Intro to Robotics: Review for Exam 1

Neeloy Chakraborty

neeloyc2@illinois.edu

Agenda

- Exam Logistics
- Topic Roadmap
- Review and Practice
- QA

Exam Logistics

- Available to take between Thursday 2/12 to Sunday 2/15
- Sign up on PrairieTest: https://us.prairietest.com/
  - Let me know if you don't have access to sign up!
- 50 minutes
- Combination of multiple choice and fill in the blank (like HWs)
- Formula sheet and Python docs will be provided in exam
- And you can ask for scratch paper and pencil in CBTF

Formula Sheet and Python Docs

E1: Exam 1: Rigid Body Motions

The following formula sheets are available to you on this exam:

- Equation Sheet
- Python Reference

Question	Status	Available points	Awarded points

Formula Sheet and Python Docs

ECE 470, Spring 2023 **Exam 1 Equation Sheet**

Grubler's Formula

$$Dof = m(N - J - 1) + \sum_i f_i$$

N: number of links

J: number of joints

m: Dof of rigid body, if planner m=3, if spatial m=6

f_i: number of freedom from joint i

2D Rotation matrix

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

3D Rotation matrix

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

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

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

3×3 skew-symmetric matrix representation

$$[x] = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}$$

Angular velocity in the fixed frame and body frame

$$[\omega_s] = \dot{R}R^T$$
$$[\omega_b] = R^T\dot{R}$$

Rotation matrix in exponential coordinate representation

ECE 470 Python Commands Reference Sheet

This document is intended to be a quick reference with some useful Python commands to help with the homework assignments, exams, and labs involved with ECE 470: Introduction to Robotics at the University of Illinois at Urbana-Champaign. For further reference, check out the official documentation for the various packages talked about here.

Importing Libraries

In order to use the libraries talked about in this document, you'll need to include them in your code. This can be done using the `import` command in Python. Add the following lines to the beginning of your code to import everything the same way I have.

```
import numpy as np
from scipy.linalg import expm, logm
```

Create an Array

This first section describes a few useful methods for creating arrays in Python.

np.array()

Creates an array with desired entries. It's done as a list of lists, where the innermost lists make up the rows and they are stacked in sequential order. Below are a few examples:

$$\texttt{np.array([[a, b, c]])} = \begin{bmatrix} a & b & c \end{bmatrix}$$

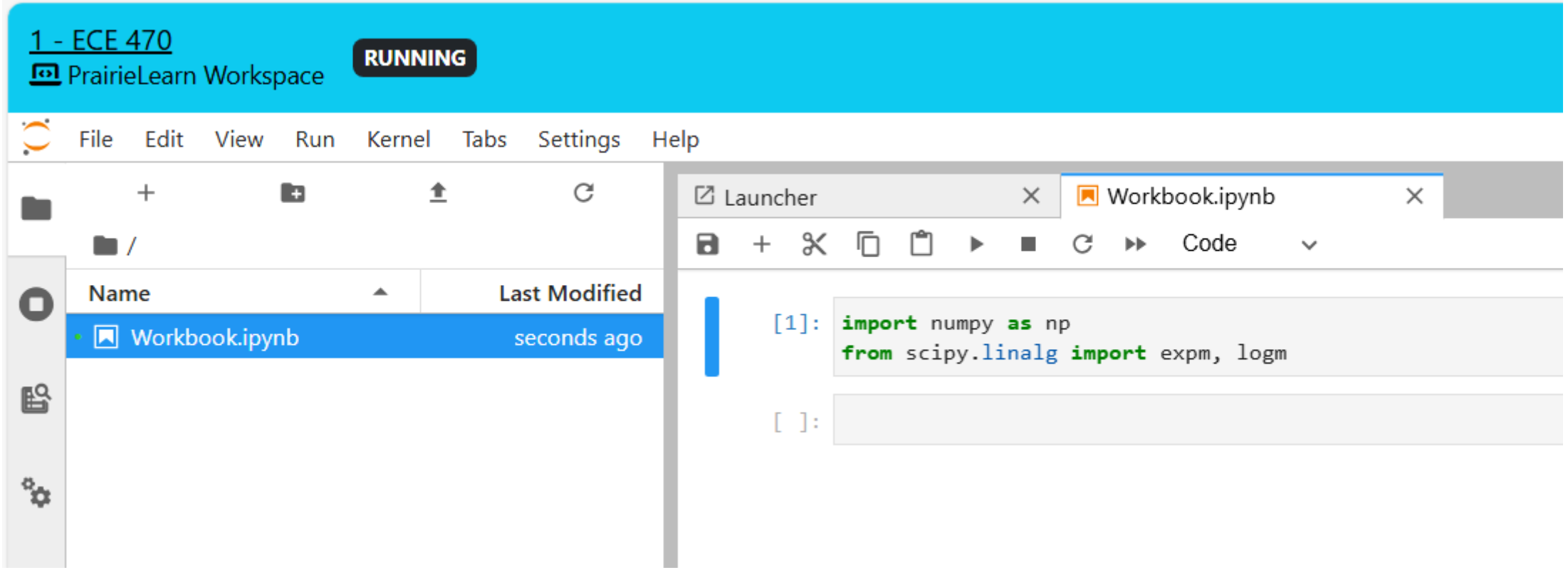
$$\texttt{np.array([[a], [b], [c]])} = \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

$$\texttt{np.array([[a, b], [c, d]])} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

np.zeros()

Creates an array of all zeros. If the input is just an integer, say n the output will be an array of size $1 \times n$. If the integer is a tuple (or a list of numbers), say $m \times n$, the output will be an array of size $m \times n$.

