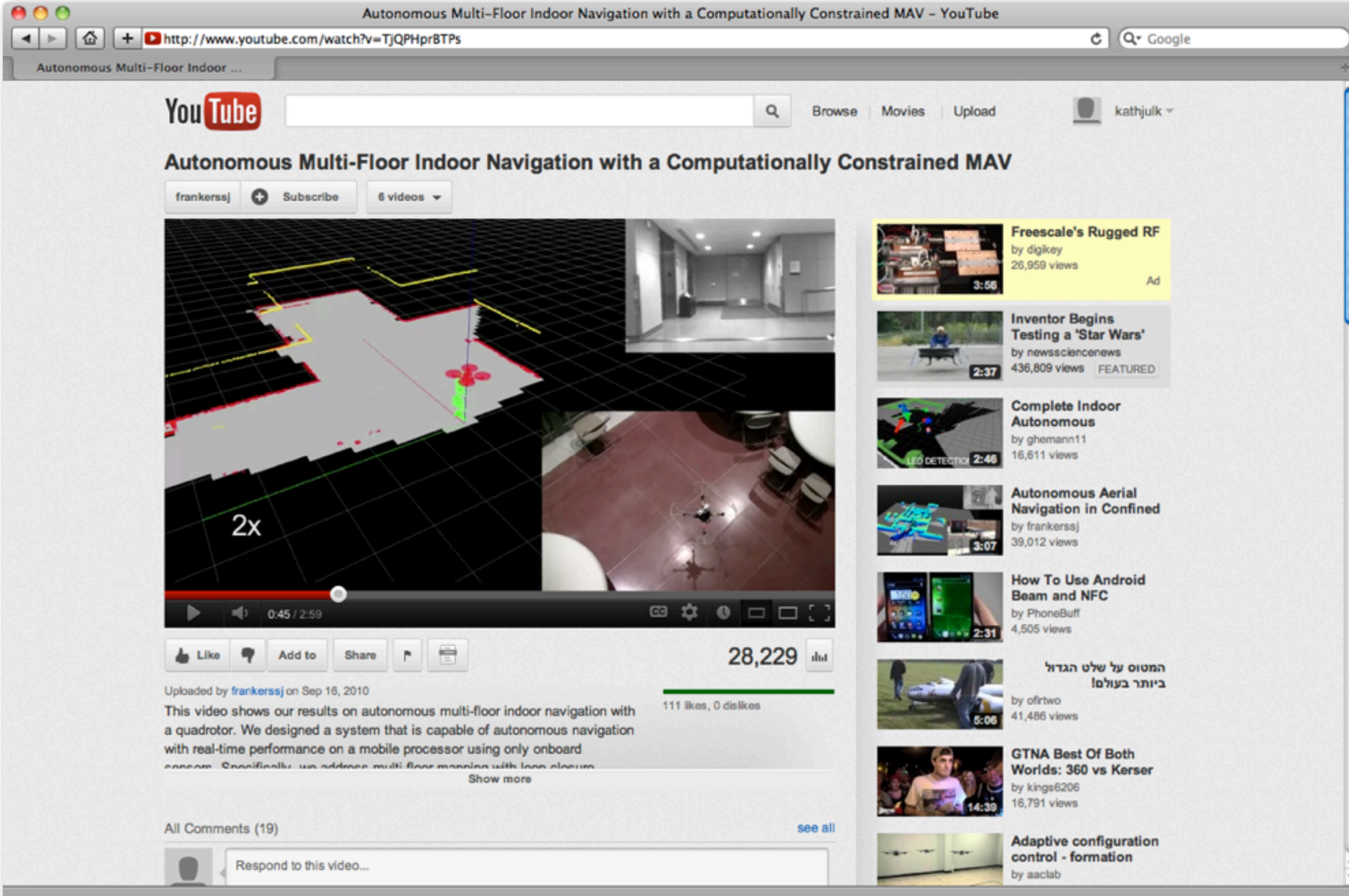
MEAM 520

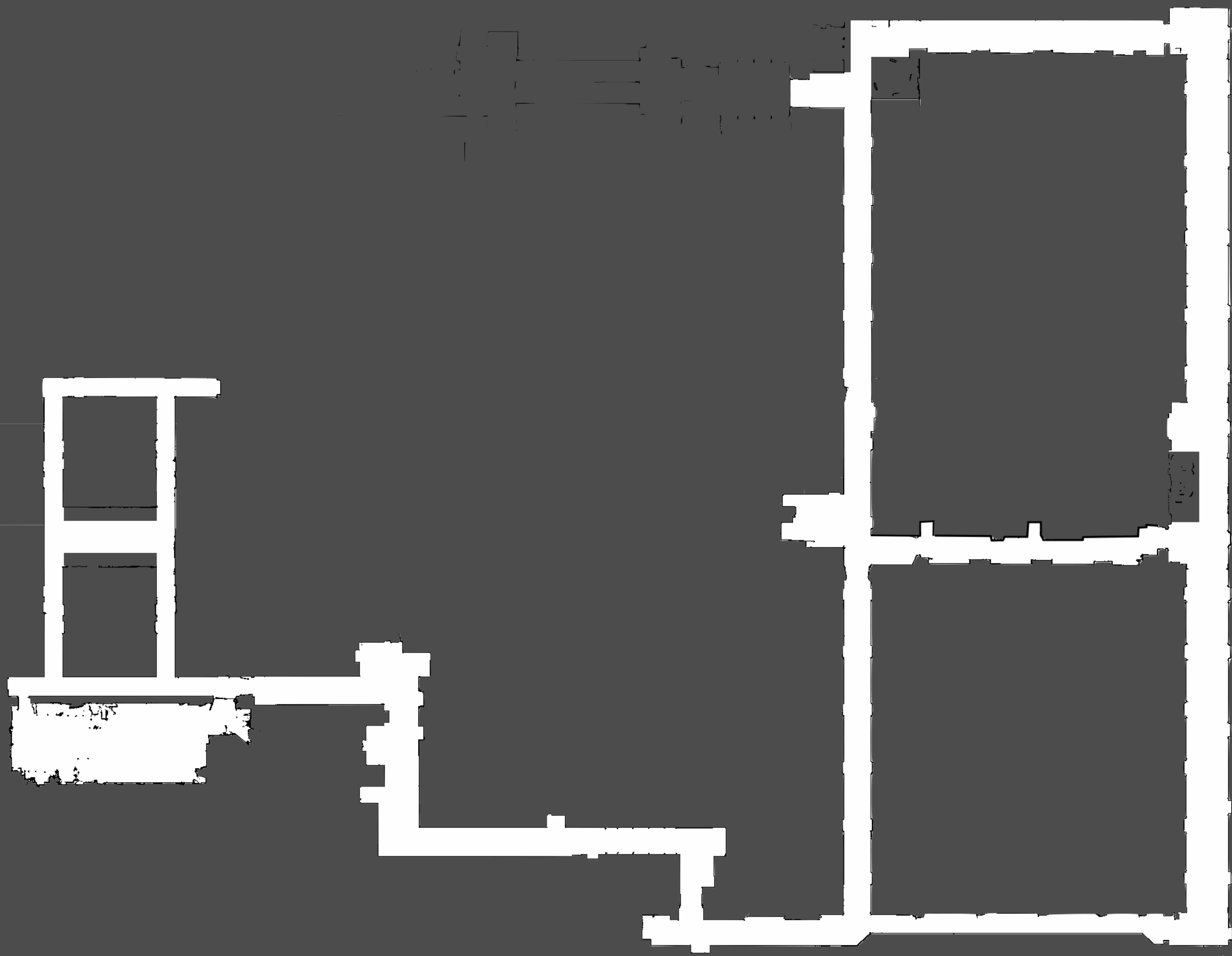
Homogenous Transformations

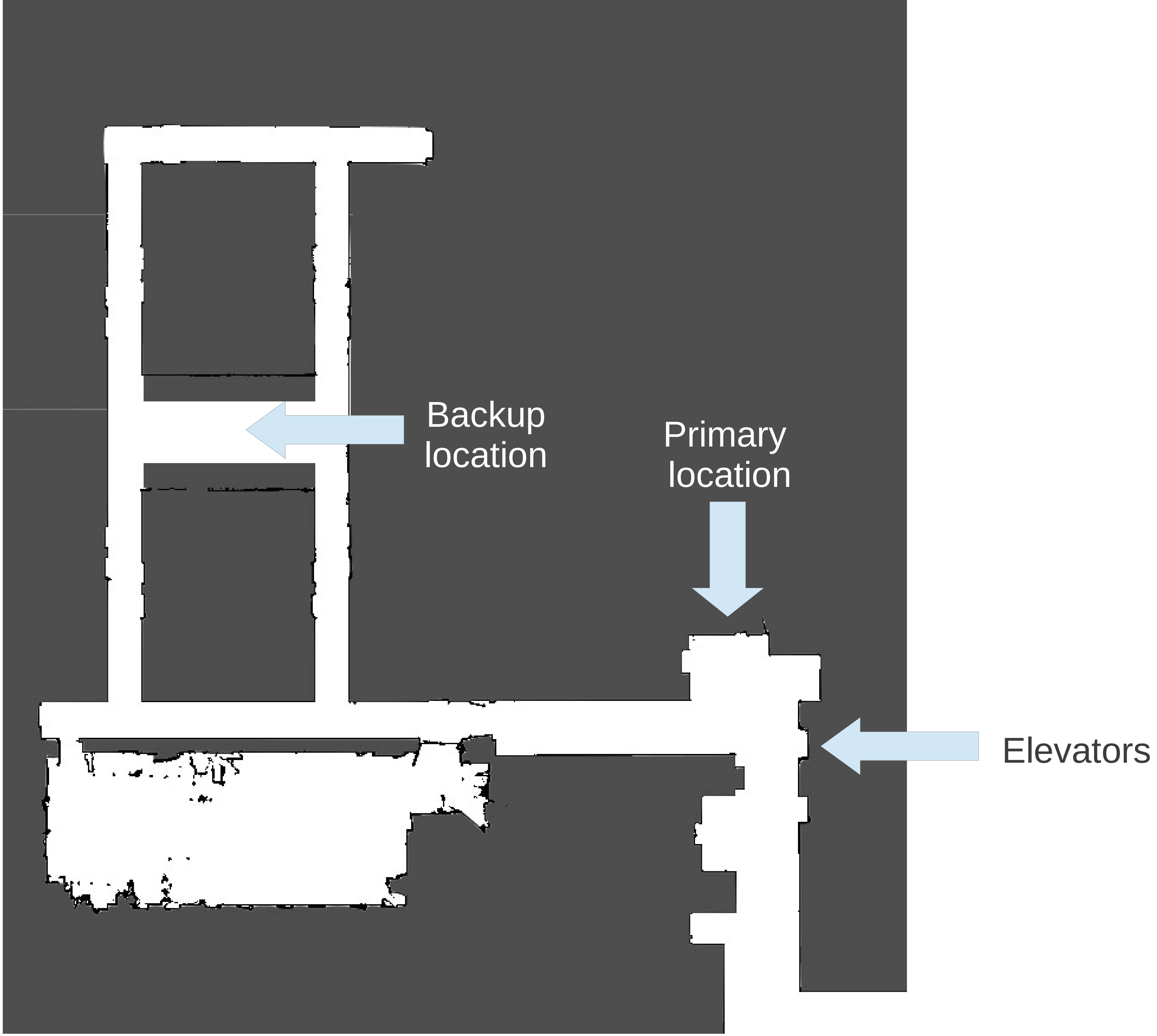
Katherine J. Kuchenbecker, Ph.D.
General Robotics, Automation, Sensing, and Perception Lab (GRASP)
MEAM Department, SEAS, University of Pennsylvania

Lecture 3: September 13, 2012



Autonomous Multi-Floor Indoor Navigation with a Computationally Constrained MAV - YouTube
http://www.youtube.com/watch?v=TjQPHprBTPs
Google
Autonomous Multi-Floor Indoor ...
YouTube
Browse
Movies
Upload
kathjulk
Autonomous Multi-Floor Indoor Navigation with a Computationally Constrained MAV
frankerssj
Subscribe
6 videos
2x
0:45 / 2:59
Like
Add to
Share
28,229
Uploaded by frankerssj on Sep 16, 2010
111 likes, 0 dislikes
This video shows our results on autonomous multi-floor indoor navigation with a quadrotor. We designed a system that is capable of autonomous navigation with real-time performance on a mobile processor using only onboard
Show more
All Comments (19)
see all
Respond to this video...
Freescale's Rugged RF
by digikey
26,959 views
Ad
3:56
Inventor Begins Testing a 'Star Wars'
by newssciencenews
436,809 views
FEATURED
2:37
Complete Indoor Autonomous
by ghemann11
16,611 views
2:46
Autonomous Aerial Navigation in Confined
by frankerssj
39,012 views
3:07
How To Use Android Beam and NFC
by PhoneBuff
4,505 views
2:31
המטוס על שלט הגדול ביותר בעולם!
by ofirtwo
41,486 views
5:06
GTNA Best Of Both Worlds: 360 vs Kerser
by kings6206
16,791 views
14:39
Adaptive configuration control - formation
by aaclab

