

# **MEAM 520**

## **Lecture 4: Homogeneous Transformations**


Cynthia Sung, Ph.D.


Mechanical Engineering & Applied Mechanics


University of Pennsylvania

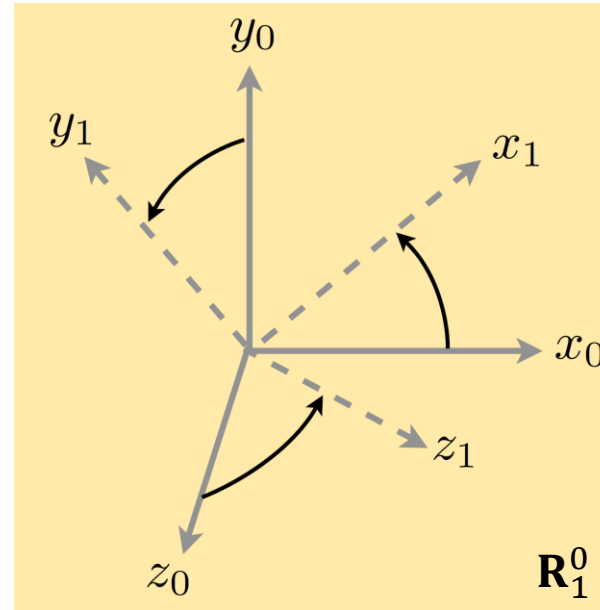
# Last Time: Rotation Matrices

$$\mathbf{R}_1^0 = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

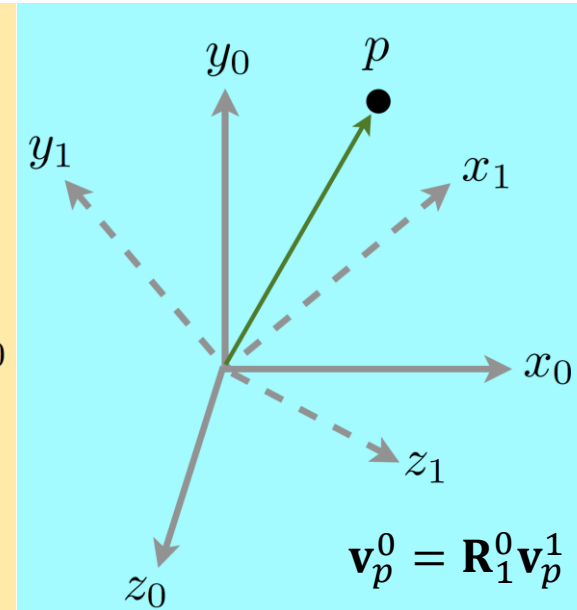

 $\hat{x}_1^0$


 $\hat{y}_1^0$

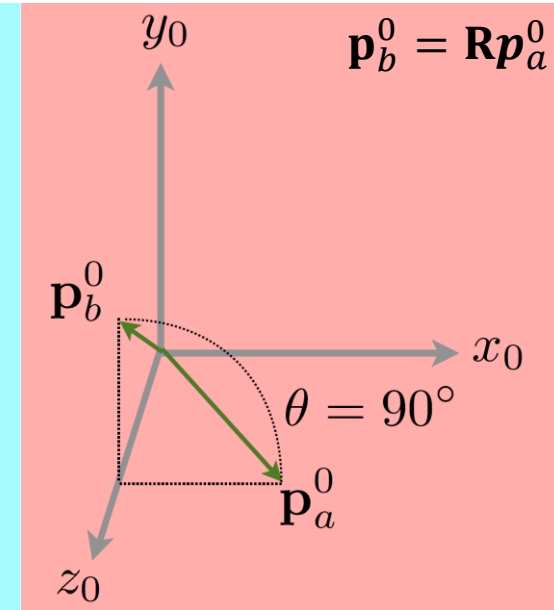

 $\hat{z}_1^0$



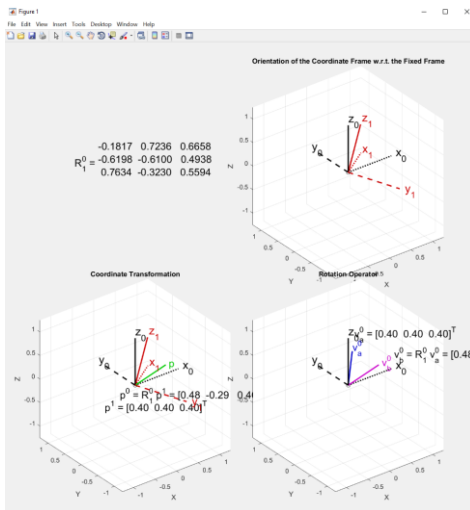
Orientation of one coordinate frame with respect to another frame



Coordinate transformation relating the coordinates of a point p in two different frames



Operator taking a vector and rotating it to yield a new vector in the same coordinate frame

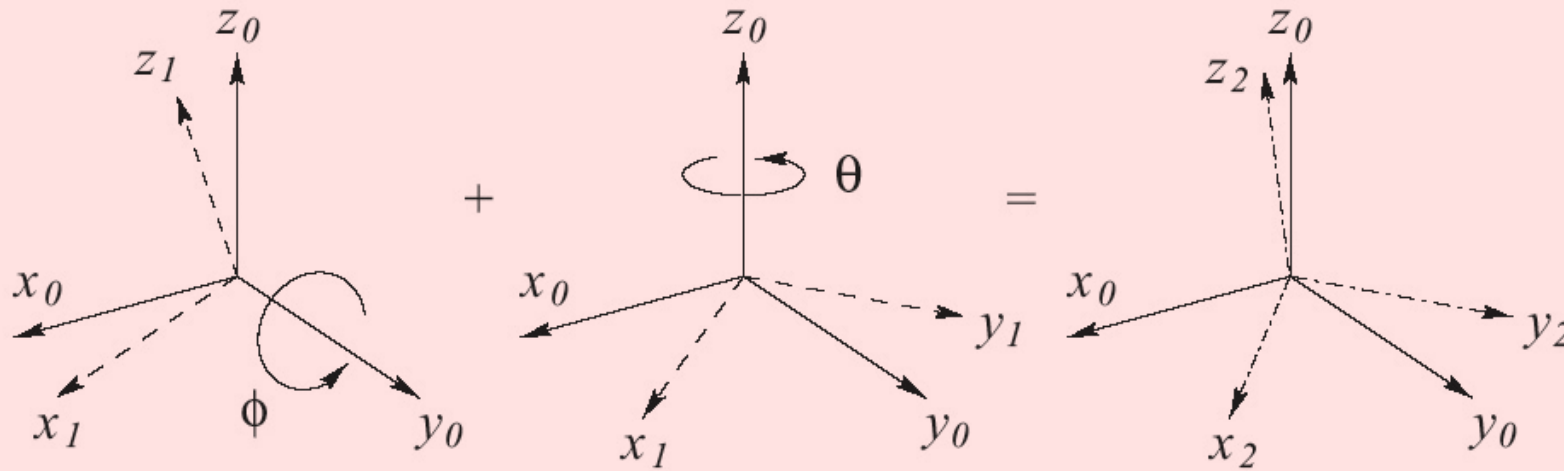


visualizeR.m

# Last Time: Multiplication Order for Rotation Matrices

Rotation about a fixed frame? **pre-multiply**

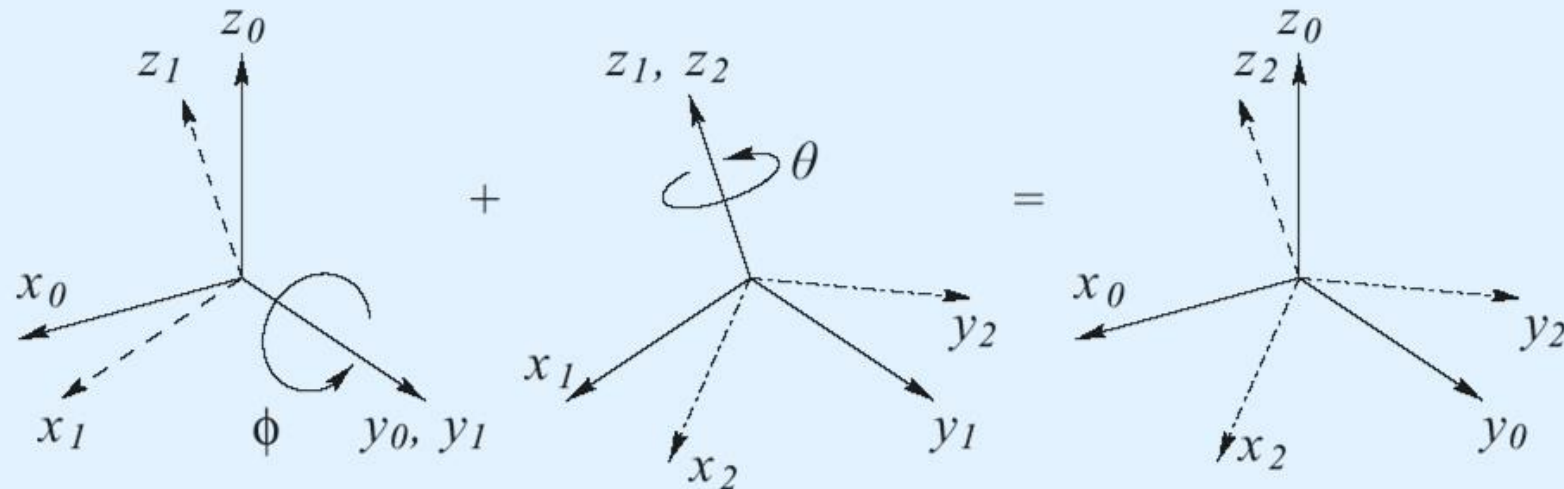
$$\mathbf{R}_2^0 = \mathbf{R}\mathbf{R}_1^0$$