Jupyter Notebook



Roadmap of what we've learned so far...

Math Review

- Matrix Operations
- Jacobians
- Differential Equations

Mechanisms

- Degrees of Freedom
- Joint Types
- Different Spaces

Rigid Body Motion

- Reference Frames
- Rotation Matrix
- Angular Velocity

Exponential Coord.

- Solve for Rotation Matrix Given Exponential Coords.
- Solve for Exp. Coords. Given Rotation Matrix

Homogeneous Transformations

- Combining Rotation and Translation

Twist and Screw

- Twist: Spatial Velocity
- Adjoint Map of Twist
- Screw to Twist
- Twist to Screw

Mechanisms

- The degrees of freedom (DoF) is the smallest number of real-valued coordinates needed to represent a robot's configuration

$$Dof = m(N - J - 1) + \sum_i f_i$$

N: number of links

J: number of joints

m: Dof of rigid body, if planner m=3, if spatial m=6

f_i: number of freedom from joint i

$$Dof = m(N - J - 1) + \sum_i f_i$$

N: number of links

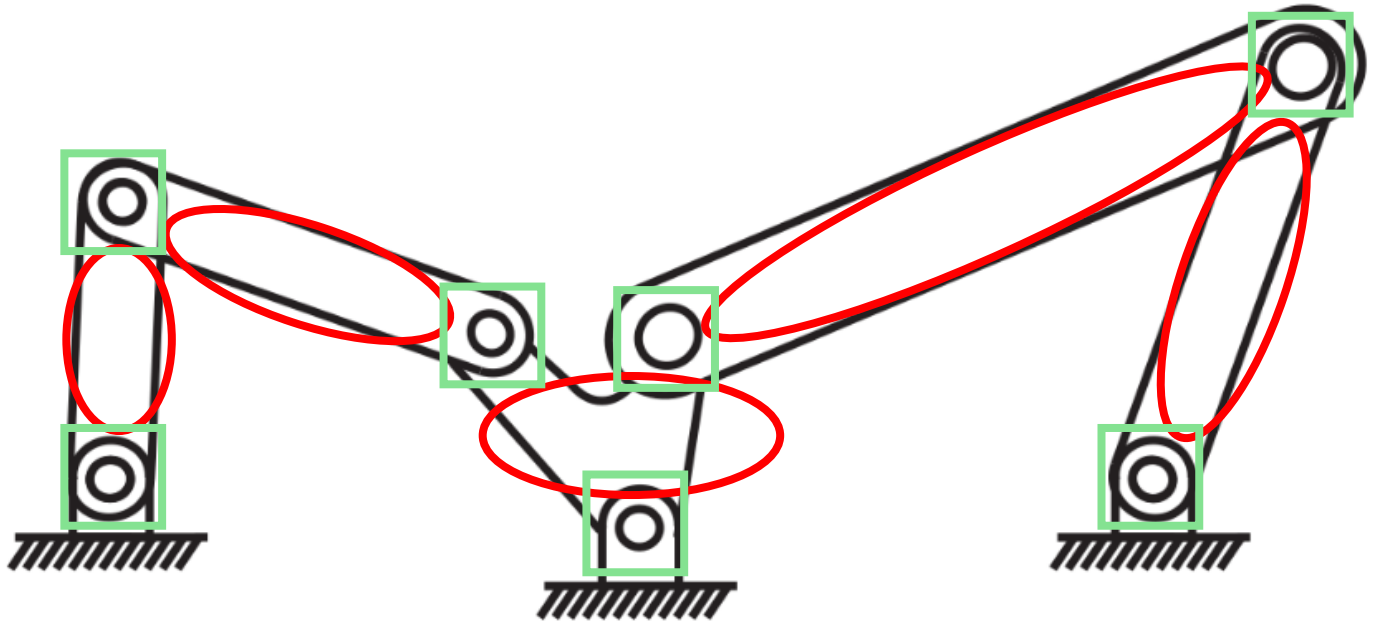
J: number of joints

m: Dof of rigid body, if planner m=3, if spatial m=6

f_i: number of freedom from joint i

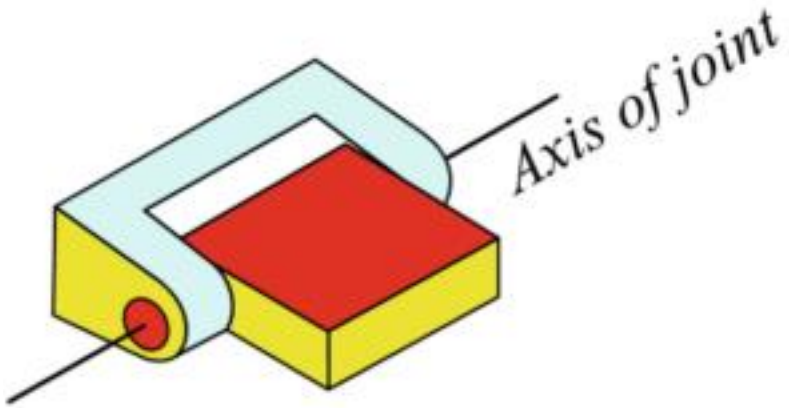
Grubler's Formula

Consider the robot depicted below. How many degrees of freedom does it have?



