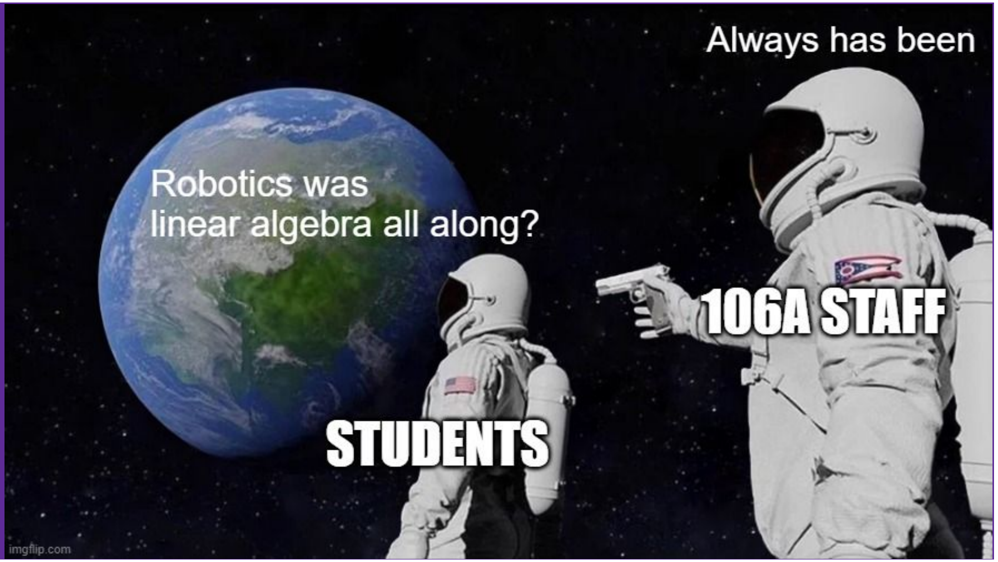
EECS/MechE/BioE C106A: Midterm 1 Review Session

Presented by Tarun Amarnath



Rigid Body Transformations

Rigid Body Transformations

- The forms of movement we discuss in this class
- 2 important qualities:
  - Length preserving

$$\| g(p) - g(q) \| = \| p - q \|$$

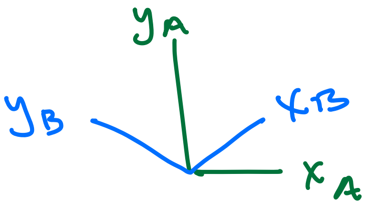
  - Orientation preserving

$$g(v \times w) = g(v) \times g(w)$$

Rotations

1st Rigid Body Transform: Rotations

- Let's say we have a world frame {A}
- There's a body attached to it {B} with its own orientation
- Rotation matrix represents the axes of {B} in terms of the axes of {A}

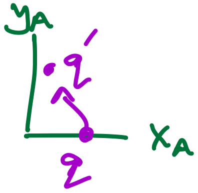


① $R_{AB} = \begin{bmatrix} x_{AB} & y_{AB} \end{bmatrix}$

x-axis of frame B in terms of Frame A

② $q' = R_{AB} \cdot q$

↳ Rotated version of q



③ $q_A = R_{AB} \cdot q_B$

↳ Represent a pt in B in A coords.

Rotations about World Axes

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} = e^{\hat{\boldsymbol{x}}\theta}$$

$$R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix} = e^{\hat{\boldsymbol{y}}\theta}$$

$$R_z(\theta) = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} = e^{\hat{\boldsymbol{z}}\theta}$$

Rodrigues' Formula

- Rotation about some **generic axis** *w* (not necessarily a world axis) by

$$R(w, \theta) = e^{\hat{\omega}\theta} = I + \hat{\omega}\sin\theta + \hat{\omega}^2(1 - \cos\theta)$$

$$\in SO(3)$$

$$w \times q = \hat{\omega} q$$

- Skew-symmetric matrix:

$$\hat{\omega} = \begin{bmatrix} 0 & -w_2 & w_1 \\ w_2 & 0 & -w_0 \\ -w_1 & w_0 & 0 \end{bmatrix} \in so(3)$$

Intrinsic (Euler) vs. Extrinsic (RPY) Rotations

- Intrinsic - rotate based on axes of **body** frame
  - Write this out from ~~right to left~~ left to right
- Extrinsic - rotate based on **world** axes
  - Write this out from ~~left to right~~ right to left

