# Computer Graphics

# 3D Transformations

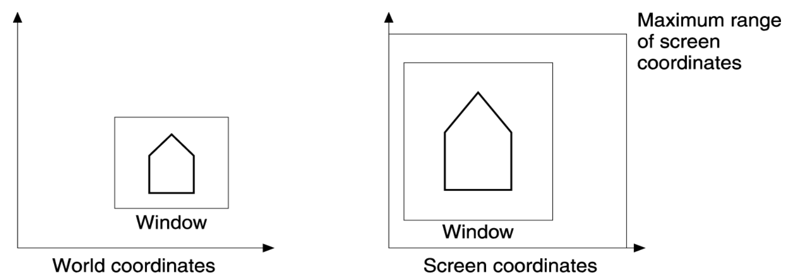
## World Window to Viewport Transformation

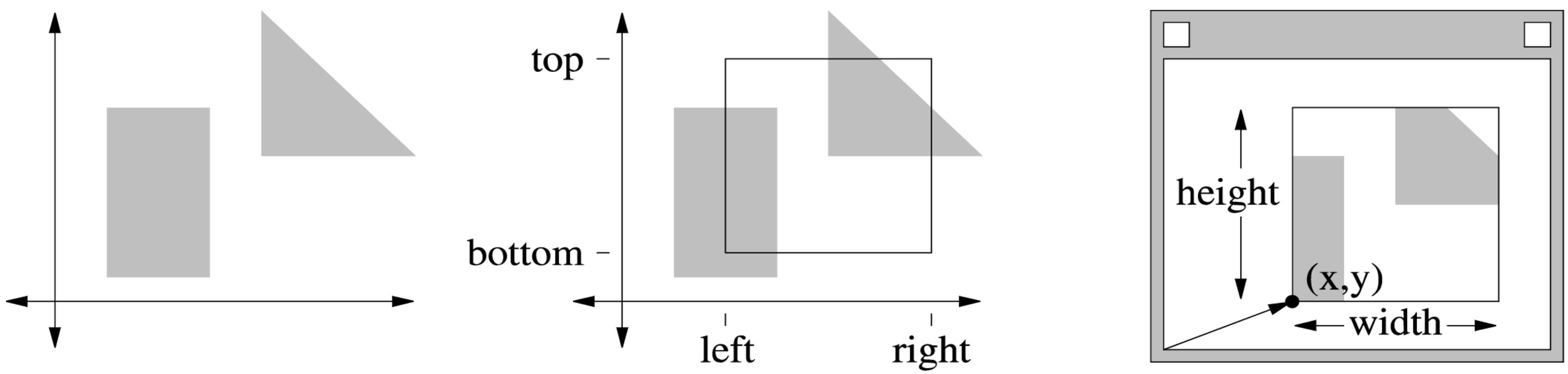
# Outline

- World window to viewport transformation
- 3D transformations
- Coordinate system transformation

# The *Window-to-Viewport* Transformation

- Problem: Screen windows cannot display the whole world (window management)
- How to transform and clip: *Objects* to *Windows* to *Screen*



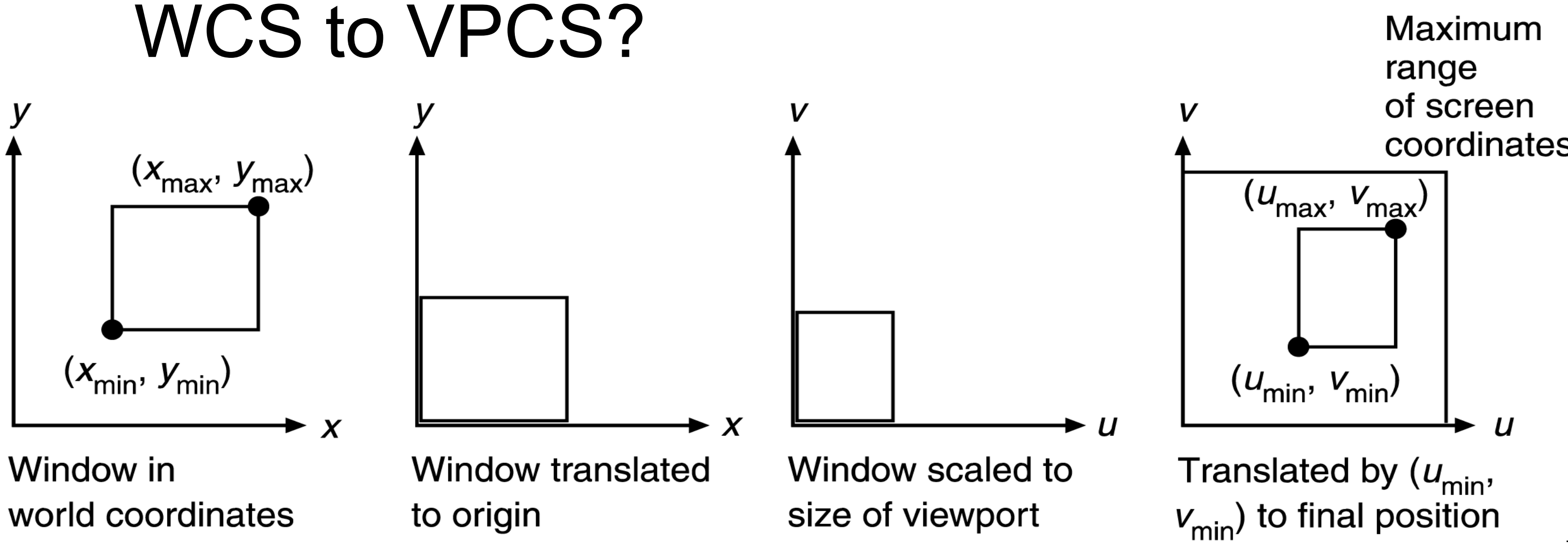

Pics/Math courtesy of Dave Mount @ UMD-CP

# *Window-to-Viewport* Transformation

- Given a window and a viewport, what is the transformation from WCS to VPCS?

