EECS/MechE/BioE C106A: Midterm 2 Review Session

The return of Prof. Tarun Amarnath!

Look at that brilliant smile

Lab

- Make sure you’re familiar with the basic setup operations!
- Nodes, topics, publishers, subscribers
- Creating packages, running programs
- Work done in labs (planning, tracking, mapping, etc.)

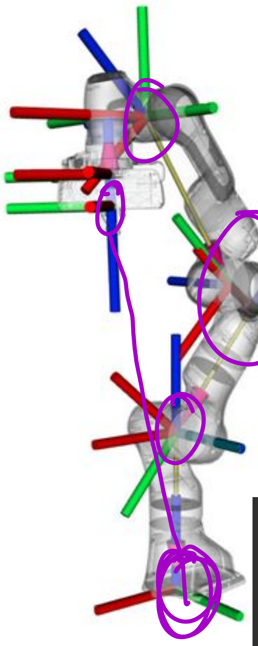
TF Tree, transforms, & RVIZ

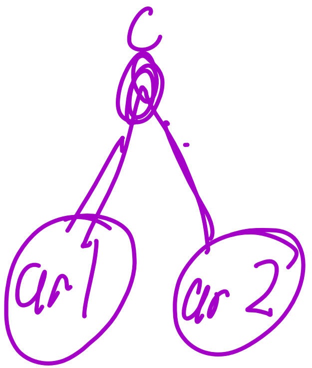
- Can perform transform between any two coordinate frames in TF tree using tf2
- How to code a transform

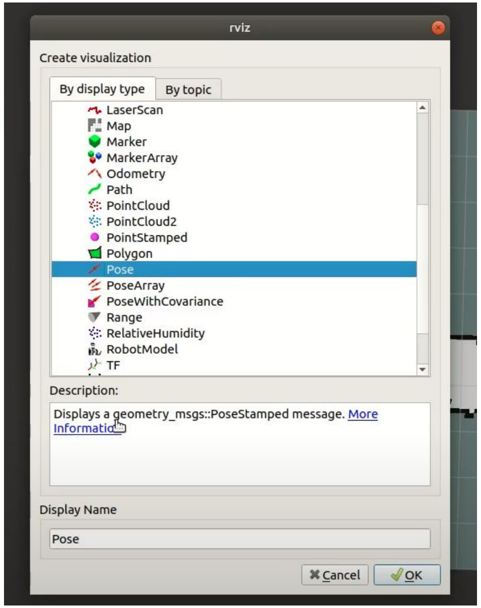
G_{ob}

```
tfBuffer = tf2_ros.Buffer()
tfListener = tf2_ros.TransformListener(tfBuffer)
while not rospy.is_shutdown():
    try:
        trans = tfBuffer.lookup_transform('odom', 'base_footprint', rospy.Time())
        print(trans)
        break
    except:
        pass
```

- We can display a bunch of objects in RViz
  - Image, TF, Robot Model, Point, Marker

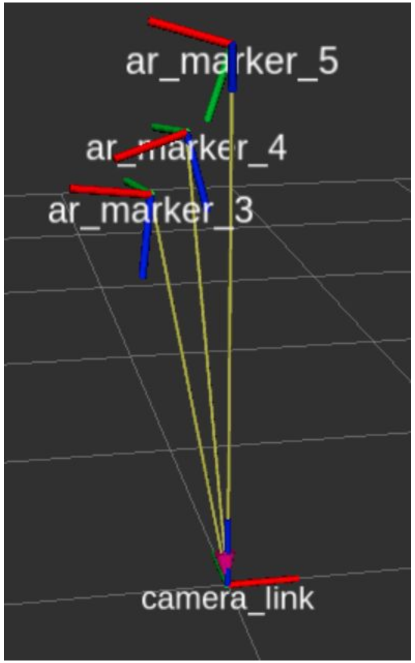






Labs - Fullstack Robotics

ar1 ar2



- Perception
  - AR Tags - Forms a TF between camera and AR tags
  - RGBD Cameras
  - RGB vs HSV

+/-10 intensity (x, y, d) brightness

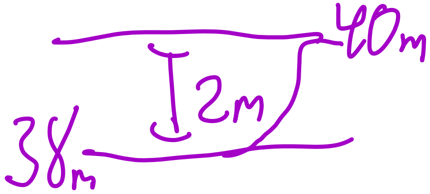
H S V

- Path Planning
  - Its ok be happy (don't worry about path planning for exam)
- Control
  - Positional vs Velocity PID control
  - **P**orptional **I**ntegral **D**erivative

+/-50

5ft

goal po



All the Past Content...

Rigid Body Transformations

- Length and orientation preserving
- Represent a movement or a change in coordinate frame
- Rotations, translations, or both (screw motion)

Homogeneous Transformation Matrices

- Compact representation
- Both rotation and translation included
- Can stack and invert

$$g = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \qquad g^{-1} = \begin{bmatrix} R^T & -R^T p \\ 0 & 1 \end{bmatrix}$$

Exponential Coordinates

- **Goal:** Create rotation and homogeneous transformation matrices as a *function of time*
- Comes from solving a differential equation
- We only need information about **how** the object moves (time is a parameter that's plugged in)

$$g(t) = e^{\hat{\xi} t} g(0)$$

(ξ, t)