Used in:  
Yaw/Pitch/Roll Angles

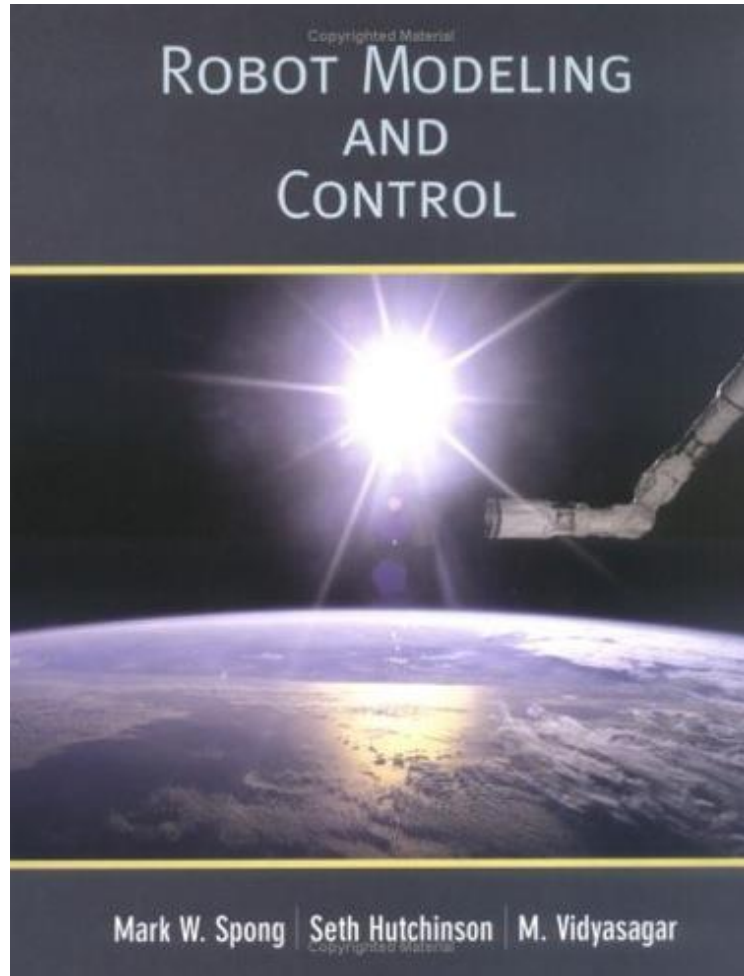
Rotation about intermediate frame? **post-multiply**

$$\mathbf{R}_2^0 = \mathbf{R}_1^0\mathbf{R}_2^1$$



Used in:  
Euler Angles

# Today: Homogeneous Transformations



## Chapter 2: Rigid Motions

- Read Sec. 2.6 - 2.8

# Lab 1 is posted (pre-lab due 9/12, lab due 9/19)

## Lab 1: Kinematic Characterization of the Lynx

MEAM 520, University of Pennsylvania

September 5, 2018

This lab consists of two portions, with a pre-lab due on **Wednesday, September 12, by midnight (11:59 p.m.)** and a lab report due on **Wednesday, September 19, by midnight (11:59 p.m.)**. Late submissions will be accepted until midnight on Saturday following the deadline, but they will be penalized by 25% for each partial or full day late. After the late deadline, no further assignments may be submitted; post a private message on Piazza to request an extension if you need one due to a special situation.

You may talk with other students about this assignment, ask the teaching team questions, use a calculator and other tools, and consult outside sources such as the Internet. To help you actually learn the material, what you submit must be your own work, not copied from any other individual or team. Any submissions suspected of violating Penn's Code of Academic Integrity will be reported to the Office of Student Conduct. When you get stuck, post a question on Piazza or go to office hours!

### Individual vs. Pair Programming

The pre-lab component of this lab must be completed and submitted individually on Canvas. For the remainder of the lab, you may work either individually or with a partner. If you do this lab with a partner, you may work with anyone you choose, but you must work with them for all parts of this assignment.

If you are in a pair, you will both turn in the same report and code (see Submission Instructions below), for which you are jointly responsible and you will both receive the same grade. Work closely with your partner throughout the lab, following these guidelines, which were adapted from "All I really needed to know about pair programming I learned in kindergarten," by Williams and Kessler, *Communications of the ACM*, May 2000. This article is available on Canvas under Files / Supplemental Material.

- Start with a good attitude, setting aside any skepticism, and expect to jell with your partner.
- Don't start alone. Arrange a meeting with your partner as soon as you can.
- Use just one setup, and sit side by side. For a programming component, a desktop computer with a large monitor is better than a laptop. Make sure both partners can see the screen.
- At each instant, one partner should be driving (writing, using the mouse/keyboard, moving the robot) while the other is continuously reviewing the work (thinking and making suggestions).
- Change driving/reviewing roles at least every 30 minutes, *even if one partner is much more experienced than the other*. You may want to set a timer to help you remember to switch.
- If you notice an error in the equation or code that your partner is writing, wait until they finish the line to correct them.
- Stay focused and on-task the whole time you are working together.
- Take a break periodically to refresh your perspective.
- Share responsibility for your project; avoid blaming either partner for challenges you run into.
- Recognize that working in pairs usually takes more time than working alone, but it produces better work, deeper learning, and a more positive experience for the participants.

## Forward Kinematics for Lynx robot

### Pre-lab:

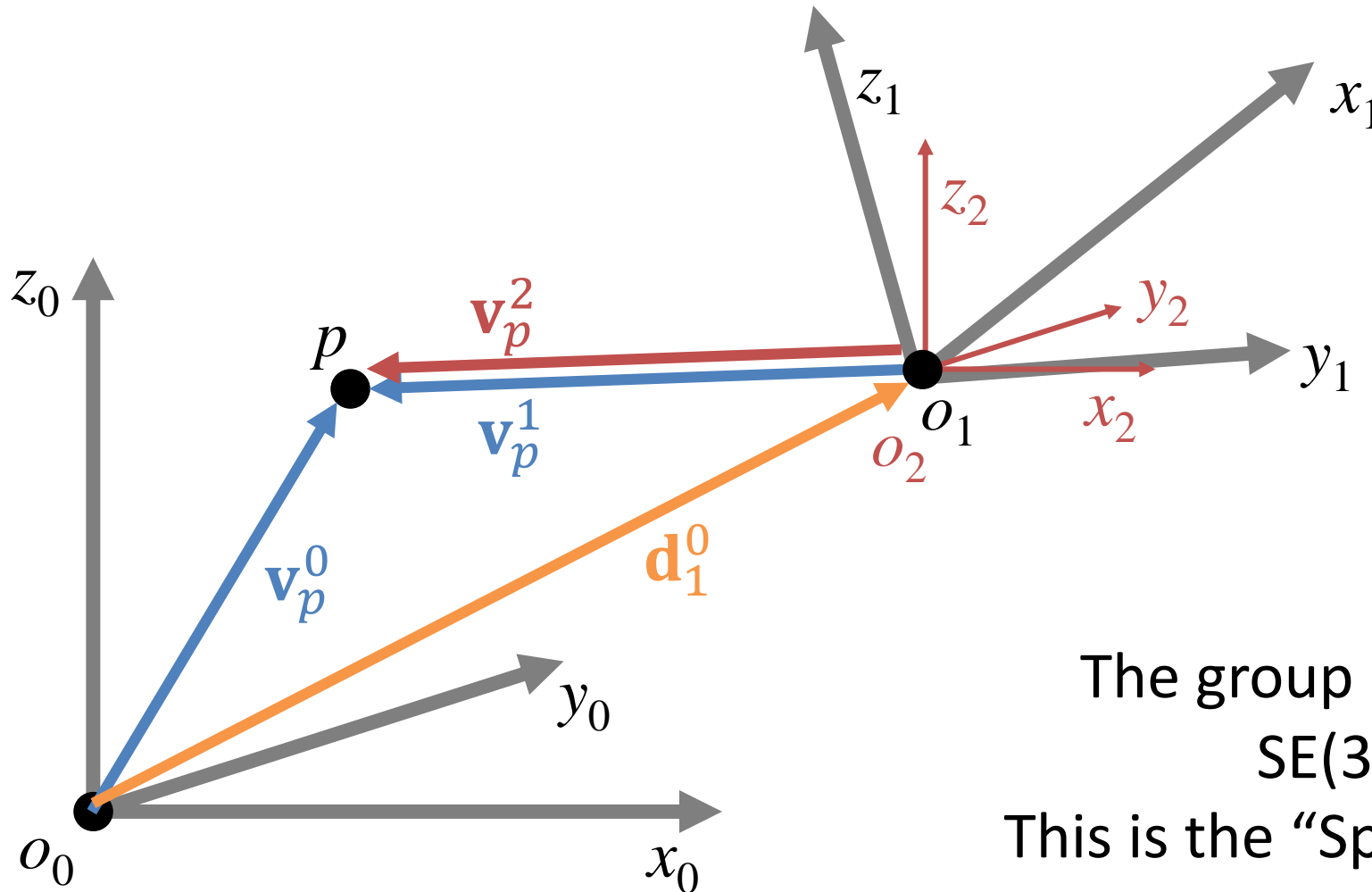
- Must be your own individual work
- Covers up to today's content