Three steps:

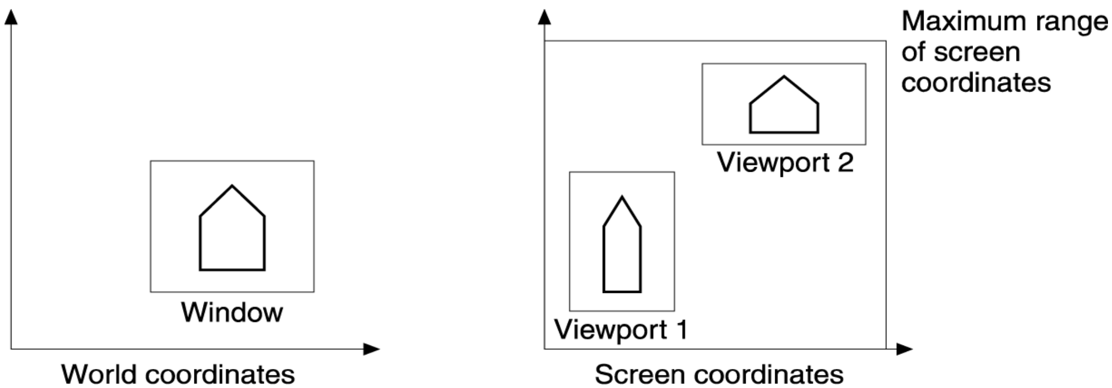
- Translate
- Scale
- Translate



-

# Transforming World Coordinates to Viewports

- 3 steps
  1. Translate
  2. Scale
  3. Translate



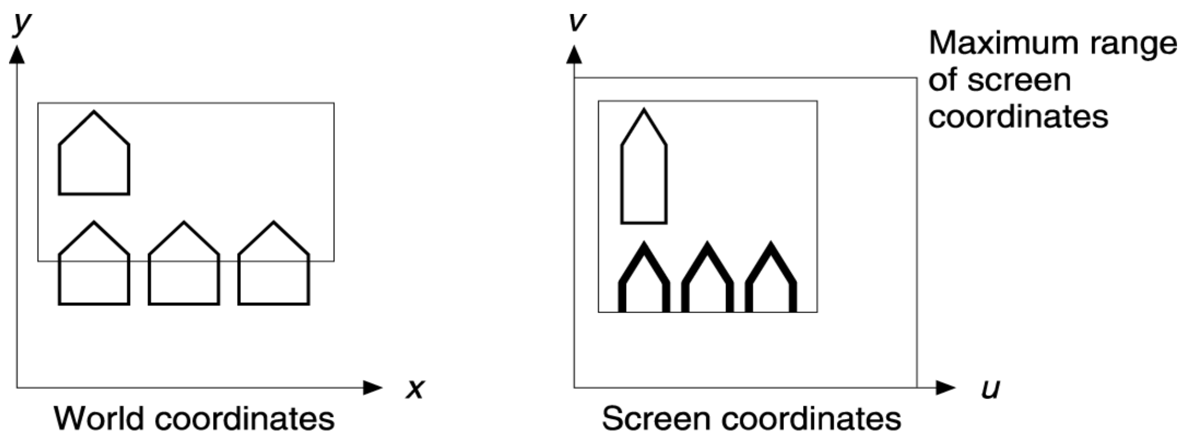

Overall Transformation:

$$M_{WV} = T(u_{min}, v_{min}) \cdot S(\frac{u_{max} - u_{min}}{x_{max} - x_{min}}, \frac{v_{max} - v_{min}}{y_{max} - y_{min}}) \cdot T(-x_{min}, -y_{min})$$

# Clipping to the Viewport

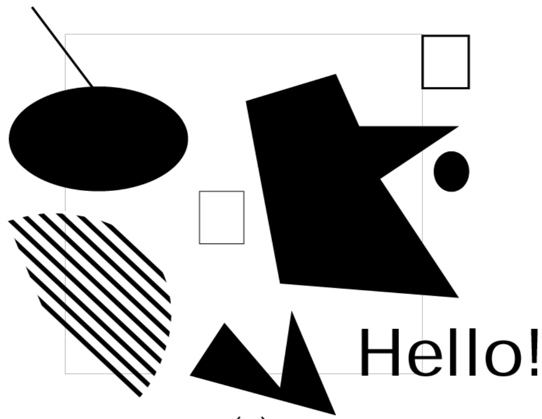
- Viewport size may not be big enough for everything
- Display only the pixels inside the viewport

- Transform lines in world
- Then clip in world
- Transform to image
- Then draw
- Do not transform pixels