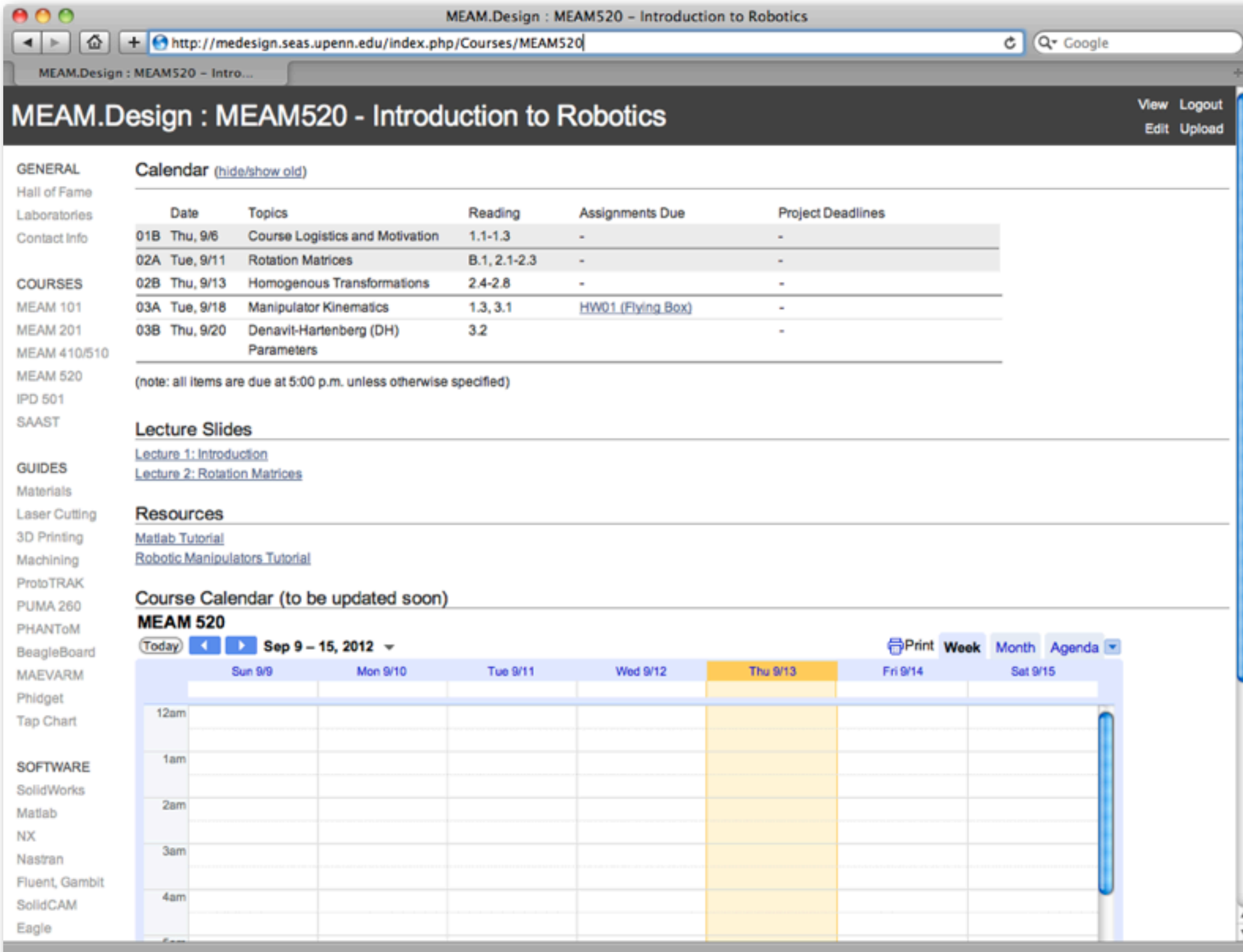



Backup location
Primary location
Elevators

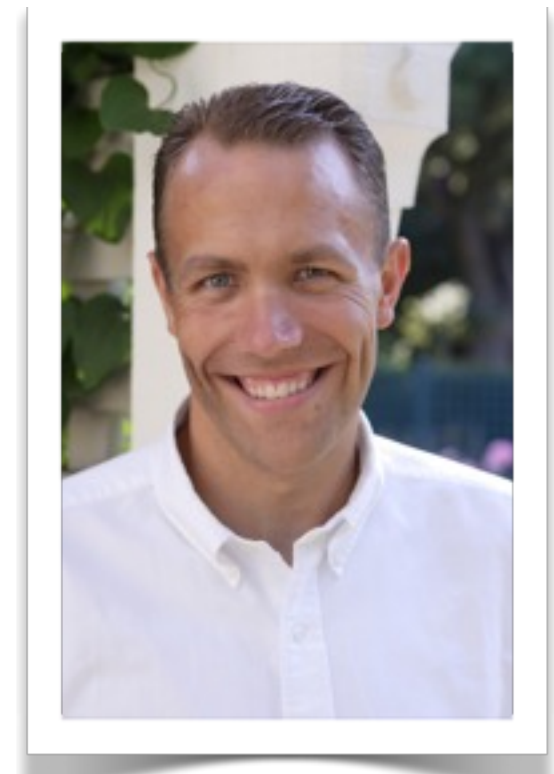
Office Hours

- Monday 1-2 pm :: Denise Wong, GRASP Conference Room
- Tuesday 1:30-2:30 pm :: Katherine Kuchenbecker, Towne 224
- Tuesday 2:30-3:30 pm :: Denise Wong, GRASP Conference Room
- Wednesday 5-6 pm :: Philip Dames, GRASP Conference Room
- Thursday 10-11 am :: Philip Dames, GRASP Conference Room
- Thursday 3-4 pm :: Katherine Kuchenbecker, Towne 224

New Course Website



Rotation Matrices

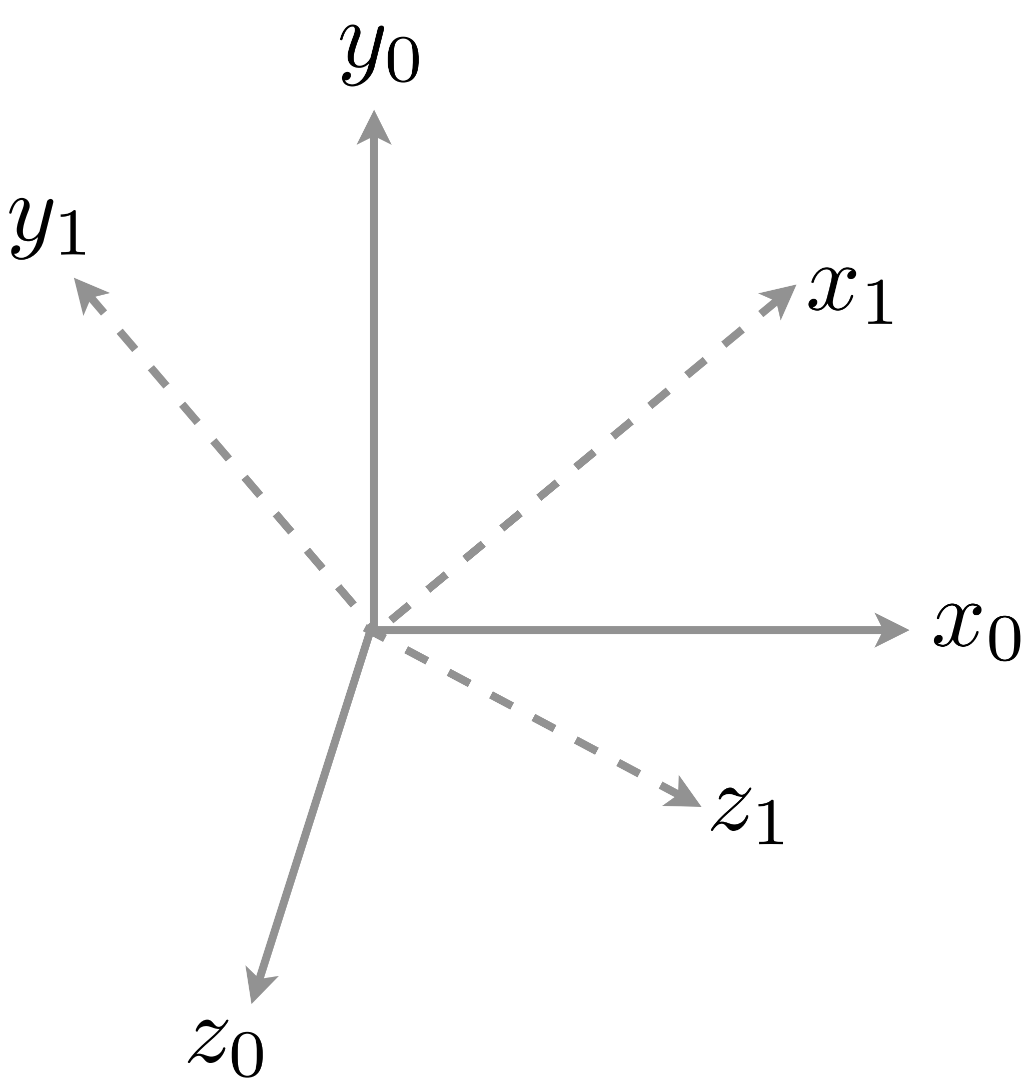


Slides created by
Jonathan Fiene

Three-Dimensional Rotation Matrix

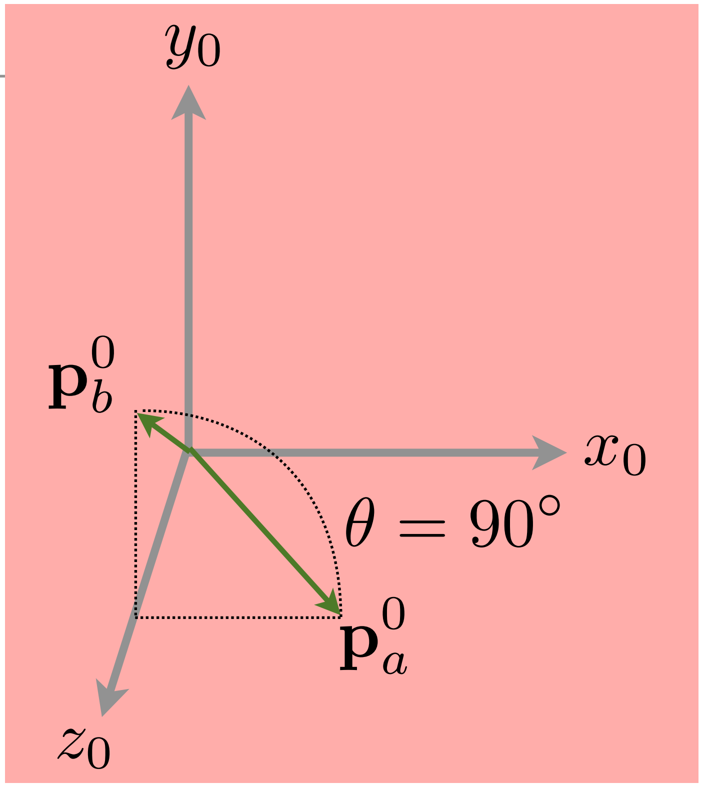
$$\mathbf{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

$$\mathbf{R}_1^0 = \begin{bmatrix} x_1 \cdot x_0 & y_1 \cdot x_0 & z_1 \cdot x_0 \\ x_1 \cdot y_0 & y_1 \cdot y_0 & z_1 \cdot y_0 \\ x_1 \cdot z_0 & y_1 \cdot z_0 & z_1 \cdot z_0 \end{bmatrix}$$



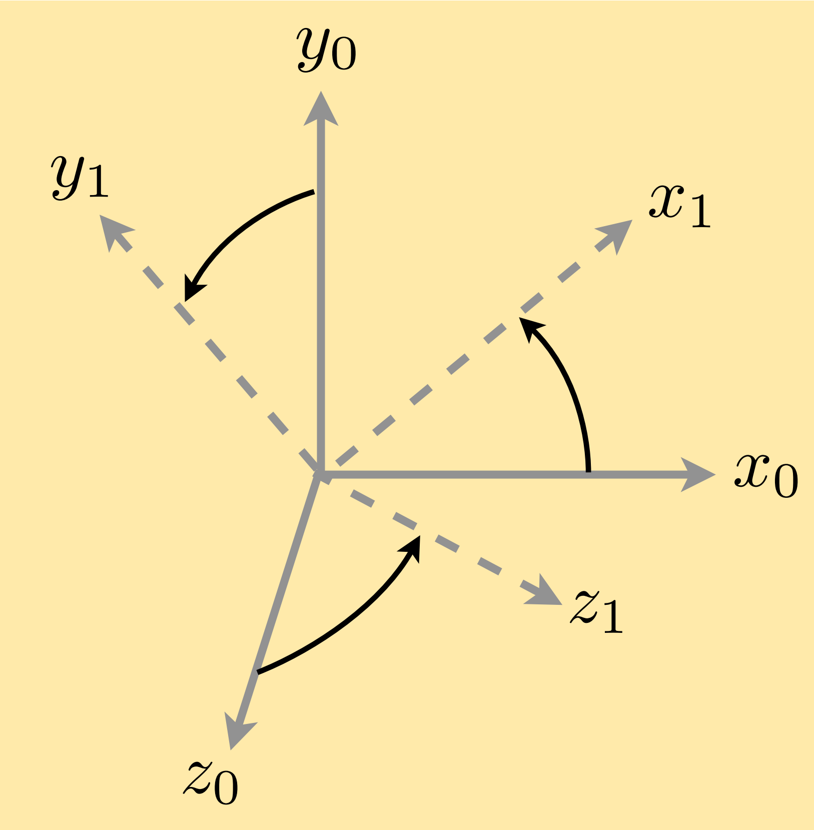
SO(3)

