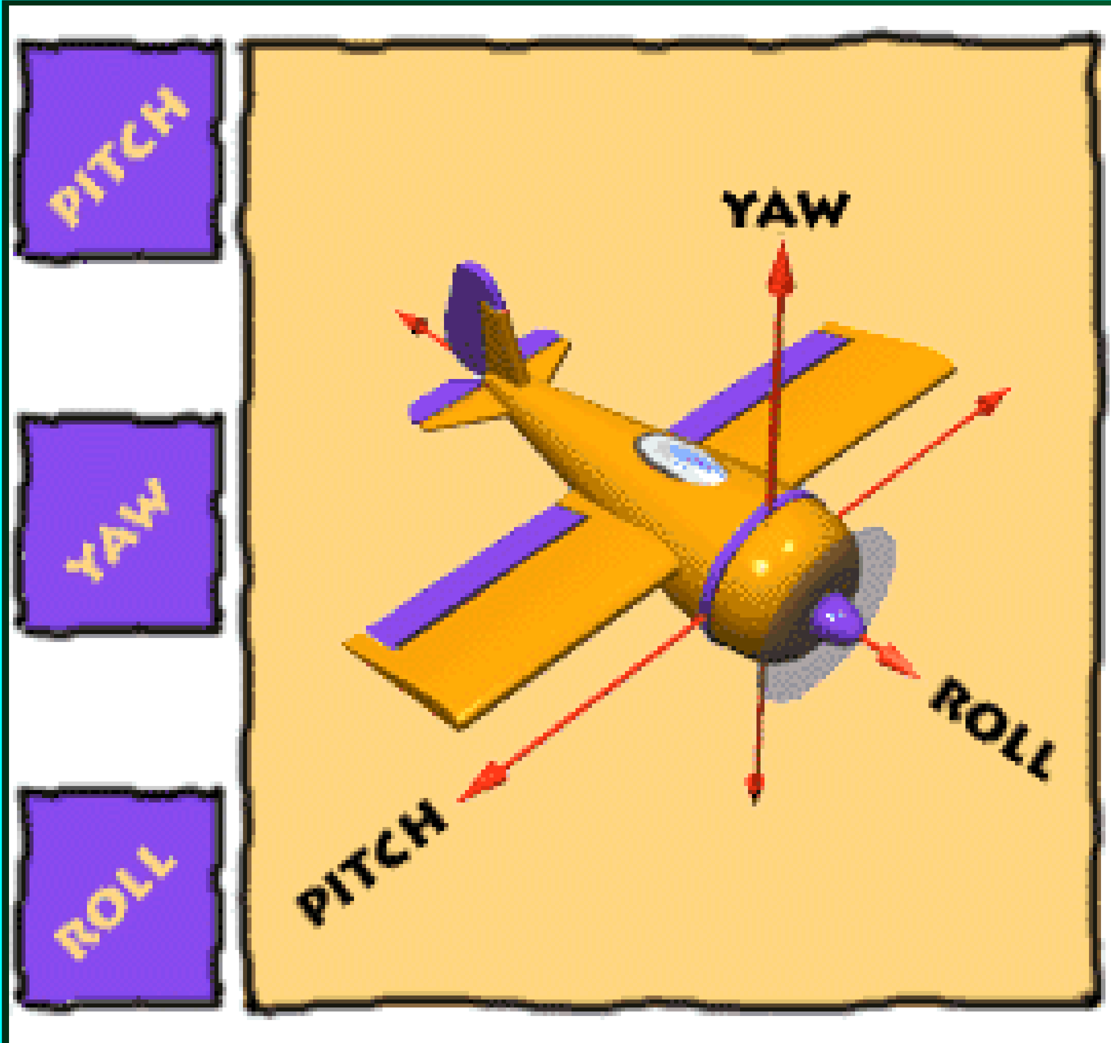
- ***Roll:*** rotation around the front-to-back axis.