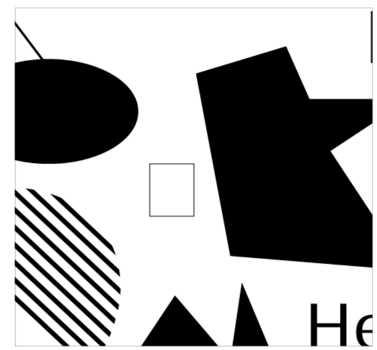


# Another Example

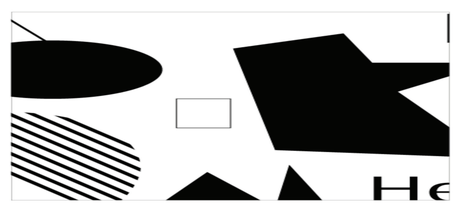
- Scan-converted
  - Lines
  - Polygons
  - Text
  - Fill regions
- Clip regions for display




(a)



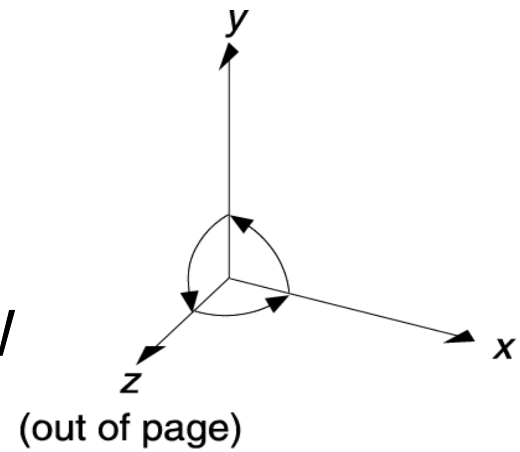
(b)



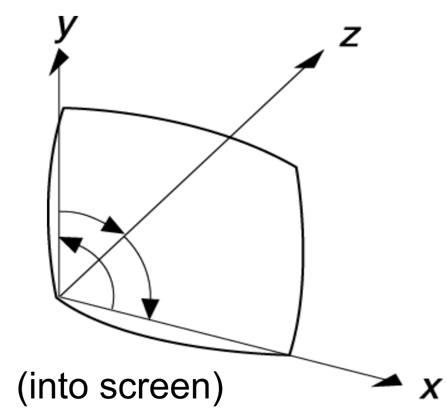
# 3D Transformations

# Representation of 3D Transformations

- $Z$ axis represents depth
- Right Handed System
  - When looking “down” at the origin, positive rotation is CCW




- Left Handed System
  - When looking “down”, positive rotation is in CW
  - More natural interpretation for displays, big *z* means “far”



# 3D Homogenous Coordinates

- Homogenous coordinates for 2D space requires 3D vectors & matrices
- Homogenous coordinates for 3D space requires 4D vectors & matrices
- *[x,y,z,w]*

$$S(s_x, s_y, s_z) = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T(d_x, d_y, d_z) = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

# 3D Transformations: Scale & Translate

- Scale
  - Parameters for each axis direction

$$S(s_x, s_y, s_z) = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- Translation

$$T(d_x, d_y, d_z) = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

# 3D Transformations: Rotation

- One rotation for each world coordinate axis

$$P' = R \cdot P$$

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

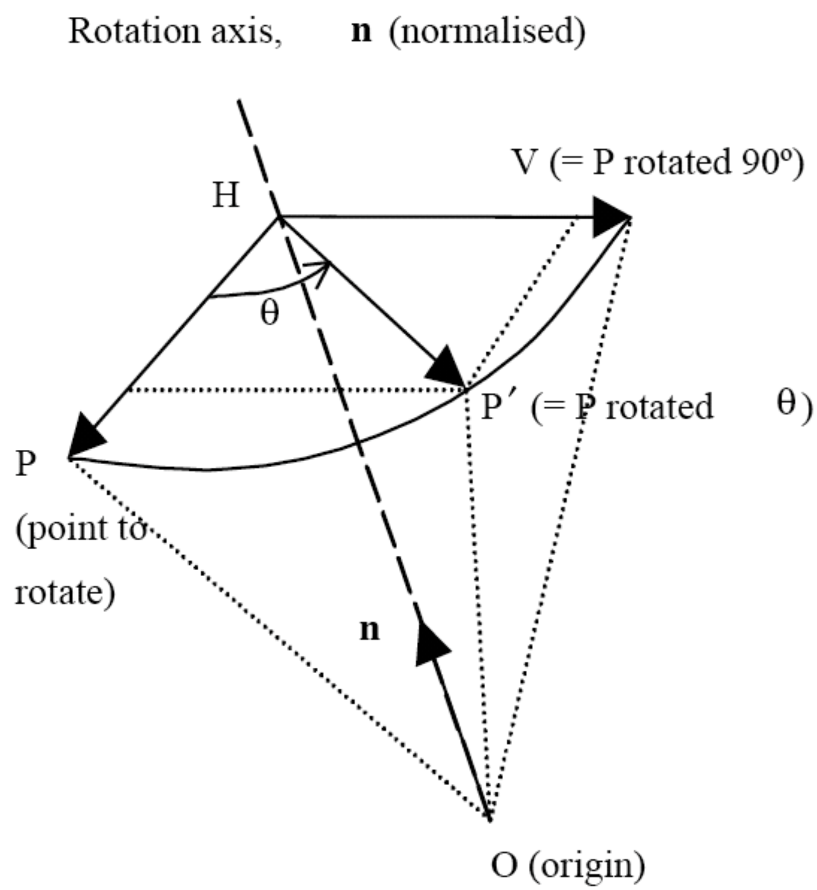
$$R_y(\theta) = \begin{bmatrix} cos\ \theta & 0 & sin\ \theta & 0 \\ 0 & 1 & 0 & 0 \\ -sin\ \theta & 0 & cos\ \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