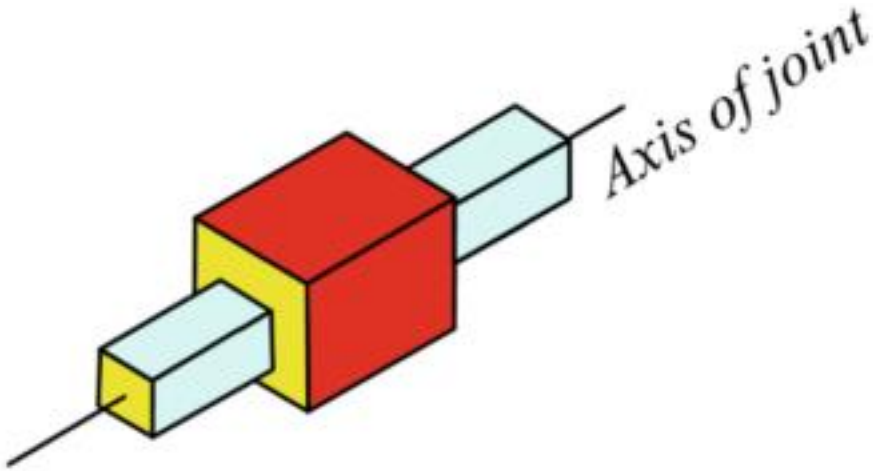
$N =$	5 + 1 (for ground) = 6	$J =$	7
$m =$	3 (since planar)	$\sum f_i =$	7 (1 per joint)
$DoF =$	3(6 – 7 – 1) + 7 = 1		

Revolute and Prismatic Joints



Revolute joint

Rotates along 1 axis
1 DoF
3 – 1 = 2 constraints in 2D
6 – 1 = 5 constraints in 3D



Prismatic joint

Translates along 1 axis
1 DoF
3 – 1 = 2 constraints in 2D
6 – 1 = 5 constraints in 3D

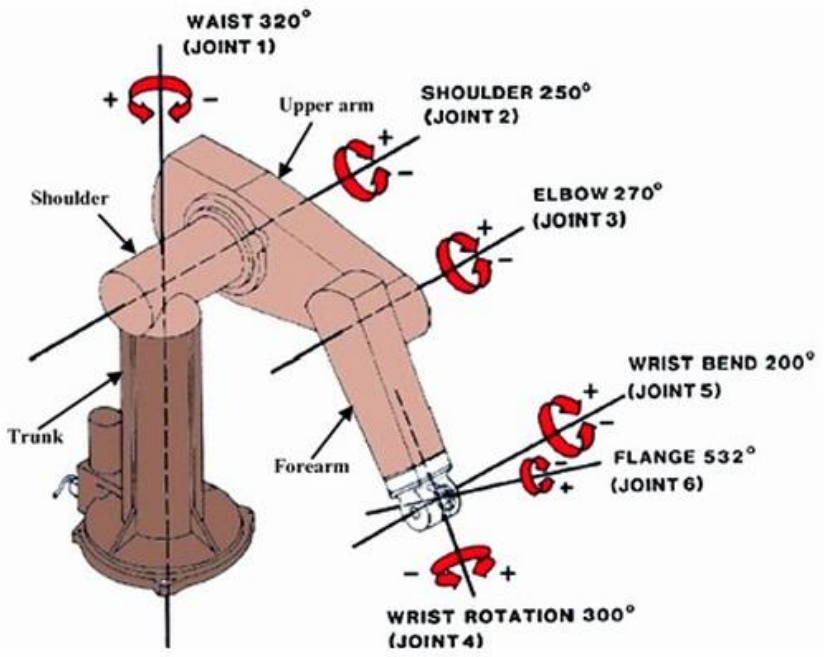
https://link.springer.com/chapter/10.1007/978-3-030-93220-6_1

Different Spaces

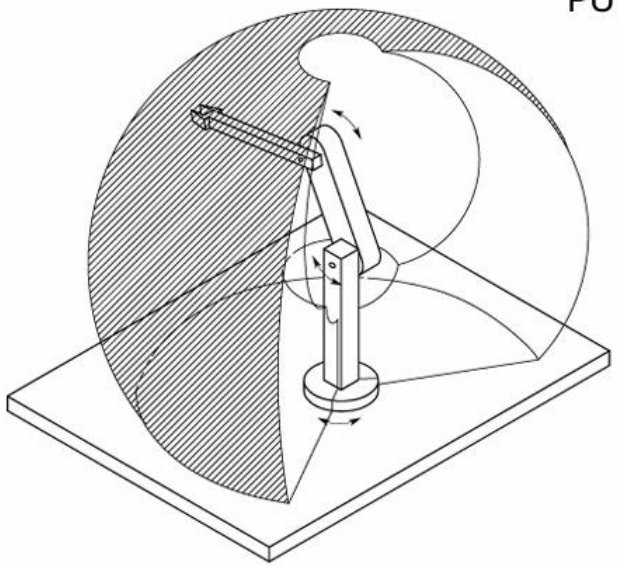
Configuration space contains all possible configurations of the robot