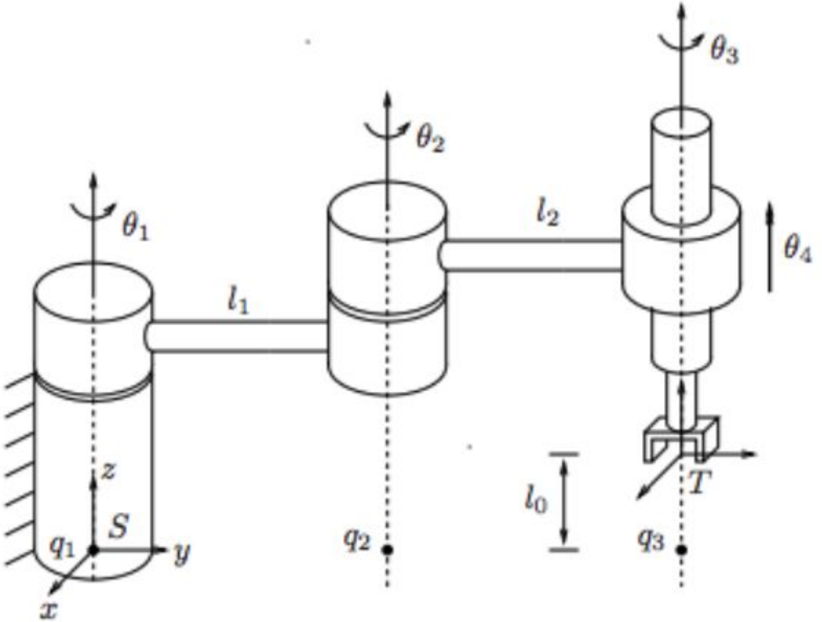
$$R(t) = e^{\hat{\omega} t} R(0)$$

(ω, t)

Forward Kinematics

- **Goal:** Find the location of the tool after a multi-joint robot arm has moved around
- Compose exp. coords

$$g_{st}(\theta_1 \ldots \theta_n) = e^{\hat{\xi}_1 \theta_1} \cdots e^{\hat{\xi}_n \theta_n} g_{st}(0)$$



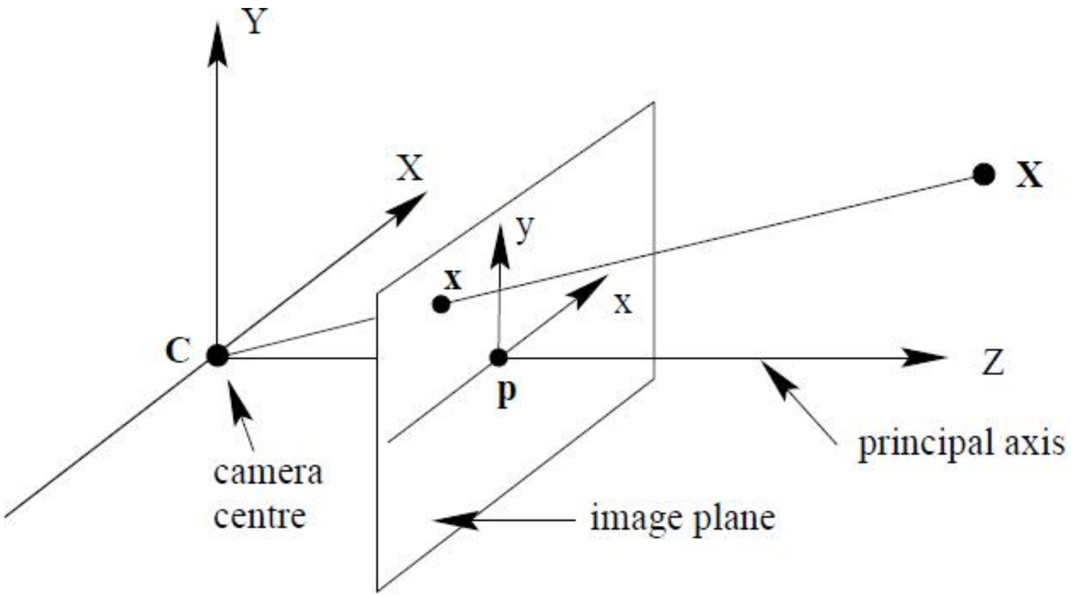
Inverse Kinematics

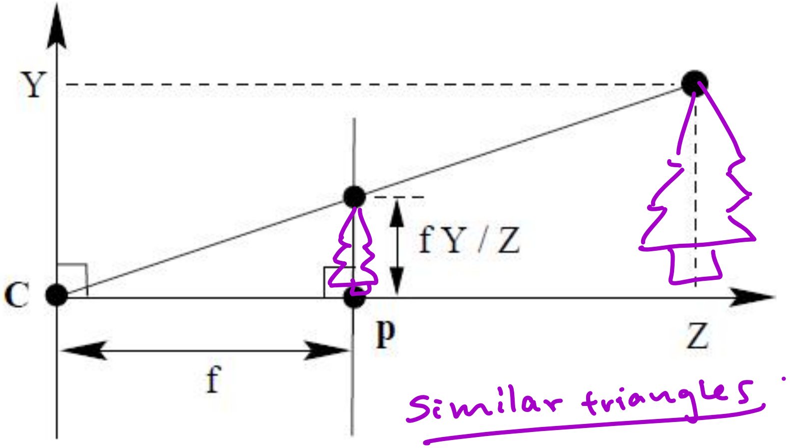
- Given: some final position
- Joint twists
- Final: Θ_s

- How do we move our robot's joints to reach a desired configuration?
- Use Paden-Kahan subproblems along with tricks (reduce problem down to simpler parts)