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

# Rotation Around an Arbitrary Axis

- Rotate a point P around axis ***n*** (x,y,z) by angle θ

$$R = \begin{bmatrix} tx^2 + c & txy + sz & txz - sy & 0 \\ txy - sz & ty^2 + c & tyz + sx & 0 \\ txz + sy & tyz - sx & tz^2 + c & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$




- c = cos(θ)
- s = sin(θ)
- t = (1 - c)

Graphics Gems I, p. 466 & 498

# 3D Transformations: Reflect & Shear

- Reflection:

about x-y plane

$$F_z = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

- Shear:

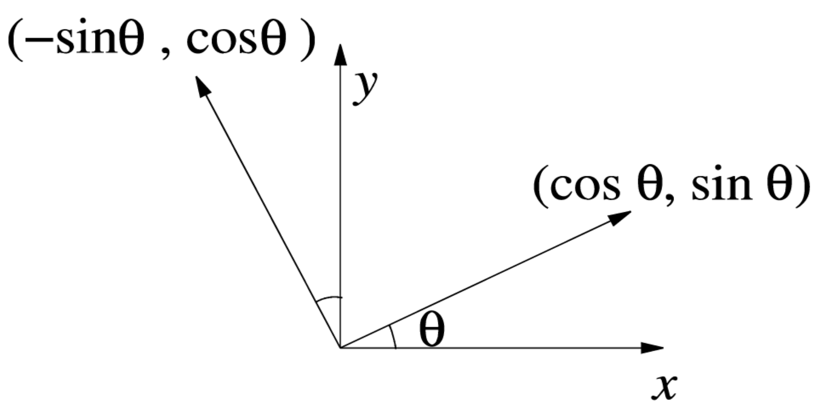
(function of z)

$$H_{xy}(\theta) = \begin{pmatrix} 1 & 0 & sh_x & 0 \\ 0 & 1 & sh_y & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

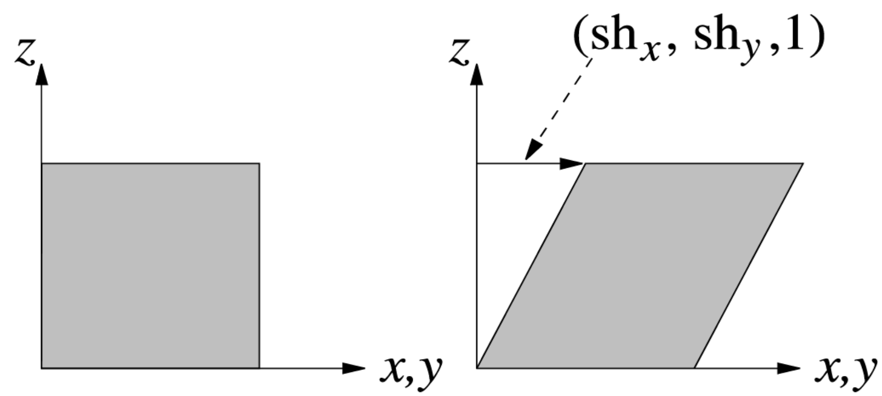


Pics/Math courtesy of Dave Mount @ UMD-CP

# Example




Rotation (about z)




Shear (orthogonal to z)



Pics/Math courtesy of Dave Mount @ UMD-CP

# Example: Composition of 3D Transformations

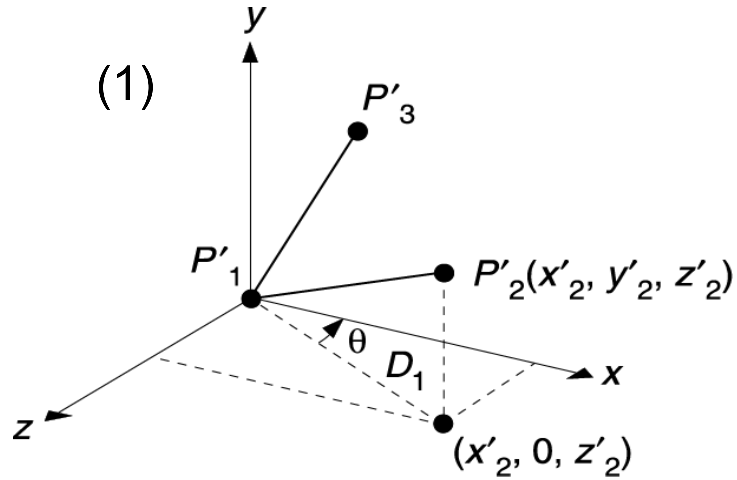
- Goal: Transform $P_1P_2$ and $P_1P_3$


(a) Initial position (b) Final position

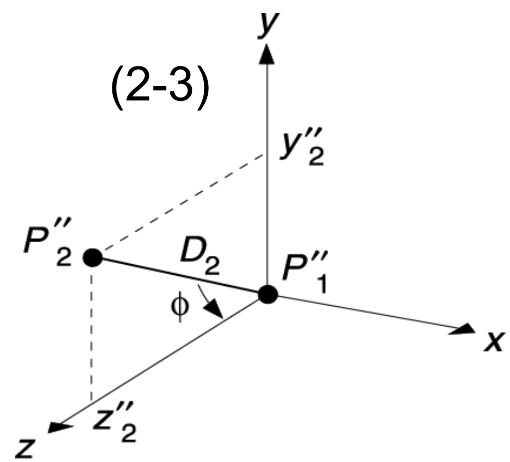
# Example (Cont.)