Special Orthogonal group of order 3



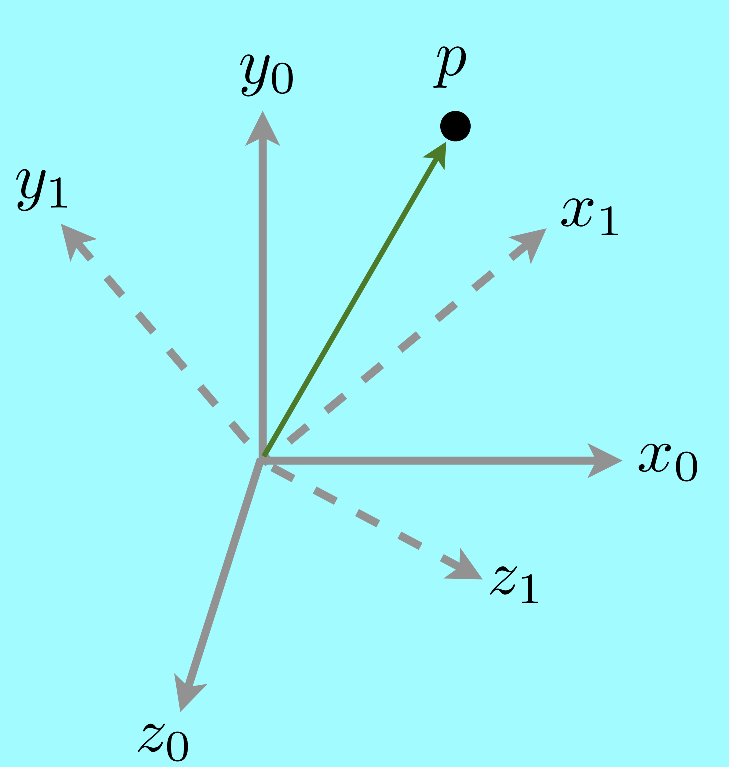
Rotation matrices serve three purposes (p. 47 in SHV):

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



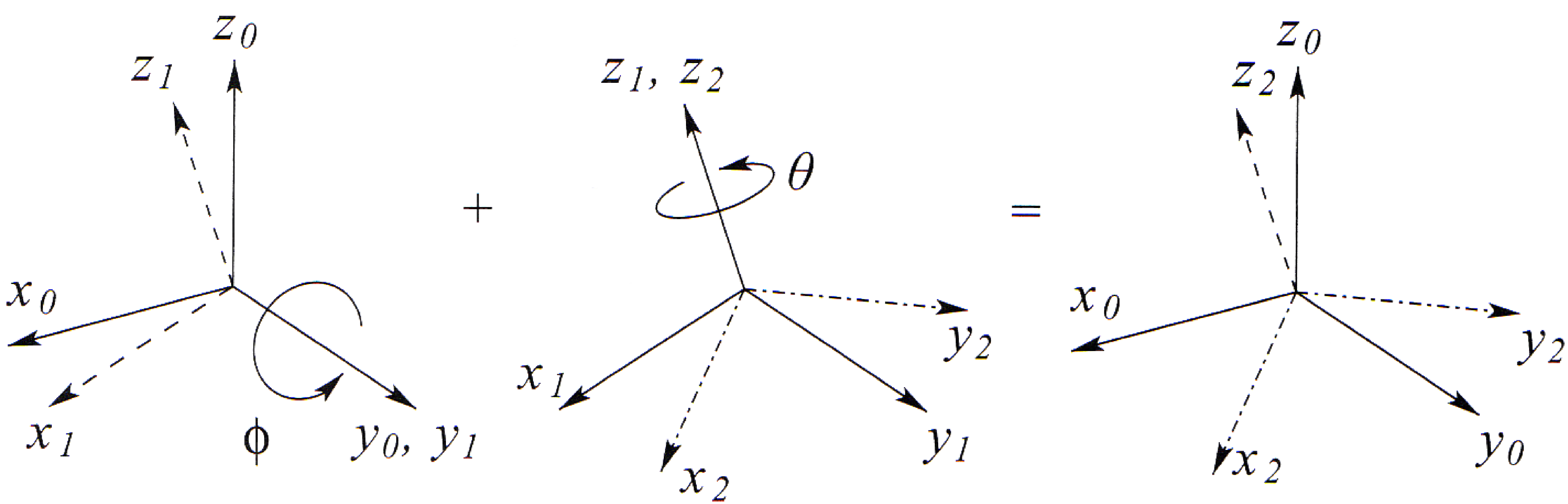
2. Orientation of a transformed coordinate frame with respect to a fixed frame

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



Composite Rotations

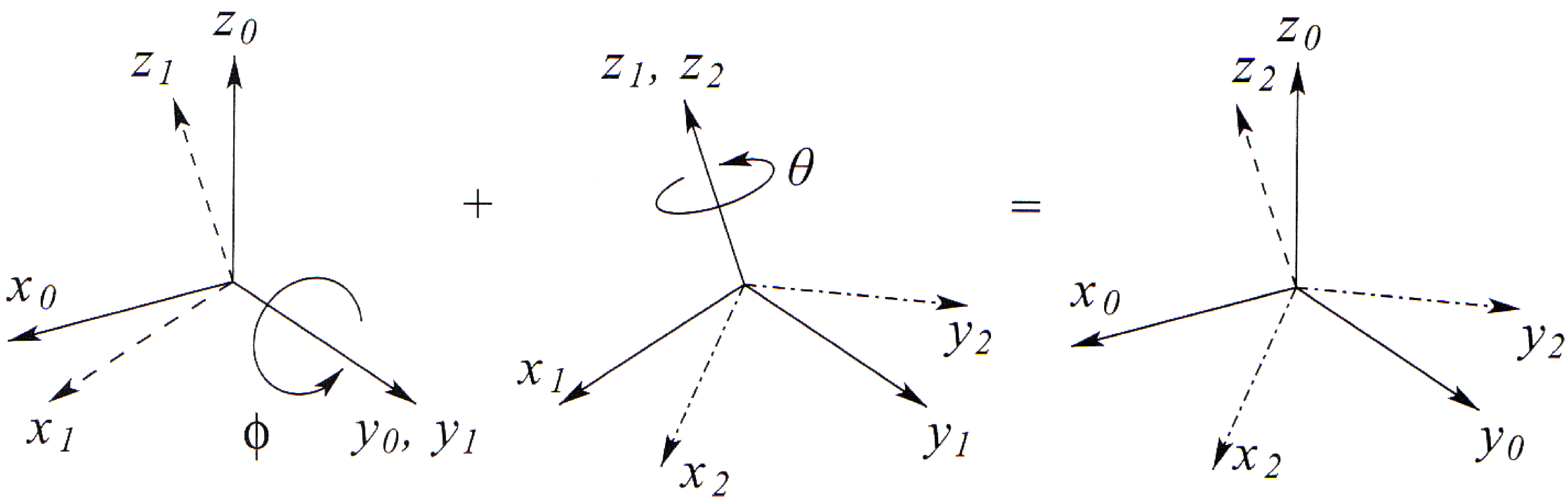
Composition of Rotations with Respect to the Current Frame



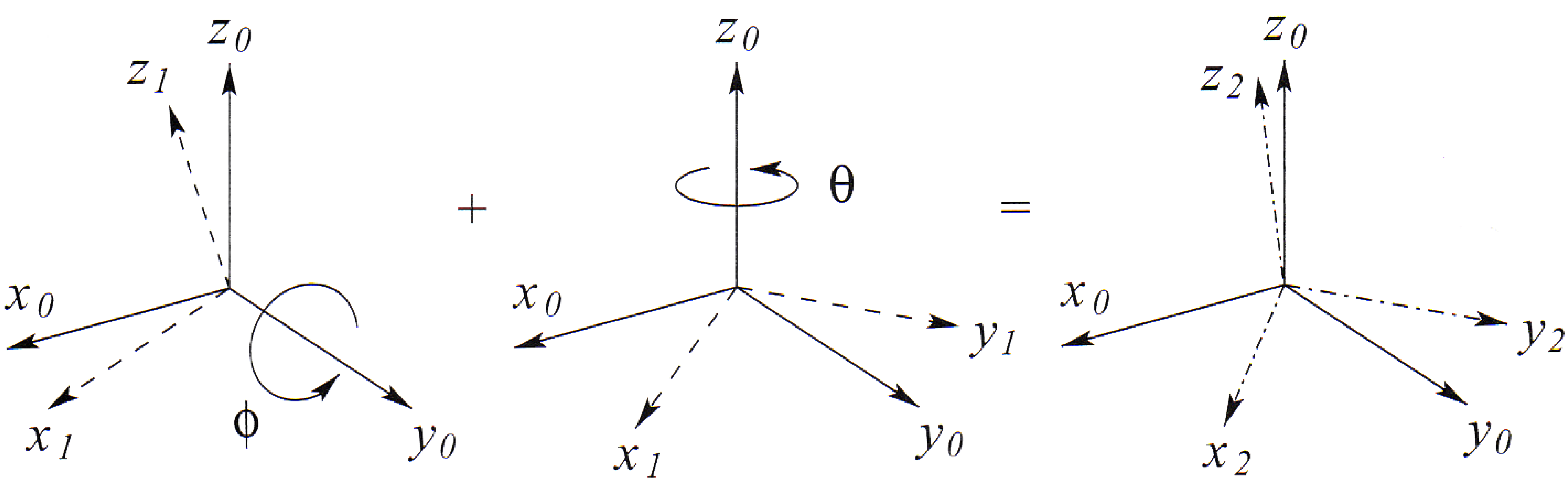
Composition of Rotations with Respect to the Current Frame

the result of a successive rotation about the current (intermediate) frame can be found by **post-multiplying** by the corresponding rotation matrix

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

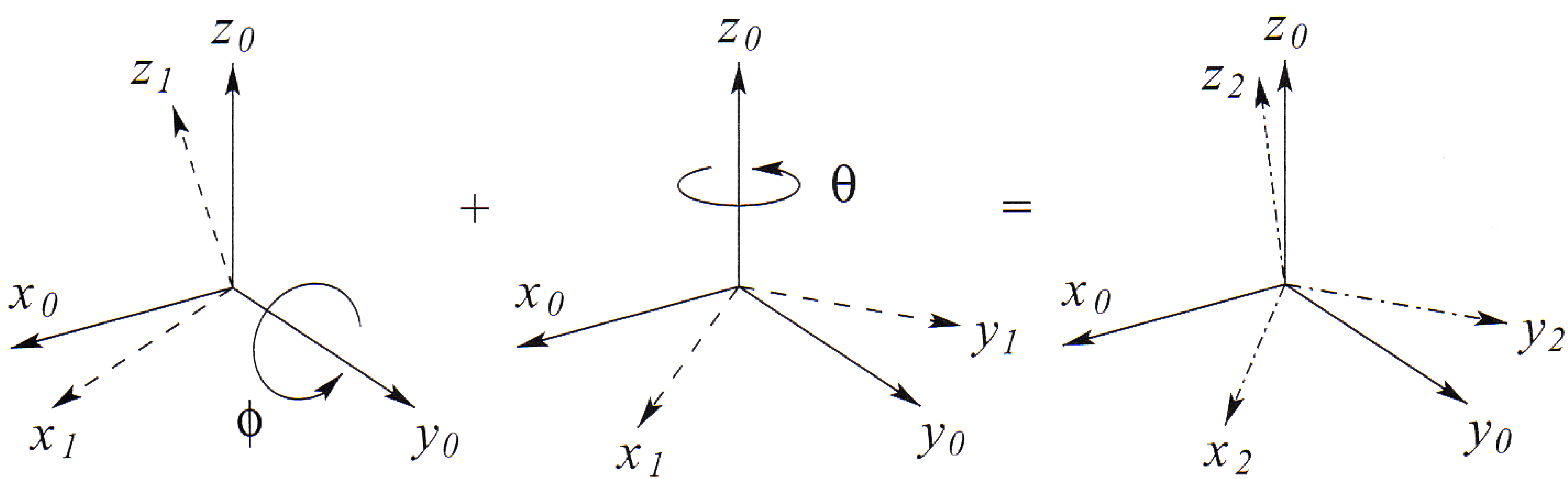


Composition of Rotations with Respect to a Fixed Frame



the result of a successive rotation about a fixed frame can be found by **pre-multiplying** by the corresponding rotation matrix

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



Note that $\mathbf{R}$ is a rotation about the original frame

You should read the book and think about this.