- **Equivalent**, depending on how you read a composition

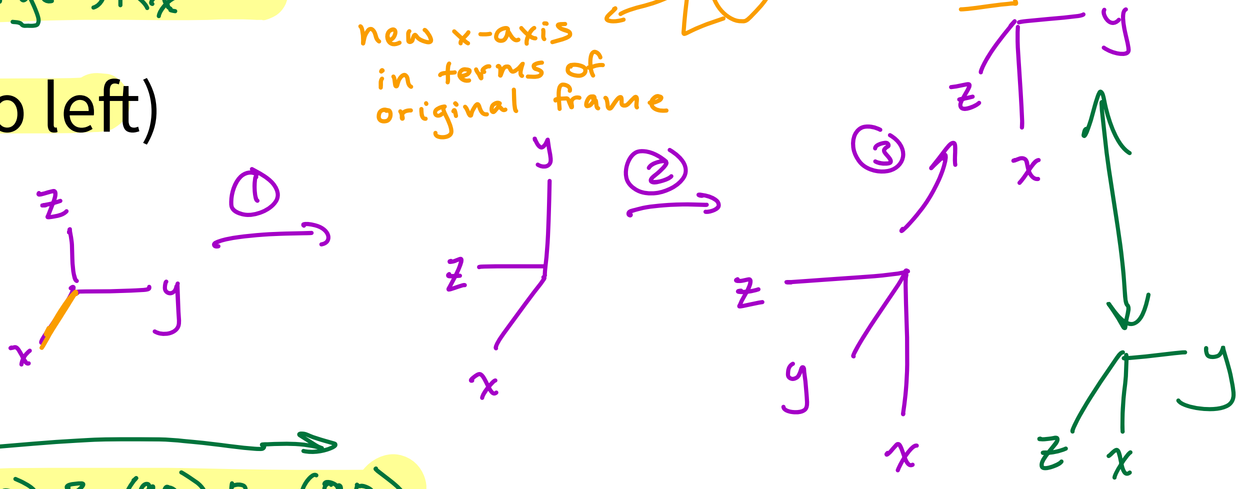
Example

$$R_{AB} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix}$$

new x-axis in terms of original frame

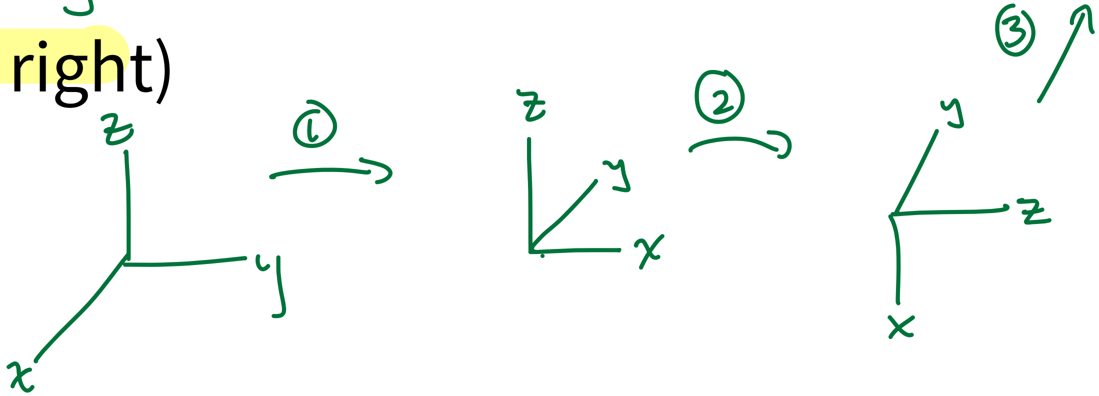
→ Extrinsic $R_z(90) R_y(90) R_x(90)$

- RPY Rotation (right to left)
  - 1 x-axis by 90°
  - 2 y-axis by 90°
  - 3 z-axis by 90°



→ Intrinsic $R_z(90) R_y(90) R_x(90)$

- Euler Rotation (left to right)
  - 1 z-axis by 90°
  - 2 *New* y-axis by 90°
  - 3 *New* x-axis by 90°