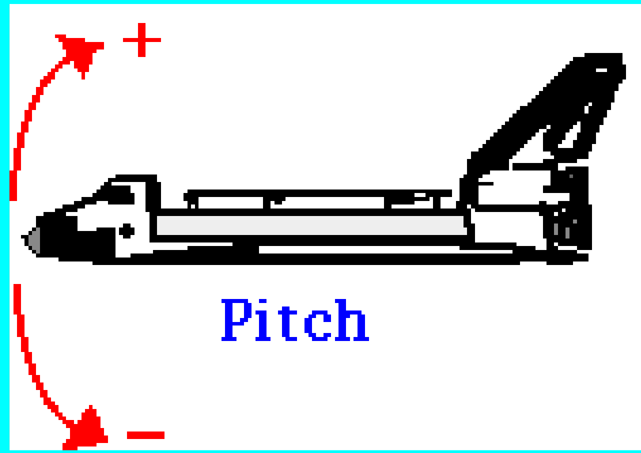




PITCH

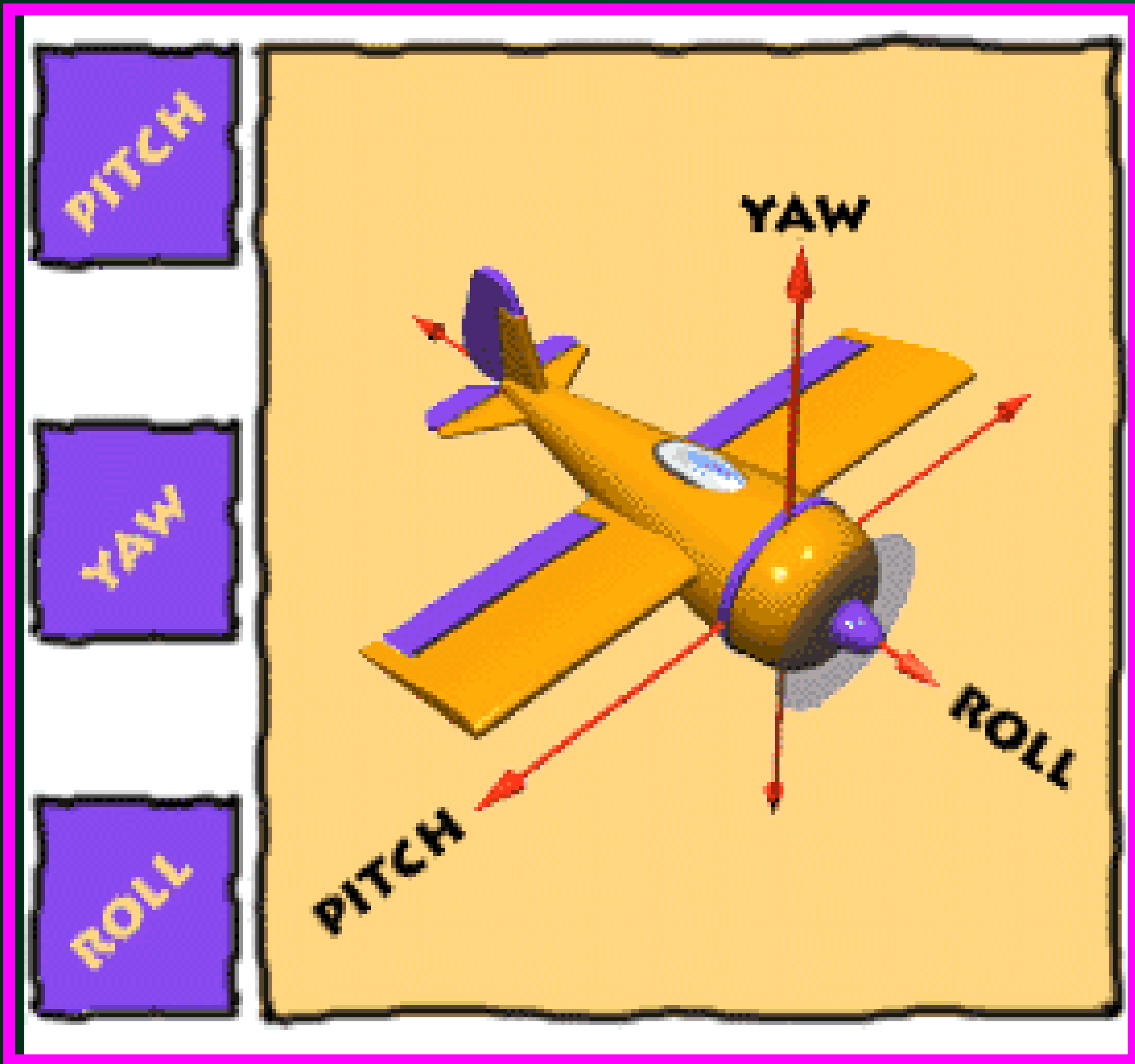
- ***PITCH:*** Rotation around the side-to-side axis