Computer Vision

Pinhole Camera Model





similar triangles

Img y → $\frac{y}{f} = \frac{Y}{Z}$

↑ Img Z

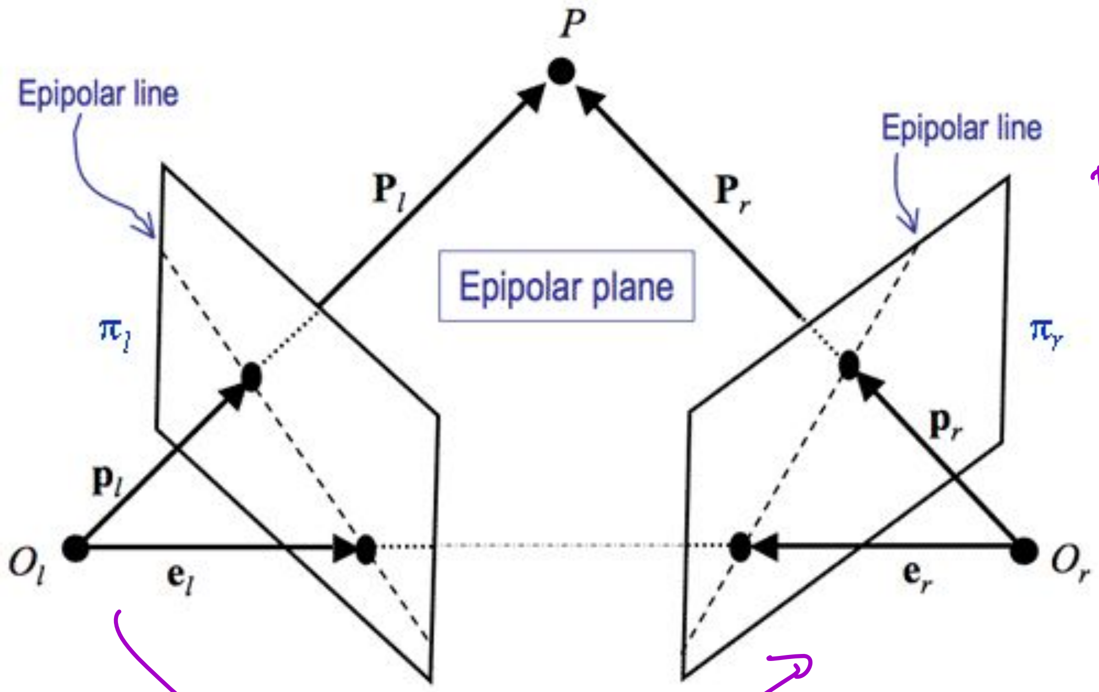
$$(X,Y,Z)^T \longrightarrow (\frac{fX}{Z}, \frac{fY}{Z})^T$$

$$K = \begin{bmatrix} f_x & \cancel{0} & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix}$$

f_θ

$z \leftarrow \lambda x = KX$

Two-View Geometry



T&R

$$(x_2)^T E x_1 = 0$$

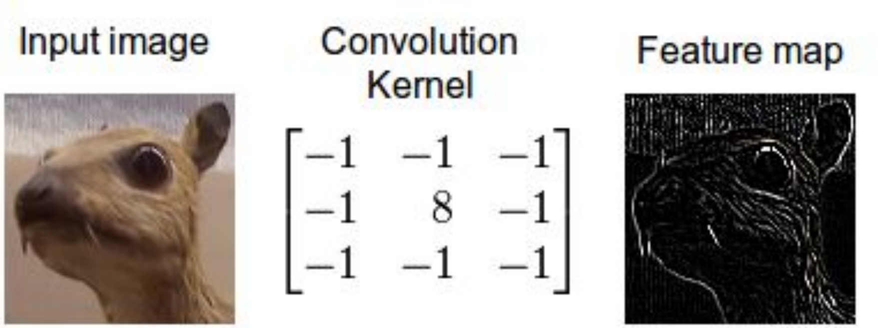
point in img 2

$= \hat{T}R$

pt in img 1

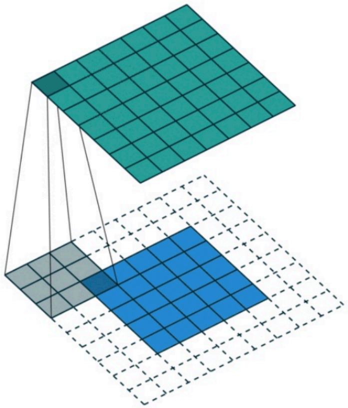
Convolutions

- Slide a kernel over some image
- Understand some information about the picture



Original image

Gaussian Blur filter applied



Velocities

What do we mean by them?

- Velocity in general is the rate of change with respect to some reference frame
- With robots, use a **stationary frame**
- Calculate the velocity of some point attached to the end effector wrt to the base

Some Important Considerations

- Spatial & body velocities - **just a coordinate shift**, tells us which coordinate system to use

$\rightarrow V^s, V^b = \begin{bmatrix} v \\ \omega \end{bmatrix}$

- Spatial and body velocities are **twists**
- Generic expressions for any point → how robot moves
- Can apply them to a specific point to determine that point's velocity

$v_q = \hat{V}^s_{AB} \, p_a$

Spatial Velocity

- Express our point in the **spatial frame**

$\rightarrow$ A frame is spatial

$$\frac{d}{dt}\left(\begin{array}{l} q_a(t) = g_{ab} \cdot q_b \\ \dot{q}_a(t) = \dot{g}_{ab} \cdot q_b \end{array}\right.$$

Switch frames

$$v_{q_a}(t) = \underbrace{\dot{g}_{ab}\, g^{-1}_{ab}}_{\hat{V}^s}\, q_a$$

$$\widehat{V}^s_{ab} := \dot{g}_{ab} g^{-1}_{ab} = \begin{bmatrix} \dot{R}_{ab} R^T_{ab} & -\dot{R}_{ab} R^T_{ab} p_{ab} + \dot{p}_{ab} \\ 0 & 0 \end{bmatrix} \qquad V^s_{ab} = \begin{bmatrix} v^s_{ab} \\ \omega^s_{ab} \end{bmatrix} = \begin{bmatrix} -\dot{R}_{ab} R^T_{ab} p_{ab} + \dot{p}_{ab} \\ (\dot{R}_{ab} R^T_{ab})^\vee \end{bmatrix}$$

Body Velocity

- Point is expressed in terms of the **body frame**

$$V_{q_b}(t) = \underbrace{g^{-1}_{ab}(t)\,\dot{g}_{ab}(t)}_{\hat{V}^{b}_{ab}}\, q_b$$

$$\widehat{V}^b_{ab} := g^{-1}_{ab}(t)\dot{g}_{ab} = \begin{bmatrix} R^T_{ab}\dot{R}_{ab} & R^T_{ab}\dot{p}_{ab} \\ 0 & 0 \end{bmatrix} \qquad V^b_{ab} = \begin{bmatrix} v^b_{ab} \\ \omega^b_{ab} \end{bmatrix} = \begin{bmatrix} R^T_{ab}\dot{p}_{ab} \\ (R^T_{ab}\dot{R}_{ab})^\vee \end{bmatrix}$$

Interpreting Velocities as Twists

- Can break them apart into *v* and *w* components
- Calculate each one separately

Quantity	Interpretation
ω_{ab}^s	Angular velocity of B wrt frame A, viewed from A.
v_{ab}^s	Velocity of a (possible imaginary) point attached to B traveling through the origin of A wrt A, viewed from A.
ω_{ab}^b	Angular velocity of B wrt frame A, viewed from B.
v_{ab}^b	Velocity of origin of B wrt frame A, viewed from B.