Special Orthogonal Matrices

- 3D rotation matrices fall into the SO(3) **group**

Definition:

$$SO(3) = \{R \in \mathbb{R}^{3\times3} \,|\, R^T R = I, \det R = 1\}$$

and

$$SO(n) = \{R \in \mathbb{R}^{n\times n} \,|\, R^T R = I, \det R = 1\}$$

- Orthogonal matrices that follow the right-hand rule

$$R^T = R^{-1}$$

Groups

$g_1, g_2 \in G$

- Multiplication maps into itself

$R_1 \cdot R_2 \rightarrow$ rotation matrix

$g_1 \cdot g_2 \in G$

- Identity operative

Unique $i \in G$: $g \cdot i = i \cdot g = g \quad \forall g \in G$

- Inverse

$g \cdot g^{-1} = g^{-1} \cdot g = i$

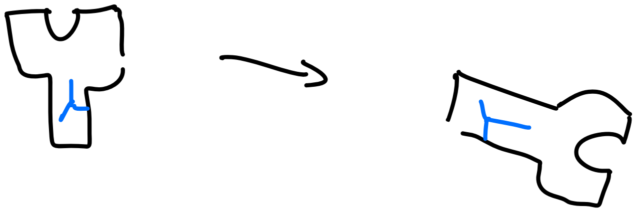
- Associativity

$g_1 (g_2 \cdot g_3) = (g_1 \cdot g_2) \cdot g_3$

Rotation *and* Translation

General Rigid Body Transformations

- **Goal:** express rotation and translation
- (points translate, vectors simply rotate)



$$g = R \cdot x + p$$

Rotate & translate

- Homogeneous coordinates:

$$p = \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} \qquad \vec{v} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ 0 \end{bmatrix}$$

Points have a 1

vecs have a 0

Homogeneous Transformation Matrices

- Compact representation
- Both rotation and translation included

$$g = \begin{bmatrix} R^{\in \mathbb{R}^{3\times 3}} & t^{\in \mathbb{R}^{3\times 1}} \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times 4}$$

$$g \cdot p = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ 1 \end{bmatrix} = \begin{bmatrix} Rp + t \\ 1 \end{bmatrix} \checkmark$$

$$g \cdot v = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ 0 \end{bmatrix} = Rv + 0 = \begin{bmatrix} Rv \\ 0 \end{bmatrix}$$

- Can stack and invert

$$g_{AC} = g_{AB} \cdot g_{BC}$$

$$g_{AC} = g_{CA}^{-1}$$

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

SE(3) is a group

$$g = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \in SE(3)$$

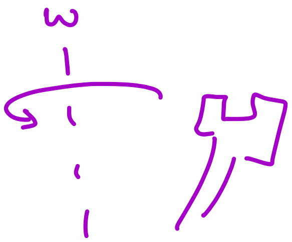
$$SE(3) = \left\{ \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{4\times4} \;\middle|\; t \in \mathbb{R}^3,\ R \in SO(3) \right\}$$

Exponential Coordinates

Exponential Coordinates

- **Goal:** Create rotation and homogeneous transformation matrices as a *function of time*

Know how our robot moves



Find $R(\omega, \theta)$

known

unknown

- Comes from solving a differential equation → $\dot{p}(t) = A\,p(t)$
- We only need information about **how** the object moves (time is a parameter that's plugged in)

$p(t) = e^{At}\,p(0)$

Exp. Coordinates for Rotation Matrices

- **Axis of rotation** creates a *rotation matrix*

$\|\omega\| = 1 : \quad \theta = t$

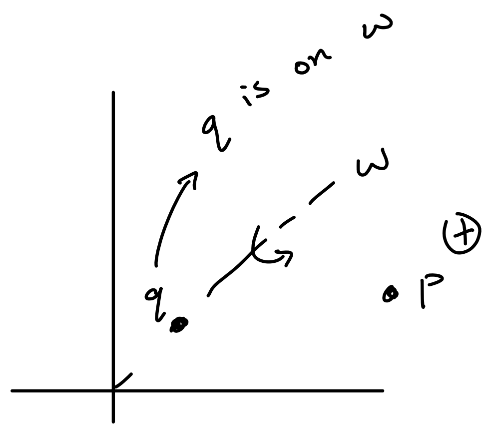
$\|\omega\| \neq 1 : \quad \theta \neq t$