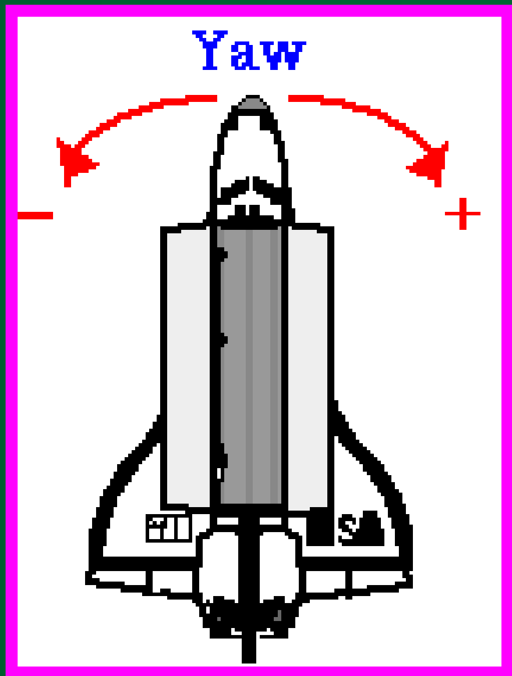




YAW

- ***YAW:*** Rotation around the vertical axis.

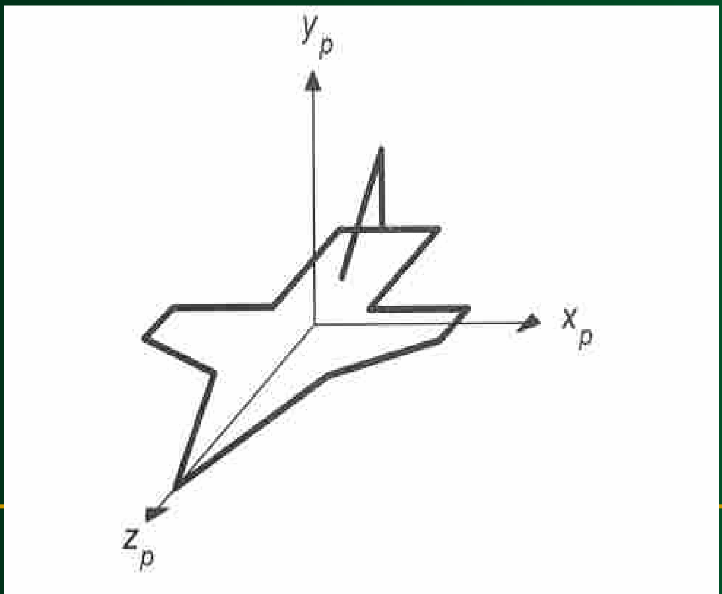


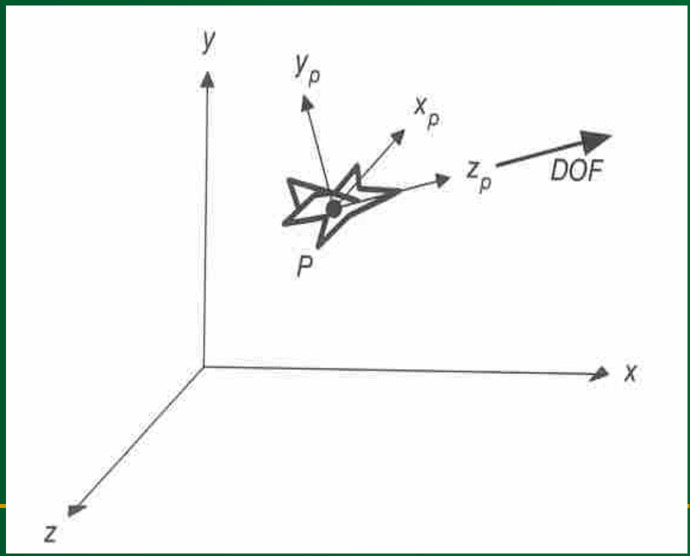


An Example of the Airplane

An Example of the Airplane

- Consider the following example. An airplane is oriented such that its nose is pointing in the positive z direction, its right wing is pointing in the positive x direction, its cockpit is pointing in the positive y direction. We want to transform the airplane so that it heads in the direction given by the vector DOF (direction of flight), is centre at P, and is not banked.





An Example of the Airplane