Workspace is the space that the end effector can actually reach

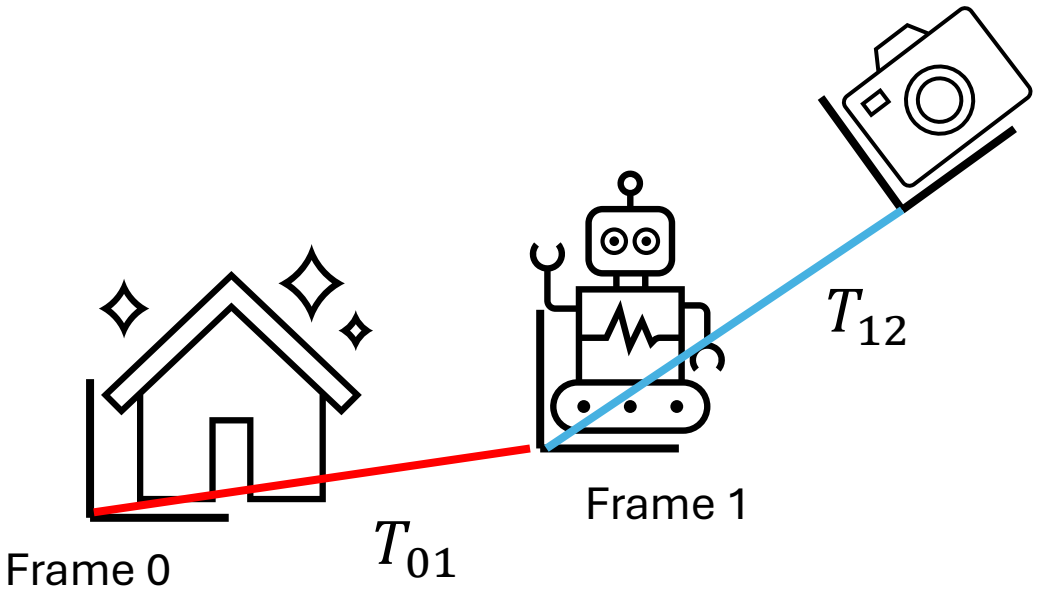
Task space is the space in which the robot's task can be naturally expressed

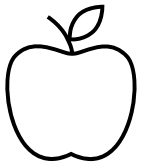


PUMA 560



Reference Frames





p_2

Point expressed in frame 2

Point expressed in frame 0 = $T_{01}T_{12}p_2$

We usually refer to the fixed frame as *s* and the body frame as *b*

Find the coordinates of a point in a different frame

View

As in the lecture example, we have a robot equipped with a sensor. The sensor detects an obstacle at location p_{sensor} in the sensor's frame of reference.

$$p_{sensor} = \begin{bmatrix} 0.14800855 \\ 0.77770102 \\ -0.97742119 \end{bmatrix}$$

In a frame of reference fixed to the robot, the sensor's pose is:

$$B = \begin{bmatrix} -0.75687540 & 0.23259733 & 0.61076847 & -0.95712585 \\ 0.29343167 & -0.71410204 & 0.63557544 & -0.01909166 \\ 0.58398415 & 0.66027022 & 0.47223484 & 0.77973625 \\ 0.00000000 & 0.00000000 & 0.00000000 & 1.00000000 \end{bmatrix}$$

$$p_{global} = ABp_{sensor}$$

And in a global reference frame, the robot's reference frame is at pose:

$$A = \begin{bmatrix} 0.81790325 & 0.11007091 & -0.56472884 & 0.24637657 \\ 0.27105327 & 0.79206968 & 0.54695133 & 0.13113728 \\ 0.50750802 & -0.60042487 & 0.61800128 & -0.42045502 \\ 0.00000000 & 0.00000000 & 0.00000000 & 1.00000000 \end{bmatrix}$$

Find the location of the obstacle in the global frame. (Make sure you round your answer up.)

$p_{global} =$ matrix (4 digits after decimal)

Rotation Matrices

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

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

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

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

$a = Rb$ will be b after applying rotation matrix R designed to rotate around an axis ω by angle θ

Rotation Matrices

$a = Rb$ will be b after applying rotation matrix R designed to rotate around an axis ω by angle θ

R is a part of **SO**(3), which gives it some special properties:

R is a 3x3 real-valued matrix

$R^T R = I$, which means that $R^{-1} = R^T$ (orthonormal matrix)

$\det(R) = 1$ (right-handed)

Do rotation matrices commute?

If A and B are rotation matrices, generally $AB \neq BA$

Rotation Commutation

Consider the two rotation matrices A and B:

$$\begin{bmatrix} 1.00 & 0.00 & 0.00 \\ 0.00 & 0.60 & -0.80 \\ 0.00 & 0.80 & 0.60 \end{bmatrix}$$