Angular velocity

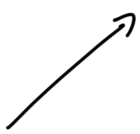
$$R(t) = \underbrace{e^{\hat{\omega}\theta}}_{\text{Use Rodrigues Formula}} R(0)$$

★ Use Rodrigues Formula

Exp. coords $= (\omega, \theta)$



$\vec{v} = p - o$



Linear velocity of p rotating about ω

$$\dot{p} = \omega \times (p - q)$$

$$= \omega \times p - \omega \times q$$

$$= \hat{\omega} p - \omega \times q$$

Write in homogeneous coords

$$\underbrace{\begin{bmatrix} \dot{p} \\ 0 \end{bmatrix}}_{\text{velocity vector}} = \underbrace{\begin{bmatrix} \hat{\omega} & -\omega \times q \\ 0 & 0 \end{bmatrix}}_{\hat{\xi}} \begin{bmatrix} p \\ 1 \end{bmatrix}$$

→ In the form of $\dot{p} = A \cdot p$

$$p = e^{At} \, p(0)$$

Twist : $\hat{\xi} \longrightarrow \hat{\xi} = \begin{bmatrix} \hat{\omega} & v \\ 0 & 0 \end{bmatrix}$

Twist coordinate: ξ

Exp. Coords for General Rigid Body Motion

- **Twists** generate *homogeneous transformation matrices*

$$\xi = \begin{bmatrix} v \\ \omega \end{bmatrix} \in \mathbb{R}^{6\times 1} \qquad g = e^{\hat{\xi}\theta} \qquad p(t) = e^{\hat{\xi}\theta} p(0)$$

- Pure rotation:

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

q: point on axis of rotation

- Pure translation:

$$\xi = \begin{bmatrix} v \\ 0 \end{bmatrix}$$

- Screw Motion:

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

h: Ratio of translation : rotation

Exp. coords: (ξ, θ)

Reference Formulas

- Rotation and Translation

Rodrigues'

$$e^{\hat{\boldsymbol{\xi}}\theta} = \begin{bmatrix} e^{\hat{\boldsymbol{\omega}}\theta} & \left(\mathbb{I}_3 - e^{\hat{\boldsymbol{\omega}}\theta}\right)\left(\underline{\boldsymbol{\omega}} \times \boldsymbol{v}\right) + \boldsymbol{\omega}\boldsymbol{\omega}^T\boldsymbol{v}\theta \\ \mathbf{0} & 1 \end{bmatrix}$$

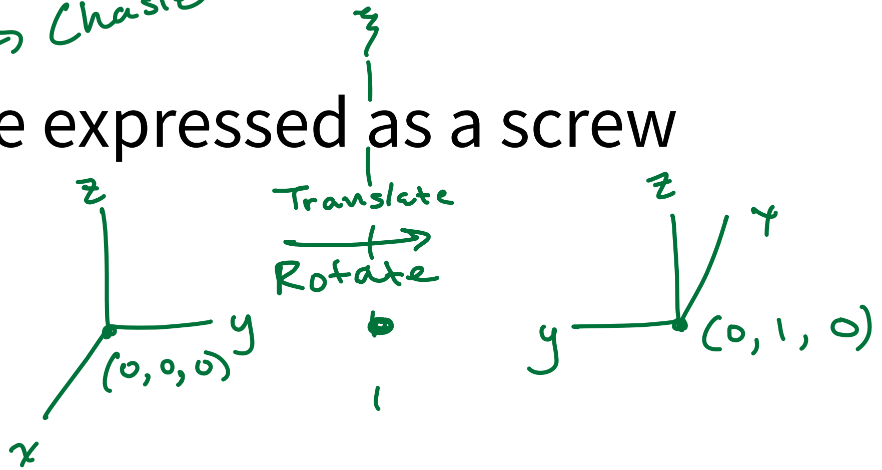
- Pure Translation

$$e^{\hat{\boldsymbol{\xi}}\theta} = \begin{bmatrix} \mathbb{I}_3 & \boldsymbol{v}\theta \\ \mathbf{0} & 1 \end{bmatrix}$$

More on Screw Motion

→ Chasles' Theorem

- Any rotation + translation can be expressed as a screw about a single axis
- Axis *w*
- Magnitude of rotation
- Pitch *h* (ratio of translation : rotation)
  - h = 0: pure rotation
  - h = infinite: pure translation