- First we are to rotate the positive z_p direction into the direction of DOF, which gives us the third column of the rotation matrix: DOF / |DOF|. The x_p axis must be transformed into a horizontal vector perpendicular to DOF – that is in the direction of y×DOF. The y_p direction is then given by $x_p \times z_p$ = DOF ×(y × DOF).

$$R = \left[\frac{y \times DOF}{\left| y \times DOF \right|} \quad \frac{DOF \times (y \times DOF)}{\left| DOF \times (y \times DOF) \right|} \quad \frac{DOF}{\left| DOF \right|} \right]$$

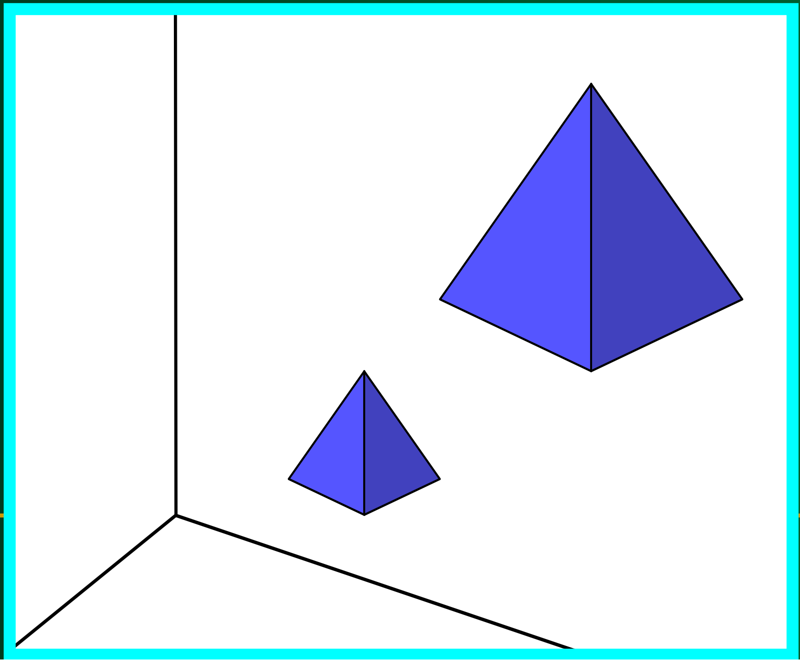
3D Scaling

3D Scaling

- **About origin:** Changes the size of the object and repositions the object relative to the coordinate origin.