Adjoints

What are they?

- Like a g matrix for twists!
- Change coordinate frames if we have a twist
- Because velocities are also twists, we can use adjoints to switch between spatial and body velocities

$$\widehat{\xi'} = g\widehat{\xi}g^{-1}$$

$$V^{s}_{AB} = Ad_{g_{ab}} \cdot V^{b}$$

$$\xi' = Ad_g\xi$$

Formulas

$$\underbrace{\begin{bmatrix} R_{ab} & \widehat{p}_{ab} R_{ab} \\ 0 & R_{ab} \end{bmatrix}}_{:= Ad_{g_{ab}}}$$

$$Ad_{g_{ab}}^{-1} = \begin{bmatrix} R_{ab}^T & -R_{ab}^T \widehat{p} \\ 0 & R_{ab}^T \end{bmatrix}$$

$$V_{ac}^s = V_{ab}^s + Ad_{g_{ab}} V_{bc}^s$$

$$V_{ac}^b = Ad_{g_{bc}^{-1}} V_{ab}^b + V_{bc}^b$$

$$Ad_{g_{ab}} \cdot Ad_{g_{bc}}$$

$$= Ad_{(g_{ab} g_{bc})}$$

Jacobians and Singularities

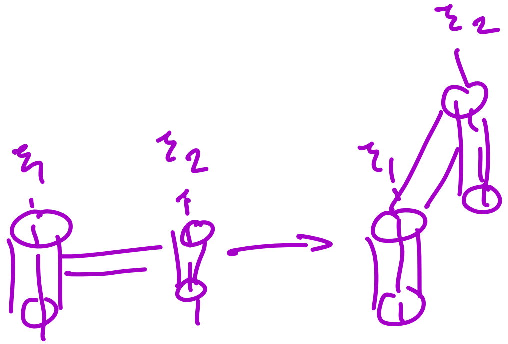
Motivation

- We want to get the velocity of our **end effector**
- However, our **sensors** give us the **velocities of our links**
- Jacobian allows us to go from **link velocities → end effector velocity**

$$V_{st}^{s} = J_{st}^{s}(\theta)\dot{\theta}$$

→ vector, how fast each link moves

Spatial Jacobian



- Gets us to the spatial velocity
- Columns of the Jacobian:
  - Twists of each of the links of the robot
  - In their *current* positions (i.e. not at 0 position, unlike FK)
  - Expressed in spatial coordinates
- Column represents derivative of end effector position wrt each of the links

Formulas

$$J_{st}^s(\theta) = \begin{bmatrix} (\frac{\partial g_{st}}{\partial \theta_1})^\vee & \cdots & (\frac{\partial g_{st}}{\partial \theta_n})^\vee \end{bmatrix}$$
$$= \begin{bmatrix} \xi_1 & \xi_2' & \cdots & \xi_n' \end{bmatrix}$$

$$\xi_i' = Ad_{(e^{\widehat{\xi}_1 \theta_1} \ldots e^{\widehat{\xi}_{i-1} \theta_{i-1}})} \xi_i$$

$$v_{q_s} = \widehat{V}_{st}^s q_s = (J_{st}^s(\theta)\dot{\theta})^\wedge q_s$$

Body Jacobian

- Analogous to ~~spatial Jacobian~~ body velocity
- Gets us the body velocity, instead of the spatial velocity
- Each of the twists are represented in the body frame instead

$$J_{st}^{b}(\theta) = \begin{bmatrix} \xi_1^{\dagger} & \xi_2^{\dagger} & \dots & \xi_n^{\dagger} \end{bmatrix}$$

$$\xi_i^{\dagger} = Ad^{-1}_{(e^{\widehat{\xi}_{i+1}\theta_{i+1}} \dots e^{\widehat{\xi}_n \theta_n} g_{st}(0))} \xi_i$$

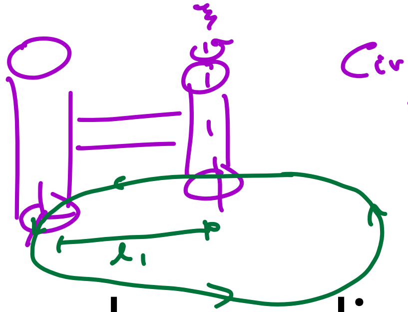
$$v_{q_b} = \widehat{V}_{st}^{b} q_b = (J_{st}^{b}(\theta)\dot{\theta})^{\wedge} q_b$$

Conversion

- Jacobians are composed of twists
- Can use the adjoint to move between them!
    - Adjoint is invertible, can go the other way as well

$$J_{st}^{s}(\theta) = Ad_{g_{st}(\theta)} J_{st}^{b}(\theta)$$

Finding the Jacobian



Circle method: performing a cross product $(-\omega \times q)$

$$-\omega \times q = \begin{bmatrix} l_1 \dot{\theta} \\ 0 \\ 0 \end{bmatrix}$$

$$\omega = \begin{bmatrix} 0 \\ 0 \\ \dot{\theta} \end{bmatrix}$$

- Can find the twists making up the columns directly by finding and applying adjoint transformation

- Alternatively, we can calculate the new positions of each of the *v* and *w* components that make up the twists

- Find new ω, ω'
- Find new q, q'

$$\xi = \begin{bmatrix} -\omega' \times q' \\ \omega' \end{bmatrix}$$

Singularities

$$V_{st}^{s} = J_{st}^{s}(\theta)\dot{\theta}$$

- Jacobian drops in rank
- We can't reach all of the velocities that we *should* be able to no matter what we set each of our link velocities to
- This is a **singular configuration**
- Would prefer to avoid being in it or near it
    - Can't achieve instantaneous motion in certain directions
    - Could require significant amounts of force in certain directions around that area

- Manipulability measure → product of singular values of Jacobian

$$\text{6 rows}\left\{\begin{bmatrix} \xi_1 & \cdots & \xi_7 \end{bmatrix}\right.$$

- Mess up error tracking

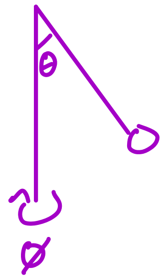
T

Dynamics

Forces!