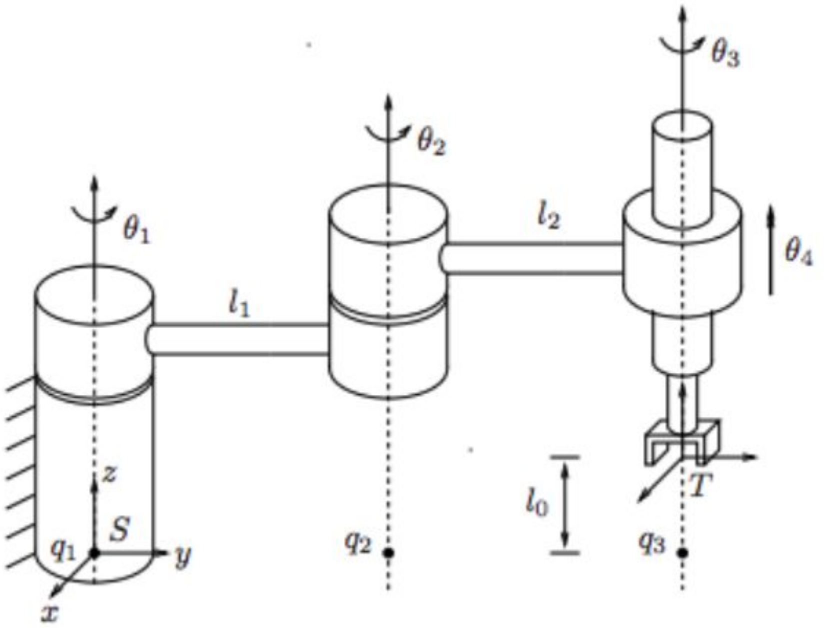
$$\xi = \begin{bmatrix} -\omega \times q \\ \omega \end{bmatrix}$$

$$\omega = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \qquad q = \begin{bmatrix} 0 \\ 0.5 \\ 0 \end{bmatrix}$$

Forward Kinematics

Multi-Link Arms

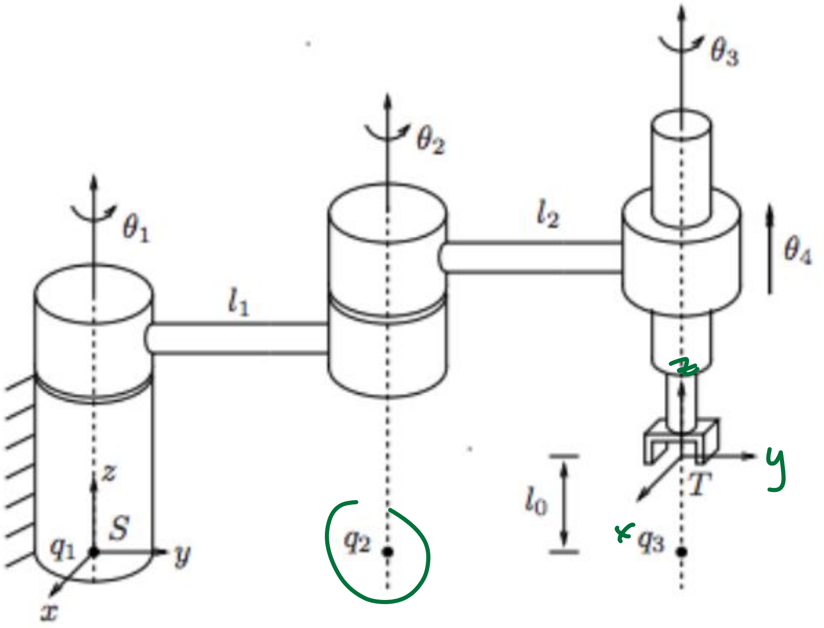
- **Goal:** Find the location of the tool after a multi-joint robot arm has moved around



Composition of Twists

1. Find twists of each joint in the reference configuration
   - All joint angles are 0
   - All vectors expressed in the *world frame*
2. Find starting position of tool
3. Put it together in exponential coordinates

$$e^{\hat{\xi}_1\theta_1} e^{\hat{\xi}_2\theta_2} \cdots e^{\hat{\xi}_n\theta_n} \cdot g_{st}(0) = g_{st}(\theta_1 \dots \theta_n)$$