- Process
  1. Translate $P_1$ to $(0,0,0)$
  2. Rotate about $y$
  3. Rotate about $x$
  4. Rotate about $z$

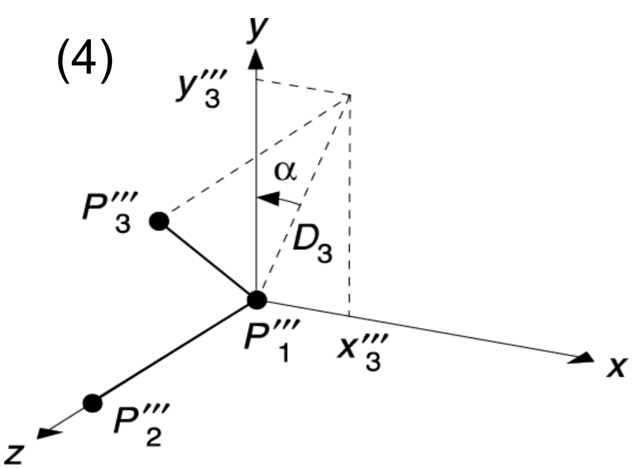
(1)




(2-3)




(4)

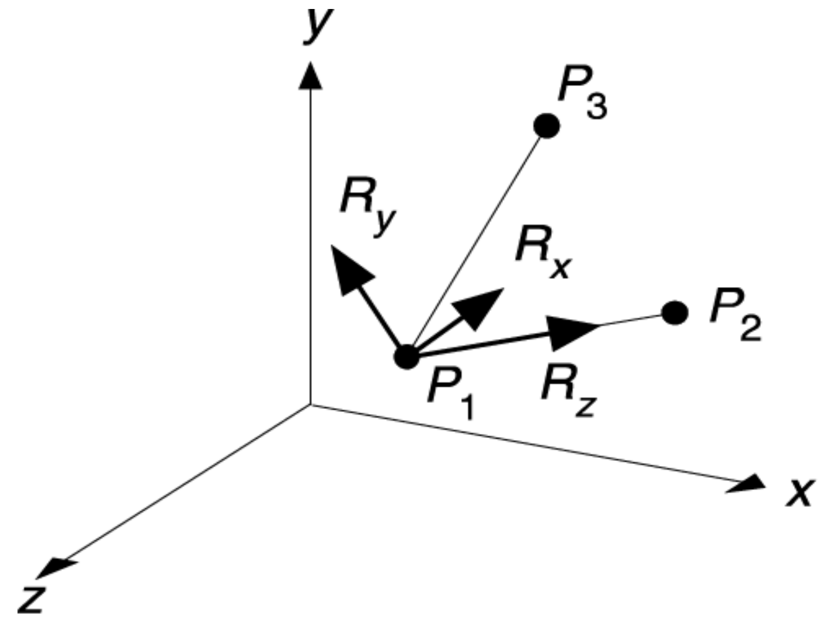





1994 Foley/VanDam/Finer/Huges/Phillips ICG

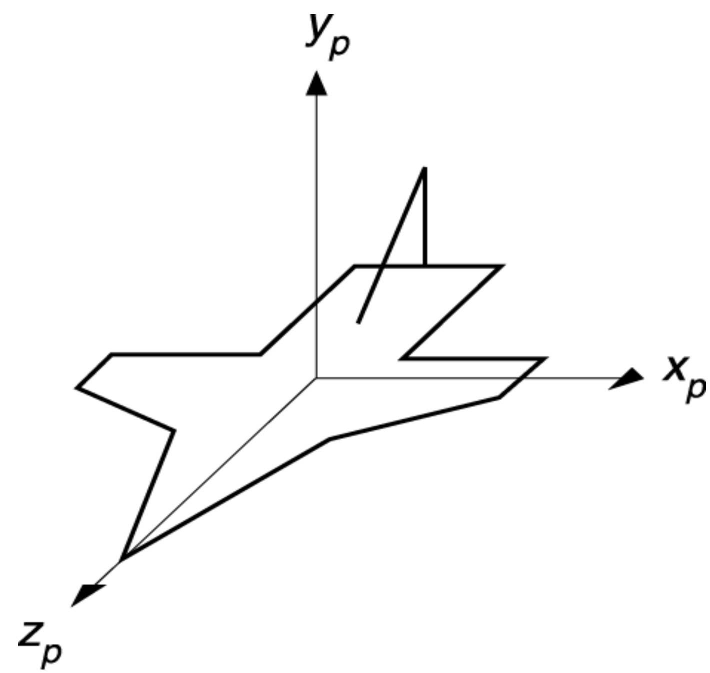
# Final Result

- What we've really done is transform the local coordinate system $R_x$, $R_y$, $R_z$ to align with the origin $x,y,z$