- In real life, we’re trying to control our robot by applying some force to its joints
- Need to get the **dynamics** of our system
- The forces in each direction so that we know exactly what to apply to achieve our trajectory

Use Energy!



- Forces can be difficult
  - When there are multiple reference frames, particularly rotating ones, in play
  - End up with many complicated terms
  - Sometimes have several “imaginary” forces to balance equations’
- Energy is nice!
  - Scalars
  - Only depends on current state of the object
  - ~~Invariant to coordinate frame~~ - choose any one

    Doesn't matter which coord frame chosen (stay consistent)

Method

1. Choose state → Generalized coords q
2. Kinetic energy → Use $q, \dot{q}$
3. Potential energy
4. Lagrangian $L = T - V$
5. Equations of motion (convert to forces)

$$\underline{v} = \frac{d}{dt}\frac{dL}{d\dot{q}} - \frac{dL}{dq}$$

State

- Depends on the problem at hand
- Choose minimal representation needed *or* the representation that makes it easiest to determine what forces to apply
- Usually p, theta, or something similar

x, y

Kinetic Energy

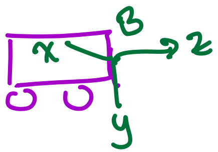
- Translational

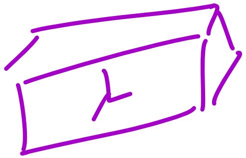
$$\frac{1}{2} m \|v\|_2^2$$

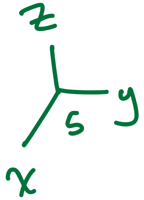
- Rotational

$$\frac{1}{2} \omega^T \mathcal{I} \omega$$

$$\frac{1}{2} I \dot{\theta}^2 \rightarrow \text{simpler}$$







$$V^b = \begin{bmatrix} 0 \\ 0 \\ 60 \end{bmatrix}$$

$$V^s = \begin{bmatrix} 0 \\ 60 \\ 0 \end{bmatrix}$$

in body frame

$$\rightarrow \quad T = \frac{1}{2} V_i^{b^T} \mathcal{M}_i V_i^b$$

Identity

$$\mathcal{M}_i = \begin{bmatrix} m I_3 & 0 \\ 0 & \mathcal{I} \end{bmatrix}$$

inertia matrix

Potential Energy

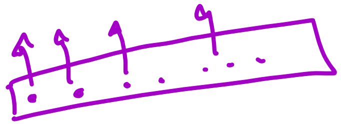
- Gravitational

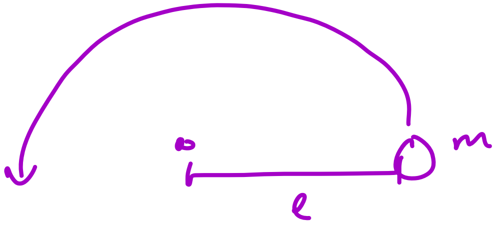
$mg\underbrace{h}$ → height from 0 position of our chosen coord frame

- Spring

$\frac{1}{2}kx^2$

Lagrangian





* scalar

$$L = T - V = \sum T_i - \sum V_i$$

$$\frac{1}{2} m (l\dot{\theta})^2$$

$$= \frac{1}{2} ml^2 \dot{\theta}^2$$

$= \text{Inertia}$

Equations of Motion

$$\frac{\partial L}{\partial \dot{q}_1}, \frac{\partial L}{\partial \dot{q}_2} \dots$$

$$\Upsilon = \frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q}$$

vector w/ same dims as generalized coords

Generalized forces

External - comes from motors, friction, etc.

Separation (Optional)

$$\Upsilon = \underbrace{M(q)}_{\text{Mass / inertia matrix}}\ddot{q} + \underbrace{C(q,\dot{q})}_{\text{Coriolis matrix}}\dot{q} + G(q)$$

Constant forces
ex. gravity

"Imaginary" forces from rotating coord frames

Control

Dynamical Systems

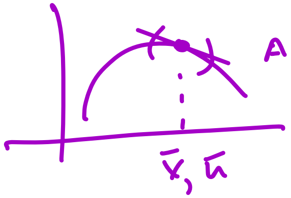
$$\dot{x} = f(x, u)$$

- Equations used to represent our system based on current state and input
- Often in the form of a differential equation
- Generated with knowledge of dynamics
  - State evolves as a result of forces
  - Input is the forces we add into the system

*Equil. Point: $\dot{x} = 0$

Linearization

$$\dot{x} = f(x,u) \xrightarrow{\text{convert to a linear system}} \dot{x} \approx A\Delta x + B\Delta u + f(\bar{x},\bar{u})$$



Approximate function as linear in a small area about the linearization point

$$\underbrace{f(x,u) \approx \underbrace{\left.\frac{df}{dx}\right|_{\bar{x},\bar{u}}}_{\text{Like our slope}} (x-\bar{x}) + \left.\frac{df}{du}\right|_{\bar{x},\bar{u}} (u-\bar{u})}_{Mx} + \underbrace{f(\bar{x},\bar{u})}_{B}$$

LTI Systems

$$\dot{x} = Ax + Bu$$

Controllability

$$\boxed{\dot{x} = Ax + Bu}$$

Assume WLOG $x_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$

$x_1 = Ax_0 + Bu_1$
$= Bu_1$
$x_2 = Ax_1 + Bu_2$
$= A(Bu_1) + Bu_2$

$x_3 = Ax_2 + Bu_3$
$= A(A(Bu_1) + Bu_2) + Bu_3$
$= A^2Bu_1 + ABu_2 + Bu_3$

we can go in span$\left(\begin{matrix} A^2B & AB \\ & B \end{matrix}\right)$

Can we get our system to behave the way we want?