### Lab:

- You are allowed to do this in pairs, but it is not required.
- Covers up to next Tuesday's content

# Rigid Motions

combine pure **translation** and pure **rotation**



$$\mathbf{v}_p^0 = \mathbf{v}_p^2 + \mathbf{v}_{o_2}^0$$

$$\mathbf{v}_p^0 = \mathbf{R}_1^2 \mathbf{v}_p^1 + \mathbf{d}_1^0$$

$$\mathbf{v}_p^0 = \mathbf{R}_1^0 \mathbf{v}_p^1 + \mathbf{d}_1^0$$

The group of all rigid motions is  
 $\text{SE}(3) = \mathbb{R}^3 \times \text{SO}(3)$ .  
This is the “Special Euclidean Group.”

# Homogeneous Transforms

$$\mathbf{v}_p^0 = \mathbf{R}_1^0 \mathbf{v}_p^1 + \mathbf{d}_1^0$$

$$\begin{bmatrix} \mathbf{v}_p^0 \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_1^0 & \mathbf{d}_1^0 \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{v}_p^1 \\ 1 \end{bmatrix}$$

homogeneous representation  
of a vector

homogeneous transformation matrix

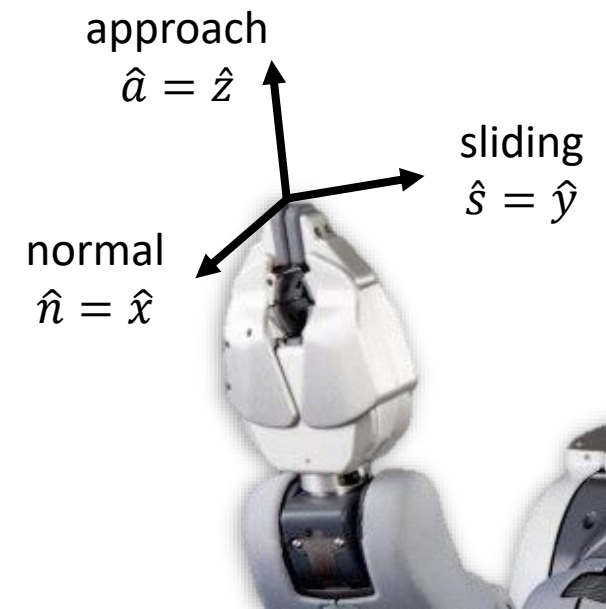
# Homogeneous Transformation Matrix

A **homogeneous transformation** is a matrix representation of rigid motion, defined as

$$\mathbf{H} = \begin{bmatrix} \mathbf{R} & \mathbf{d} \\ \mathbf{0} & 1 \end{bmatrix}$$

where  $\mathbf{R}$  is the 3x3 rotation matrix and  $\mathbf{d}$  is the 3x1 translation vector

$$\mathbf{H} = \begin{bmatrix} \mathbf{R} & \mathbf{d} \\ \text{three zeros} & \text{one one} \end{bmatrix} = \begin{bmatrix} n_x & s_x & a_x & d_x \\ n_y & s_y & a_y & d_y \\ n_z & s_z & a_z & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$





# Homogeneous Representation

A **homogeneous representation of a vector** is formed by concatenating the original vector with a unit scalar

$$\mathbf{P} = \begin{bmatrix} \mathbf{p} \\ 1 \end{bmatrix}$$

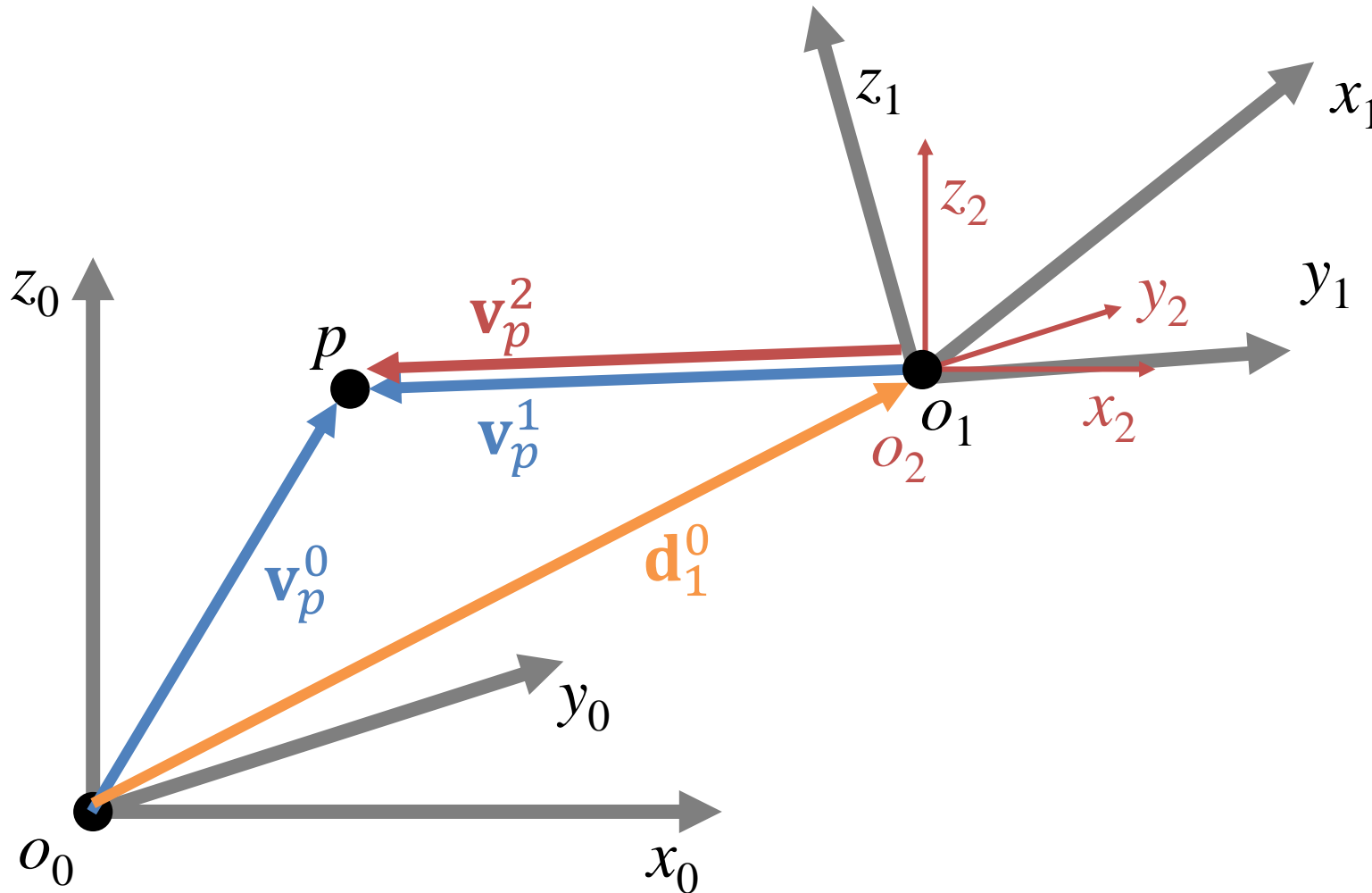
where  $\mathbf{p}$  is the 3x1 vector

$$\mathbf{P} = \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}$$

$\mathbf{v}_p^1$

# Rigid Motions

combine pure **translation** and pure **rotation**



$$\mathbf{v}_p^0 = \mathbf{R}_1^0 \mathbf{v}_p^1 + \mathbf{d}_1^0$$

$$\mathbf{v}_p^0 = \begin{bmatrix} \mathbf{R}_1^0 & \mathbf{d}_1^0 & \mathbf{v}_p^1 \end{bmatrix} = \begin{bmatrix} n_x & s_x & a_x & d_x \\ n_y & s_y & a_y & d_y \\ n_z & s_z & a_z & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}$$