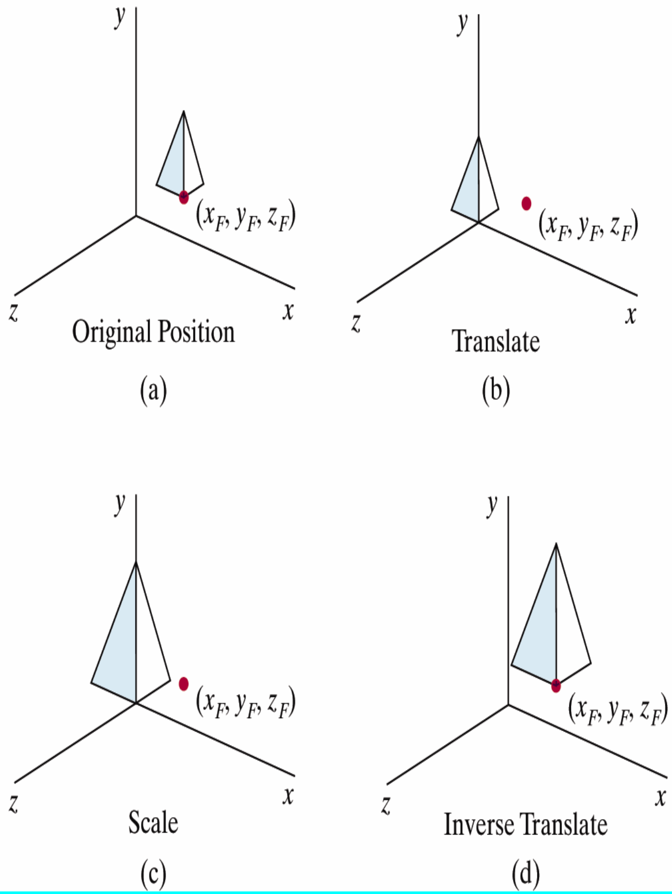
$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

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



3D Scaling

- **About any fixed point:**



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

Composite

3D Transformations

Composite 3D Transformations

- **Same way as in two dimensions:**
  - Multiply matrices
  - Rightmost term in matrix product is the first transformation to be applied

3D Reflections

3D Reflections

- **About an axis:** equivalent to 180° rotation about that axis

3D Reflections

About a plane:

- A reflection through the **xy** plane:

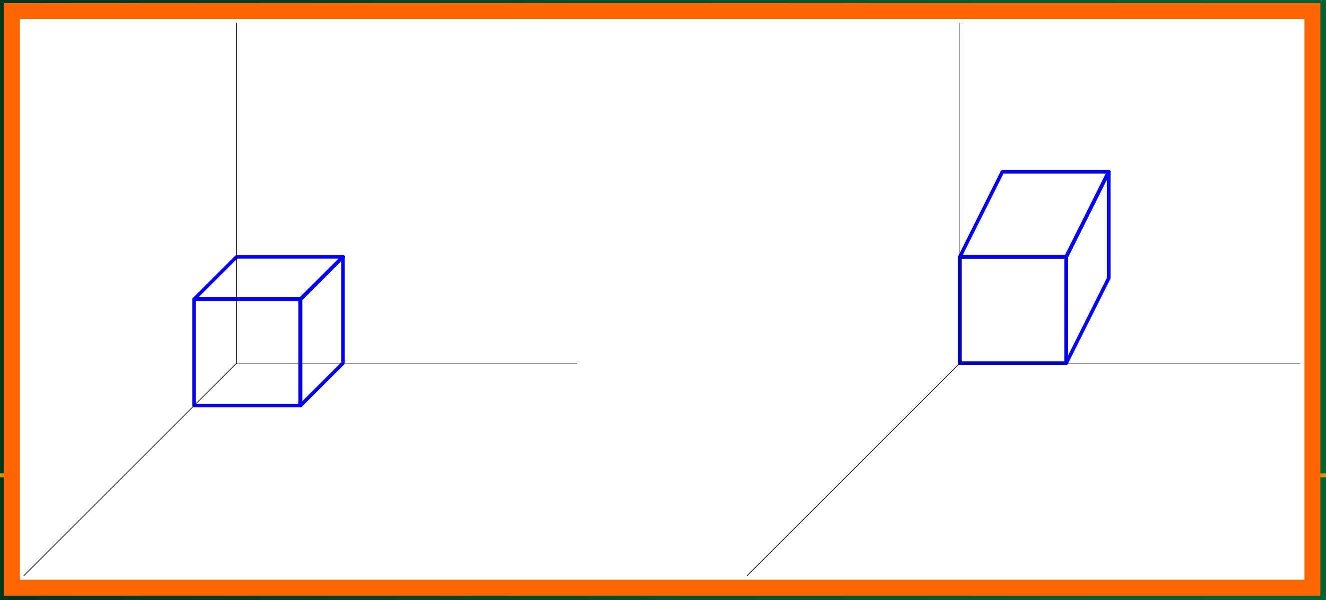
$$\begin{bmatrix} x \\ y \\ -z \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- A reflections through the **xz** and the **yz** planes are defined similarly.