Controllability Matrix: $Q = \begin{bmatrix} B & AB & \cdots & A^{n-1}B \end{bmatrix}$

n is the dim. of our state

Controllable within the span of Q

Q full rank: fully controllable

ex.

$$Q = \begin{bmatrix} 1 & 2 \\ 1 & 2 \end{bmatrix} \qquad \text{span}(Q) = \left\{ \begin{bmatrix} 1 \\ 1 \end{bmatrix} \right\}$$

we can get system to go from $\begin{bmatrix} 4 \\ 5 \end{bmatrix} \rightarrow \begin{bmatrix} 7 \\ 8 \end{bmatrix}$

we can't $\begin{bmatrix} 4 \\ 5 \end{bmatrix} \rightarrow \begin{bmatrix} 4 \\ 4 \end{bmatrix}$

$x_f - x_i$ is not in span(Q)

Stability

LTI system $\dot{x} = Ax + Bu$ stable:

All $Re(eig(A)) < 0$

Stabilizable if x is controllable in the direction of the eigenvectors w/ nonnegative eigenvalues

PID Control

- Used to error correct and can follow trajectory to some small extent
- Model-free control - only need to know error, not system equations

Main error correction

Fix steady-state error

Reduces oscillation

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \dot{e}(t)$$

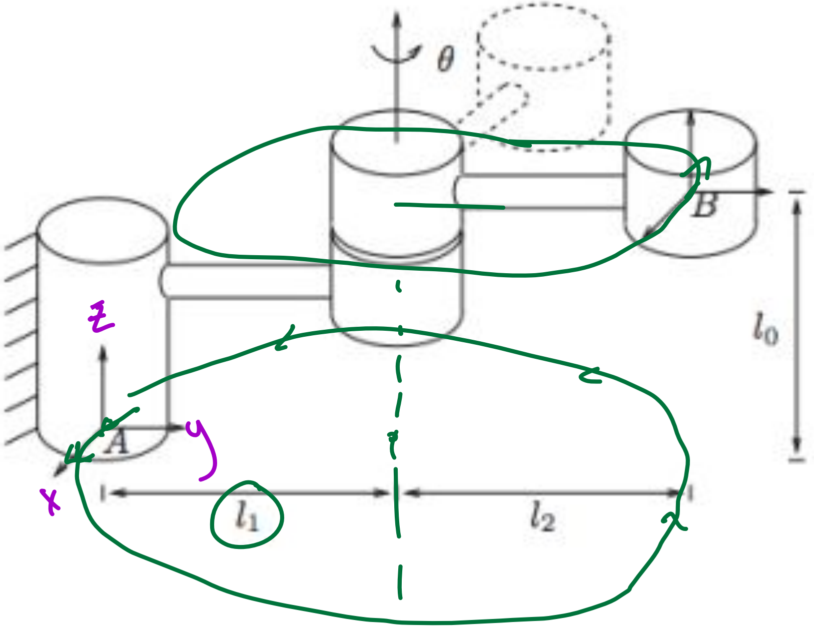
The Terms

- Proportional
  - Workhorse
  - Applies input that pulls state towards desired trajectory
- Derivative
  - Dampens proportional response
  - Prevents oscillation and overcorrection
  - Allows for convergence
- Integral
  - Corrects steady-state error because of constant forces like g

Feedback Linearization

- Incorporate error into the input term of a linear system
- Set up the equations so that error converges to 0

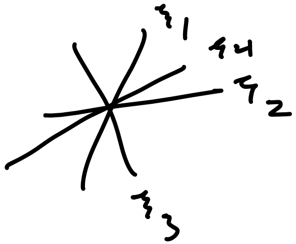
Calculate Spatial + Body Velocity



Spatial: $\begin{bmatrix} l_1\dot{\theta} \\ 0 \\ 0 \\ 0 \\ 0 \\ \dot{\theta} \end{bmatrix}$

Body: $\begin{bmatrix} -l_2\dot{\theta} \\ 0 \\ 0 \\ 0 \\ 0 \\ \dot{\theta} \end{bmatrix}$

Show that a manipulator with 4 intersecting revolute joints will have a singularity.