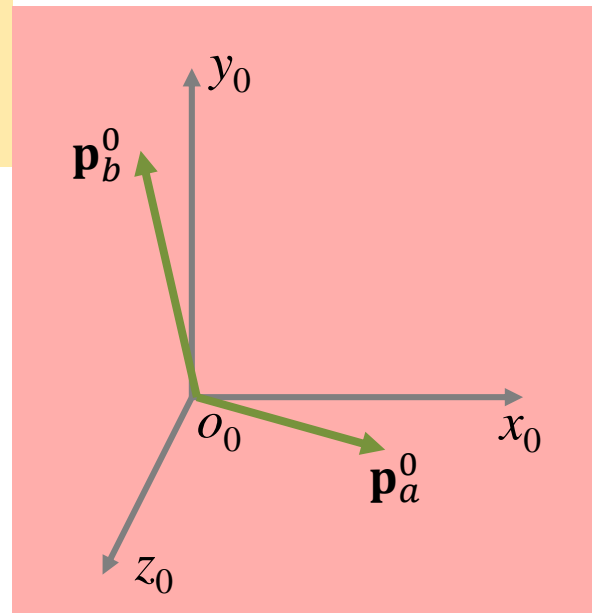
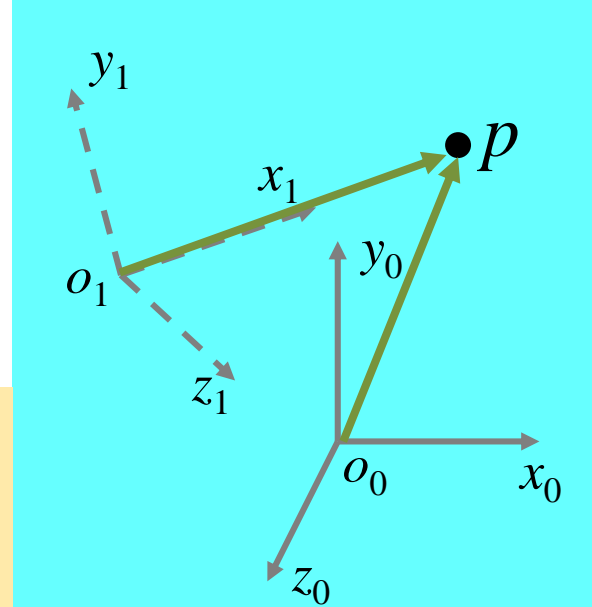
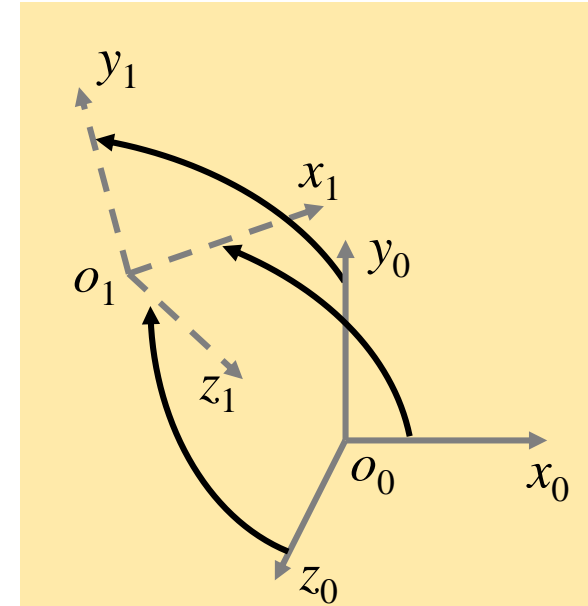
$$\mathbf{P}^0 = \mathbf{H}_1^0 \mathbf{P}^1$$

$$\mathbf{P}^0 = \mathbf{H}_1^0 \mathbf{P}^1$$

# Transformation Matrices

Serve 3 purposes:

1. Coordinate transformations relating coordinates of a point  $p$  in two different frames
2. Orientation of a transformed coordinate frame with respect to a fixed frame
3. Operator taking a vector and transforming it to yield a new vector in the same coordinate frame

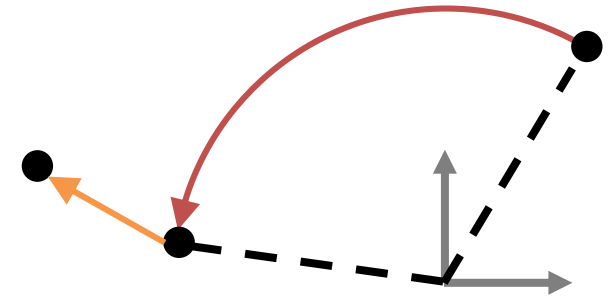


$$\mathbf{B} = \mathbf{H}\mathbf{A}$$

# Rigid Motions (Operator Interpretation)

Homogeneous transformations perform **rotation then translation**

$$\mathbf{v}_p^0 = \mathbf{R}\mathbf{v}_p^1 + \mathbf{d}$$

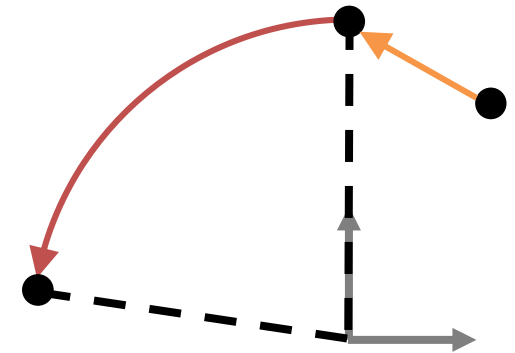


What would **translation then rotation** look like?

$$\mathbf{v}_p^0 = \mathbf{R}(\mathbf{v}_p^1 + \mathbf{d})$$

$$\mathbf{v}_p^0 = \mathbf{R}\mathbf{v}_p^1 + \mathbf{R}\mathbf{d}$$

$$\mathbf{H} = \begin{bmatrix} \mathbf{R} & \mathbf{R}\mathbf{d} \\ \mathbf{0} & 1 \end{bmatrix}$$

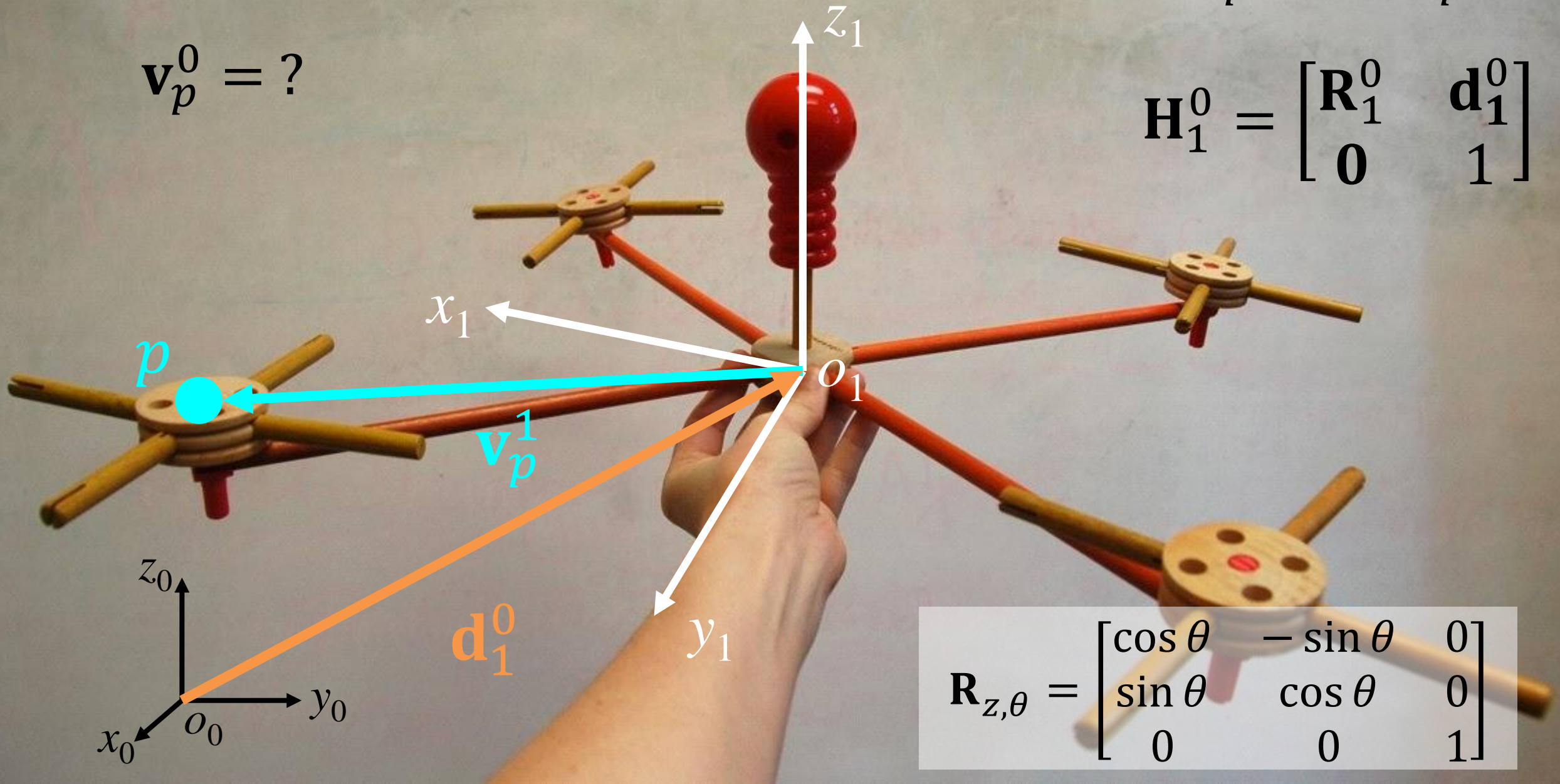


# Example with quadrotor model

$$\mathbf{v}_p^0 = ?$$

$$\mathbf{v}_p^0 = \mathbf{H}_1^0 \mathbf{v}_p^1$$

$$\mathbf{H}_1^0 = \begin{bmatrix} \mathbf{R}_1^0 & \mathbf{d}_1^0 \\ \mathbf{0} & 1 \end{bmatrix}$$