$0, l_1, 0$

ξ_1: $\omega = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ $\quad q = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$

$$\xi_1 = \begin{bmatrix} -\omega \times q \\ \omega \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

ξ_2: $\omega = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ $\quad q = \begin{bmatrix} 0 \\ l_1 \\ 0 \end{bmatrix}$

$$\xi_2 = \begin{bmatrix} -\omega \times q \\ \omega \end{bmatrix} = \begin{bmatrix} l_1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\xi_3 = \begin{bmatrix} l_1 + l_2 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T$$

$$\xi_4 = \begin{bmatrix} v \\ 0 \end{bmatrix} \qquad v = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

$$\xi_4 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Identity rotation

$$g_{st}(0) = \begin{bmatrix} I & \begin{pmatrix} 0 \\ l_1 + l_2 \\ l_0 \end{pmatrix} \\ 0 & 1 \end{bmatrix}$$

$$e^{\hat{\xi}_1 \theta_1} \cdots e^{\hat{\xi}_4 \theta_4} g_{st}(0) = g_{st}(\theta_1 \ldots \theta_4)$$

Notes

- **Order matters**: joints on the right cannot affect the position of joints on the left $\rightarrow$ write $\theta_1 \rightarrow \theta_4$
- Forward kinematics (calculated using the world frame) will deliver the **same result** as composing homogeneous transformation matrices (see Discussion 3, Problem 3)

Inverse Kinematics

The Goal

- How do we move our robot's joints to reach a desired configuration?
- Given:
  - Where we want to go $= g_d$
  - Details about the robot
    - Twists $= \xi_1 \cdots \xi_n$
    - Starting Configuration $\rightarrow g_{st}(0)$
- Desired: $\theta_1 \cdots \theta_n$

Padan-Kahan Subproblems

- We have proven the solutions to some basic inverse kinematics problems → the subproblems
- Can we reduce our much more complicated robotics problem down to these?

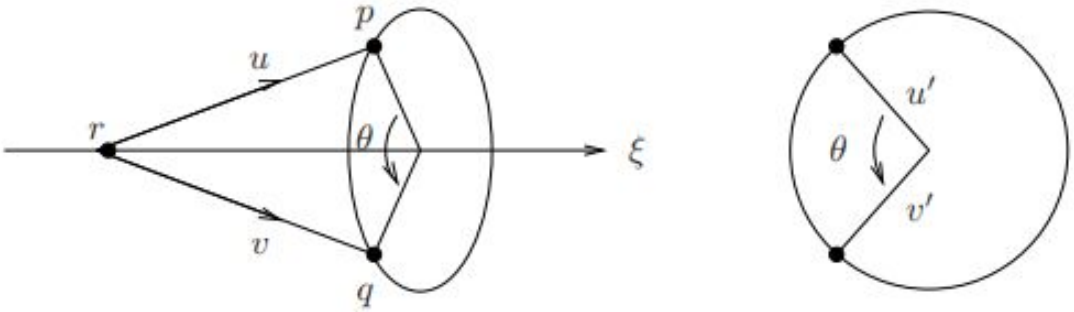
★ Only revolute joints

Subproblem 1

- Rotation about some fixed axis
  - We specify (p, q)
  - Find theta
- <= 1 solution

$e^{\hat{\xi}\theta} p = q$

≤ 1 soln

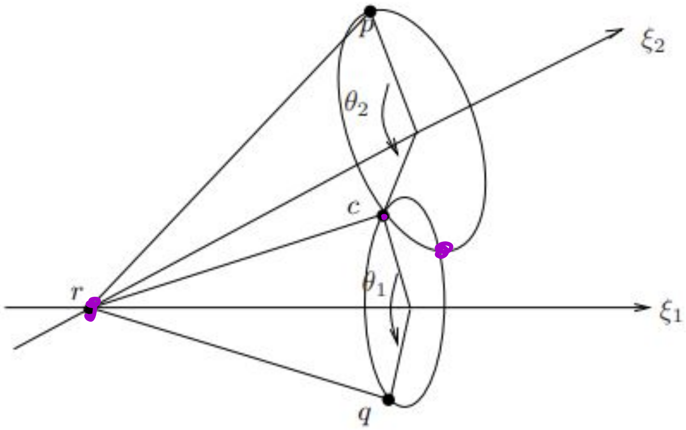


Subproblem 2

- Rotate about 2 intersecting axes
  - We specify (p, q)
  - Find theta
- <= 2 solutions