Parameterizing Rotations

Parameterization of Rotations

$$\mathbf{R}_1^0 = \begin{bmatrix} x_1 \cdot x_0 & y_1 \cdot x_0 & z_1 \cdot x_0 \\ x_1 \cdot y_0 & y_1 \cdot y_0 & z_1 \cdot y_0 \\ x_1 \cdot z_0 & y_1 \cdot z_0 & z_1 \cdot z_0 \end{bmatrix}$$

in three dimensions, no more than 3 values are needed to specify an arbitrary rotation

which means that the 9-element rotation matrix has at least 6 redundancies

---

numerous methods have been developed to represent rotation/orientation with less redundancy

Euler Angles

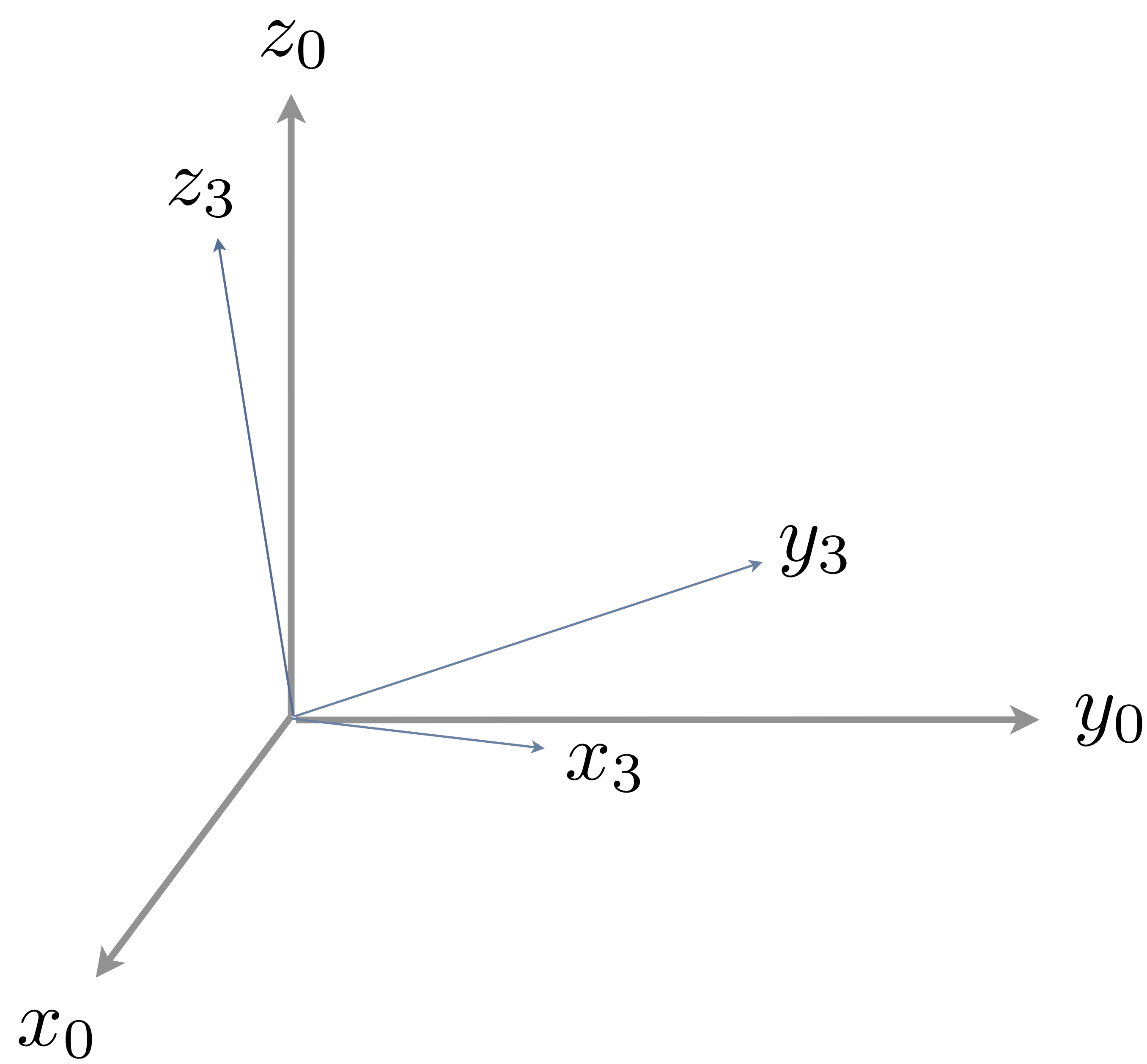
Roll, Pitch, Yaw Angles

Axis/Angle Representation

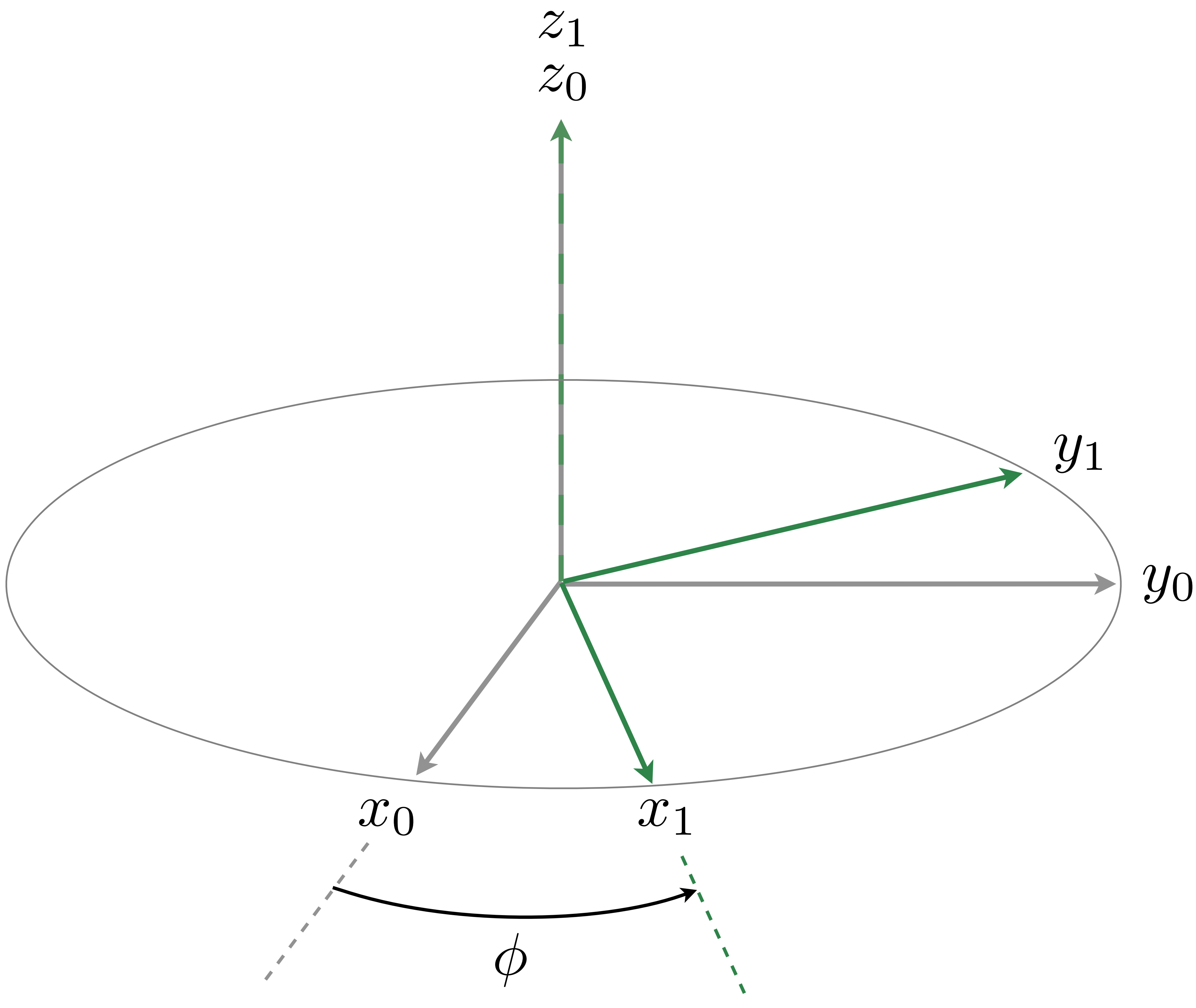
Conventions vary, so always check definitions!

Euler Angles

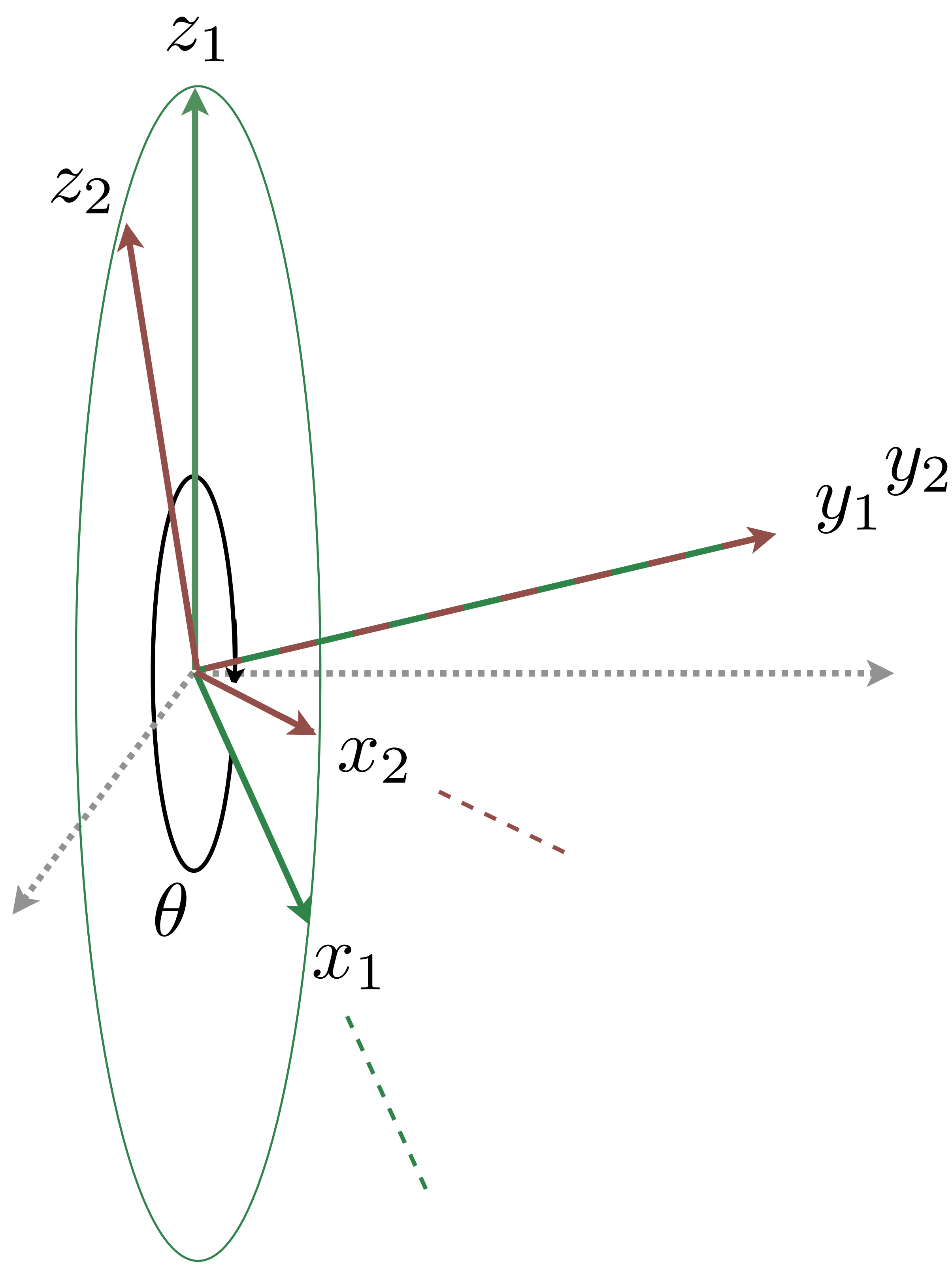
Define a set of three **intermediate** angles, ϕ, θ, ψ , to go from $0 \longrightarrow 3$



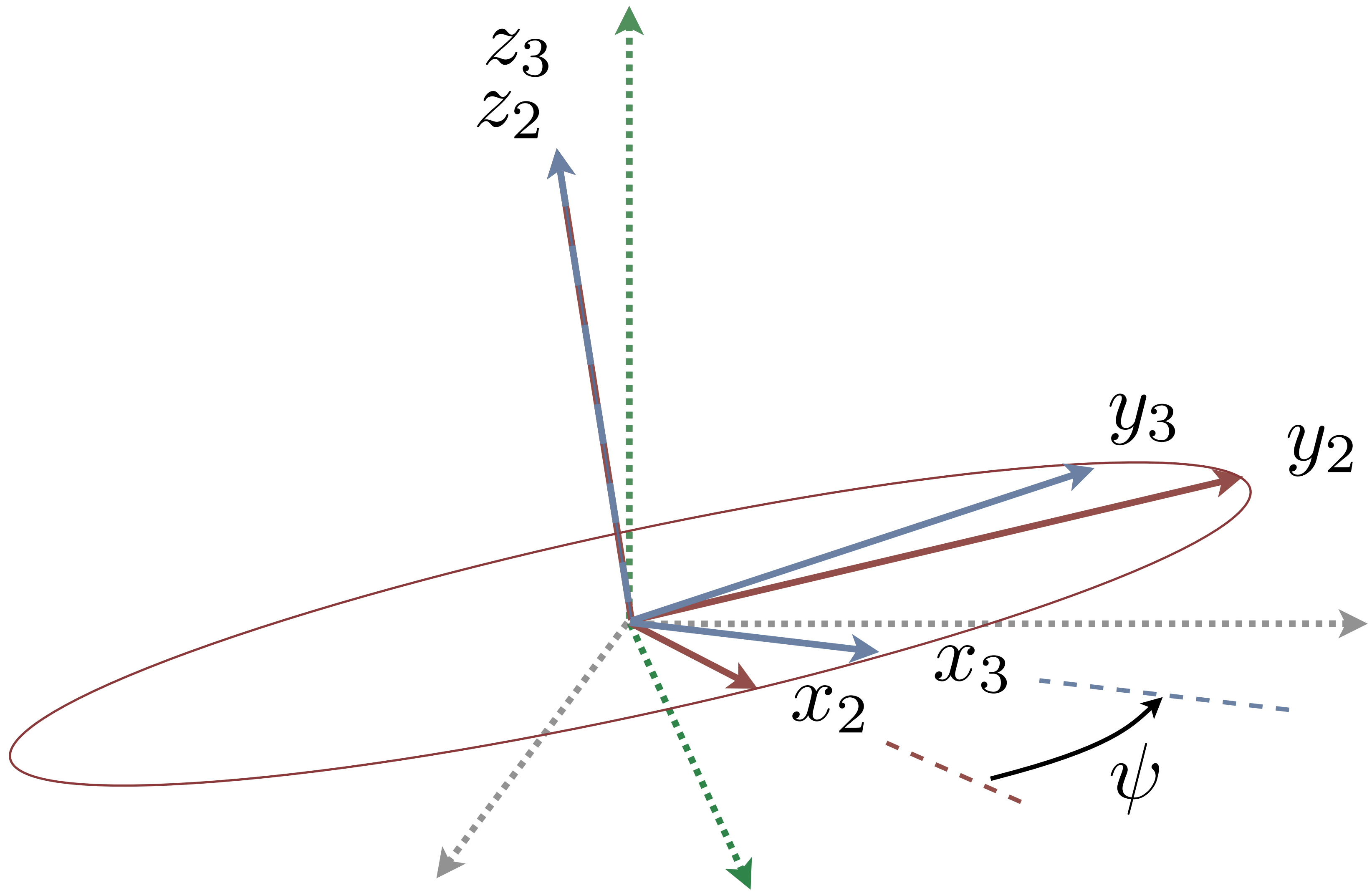
step 1: rotate by ϕ about z_0



step 2: rotate by θ about y_1



step 3: rotate by ψ about z_2



Euler Angles to Rotation Matrices

(**post**-multiply using the **basic rotation matrices**)