$$\mathbf{R}_{z,\theta} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

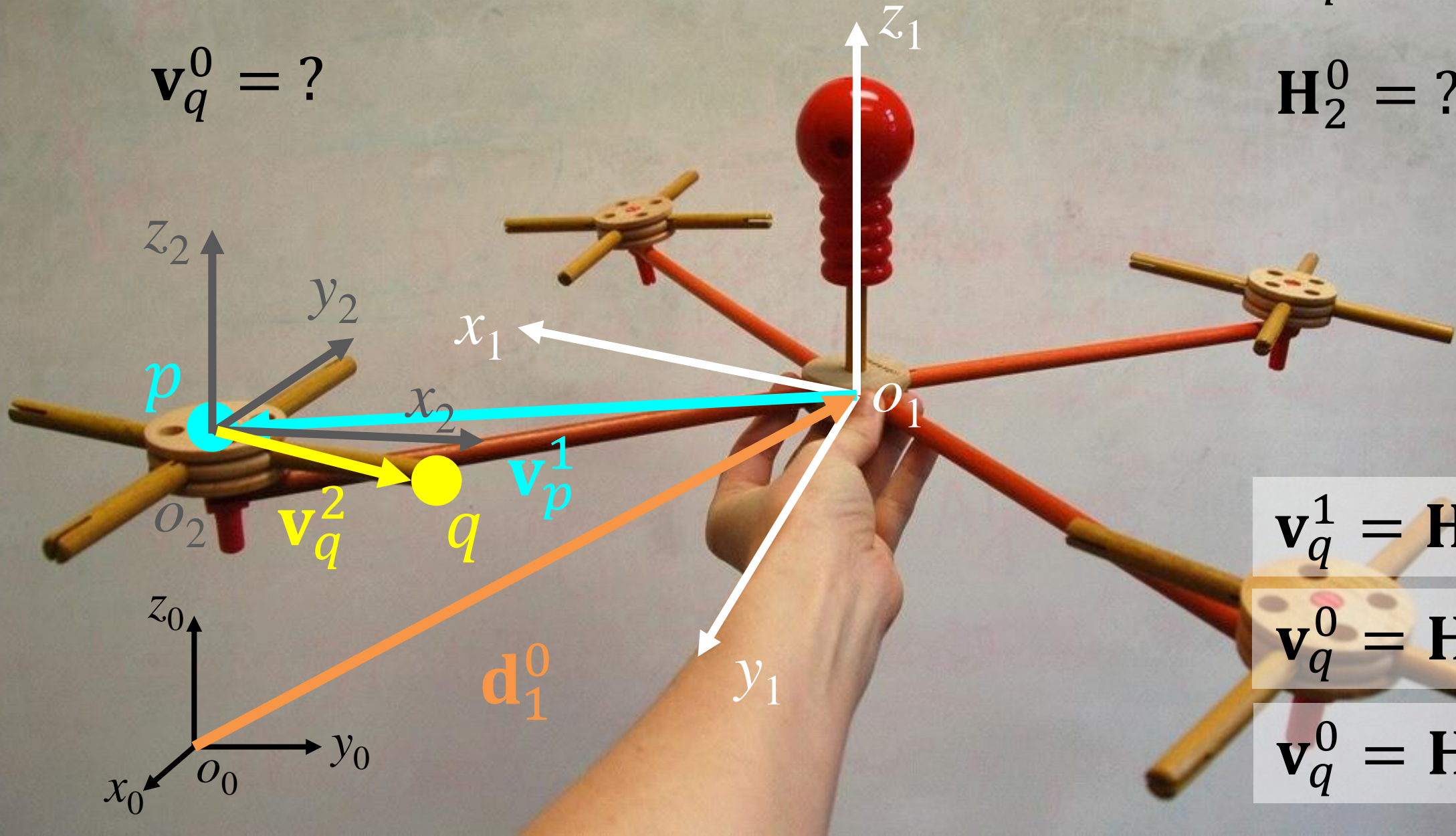


# Example with quadrotor model

$$\mathbf{v}_q^0 = ?$$

$$\mathbf{v}_q^0 = \mathbf{H}_2^0 \mathbf{v}_q^2$$

$$\mathbf{H}_2^0 = ?$$



$$\mathbf{v}_q^1 = \mathbf{H}_2^1 \mathbf{v}_q^2$$

$$\mathbf{v}_q^0 = \mathbf{H}_1^0 \mathbf{v}_q^1$$

$$\mathbf{v}_q^0 = \mathbf{H}_1^0 \mathbf{H}_2^1 \mathbf{v}_q^2$$

# Compositions of Homogeneous Transformations

Composition of multiple transforms is the same as for rotation matrices

**post-multiply** when successive transformations  
are relative to the intermediate frame

$$\mathbf{H}_2^0 = \mathbf{H}_1^0 \mathbf{H}_2^1$$

**pre-multiply** when successive transformations  
are relative to the fixed world frame

$$\mathbf{H}_2^0 = \mathbf{H} \mathbf{H}_1^0$$

## Composition via an Intermediate Frame

$$\begin{aligned}\mathbf{H}_2^0 &= \mathbf{H}_1^0 \mathbf{H}_2^1 = \begin{bmatrix} \mathbf{R}_1^0 & \mathbf{d}_1^0 \\ \mathbf{0} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_2^1 & \mathbf{d}_2^1 \\ \mathbf{0} & 1 \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{R}_2^0 & \mathbf{R}_1^0 \mathbf{d}_2^1 + \mathbf{d}_1^0 \\ \mathbf{0} & 1 \end{bmatrix}\end{aligned}$$

Q: What is the inverse of a homogeneous transformation?

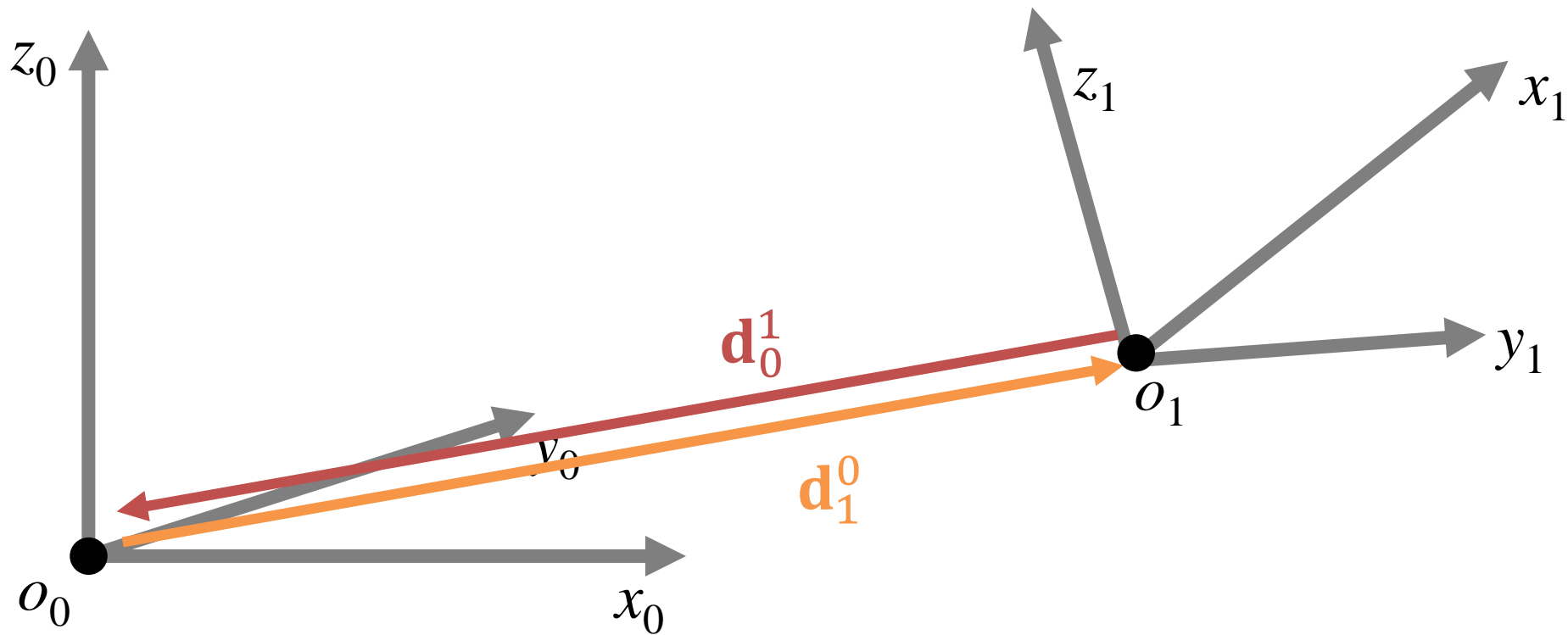


## Inverse Transform

$$\mathbf{H}_0^1 = \begin{bmatrix} \mathbf{R}_0^1 & \mathbf{d}_0^1 \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} & & & \\ & & & \\ & & & \\ & \mathbf{0} & & 1 \end{bmatrix}$$