3D Shearing

3D Shearing

- **Modify object shapes**
- **Useful for perspective projections:**
  - E.g. draw a cube (3D) on a screen (2D)
  - Alter the values for **x** and **y** by an amount proportional to the distance from z_{ref}



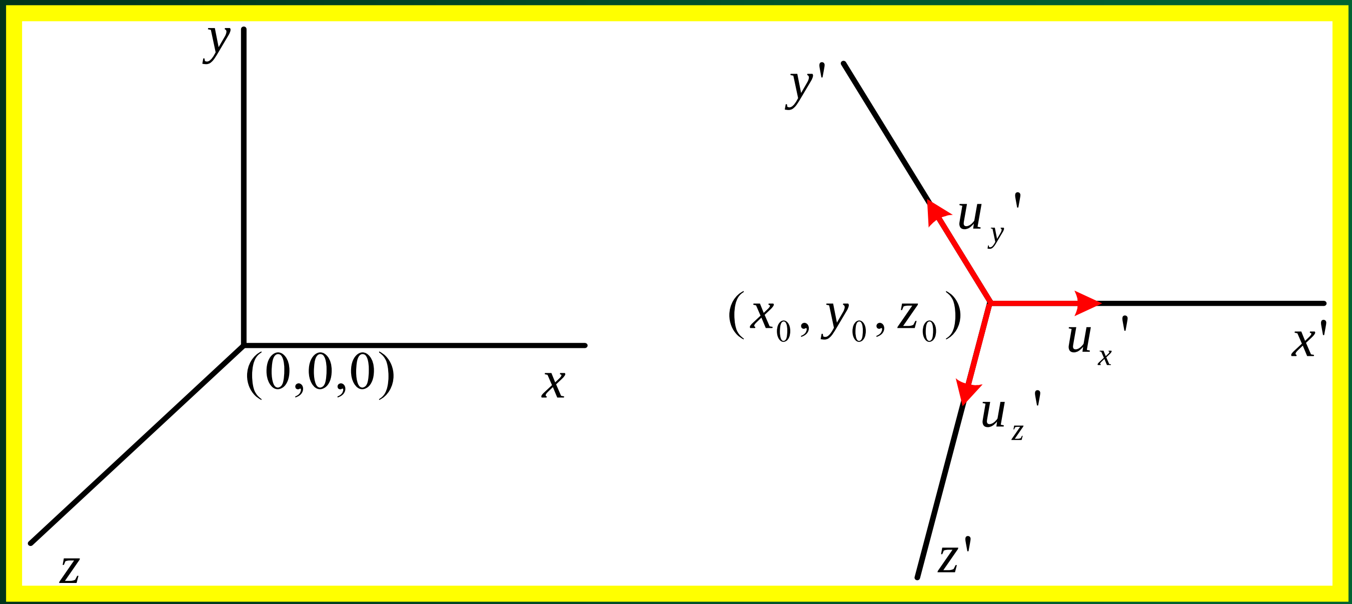
3D Shearing

$$M_{zshear} = \begin{bmatrix} 1 & 0 & sh_{zx} & -sh_{zx} \cdot z_{ref} \\ 0 & 1 & sh_{zy} & -sh_{zy} \cdot z_{ref} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Transformations Between Coordinate Systems

Transformations Between Coordinate ystems

- Translate such that the origins overlap
- Rotate such that the axes overlap



$$R \cdot T = \begin{bmatrix} u'_{x1} & u'_{x2} & u'_{x3} & 0 \\ u'_{y1} & u'_{y2} & u'_{y3} & 0 \\ u'_{z1} & u'_{z2} & u'_{z3} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot T(-x_0, -y_0, -z_0)$$