WLOG, assume these joints are @ origin

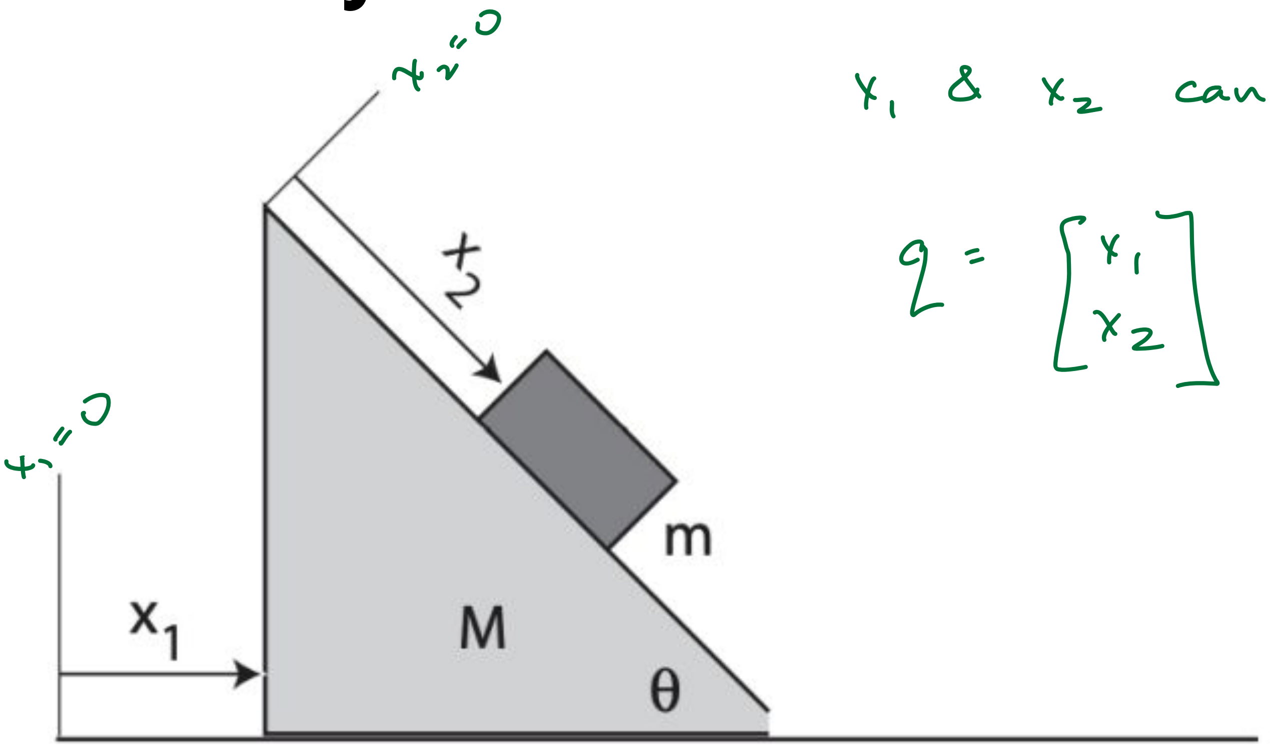
$\hookrightarrow -\omega \times q = -\omega \times 0 = 0$

$$J^s = \begin{bmatrix} \xi_1 & \xi_2' & \xi_3' & \xi_4' \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & 0 & 0 \\ \omega_1 & \omega_2' & \omega_3' & \omega_4' \end{bmatrix}}_{\text{Max rank of 3}}$$

We have 4 joints though

$\hookrightarrow$ Jacobian can't be rank 4

Calculate the dynamics



x_1 & x_2 can move

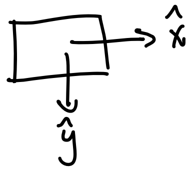
$$q = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Figure 1: Image sourced from http://www.dzre.com/alex/P441/lectures/lec_18.pdf

① $q = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

② KE Ramp moving $\rightarrow$ $\frac{1}{2} M \dot{x}_1^2$

Object moving $\rightarrow$ velocities from $\dot{x}_1$ & $\dot{x}_2$



$v_{\hat{x}} = \dot{x}_1 + \dot{x}_2 \cos\theta$

$v_{\hat{y}} = \dot{x}_2 \sin\theta$

$T_m = \frac{1}{2} m (\dot{x}_1 + \dot{x}_2 \cos\theta)^2 + \frac{1}{2} m (\dot{x}_2 \sin\theta)^2$

Expanding expression

$\sin^2 + \cos^2 = 1$

$= \frac{1}{2} m (\dot{x}_1^2 + 2\dot{x}_1\dot{x}_2 \cos\theta + \dot{x}_2^2)$

$T = T_M + T_m$

③ PE: box on ramp

$V_g = -mg(\underbrace{x_2 \sin\theta}_{\text{height}})$

④ Lagrangian

$L = T - V$

$= T_M + T_m - V$

$= \frac{1}{2} M \dot{x}_1^2 + \frac{1}{2} m (\dot{x}_1^2 + 2\dot{x}_1\dot{x}_2 \cos\theta + \dot{x}_2^2) + mg x_2 \sin\theta$

⑤ $\gamma = \frac{d}{dt}\frac{dL}{d\dot{q}} - \frac{dL}{dq}$

$\frac{dL}{d\dot{x}_1} = M\dot{x}_1 + m\dot{x}_1 + m\dot{x}_2 \cos\theta \rightarrow \frac{d}{dt} = M\ddot{x}_1 + m\ddot{x}_1 + m\ddot{x}_2 \cos\theta$

$\frac{dL}{d\dot{x}_2} = m\dot{x}_1 \cos\theta + m\dot{x}_2 \longrightarrow \frac{d}{dt} = m\ddot{x}_1 \cos\theta + m\ddot{x}_2$

$$\frac{dL}{dx_1} = 0 \qquad \frac{dL}{dx_2} = mg\sin\theta$$

$$\begin{bmatrix} \tau_{x_1} \\ \tau_{x_2} \end{bmatrix} = \begin{bmatrix} M\ddot{x}_1 + m\ddot{x}_1 + m\ddot{x}_2\cos\theta \\ m\ddot{x}_1\cos\theta + m\ddot{x}_2 + mg\sin\theta \end{bmatrix}$$

Say we have a system with x_1, x_2, u_1, u_2. Linearize the system at an equilibrium point of (0, 0), and analyze the stability and controllability:

$$\dot{x}_1 = 2x_1^2 + 3x_2 + u_1^2 = f_1$$
$$\dot{x}_2 = \sin x_1 + x_2 = f_2$$

$$f(x,u) - f(\bar{x},\bar{u}) = \delta f \approx \underbrace{\left.\frac{\partial f}{\partial x}\right|_{\substack{x=\bar{x}\\u=\bar{u}}}}(x-\bar{x}) + \left.\frac{\partial f}{\partial u}\right|_{\substack{x=\bar{x}\\u=\bar{u}}}(u-\bar{u})$$

$$\left.\frac{\partial f}{\partial x}\right|_{\substack{x=\bar{x}\\u=\bar{u}}} = \left.\begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix}\right|_{\substack{x=\bar{x}\\u=\bar{u}}} = \left.\begin{bmatrix} 4x_1 & 3 \\ \cos x_1 & 1 \end{bmatrix}\right|_{\substack{x=\bar{x}=0\\u=\bar{u}=0}} = \begin{bmatrix} 0 & 3 \\ 1 & 1 \end{bmatrix}$$