## Inverse Transform

$$\mathbf{H}_0^1 = \begin{bmatrix} \mathbf{R}_0^1 & \mathbf{d}_0^1 \\ \mathbf{0} & 1 \end{bmatrix} = \begin{bmatrix} \underbrace{(\mathbf{R}_1^0)^{-1}} & \underbrace{?} \\ (\mathbf{R}_1^0)^\top & -(\mathbf{R}_1^0)^\top \mathbf{d}_1^0 \\ \mathbf{0} & 1 \end{bmatrix}$$



## Other Properties of Homogeneous Transforms

$$\mathbf{H} = \begin{bmatrix} \mathbf{R} & \mathbf{d} \\ \mathbf{0} & 1 \end{bmatrix}$$

- $\|\mathbf{HP} - \mathbf{HQ}\| = \|\mathbf{P} - \mathbf{Q}\|$     Why?  $\|\mathbf{HP} - \mathbf{HQ}\| = \|(\mathbf{R}\mathbf{v}_p + \mathbf{d}) - (\mathbf{R}\mathbf{v}_q + \mathbf{d})\| = \|\mathbf{R}(\mathbf{v}_p - \mathbf{v}_q)\|$

Geometric interpretation: **H** is distance-preserving

- $\langle \mathbf{HP} - \mathbf{HQ}, \mathbf{HS} - \mathbf{HQ} \rangle = \langle \mathbf{P} - \mathbf{Q}, \mathbf{S} - \mathbf{Q} \rangle$

Geometric interpretation: **H** is angle-preserving

# Basic Homogeneous Transformations

$$\text{Trans}_{x,a} = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Trans}_{y,b} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Trans}_{z,c} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

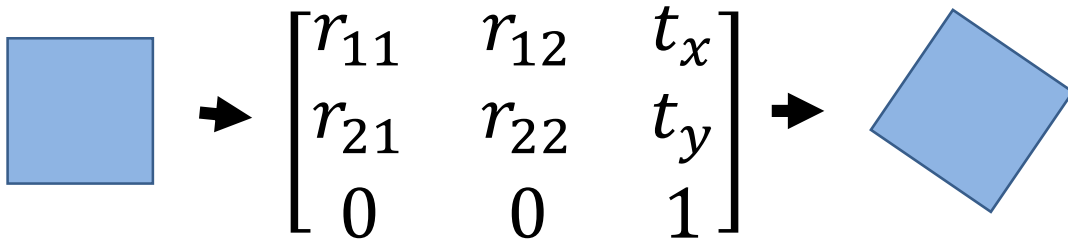
$$\text{Rot}_{x,\alpha} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_\alpha & -s_\alpha & 0 \\ 0 & s_\alpha & c_\alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\text{Rot}_{y,\beta} = \begin{bmatrix} c_\beta & 0 & s_\beta & 0 \\ 0 & 1 & 0 & 0 \\ -s_\beta & 0 & c_\beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

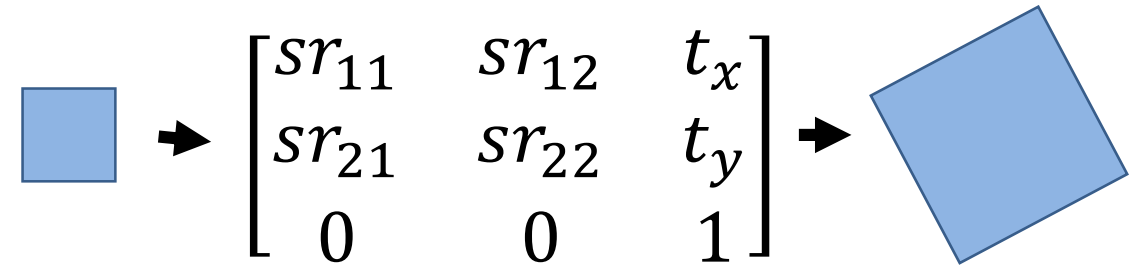
$$\text{Rot}_{z,\gamma} = \begin{bmatrix} c_\gamma & -s_\gamma & 0 & 0 \\ s_\gamma & c_\gamma & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

# Transformation Matrices (in 2D)

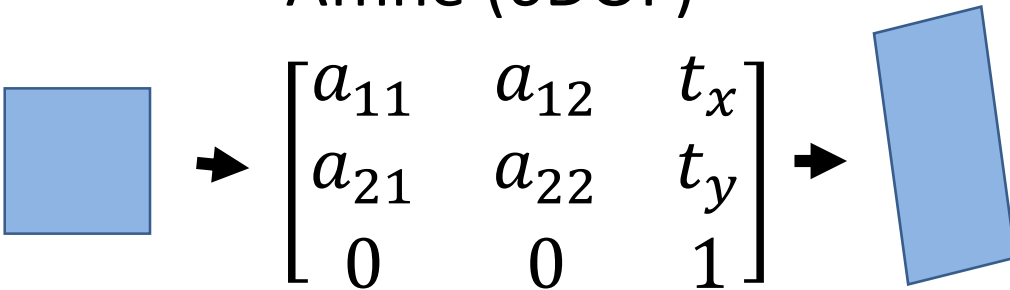
Homogeneous (3DOF)



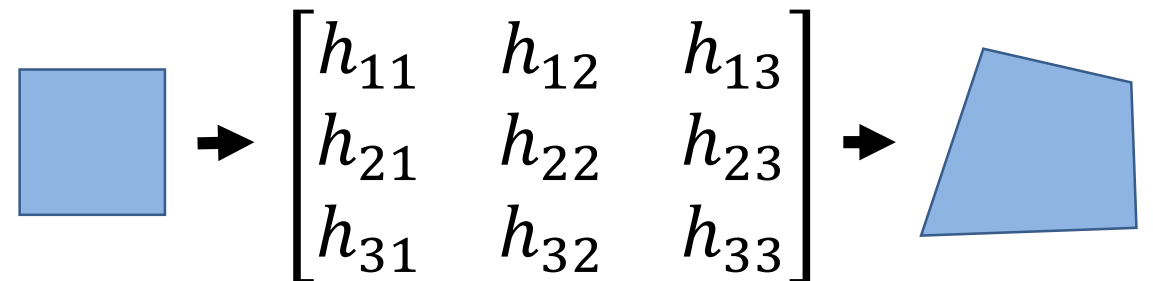
Similarity (4DOF)



Affine (6DOF)

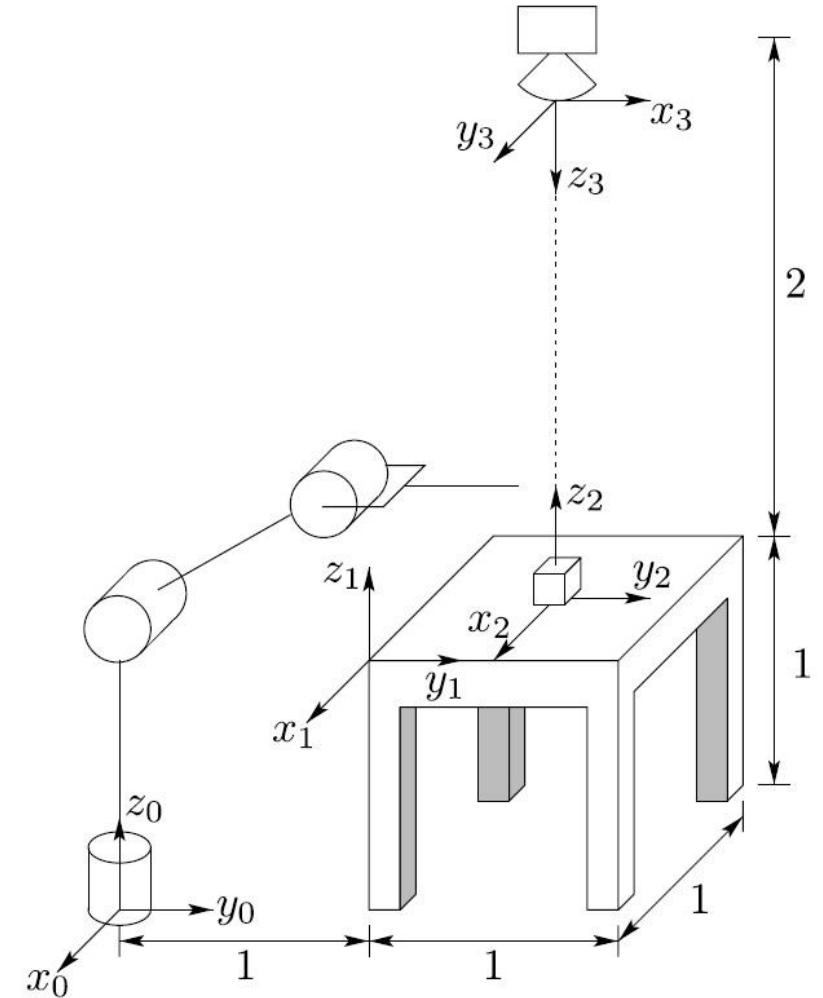


Projective (8DOF)