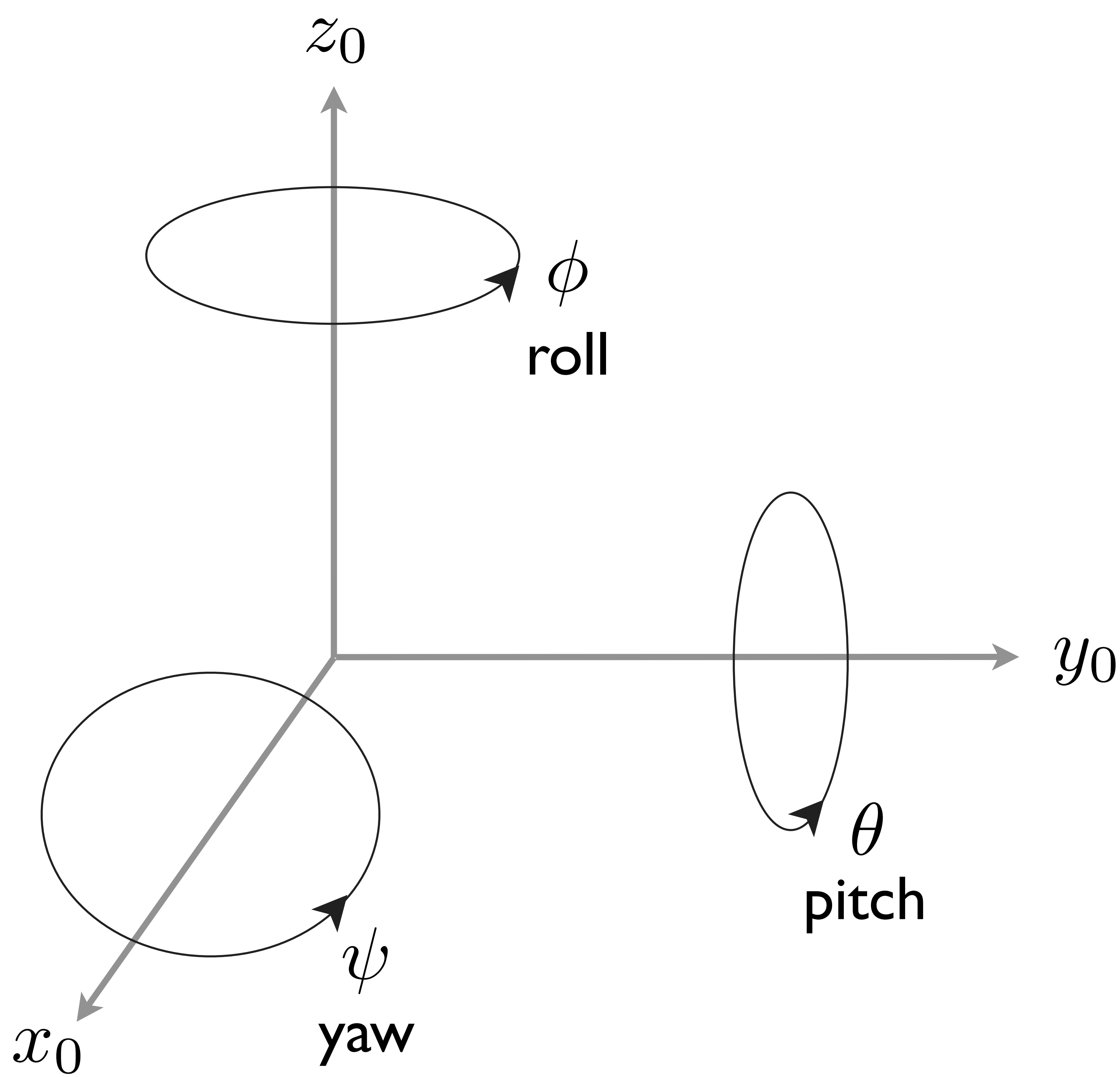
$$\mathbf{R} = \mathbf{R}_{z,\phi}\, \mathbf{R}_{y,\theta}\, \mathbf{R}_{z,\psi}$$

$$= \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} c_\phi c_\theta c_\psi - s_\phi s_\psi & -c_\phi c_\theta s_\psi - s_\phi c_\psi & c_\phi s_\theta \\ s_\phi c_\theta c_\psi + c_\phi s_\psi & -s_\phi c_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta \\ -s_\theta c_\psi & s_\theta s_\psi & c_\theta \end{bmatrix}$$

Roll, Pitch, Yaw Angles

defined as a set of three angles about a **fixed** reference



Roll, Pitch, Yaw Angles to Rotation Matrices

(**pre**-multiply using the **basic rotation matrices**)

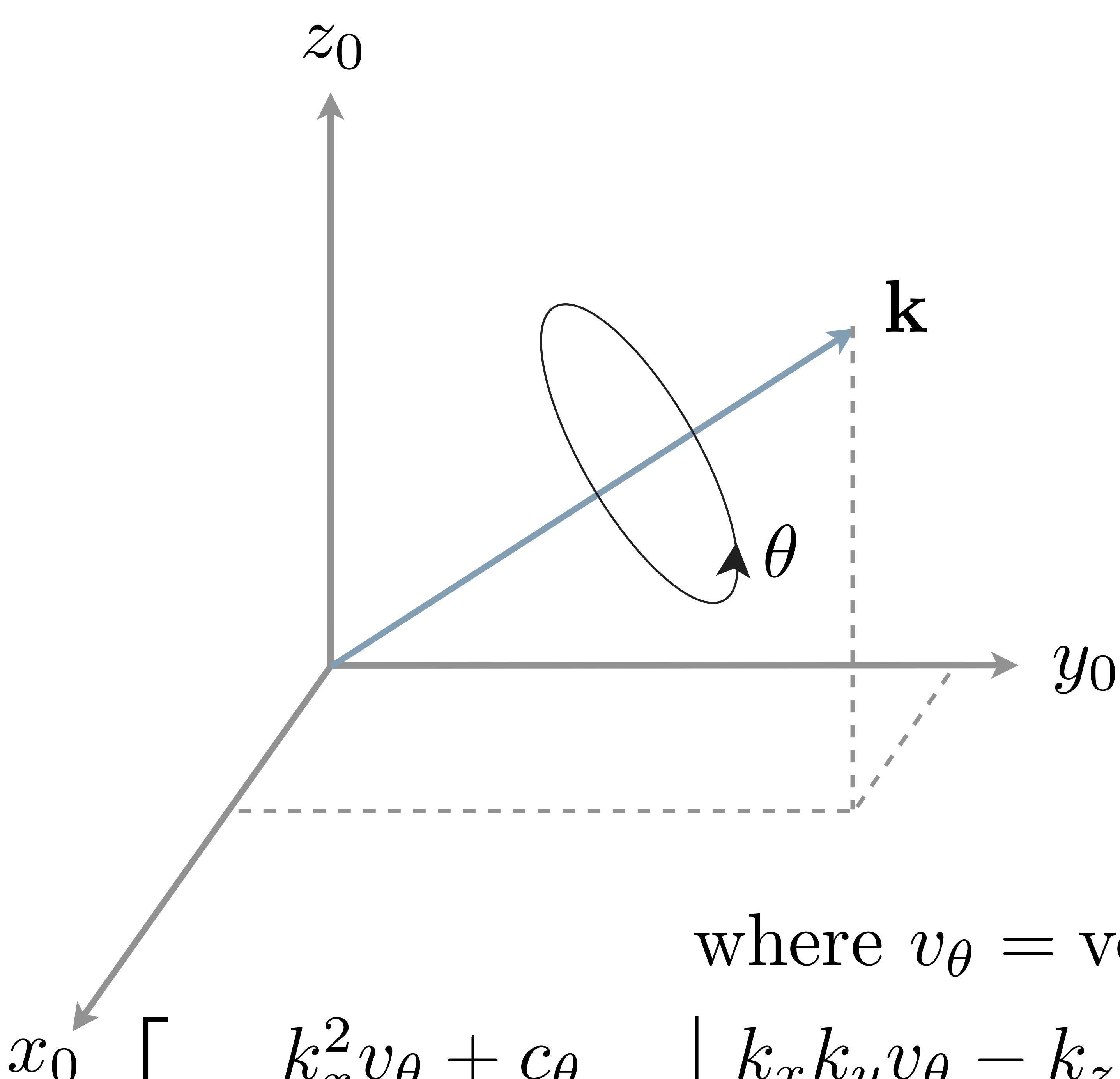
$$\mathbf{R} = \mathbf{R}_{z,\phi}\,\mathbf{R}_{y,\theta}\,\mathbf{R}_{x,\psi}$$

$$= \begin{bmatrix} c_\phi & -s_\phi & 0 \\ s_\phi & c_\phi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_\theta & 0 & s_\theta \\ 0 & 1 & 0 \\ -s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\psi & -s_\psi \\ 0 & s_\psi & c_\psi \end{bmatrix}$$

$$= \begin{bmatrix} c_\phi c_\theta & c_\phi s_\theta s_\psi - s_\phi c_\psi & s_\phi s_\psi + c_\phi s_\theta c_\psi \\ s_\phi c_\theta & s_\phi s_\theta s_\psi + c_\phi c_\psi & s_\phi s_\theta c_\psi - c_\phi s_\psi \\ -s_\theta & c_\theta s_\psi & c_\theta c_\psi \end{bmatrix}$$