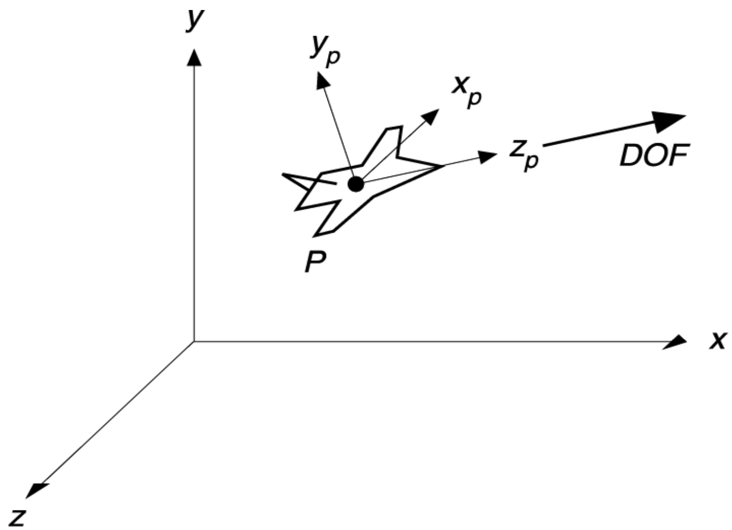
# Example 2: Composition of 3D Transformations

- Airplane defined in *x,y,z*
- Problem: want to point it in Dir of Flight (DOF) centered at point *P*
- Note: DOF is a vector
- Process:
  - Rotate plane
  - Move to P



# Example 2 (cont.)

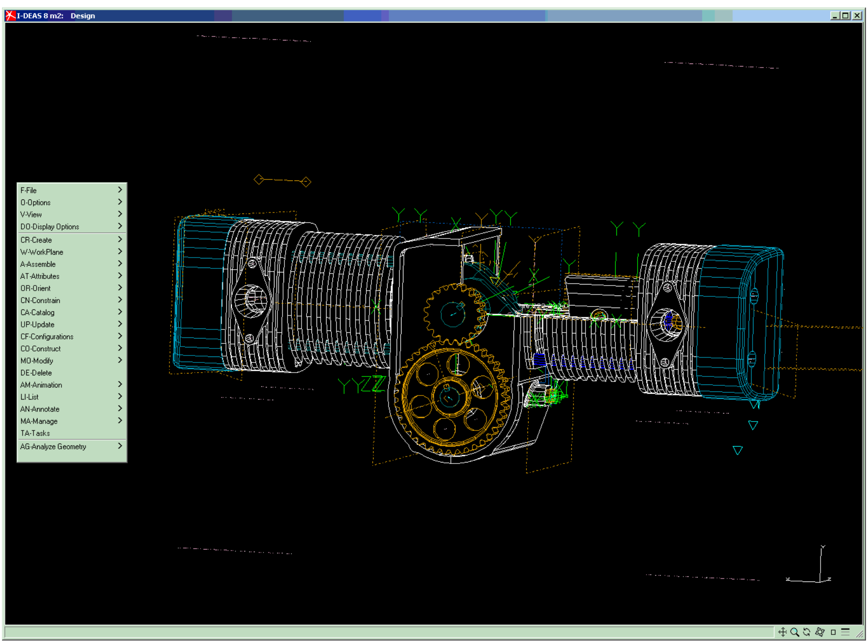
- $Z_p$ axis to be *DOF*
- $X_p$ axis to be a horizontal vector perpendicular to *DOF*
  - $y$ x *DOF*
- $Y_p$, vector perpendicular to both $Z_p$ and $X_p$ (i.e.$Z_p$ x $X_p$)

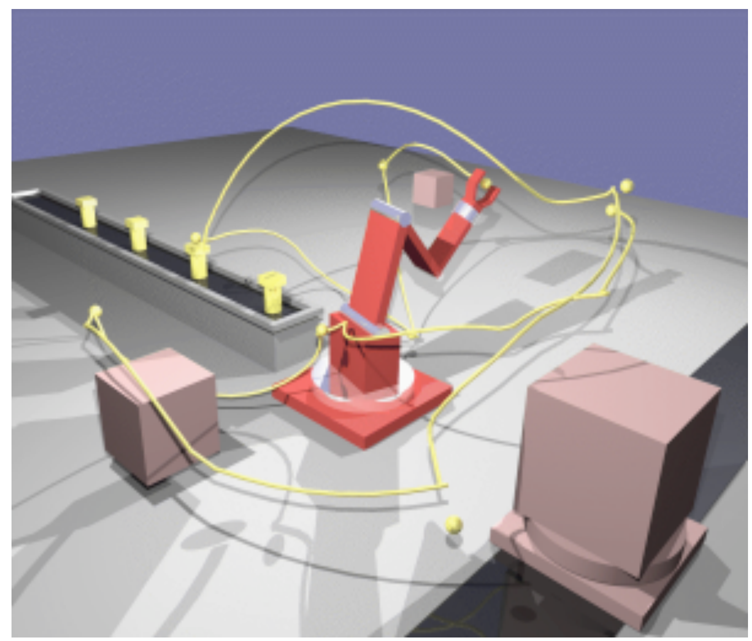



$$R = \begin{bmatrix} & & & 0 \\ |y \times DOF| & |DOF \times (y \times DOF)| & |DOF| & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

# Transformations to Change Coordinate Systems

- Issue: the world has many different relative frames of reference
- How do we transform among them?
- Example: CAD Assemblies & Animation Models