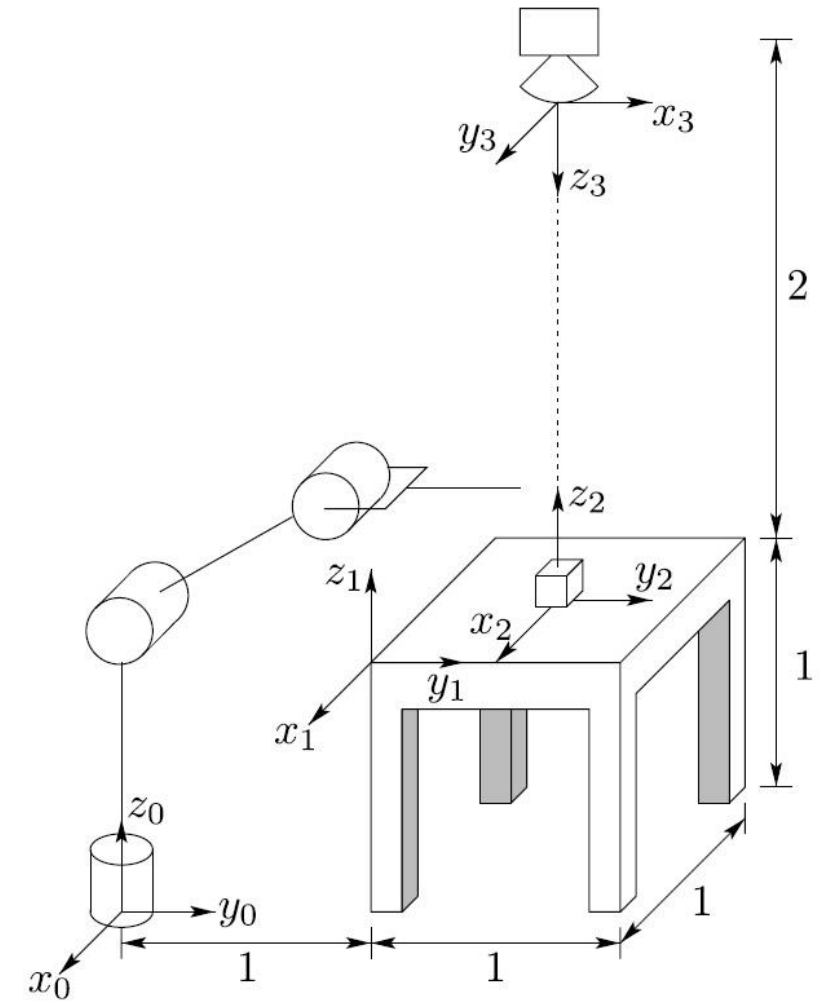
## Practice:

1. Write the homogeneous transformations relating frame 0, frame 1, and frame 2 to the camera frame 3.
2. Find the homogeneous transformation relating cube frame 2 to base frame 0.
3. Write an expression for the homogeneous transformation relating cube frame 2 to robot frame 0 as a function of the others.
4. Check this.



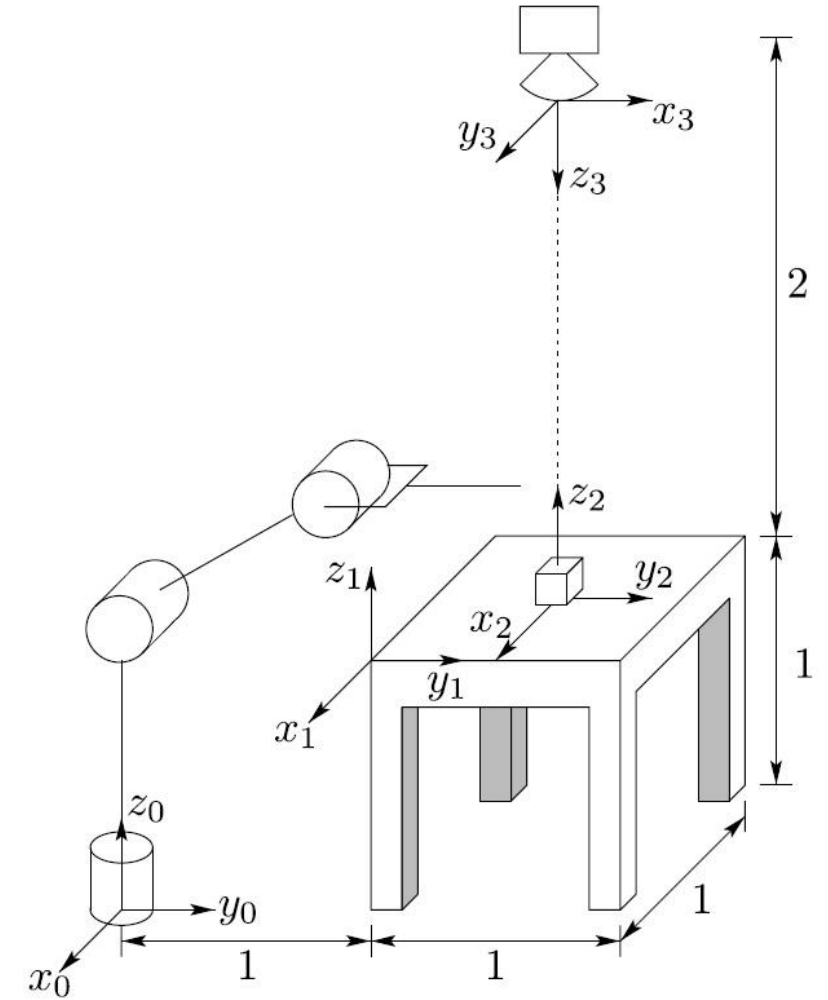
## Practice:

Write the homogeneous transformations relating frame 0, frame 1, and frame 2 to the camera frame 3.



## Practice:

Find the homogeneous transformation relating cube frame 2 to base frame 0.



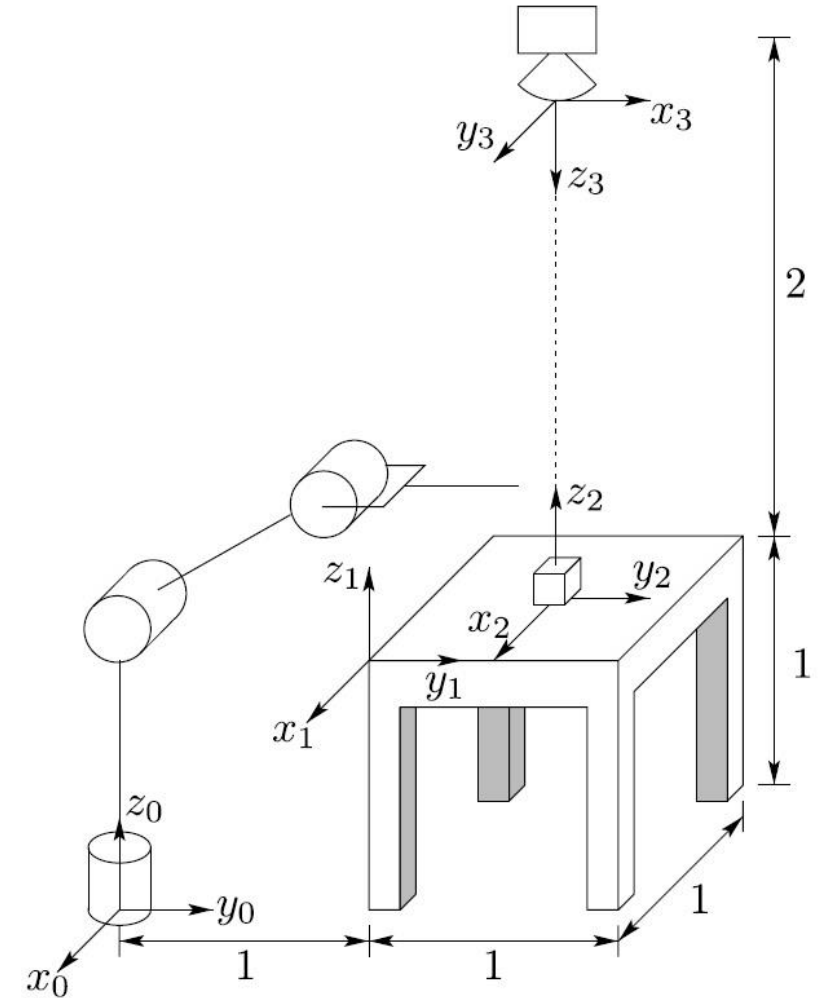


## Practice:

Write an expression for the homogeneous transformation relating cube frame 2 to base frame 0 as a function of the others.