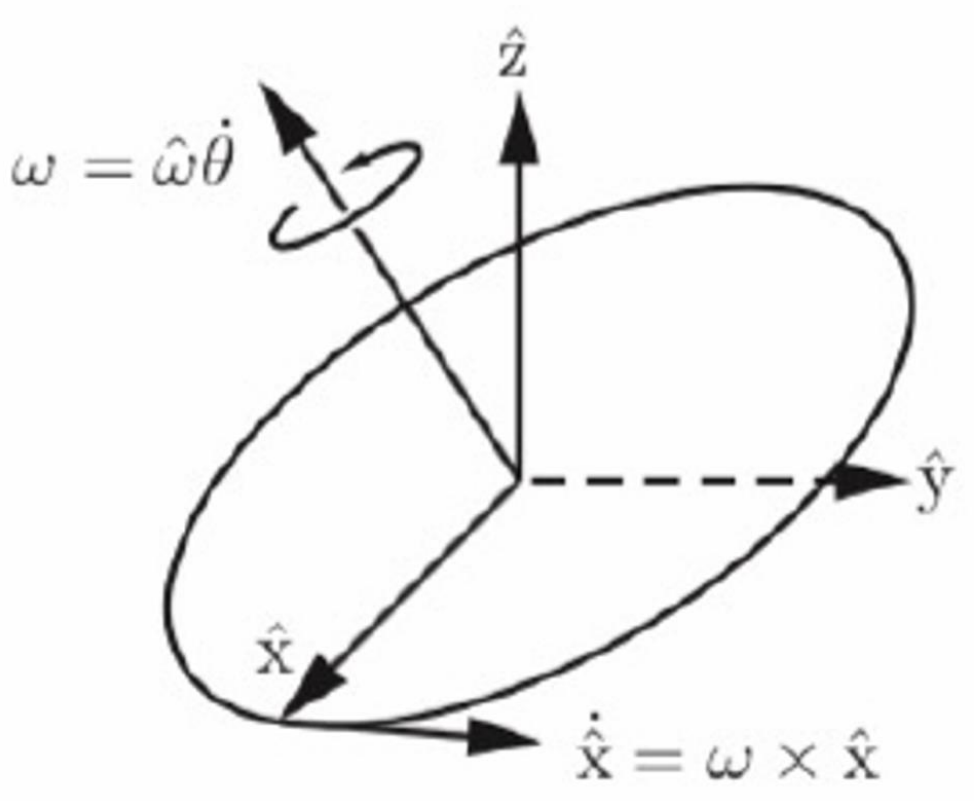
$$\begin{bmatrix} -0.01 & 0.00 & 1.00 \\ 0.00 & 1.00 & 0.00 \\ -1.00 & 0.00 & -0.01 \end{bmatrix}$$

What is AB? What is BA? Do these matrices commute?

$AB =$	matrix (rtol=0.01, atol=1e-08)
$BA =$	matrix (rtol=0.01, atol=1e-08)

- ○ (a) Yes, they commute!
- ○ (b) No, they do not commute!

Angular Velocity



Suppose $\hat{x}, \hat{y}, \hat{z}$ denotes the basis of our frame, $\hat{\omega}$ is an axis of rotation, and $\dot{\theta}$ is our rotational velocity

$$\dot{\hat{x}} = \omega \times \hat{x}$$
$$\dot{\hat{y}} = \omega \times \hat{y}$$
$$\dot{\hat{z}} = \omega \times \hat{z}$$

The cross product produces a vector orthogonal to the other two

Angular Velocity

Suppose $\hat{x}, \hat{y}, \hat{z}$ denotes the basis of our frame, $\hat{\omega}$ is an axis of rotation, and $\dot{\theta}$ is our rotational velocity

Let $R = R_{sb}$ be the orientation of our body frame with respect to the fixed frame

$$\dot{\hat{x}} = \omega \times \hat{x}$$
$$\dot{\hat{y}} = \omega \times \hat{y}$$
$$\dot{\hat{z}} = \omega \times \hat{z}$$

$$\dot{R} = [\omega \times \hat{x} \quad \omega \times \hat{y} \quad \dot{\hat{z}} = \omega \times \hat{z}]$$

Computing cross products is kinda hard!

Skew-Symmetric Matrix and $[\omega]$

For a given rotation axis ω, we define $[\omega]$ as a 3×3 skew-symmetric matrix which lives in so(3):

$$[x] = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}$$

Then, $\dot{R} = \omega \times R = [\omega]R$ (for R_{sb} and ω_s)

Properties of Skew-Symmetric Matrices

If ω is a 3D real-valued vector and R is in **SO**(3), then $R[\omega]R^T = [R\omega]$