$$e^{\hat{\xi}_1 \theta_1} e^{\hat{\xi}_2 \theta_2} p = q$$

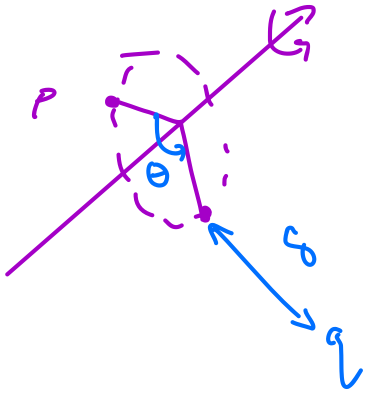
$\leq$ 2 sols



Subproblem 3

* Only revolute joints

- Move to a specified distance from another point
  - Specify (p, q, delta)
  - Find theta
- <= 2 solutions



$$\| e^{\hat{\xi}\theta} \cdot p - q \| = \delta$$

≤ 2 solutions

Prismatic joint solution can take this form, but need to use geometry to solve

Extrapolating

- We know we can solve these simple IK problems to find theta
- But what about the complex robot arm?

$$e^{\hat{\xi}_1 \theta_1} \cdots e^{\hat{\xi}_6 \theta_6} g_{st}(0) = g_d$$

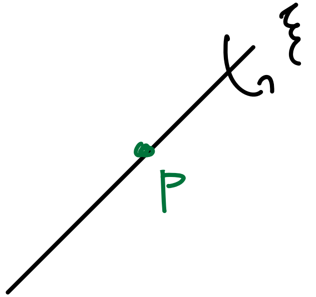
- Solution: repeatedly apply subproblems using convenient points
- Some tricks to help us out

Rearrange

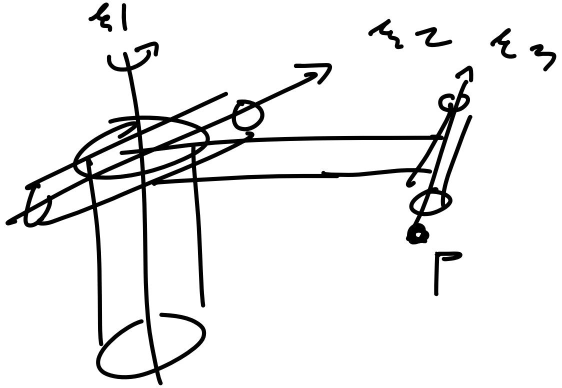
$$\downarrow$$

$$e^{\hat{\xi}_1 \theta_1} \cdots e^{\hat{\xi}_6 \theta_6} = g_d \, g_{st}^{-1}(0) =: g_1$$

Trick #1: Eliminate on the RHS



$$e^{\hat{\xi}\theta} p = p$$



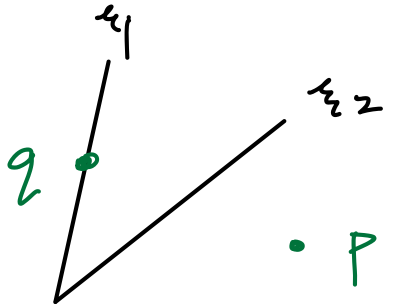
$$e^{\hat{\xi}_1\theta_1} e^{\hat{\xi}_2\theta_2} e^{\hat{\xi}_3\theta_3} p = g \cdot p$$

- p is on ξ_3

$$e^{\hat{\xi}_1\theta_1} e^{\hat{\xi}_2\theta_2} p = g p$$

$\longrightarrow$ Subproblem 2

Trick #2: Eliminate on the LHS Using Norms



$$e^{\hat{\xi}_1\theta_1} e^{\hat{\xi}_2\theta_2} = g$$

- q on ξ_1
- p <u>not</u> on ξ_2

ξ_1 doesn't affect q

$$\| e^{\hat{\xi}_1\theta_1} e^{\hat{\xi}_2\theta_2} p - q \| = \| gp - q \|$$

$$\| e^{\hat{\xi}_1\theta} (e^{\hat{\xi}_2\theta_2} p - q) \| = \| gp - q \|$$

$e^{\hat{\xi}_1\theta_1}$ won't affect norm

$$\| e^{\hat{\xi}_2\theta_2} p - q \| = \underline{\| gp - q \|} \rightarrow \text{known} = \delta$$