$$\mathbf{H}_0^3 = \begin{bmatrix} 0 & 1 & 0 & -1.5\text{m} \\ 1 & 0 & 0 & 0.5\text{m} \\ 0 & 0 & -1 & 3\text{m} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{H}_1^3 = \begin{bmatrix} 0 & 1 & 0 & -0.5\text{m} \\ 1 & 0 & 0 & 0.5\text{m} \\ 0 & 0 & -1 & 2\text{m} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{H}_2^3 = \begin{bmatrix} 0 & 1 & 0 & 0\text{m} \\ 1 & 0 & 0 & 0\text{m} \\ 0 & 0 & -1 & 2\text{m} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{H}_2^0 = \begin{bmatrix} 1 & 0 & 0 & -0.5\text{m} \\ 0 & 1 & 0 & 1.5\text{m} \\ 0 & 0 & 1 & 1\text{m} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



# Practice:

Check

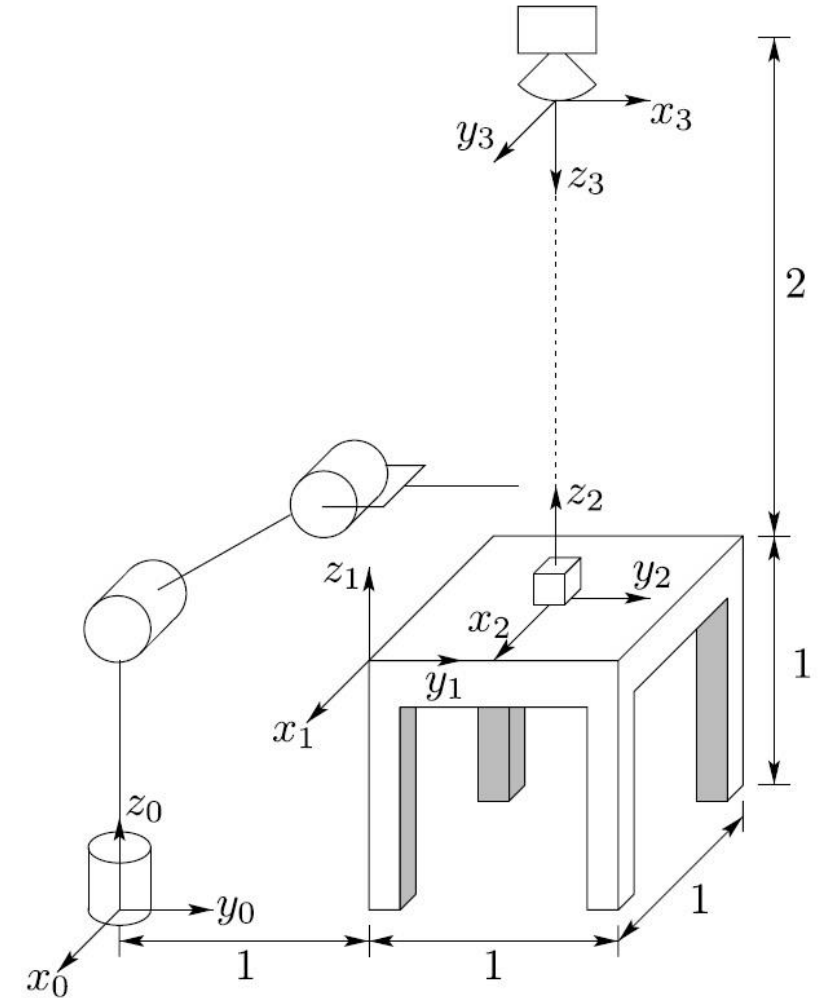
$$\mathbf{H}_0^3 = \begin{bmatrix} 0 & 1 & 0 & -1.5\text{m} \\ 1 & 0 & 0 & 0.5\text{m} \\ 0 & 0 & -1 & 3\text{m} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{H}_1^3 = \begin{bmatrix} 0 & 1 & 0 & -0.5\text{m} \\ 1 & 0 & 0 & 0.5\text{m} \\ 0 & 0 & -1 & 2\text{m} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{H}_2^3 = \begin{bmatrix} 0 & 1 & 0 & 0\text{m} \\ 1 & 0 & 0 & 0\text{m} \\ 0 & 0 & -1 & 2\text{m} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{H}_2^0 = \begin{bmatrix} 1 & 0 & 0 & -0.5\text{m} \\ 0 & 1 & 0 & 1.5\text{m} \\ 0 & 0 & 1 & 1\text{m} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{H}_2^0 = (\mathbf{H}_0^3)^{-1} \mathbf{H}_2^3$$



# Computational Considerations

What is the difference between:

$$\mathbf{P}^0 = \mathbf{H}_1^0 \mathbf{P}^1 \quad \text{and} \quad \mathbf{p}_v^0 = \mathbf{R}_1^0 \mathbf{p}_v^1 + \mathbf{d}_1^0$$

# Computational Considerations

What is the difference between:

$$\mathbf{P}^0 = \mathbf{H}_1^0 \mathbf{P}^1$$

and

practical systems use this

$$\mathbf{p}_v^0 = \mathbf{R}_1^0 \mathbf{p}_v^1 + \mathbf{d}_1^0$$

4 elements x (4 mults + 3 adds)  
28 operations

3 elements x (3 mults + 3 adds)  
18 operations

multiplying by 0s and 1s is wasteful

# Computational Considerations

What is the difference between:

$$\mathbf{p}_v^0 = \left( \mathbf{R}_1^0 (\mathbf{R}_2^1 \mathbf{R}_3^2) \right) \mathbf{p}_v^3 \quad \text{and} \quad \mathbf{p}_v^0 = \mathbf{R}_1^0 \left( \mathbf{R}_2^1 (\mathbf{R}_3^2 \mathbf{p}_v^3) \right)$$

# Computational Considerations

What is the difference between:

Use this if **R** matrices constant

$$\mathbf{p}_v^0 = \left( \mathbf{R}_1^0 (\mathbf{R}_2^1 \mathbf{R}_3^2) \right) \mathbf{p}_v^3$$

multiplying 3x3 matrices takes  
27 mults and 18 adds

105 operations

Use this if **R** matrices change

and

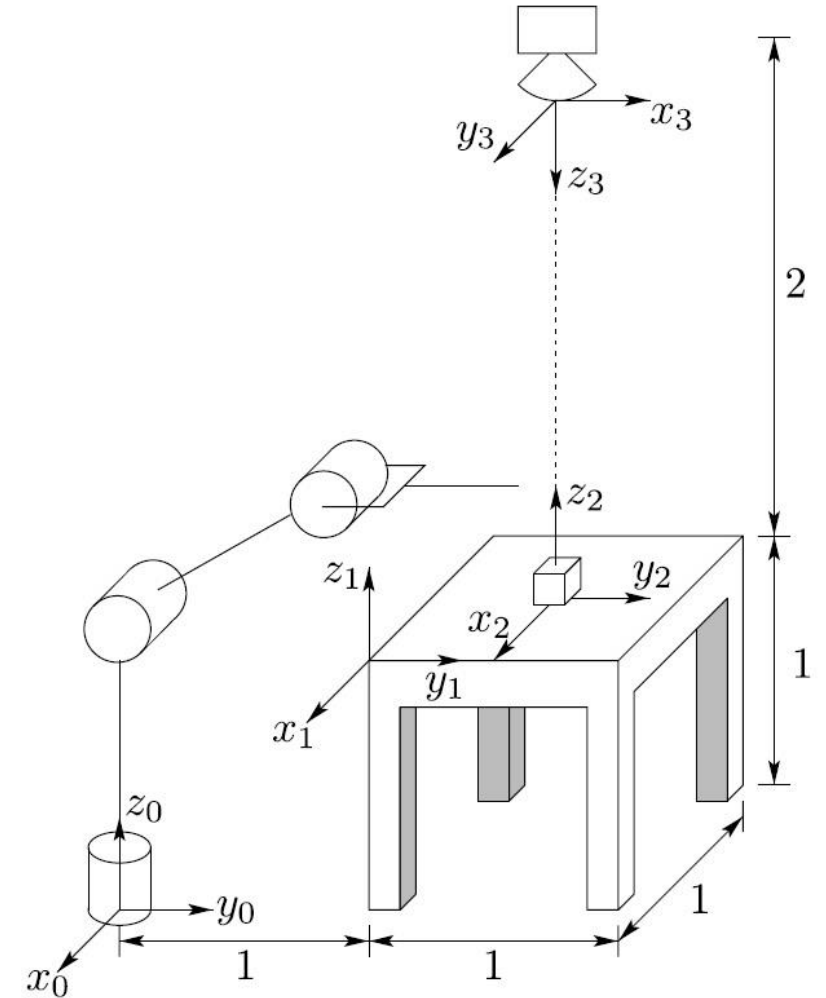
$$\mathbf{p}_v^0 = \mathbf{R}_1^0 \left( \mathbf{R}_2^1 (\mathbf{R}_3^2 \mathbf{p}_v^3) \right)$$

multiplying a 3x3 matrix by a 3x1 vector takes  
9 mults and 6 adds

45 operations

# Example

What is the best way to organize the computation to minimize the calculation effort?



We'll never worry about speed in lab, so it's fine to use  $\mathbf{P}^0 = \left( \mathbf{H}_1^0 (\mathbf{H}_2^1 \mathbf{H}_3^2) \right) \mathbf{P}^3$

# Next time: Forward Kinematics for Serial Manipulators

## Chapter 3: Forward and Inverse Kinematics

- Read Sec. 3.intro - 3.2