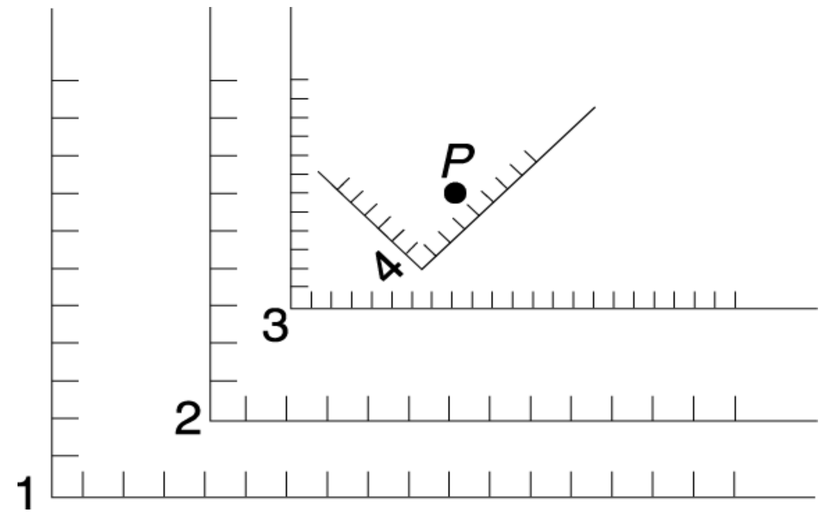
# Transformations to Change Coordinate Systems

- 4 coordinate systems
  1 point *P*

$$M_{1\leftarrow 2} = T(4,2)$$

$$M_{2\leftarrow 3} = T(2,3) \bullet S(0.5,0.5)$$

$$M_{3\leftarrow 4} = T(6.7,1.8) \bullet R(45^\circ)$$




$$M_{i\leftarrow k} = M_{i\leftarrow j} \cdot M_{j\leftarrow k}$$

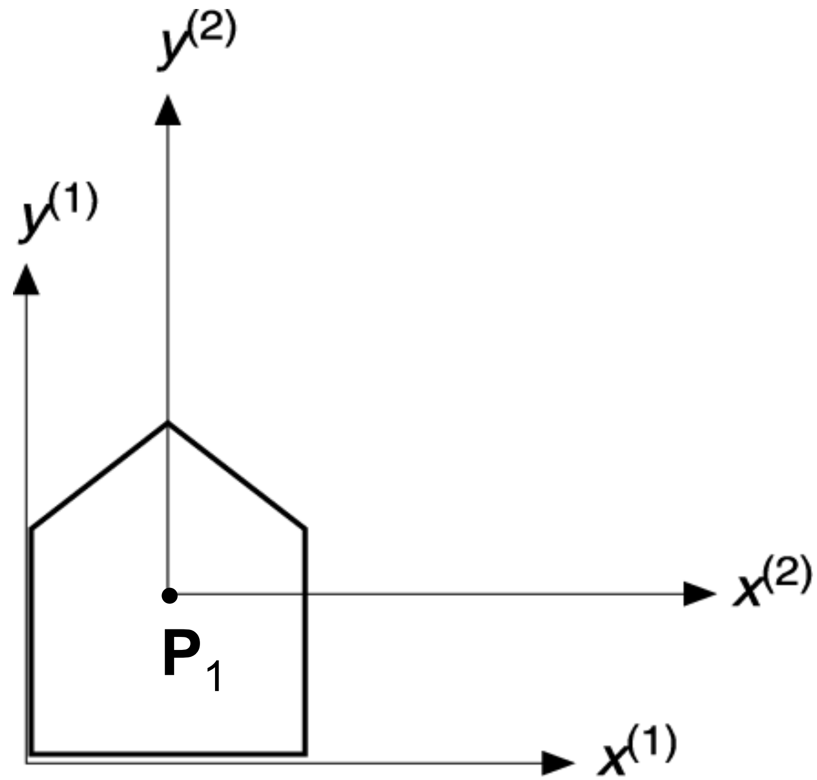
# Coordinate System Example (1)