Suppose $R = R_{sb}$ and we want to solve for $[\omega_b]$ given $[\omega_s]$

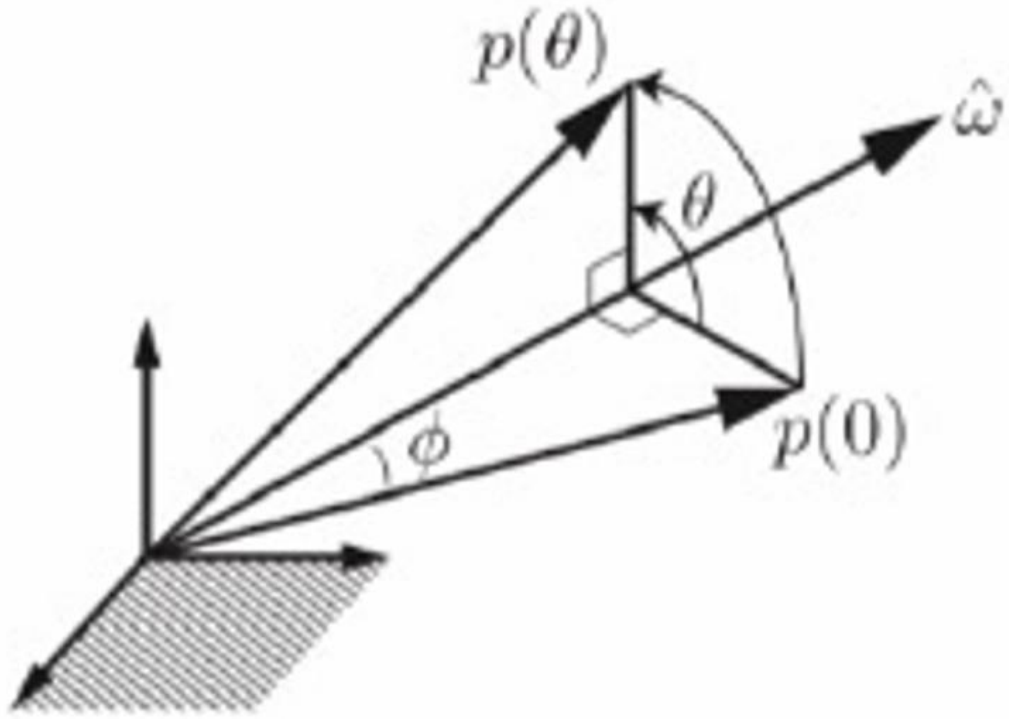
$$[\omega_b] = [R^T\omega_s] = R^T[\omega_s]R = R^T\dot{R}R^TR = R^T\dot{R}$$

$$\dot{R} = [\omega_s]R = R[\omega_b]$$

Exponential Coordinates

$$\dot{R} = [\omega_s]R = R[\omega_b]$$

Since we have a differential equation of the form $\dot{x} = Ax$, we can solve for x using the matrix exponential!



$$R(\omega, \theta) = e^{[\omega_s]\theta}R(0) = R(0)e^{[\omega_b]\theta}$$

scipy.linalg.expm()

Matrix Logarithm

scipy.linalg.logm()

The matrix exponential gives us the rotation matrix given exponential coordinates

The matrix logarithm gives us the exponential coordinates given the rotation matrix

$$tr(R) = r_{11} + r_{22} + r_{33} = 1 + 2cos\theta$$

$$[\hat{\omega}] = \frac{1}{2sin\theta}(R^T - R)$$

Homogeneous Transformations and **SE**(3)

The **special Euclidean group** $\boldsymbol{SE(3)}$ is the set of 4×4 matrices of the form:

$$T = T(R,p) = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}$$

numpy.linalg.inv()

- The inverse of T is $T^{-1} = \begin{bmatrix} R^{\mathsf{T}} & -R^{\mathsf{T}}p \\ 0 & 1 \end{bmatrix} \in SE(3)$
- If T_1 and $T_2 \in SE(3)$, then $T_1T_2 \in SE(3)$

Spatial Twist

While rotation matrices capture rotation, homogeneous transformations capture both rotation and translation

We want to find an operator that governs the ODE for the pose of the robot body

$$\dot{R} = [\omega_s]R = R[\omega_b]$$

$$\dot{T} = [V_s]T = T[V_b]$$

Spatial Twist

Twist V is a 6D real-valued vector containing the angular and linear velocity

$$\mathcal{V} = \begin{bmatrix} \omega \\ v \end{bmatrix} \quad [\mathcal{V}] = \begin{bmatrix} [\omega] & v \\ 0 & 0 \end{bmatrix}$$

$$\mathcal{V}_s = [Ad_T]\mathcal{V}_b = \begin{bmatrix} R & 0 \\ [p]R & R \end{bmatrix} \mathcal{V}_b$$

Find a twist given linear and angular velocity

View...

Given the angular velocity $w^b_{s,b}$ and linear velocity $v^b_{s,b}$ of the body frame with respect to the space frame in the coordinates of the body frame, find the body twist $\mathcal{V}^b_{s,b}$.

$$\mathcal{V} = \begin{bmatrix} \omega \\ v \end{bmatrix} \qquad [\mathcal{V}] = \begin{bmatrix} [\omega] & v \\ 0 & 0 \end{bmatrix}$$