Axis/Angle Representation

rotation by an angle about an axis in space



$$\mathbf{k} = \begin{bmatrix} k_x \\ k_y \\ k_z \end{bmatrix}$$

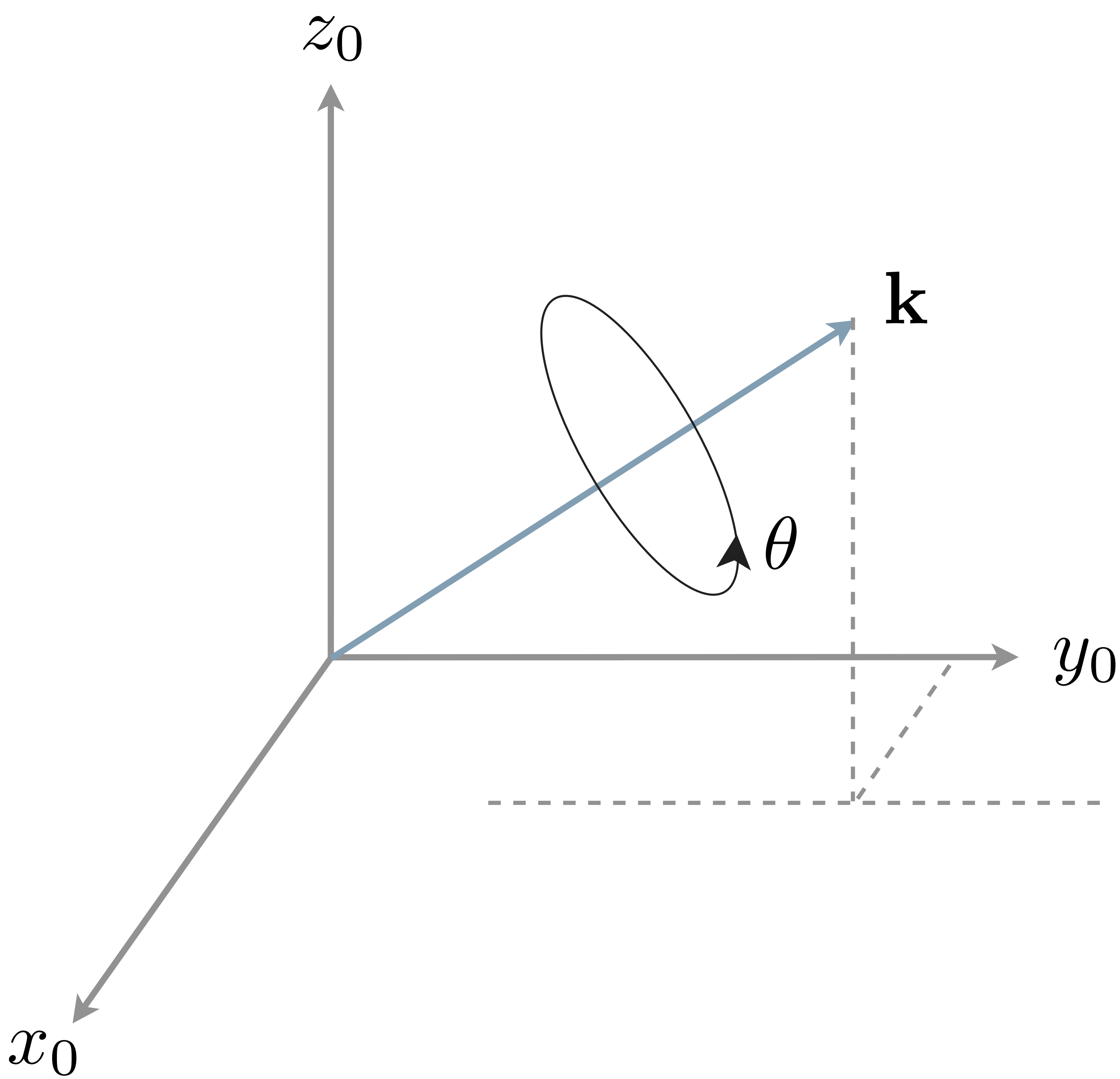
$$k_x^2 + k_y^2 + k_z^2 = 1$$

$$\text{where } v_\theta = \operatorname{vers}\theta = 1 - \cos\theta$$

$$R_{k,\theta} = \left[\begin{array}{c|c|c} k_x^2 v_\theta + c_\theta & k_x k_y v_\theta - k_z s_\theta & k_x k_z v_\theta + k_y s_\theta \\ k_x k_y v_\theta + k_z s_\theta & k_y^2 v_\theta + c_\theta & k_y k_z v_\theta - k_x s_\theta \\ k_x k_z v_\theta - k_y s_\theta & k_y k_z v_\theta + k_x s_\theta & k_z^2 v_\theta + c_\theta \end{array}\right]$$

Axis/Angle Representation

any rotation matrix can be represented this way!

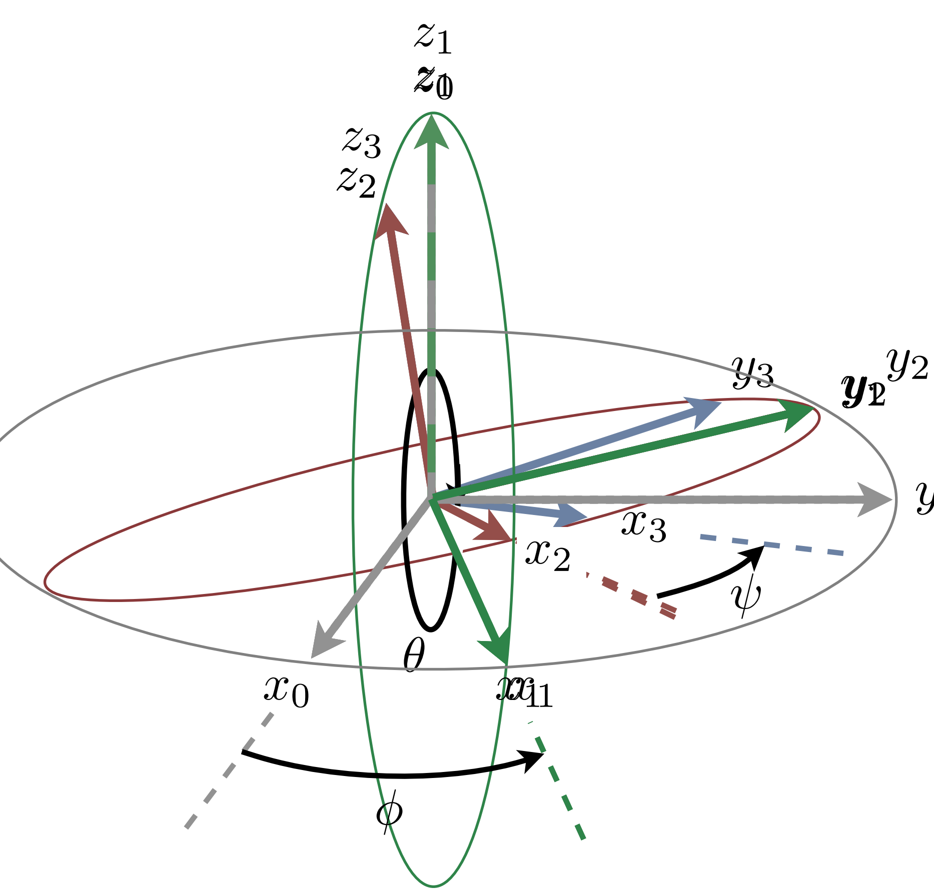


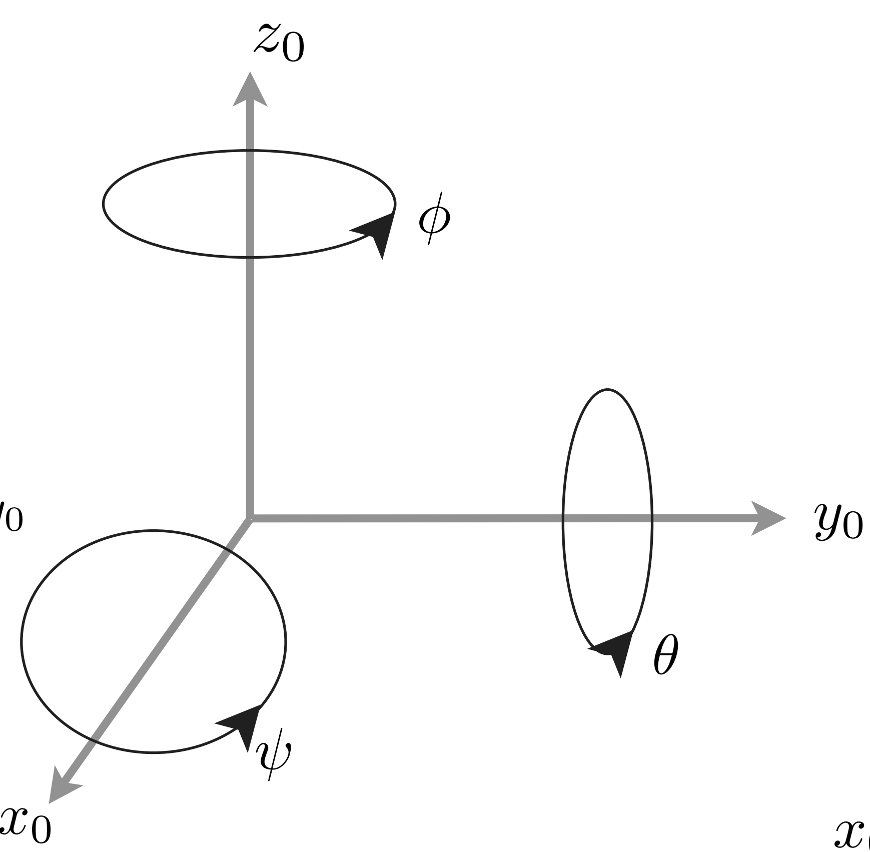
$$R = R_{k,\theta}$$

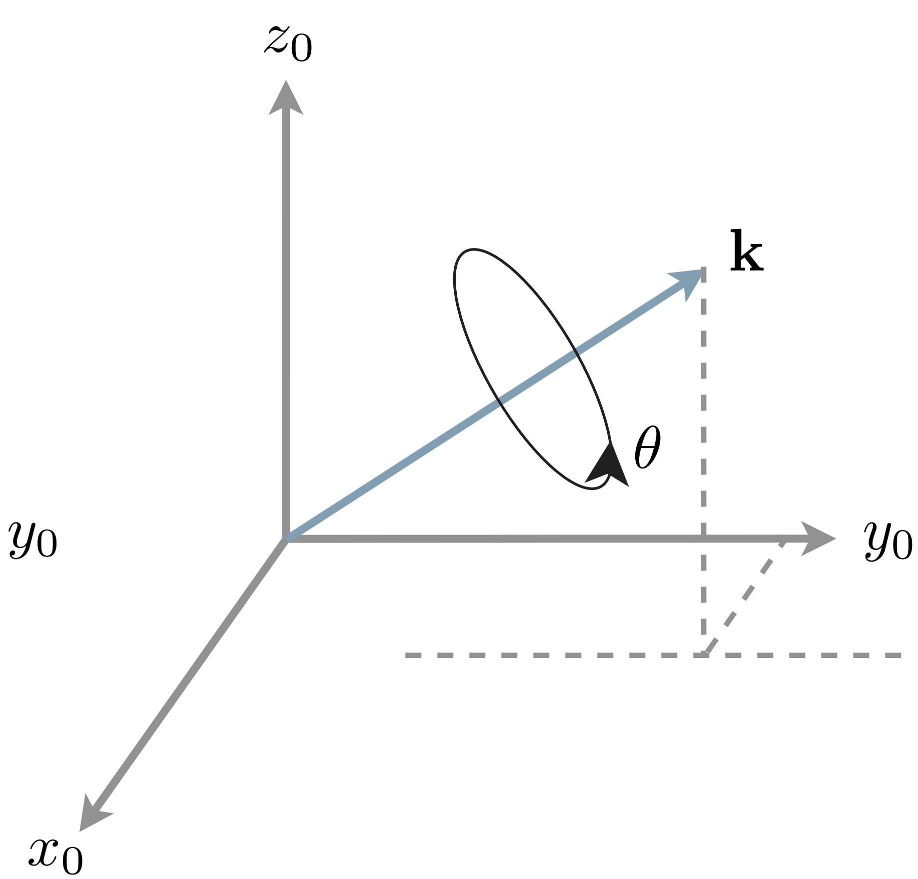
$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}$$

$$\theta = \cos^{-1}\left(\frac{r_{11} + r_{22} + r_{33} - 1}{2}\right) \qquad k = \frac{1}{2\sin\theta}\begin{bmatrix} r_{32} - r_{23} \\ r_{13} - r_{31} \\ r_{21} - r_{12} \end{bmatrix}$$

Talk to the person next to you.
Explain one of the three parameterization approaches to your partner, then switch.
Talk about the third one together.