$$\frac{\partial f}{\partial u} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_1}{\partial u_2} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} \end{bmatrix}\Bigg|_{\substack{x=\bar{x}\\u=\bar{u}}} = \begin{bmatrix} 2u_1 & 0 \\ 0 & 0 \end{bmatrix}\Bigg|_{\substack{x=\bar{x}\\u=\bar{u}}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

$$\dot{x}_1 = 2x_1^2 + 3x_2 + 2u_1 = f_1$$
$$\dot{x}_2 = \sin x_1 + x_2 = f_2$$

$$f(x,u) - f(\bar{x},\bar{u}) = \delta f \approx \underbrace{\frac{\partial f}{\partial x}\Big|_{\substack{x=\bar{x}\\u=\bar{u}}}}(x-\bar{x}) + \frac{\partial f}{\partial u}\Big|_{\substack{x=\bar{x}\\u=\bar{u}}}(u-\bar{u})$$

$$\frac{\partial f}{\partial x}\Bigg|_{\substack{x=\bar{x}\\u=\bar{u}}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix}\Bigg|_{\substack{x=\bar{x}\\u=\bar{u}}} = \begin{bmatrix} 4x_1 & 3 \\ \cos x_1 & 1 \end{bmatrix}\Bigg|_{\substack{x=\bar{x}=0\\u=\bar{u}=0}} = \begin{bmatrix} 0 & 3 \\ 1 & 1 \end{bmatrix}$$

$$\frac{\partial f}{\partial u}\Bigg|_{\substack{x=\bar{x}\\u=\bar{u}}} = \begin{bmatrix} \frac{\partial f_1}{\partial u_1} & \frac{\partial f_2}{\partial u_2} \\ \frac{\partial f_2}{\partial u_1} & \frac{\partial f_2}{\partial u_2} \end{bmatrix} = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}\Bigg|_{\substack{x=\bar{x}=0\\u=\bar{u}=0}} = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}$$

FINAL LINEARIZATION:

$\bar{f}_1 = 0 + 0 + 0 \rightarrow$ Evaluated @ linearization point
$\bar{f}_2 = 0 + 0 \rightarrow$

$$\delta f = f - \bar{f} = f \approx \begin{bmatrix} 0 & 3 \\ 1 & 1 \end{bmatrix}(x-\bar{x}) + \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}(u-\bar{u})$$

$$\begin{bmatrix} x_1 - \overset{=0}{\bar{x}_1} \\ x_2 - \underset{=0}{\bar{x}_2} \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Stability

$$(0-\lambda)(1-\lambda) - (3)(1) = 0 \qquad \frac{1 \pm \sqrt{1-4(1)(-3)}}{2}$$

$$0 - \lambda + \lambda^2 - 3 = 0$$

$$\lambda = \frac{1 \pm \sqrt{13}}{2}$$

Unstable b/c we have a nonnegative eigenvalue

Controllability

$n = 2$ (we have 2 states)

$$Q = \begin{bmatrix} B & AB \end{bmatrix}$$

$$B = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}$$

$$AB = \begin{bmatrix} 0 & 3 \\ 1 & 1 \end{bmatrix}\begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 2 & 0 \end{bmatrix}$$

$$Q = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 \end{bmatrix}$$

Rank = 2

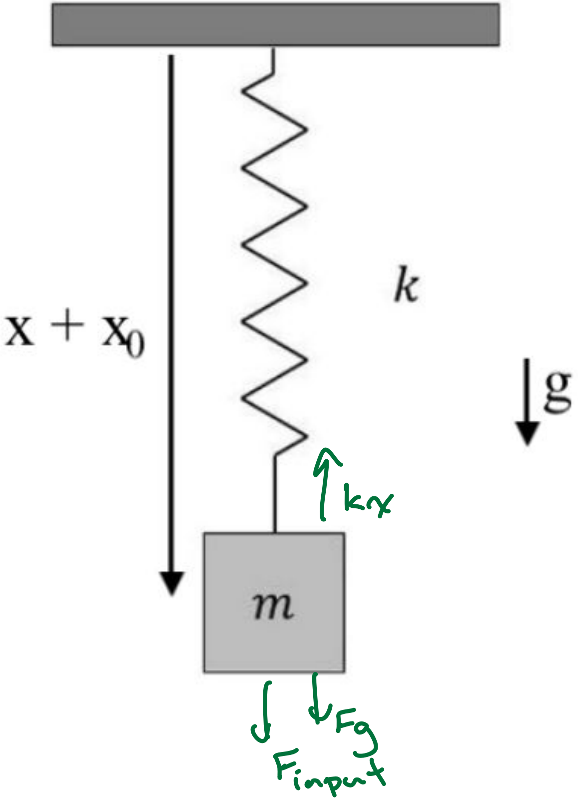
Fully rank $\Rightarrow$ Fully controllable !

Stabilizability

Can we stabilize in the direction of the eigenvector(s) w/ positive eigenvalue(s)? $\rightarrow$ can apply control input to keep system stable

$\longrightarrow$ YES b/c we are fully controllable (Q is full rank)

Calculate a control law that causes error to go to 0



$$m\ddot{x} = mg - kx + F_{input}$$

Solve for input force that achieves $\ddot{x}_d$

Add feedback term for error correction

$$F_{input} = m\ddot{x}_d - mg + kx + m(K_p e + K_d \dot{e})$$

$$m\ddot{x} = mg - kx + (m\ddot{x}_d - mg + kx + m(K_p e + K_d \dot{e}))$$

$$= m\ddot{x}_d - m\ddot{x} + m\left(K_p e + K_d \dot{e}\right)$$

$$0 = m\left(\ddot{e} + K_p e + K_d \dot{e}\right)$$

↓

error will converge to 0 ✓