Find a twist given its matrix representation

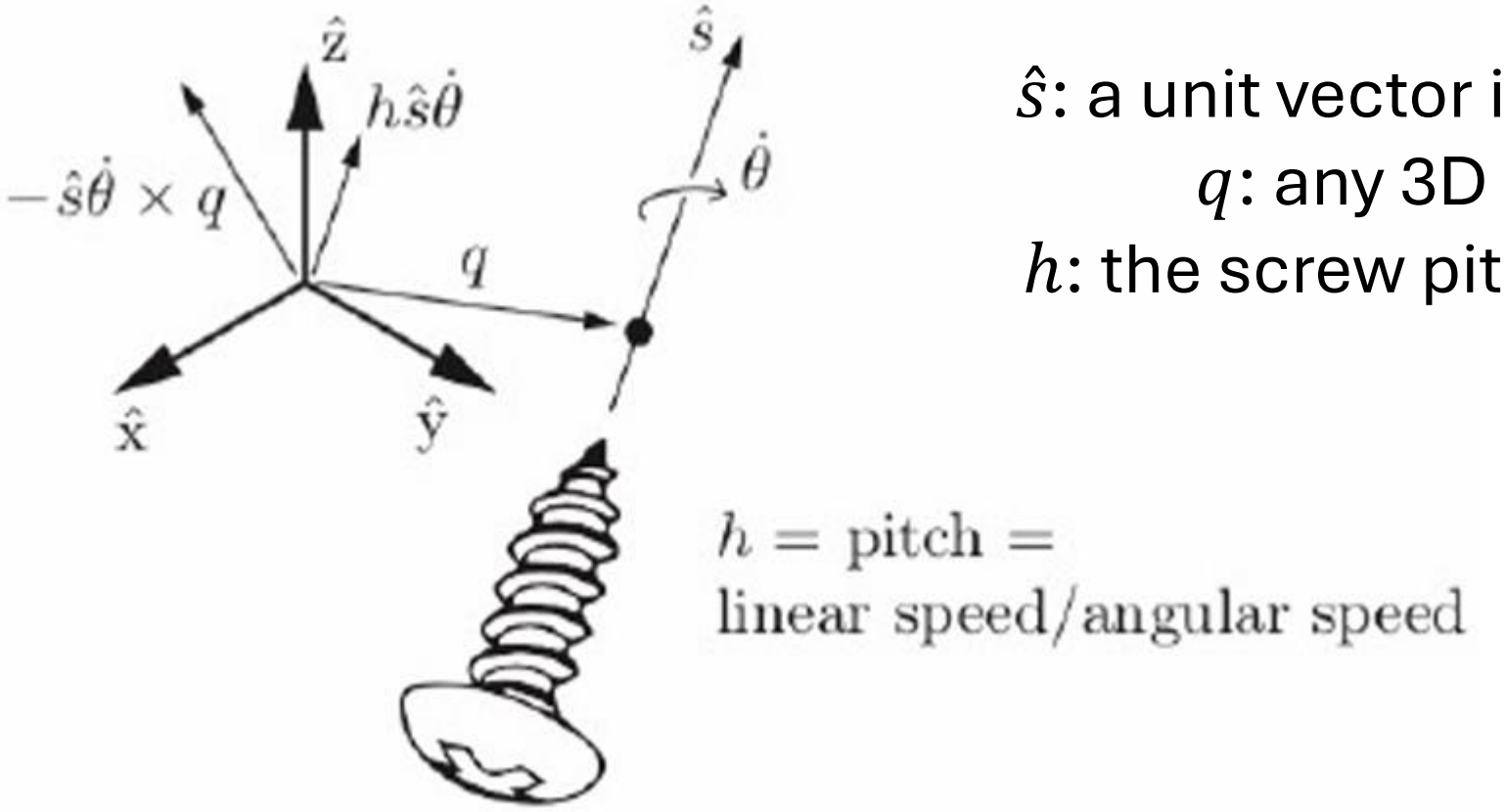
Find the twist $\mathcal{V}$ whose matrix representation is

$$[\mathcal{V}] = \begin{bmatrix} 0 & 10 & 8 & -8 \\ -10 & 0 & 9 & -8 \\ -8 & -9 & 0 & 5 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

$$\mathcal{V} = \begin{bmatrix} \omega \\ v \end{bmatrix} \qquad [\mathcal{V}] = \begin{bmatrix} [\omega] & v \\ 0 & 0 \end{bmatrix}$$

Twist in Terms of Screw

A twist V can be viewed in terms of a screw motion S



$\hat{s}$: a unit vector in the direction of the axis

q: any 3D point along the axis

h: the screw pitch (linear/angular speed)

Twist in Terms of Screw

$\hat{s}$: a unit vector in the direction of the axis
q: any 3D point along the axis
h: the screw pitch (linear/angular speed)

Screw to Twist:

$$\mathcal{V} = \begin{bmatrix} \omega \\ v \end{bmatrix} = \begin{bmatrix} \hat{s}\dot{\theta} \\ h\hat{s}\dot{\theta} - \hat{s}\dot{\theta} \times q \end{bmatrix}$$

Twist to Screw:

$$\hat{s} = \frac{\omega}{||\omega||} = \hat{\omega}, \quad \dot{\theta} = ||\omega||, \quad h = \frac{\omega^T v}{||\omega||^2}$$

$$\text{if } \omega \neq 0, \quad q = \frac{\hat{s} \times v}{\dot{\theta}}$$

Twist in Terms of Screw

A twist V for 1 second around the screw axis at angular velocity θ results in frame T

A twist V for θ seconds around the screw axis at angular velocity 1 results in frame T

2 Special Cases of Screws

$\hat{s}$: a unit vector in the direction of the axis
q: any 3D point along the axis
h: the screw pitch (linear/angular speed)

$$\mathcal{V} = \begin{bmatrix} \omega \\ v \end{bmatrix} = \begin{bmatrix} \hat{s}\dot{\theta} \\ h\hat{s}\dot{\theta} - \hat{s}\dot{\theta} \times q \end{bmatrix}$$

$$v = h\omega - \omega \times q$$

If $||\omega|| = 1$ and $h = 0$:
Pure rotation
Revolute joint

If $\omega = 0$ and $||v|| = 1$:
Pure translation
Prismatic joint

Solving the ODE $\dot{T} = [V_s]T$

scipy.linalg.expm()

Recall how we solved for the orientation R_{sb} given the angular velocity ω:

$$\dot{R} = [\omega_s]R = R[\omega_b]$$

$$R(\omega, \theta) = e^{[\omega_s]\theta}R(0) = R(0)e^{[\omega_b]\theta}$$

Similarly, solving for the pose T_{sb} after applying twist V:

$$\dot{T} = [V_s]T = T[V_b]$$

$$T(V, \theta) = e^{[V_s]\theta}T(0) = T(0)e^{[V_b]\theta}$$

Find the pose that is produced by a given screw

View...