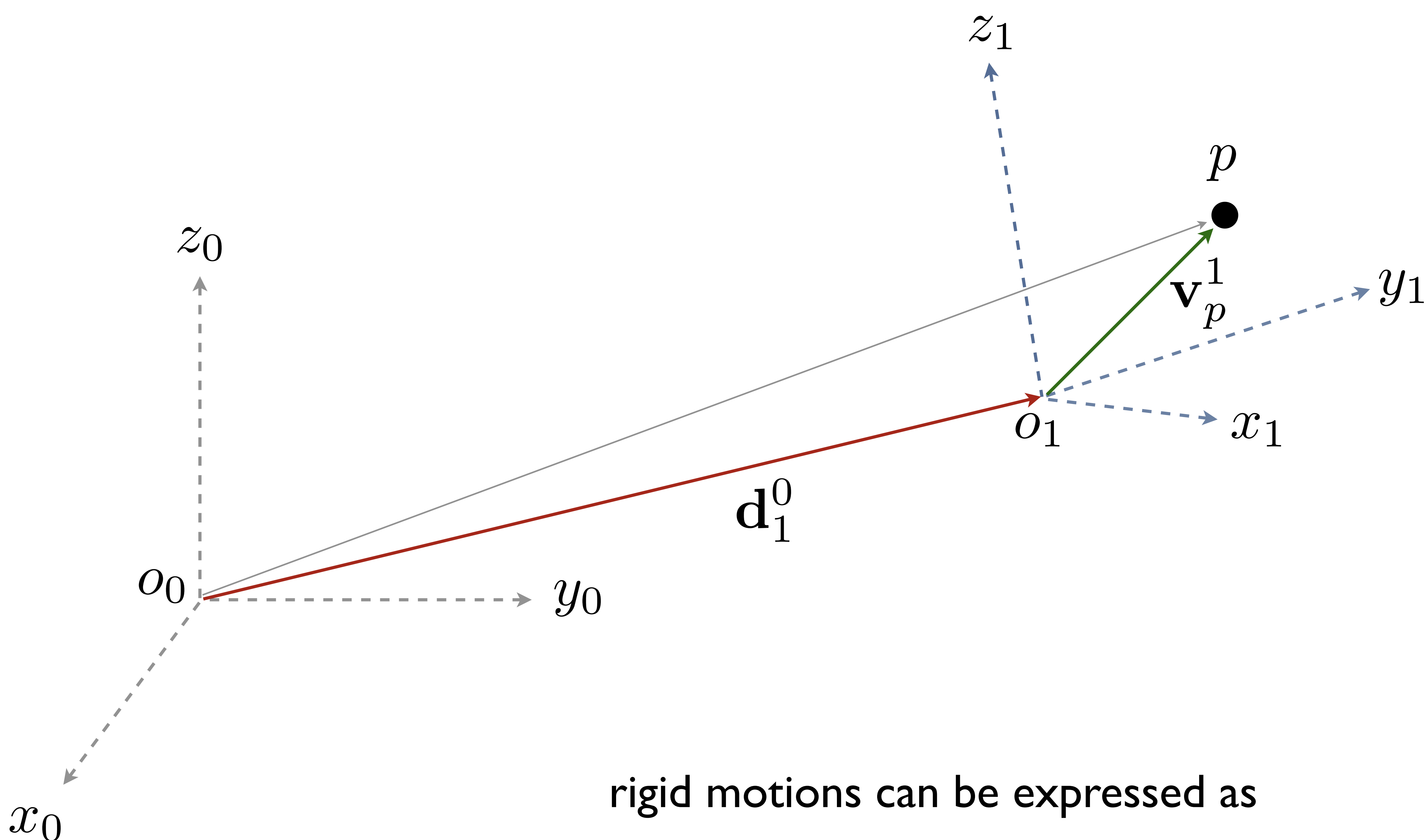






Homogeneous Transformations

a **rigid motion** couples pure translation with pure rotation



rigid motions can be expressed as

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

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

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

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

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

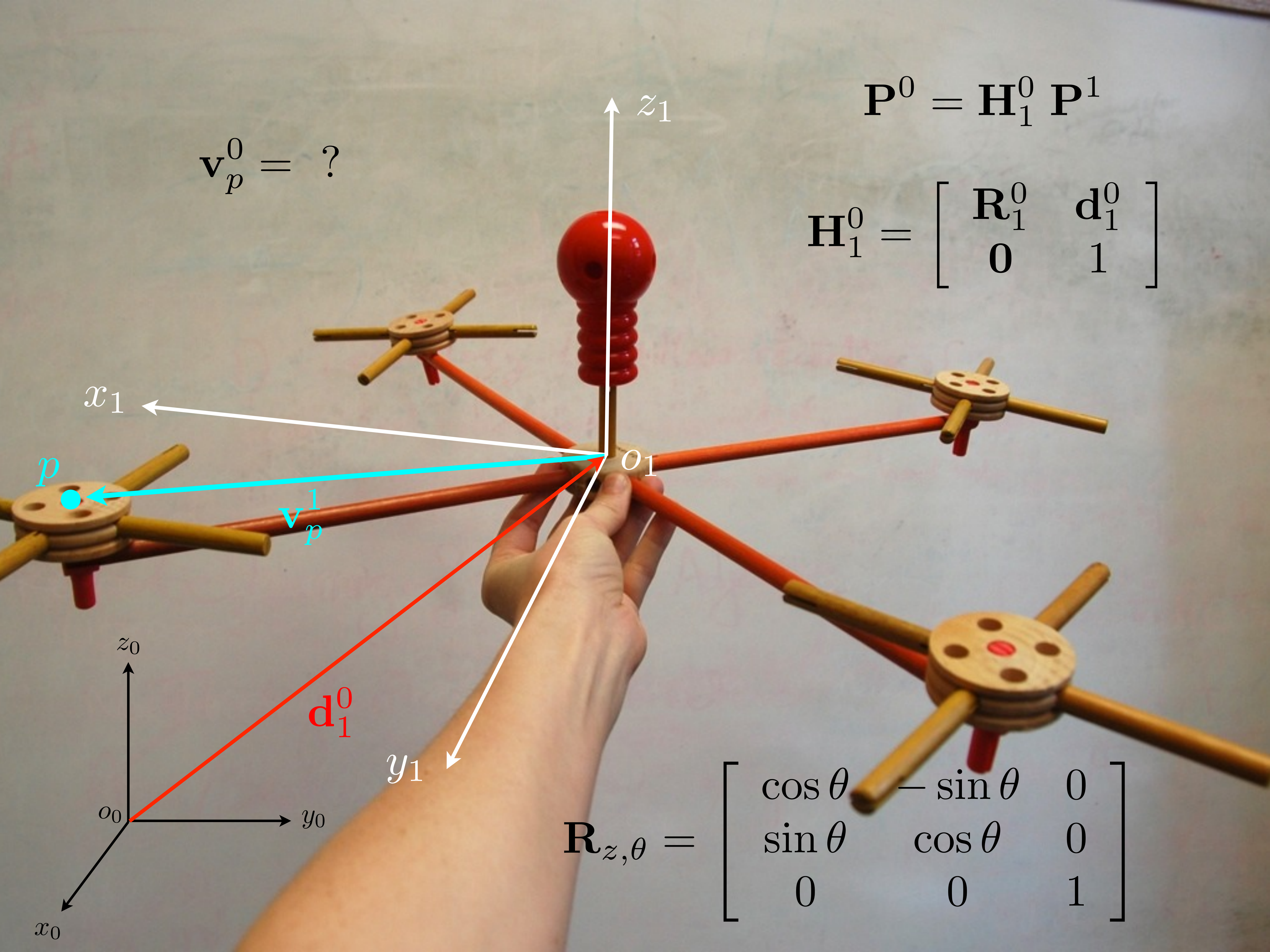
where $\mathbf{p}$ is the 3x1 vector

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

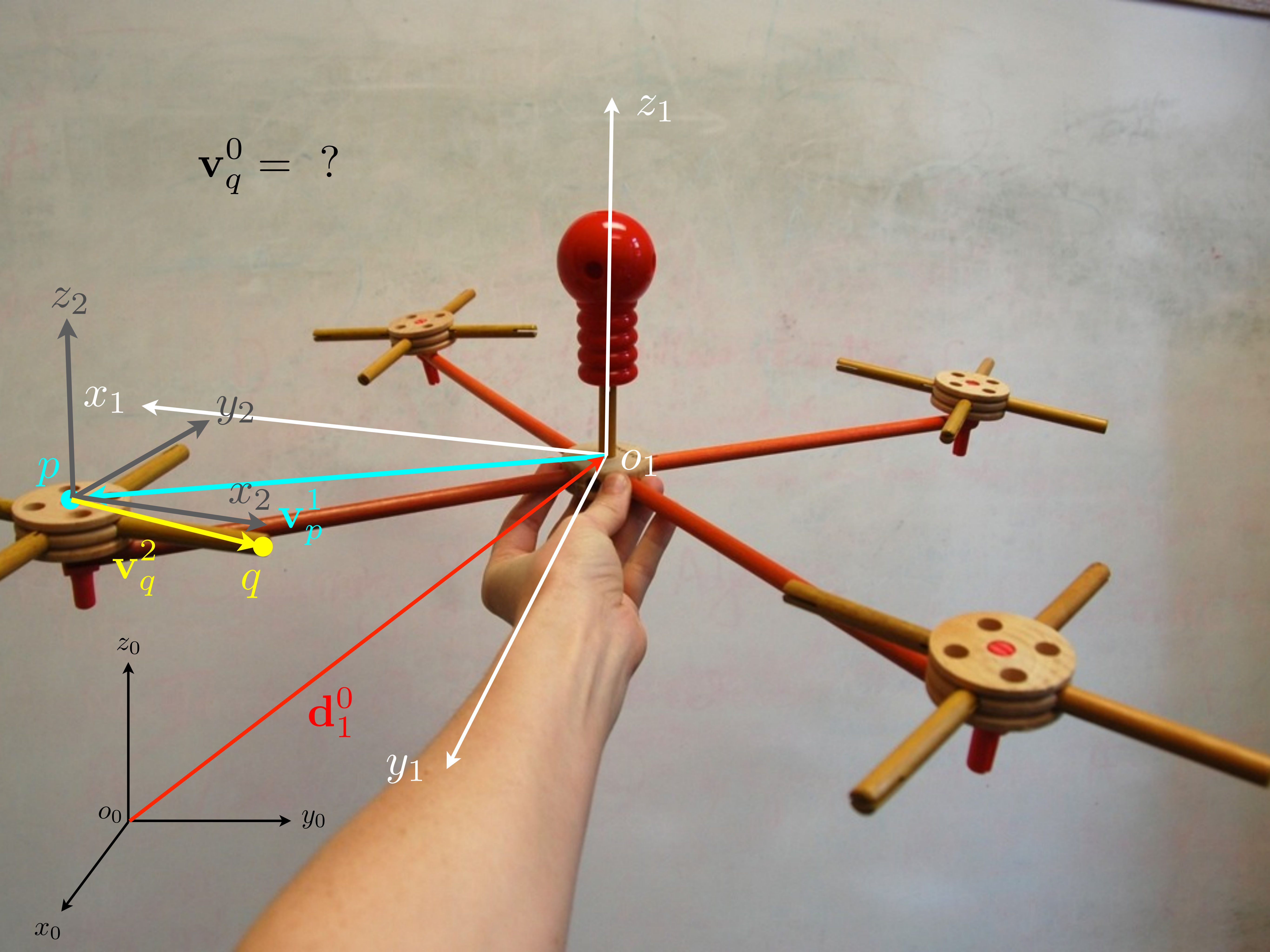
rigid body transformations are accomplished by pre-multiplying by the homogenous transform

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

{example with quadrotor model}



$\mathbf{P}^0 = \mathbf{H}_1^0 \, \mathbf{P}^1$
$\mathbf{v}_p^0 = \ ?$
$\mathbf{H}_1^0 = \begin{bmatrix} \mathbf{R}_1^0 & \mathbf{d}_1^0 \\ \mathbf{0} & 1 \end{bmatrix}$
z_1
x_1
o_1
p
$\mathbf{v}_p^1$
y_1
$\mathbf{d}_1^0$
z_0
o_0
y_0
x_0
$\mathbf{R}_{z,\theta} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$



$\mathbf{v}_q^0 = \ ?$
z_1
z_2
x_1
y_2
p
o_1
x_2
$\mathbf{v}_p^1$
$\mathbf{v}_q^2$
q
z_0
$\mathbf{d}_1^0$
y_1
o_0
y_0
x_0

composition of multiple transforms is the same as for rotation matrices:

post-multiply when successive rotations are relative to intermediate frames

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

pre-multiply when successive rotations are relative to the first fixed frame

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

Composition (intermediate frame)

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

Inverse Transform

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

Homework 1

Homework 1:
Rigid Motions and Homogeneous Transformations

MEAM 520, University of Pennsylvania
Katherine J. Kuchenbecker, Ph.D.

September 11, 2012

This assignment is due on Tuesday, September 18, by 5:00 p.m. sharp. You should aim to turn the paper part in during class that day. If you don't finish until later in the day, you can turn it in to Professor Kuchenbecker's office, Towne 224. The code must be emailed according to the instructions at the end of this document. Late submissions of either or both parts will be accepted until 5:00 p.m. on Wednesday, but they will be penalized by 25%. After that deadline, no further assignments may be submitted.

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 write down should be your own work, not copied from a peer or a solution manual.

Book Problems *(30 points)*

The first set of problems is from the textbook, *Robot Modeling and Control* by Spong, Hutchinson, and Vidyasagar (SHV). Please follow the extra clarifications and instructions when provided. Write in pencil, show your work clearly, $\boxed{\text{box your answers}}$, and staple together all pages of your assignment.

1. SHV 2-10, page 66 – Sequence of Rotations *(5 points)*
   Please specify each element of each matrix in symbolic form and show the order in which the matrices should be multiplied; as stated in the problem, you do not need to perform the matrix multiplication.
2. SHV 2-14, page 67 – Rotating a Coordinate Frame *(5 points)*
   Sketch the initial, intermediate, and final frames by reading the text in the problem. Then find R in two ways: by inspection of your sketch and by calculation. Check your solutions against one another.
3. SHV 2-23, page 68 – Axis/Angle Representation *(10 points)*
4. SHV 2-39, page 70 – Homogeneous Transformations *(10 points)*
   Treat frame $o_2x_2y_2z_2$ as being located at the center of the cube's bottom surface (as drawn in Figure 2.14), not at the center of the cube (as stated in the problem).

MATLAB Programming *(30 points)*

This class will use MATLAB to analyze and simulate robotic systems and also to control real robots. While Professor Kuchenbecker loves MATLAB, she recognizes that it can be difficult to use at the start. Even if you don't like MATLAB now, please give it a chance, and come to office hours or contact the teaching team if you feel lost or frustrated.