- Translate the House to the origin

$$M_{1\leftarrow 2} = T(x_1, y_1)$$

$$M_{2\leftarrow 1} = \left(M_{1\leftarrow 2}\right)^{-1}$$

$$= T(-x_1, -y_1)$$

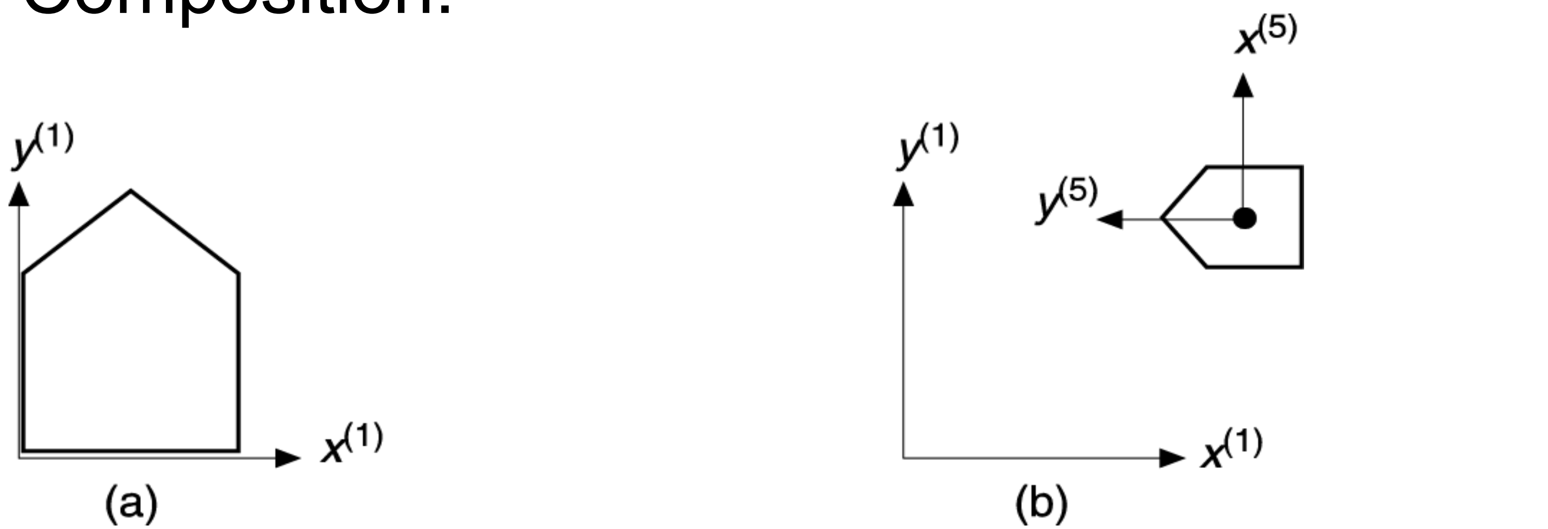


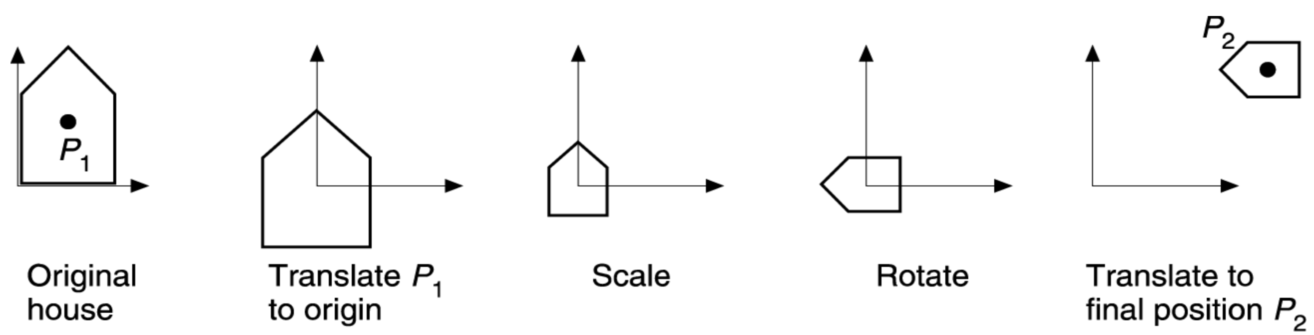

The matrix $M_{ij}$ that maps points from coordinate system j to i is the inverse of the matrix $M_{ji}$ that maps points from coordinate system j to coordinate system i.

# Coordinate System Example (2)

- Transformation Composition:

$$M_{5\leftarrow1} = M_{5\leftarrow4} \bullet M_{4\leftarrow3} \bullet M_{3\leftarrow2} \bullet M_{2\leftarrow1}$$



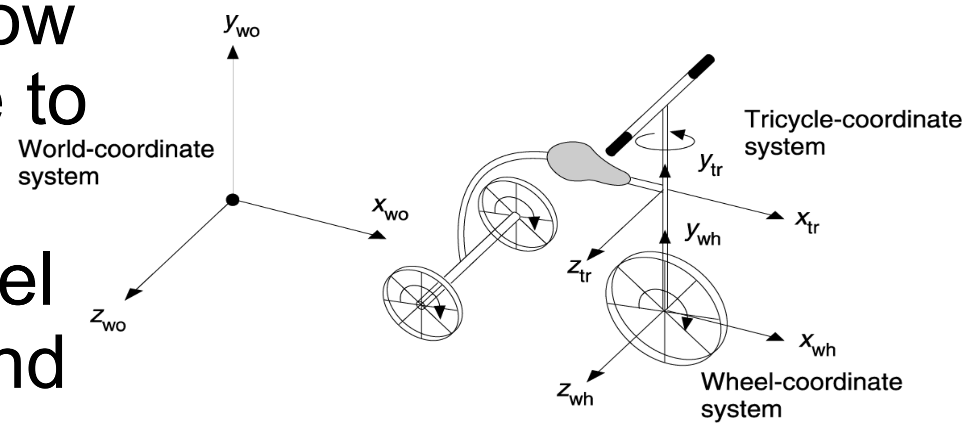



1994 Foley/VanDam/Finer/Huges/Phillips ICG

# World Coordinates and Local Coordinates

- To move the tricycle, we need to know how all of its parts relate to the WCS
- Example: front wheel rotates on the ground wrt the front wheel's z axis:

$$P^{(wo)} = T(\alpha r, 0, 0) \cdot R_z(\alpha) \cdot P^{(wh)}$$




Coordinates of *P* in wheel coordinate system:

$$P'^{(wh)} = R_z(\alpha) \cdot P^{(wh)}$$