Frame 0 is fixed in space and frame 1 is fixed in a body. The current pose of frame 1 in the coordinates of frame 0 is M. The body undergoes a spatial screw motion with axis $\mathcal{S}$ and magnitude θ . Find the new pose T_{01} of frame 1 in the coordinates of frame 0.

```
import numpy as np

M = np.array([[0.14646782, -0.54878165, 0.82303456, 0.11733897], [0.38572645, 0.79783280, 0.46333350, -2.8500
S = np.array([[0.84568872], [-0.43027893], [-0.31570656], [0.26600728], [0.48278670], [2.53120659]])
theta = 0.37859059
```

$$M = T_{01}(0)$$

$$S = \begin{bmatrix} \omega_s \\ v_s \end{bmatrix} = V_s \text{ (remember, these variables are a bit overloaded)}$$

$$[V_s] = \dot{T}T^{-1} \rightarrow \dot{T} = [V_s]T \qquad T_{01}(\theta) = \exp([V_s]\theta)\, T_{01}(0)$$

Find the pose that is produced by a given screw

View...

Frame 0 is fixed in space and frame 1 is fixed in a body. The current pose of frame 1 in the coordinates of frame 0 is M. The body undergoes a body screw motion with axis $\mathcal{B}$ and magnitude θ. Find the new pose T_{01} of frame 1 in the coordinates of frame 0.

```
import numpy as np

M = np.array([[0.92312088, -0.22311794, 0.31315529, 2.87892900], [-0.31001199, -0.91366561, 0.26288347, -
B = np.array([[0.00000000], [0.00000000], [0.00000000], [0.41835746], [0.87281427], [0.25134100]])
theta = 1.04488973
```

$$M = T_{01}(0)$$

$$B = \begin{bmatrix} \omega_b \\ v_b \end{bmatrix} = V_b \text{ (remember, these variables are a bit overloaded)}$$

$$[V_b] = T^{-1}\dot{T} \rightarrow \dot{T} = T[V_b] \qquad T_{01}(\theta) = T_{01}(0)\exp([V_b]\theta)$$

Going from a Transformation to Twist

Just like how we took the matrix logarithm of the rotation matrix to get the angular velocity, we can apply the same to a given transformation to get the twist:

$$\dot{T} = [V_s]T = T[V_b]$$

$$T(V, \theta) = e^{[V_s]\theta}T(0) = T(0)e^{[V_b]\theta}$$

$$\log\big(T(V, \theta)\big) = [V_s]\theta$$

scipy.linalg.logm()

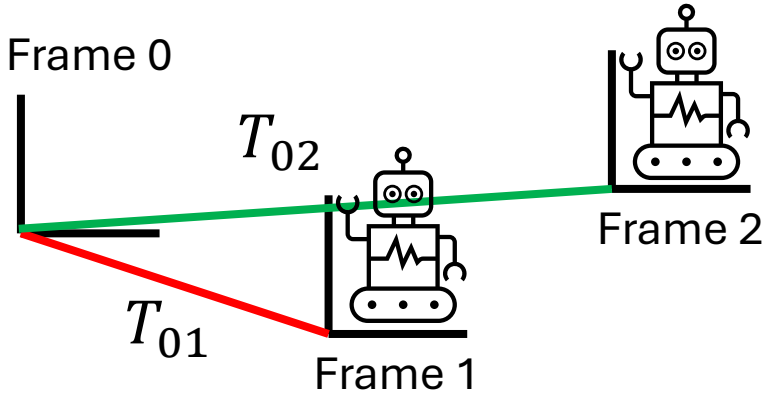
Find the spatial twist that would produce a given change in pose

View...

The current pose of frames 1 and 2 are given as follows. With what constant spatial twist $\mathcal{V}_0$ would frame 1 have to move in order to be aligned with frame 2 after one second?

```
import numpy as np

T_1in0 = np.array([[0.58260087, -0.78693103, -0.20326285, -0.16331626], [0.61107616, 0.58900330, -0.52882988, 0.21503648],
T_2in0 = np.array([[0.48872007, 0.01537159, -0.87230523, -0.05045123], [-0.48921424, 0.83269040, -0.25941498, 0.25574963],
```



$$[V_s] = \dot{T}T^{-1} \rightarrow \dot{T} = [V_s]T$$

$$T_{02} = \exp([V_s]\theta)\, T_{01}$$

$$\exp([V_s]\theta) = T_{02}T_{01}^{-1} \qquad [V_s]\theta = \log(T_{02}T_{01}^{-1})$$

Roadmap of what we've learned so far...

Math Review

- Matrix Operations
- Jacobians
- Differential Equations

Mechanisms

- Degrees of Freedom
- Joint Types
- Different Spaces

Rigid Body Motion

- Reference Frames
- Rotation Matrix
- Angular Velocity

Exponential Coord.

- Solve for Rotation Matrix Given Exponential Coords.
- Solve for Exp. Coords. Given Rotation Matrix

Homogeneous Transformations

- Combining Rotation and Translation

Twist and Screw

- Twist: Spatial Velocity
- Adjoint Map of Twist
- Screw to Twist
- Twist to Screw