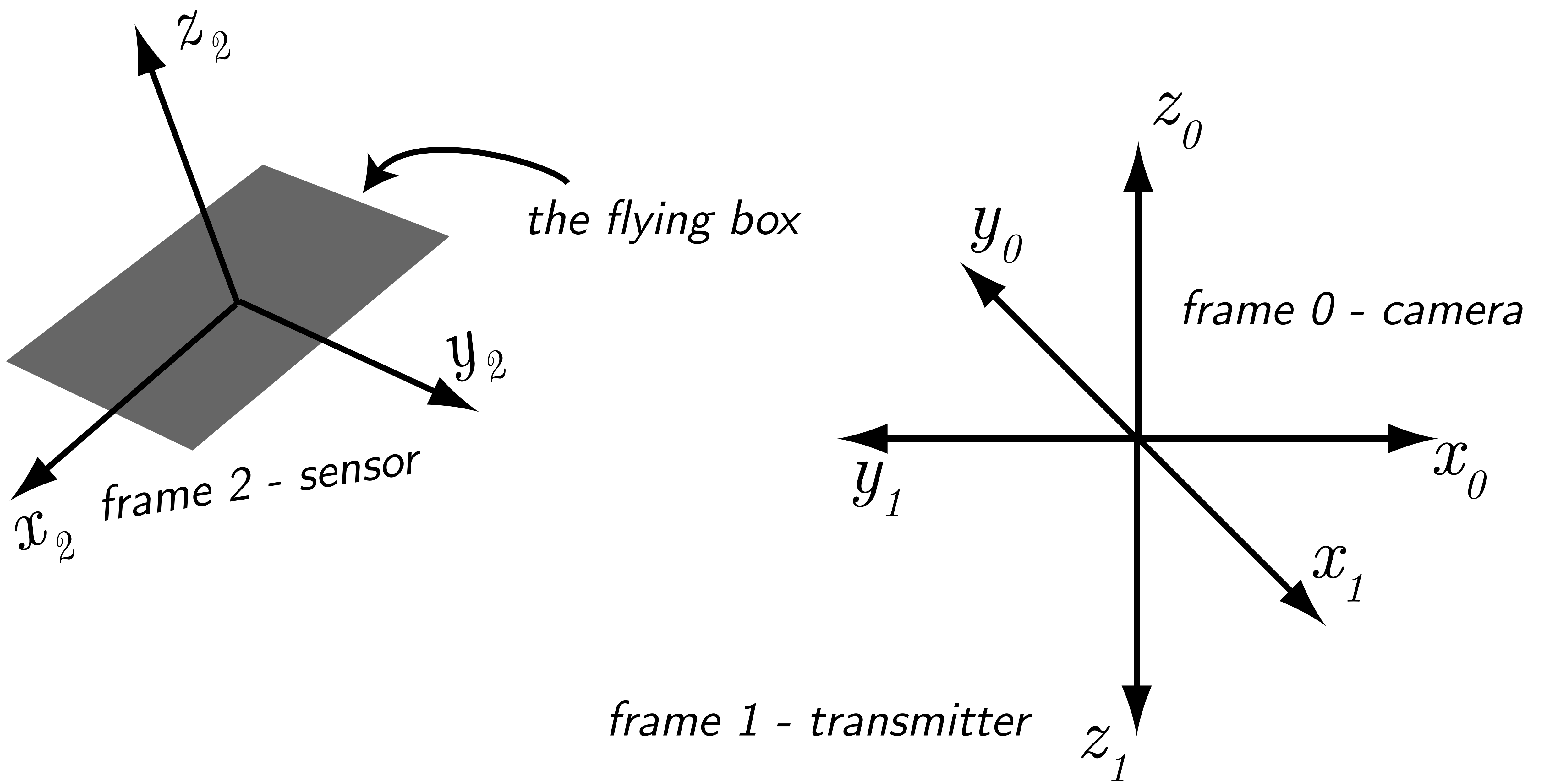
Your task for this question is to update a provided MATLAB script so that it animates the movement of rectangular block that was moved in a specific way. The motion was captured on video, and the positions and orientations of the block were recorded over time using a Ascension TrakStar magnetic motion tracking system that includes a sensor located inside the block.

1

flying_box.txt

Format for each sensor is: status x, y, z, azimuth, elevation, roll, button, quality, time

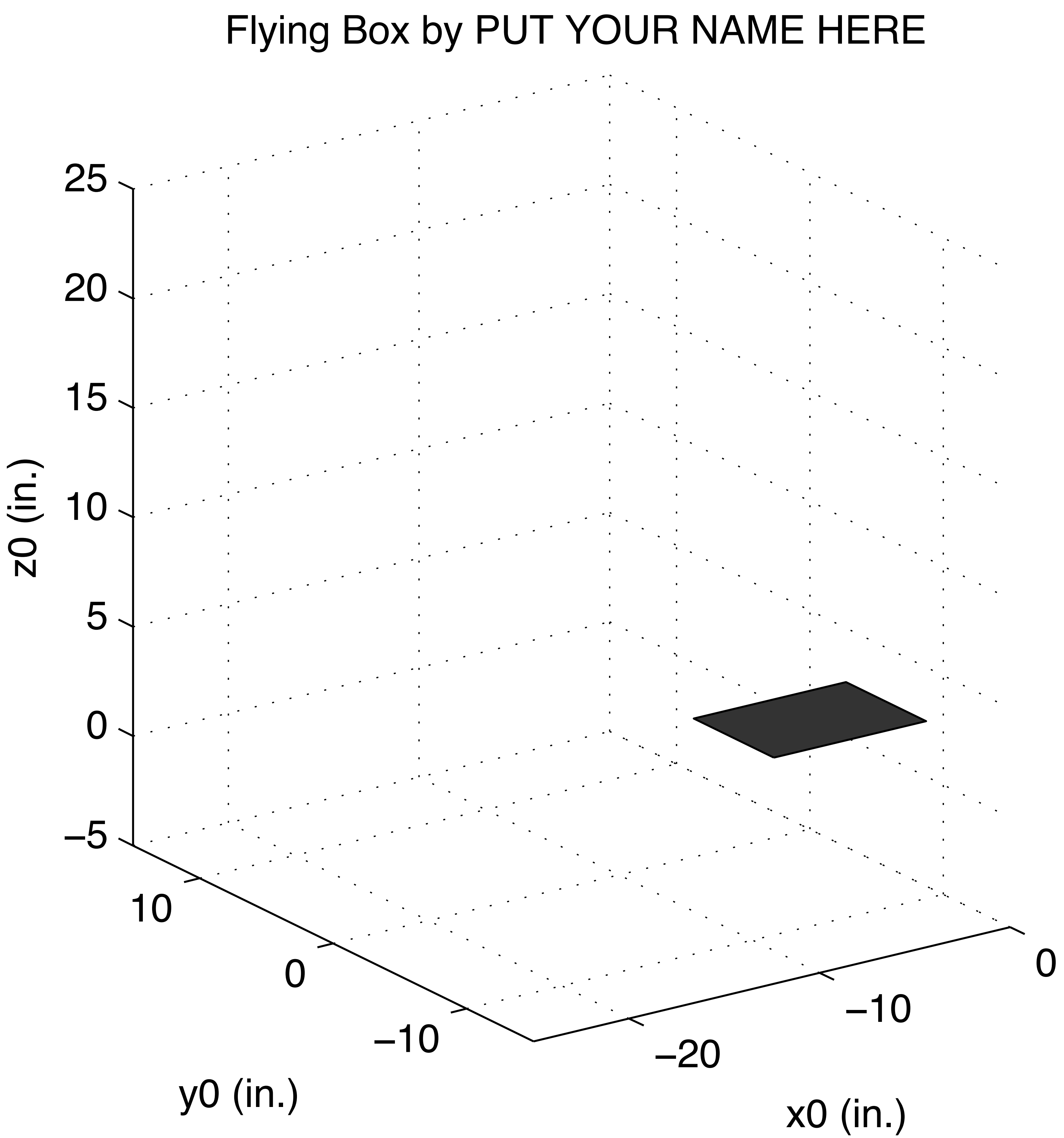
Sensor1:	0x0000	0.312	8.886	-15.579	85.703	2.694	-173.770	0	4	1347328434.211
Sensor1:	0x0000	0.312	8.881	-15.579	85.703	2.687	-173.784	0	4	1347328434.228
Sensor1:	0x0000	0.312	8.873	-15.579	85.696	2.673	-173.798	0	4	1347328434.244
Sensor1:	0x0000	0.312	8.868	-15.579	85.703	2.645	-173.820	0	4	1347328434.261
Sensor1:	0x0000	0.312	8.864	-15.583	85.718	2.624	-173.841	0	4	1347328434.277
Sensor1:	0x0000	0.312	8.859	-15.583	85.746	2.603	-173.862	0	4	1347328434.294
Sensor1:	0x0000	0.312	8.855	-15.587	85.788	2.589	-173.897	0	4	1347328434.315
Sensor1:	0x0000	-0.237	7.163	-13.179	85.729	0.776	-174.592	0	3	1347328468.754
Sensor1:	0x0000	-0.237	7.163	-13.175	85.729	0.776	-174.592	0	3	1347328468.763
Sensor1:	0x0000	-0.233	7.163	-13.175	85.729	0.783	-174.592	0	3	1347328468.775
Sensor1:	0x0000	-0.233	7.163	-13.175	85.729	0.783	-174.592	0	3	1347328468.783
Sensor1:	0x0000	-0.233	7.163	-13.175	85.729	0.790	-174.599	0	3	1347328468.800
Sensor1:	0x0000	-0.229	7.163	-13.175	85.729	0.797	-174.606	0	3	1347328468.817
Sensor1:	0x0000	-0.229	7.163	-13.175	85.729	0.804	-174.606	0	3	1347328468.833
Sensor1:	0x0000	-0.224	7.159	-13.175	85.722	0.811	-174.613	0	3	1347328468.850
Sensor1:	0x0000	-0.220	7.154	-13.179	85.721	0.825	-174.613	0	3	1347328468.867
Sensor1:	0x0000	-0.215	7.150	-13.179	85.728	0.846	-174.627	0	3	1347328468.883
Sensor1:	0x0000	-0.211	7.146	-13.184	85.736	0.860	-174.641	0	3	1347328468.900
Sensor1:	0x0000	-0.207	7.137	-13.188	85.764	0.867	-174.662	0	3	1347328468.917
Sensor1:	0x0000	-0.202	7.132	-13.188	85.778	0.874	-174.676	0	3	1347328468.933
Sensor1:	0x0000	-0.202	7.128	-13.192	85.792	0.874	-174.683	0	3	1347328468.950
Sensor1:	0x0000	-0.198	7.124	-13.192	85.799	0.874	-174.690	0	3	1347328468.967
Sensor1:	0x0000	-0.198	7.119	-13.192	85.813	0.874	-174.697	0	3	1347328468.983
Sensor1:	0x0000	-0.198	7.115	-13.197	85.820	0.867	-174.704	0	3	1347328469.000
Sensor1:	0x0000	-0.193	7.110	-13.197	85.827	0.860	-174.711	0	3	1347328469.017
Sensor1:	0x0000	-0.189	7.106	-13.201	85.834	0.846	-174.711	0	3	1347328469.033
Sensor1:	0x0000	-0.189	7.102	-13.201	85.841	0.825	-174.711	0	3	1347328469.050
Sensor1:	0x0000	-0.185	7.097	-13.201	85.855	0.804	-174.711	0	3	1347328469.067
Sensor1:	0x0000	-0.185	7.093	-13.201	85.869	0.783	-174.697	0	3	1347328469.083
Sensor1:	0x0000	-0.185	7.088	-13.201	85.876	0.762	-174.690	0	3	1347328469.100
Sensor1:	0x0000	-0.185	7.088	-13.201	85.883	0.748	-174.683	0	3	1347328469.117
Sensor1:	0x0000	-0.185	7.088	-13.201	85.883	0.741	-174.676	0	3	1347328469.133
Sensor1:	0x0000	-0.189	7.088	-13.201	85.876	0.734	-174.669	0	3	1347328469.150
Sensor1:	0x0000	-0.189	7.088	-13.201	85.869	0.727	-174.662	0	3	1347328469.167
Sensor1:	0x0000	-0.189	7.088	-13.201	85.862	0.720	-174.655	0	3	1347328469.183
Sensor1:	0x0000	-0.189	7.088	-13.201	85.848	0.706	-174.655	0	3	1347328469.200
Sensor1:	0x0000	-0.193	7.088	-13.201	85.813	0.692	-174.655	0	3	1347328469.217
Sensor1:	0x0000	-0.193	7.093	-13.206	85.715	0.650	-174.655	0	3	1347328469.233
Sensor1:	0x0000	-0.198	7.097	-13.210	85.497	0.581	-174.635	0	3	1347328469.250
Sensor1:	0x0000	-0.193	7.106	-13.219	85.062	0.490	-174.592	0	3	1347328469.267
Sensor1:	0x0000	-0.180	7.110	-13.228	84.437	0.427	-174.536	0	3	1347328469.283
Sensor1:	0x0000	-0.154	7.110	-13.236	83.650	0.399	-174.536	0	3	1347328469.300
Sensor1:	0x0000	-0.114	7.106	-13.254	82.409	0.399	-174.614	0	4	1347328469.321
Sensor1:	0x0000	-0.083	7.102	-13.267	81.540	0.420	-174.677	0	4	1347328469.333
Sensor1:	0x0000	-0.022	7.088	-13.285	79.896	0.476	-174.761	0	4	1347328469.354


z_2
the flying box
y_2
x_2
frame 2 - sensor
z_0
y_0
frame 0 - camera
y_1
x_0
x_1
frame 1 - transmitter
z_1

Editor – /Users/kuchenbe/Documents/teaching/meam 520/assignments/01 transformations/matlab/flying_box_starter.m

```
%% flying_box_starter.m
%
% This Matlab script provides the starter code for the flying box
% problem on Homework 1 in MEAM 520 at the University of Pennsylvania.
% The original was written by Professor Katherine J. Kuchenbecker in
% September of 2012. Students will modify this code to create their own
% script. Email kuchenbe@seas.upenn.edu with questions.
%
% Student Name:
%
% Change the name of this file to replace "starter" with your PennKey.  For
% example, Professor Kuchenbecker's script would be visualize_kuchenbe.m

%% SETUP

% Delete all variables from our workspace.
clear

% Load the TrakStar data recorded during the movie.
% This MATLAB data file includes time histories of the x, y, and z
% coordinates in inches, as well as time histories of the azimuth,
% elevation, and roll angles in degrees.
load flying_box;

% Open figure 1 and clear it to get ready for plotting.
figure(1)
clf

%% DEFINITIONS

% We need to keep track of three frames in this code.
%
% Frame 0 is the frame of the camera's view, with x positive to the right,
% y positive straight back, and z positive up.  The is the base frame, and
% it's what we plot in.  Its origin coincides with the origin of frame 1.
%
% Frame 1 is the frame of the TrakStar transmitter, which sits on the desk.
% It has x positive straight out, y positive to the left, and z positive
```

script | Ln 24 Col 1



Flying Box by PUT YOUR NAME HERE
25
20
15
10
5
0
−5
z0 (in.)
10
0
−10
y0 (in.)
0
−10
−20
x0 (in.)

Programming Homework Tips:

1. Write out your approach before sitting down to program.

2. Start early to give yourself time to figure things out.

Questions ?