Use SP3

Trick #3: Prismatic Joints

Get into δ form:

$$\| e^{\hat{\xi}\theta} p - q \| = \delta$$

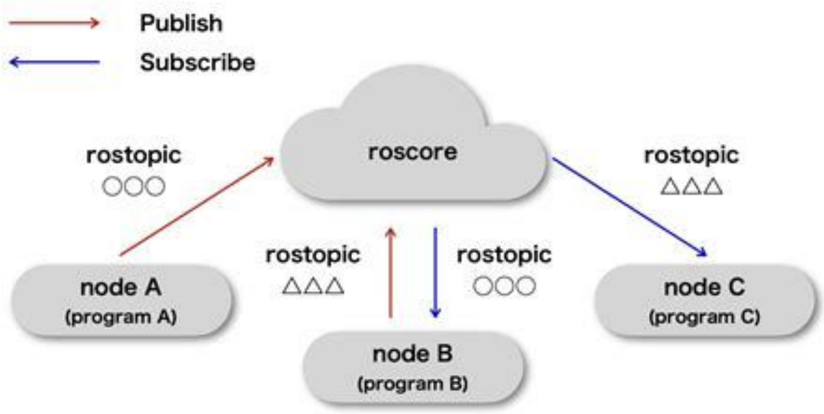
$$\delta = \underbrace{l_0}_{\text{original dist.}} + \underbrace{\theta}_{\text{moved the prismatic joint}}$$

Lab Content

ROS Overview

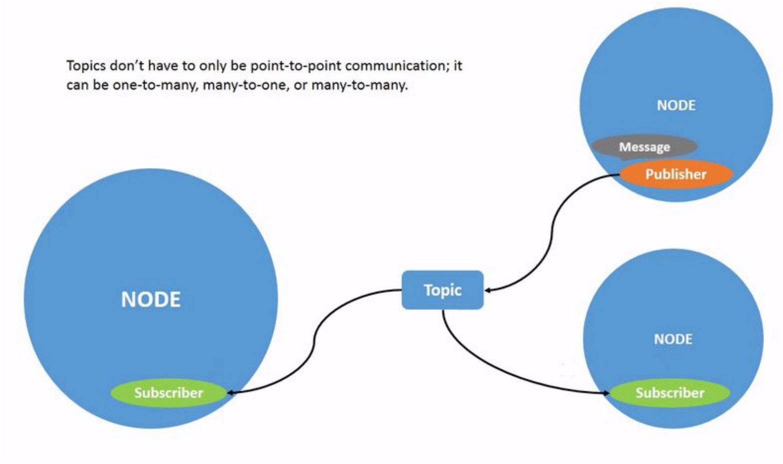
```
ros_workspaces
  lab1
    build
    devel
      setup.bash
    src
      CMakeLists.txt
      foo
        CMakeLists.txt
        package.xml
      bar
        CMakeLists.txt
        package.xml
```

- Meta-operating system for a robot
- Hardware abstraction and message passing between nodes
- Things to know:
  - Computation graph
  - How the file system looks
  - Workspace vs Package
    - Where do we place them
  - Where do packages store msg/srv definitions
  - Basic commands (ex. catkin_make)



Publishers and Subscribers

- Publishers post messages on topics, subscribers read topic
- Asynchronous, fast, One-to-many
- Things to know:
    - ROS message definitions
    - How to view various information (rostopic list/echo/etc.)
    - Python code
        - You won’t have to write your own,
          but fill-in-the-blank is definitely possible



My_msg_defn.msg

```
<< data_type1 >>  << name_1 >>
<< data_type2 >>  << name_2 >>
<< data_type3 >>  << name_3 >>
...
```

Basic Pub Sub Code

Publisher

```python
def my_pub():
    #                         Topic Name   Msg Type    Queue Size
    pub = rospy.Publisher('/my_topic', String, queue_size=10)
    r = rospy.Rate(10) # Run node at 10hz

    # Loop until the node is killed with Ctrl-C
    while not rospy.is_shutdown():
        pub_msg = String()
        pub_msg.message = "Hello World!"
        pub.publish(pub_msg)
        r.sleep()

if __name__ == '__main__':
    rospy.init_node('my_pub', anonymous=True)
    my_pub()
```

Subscriber

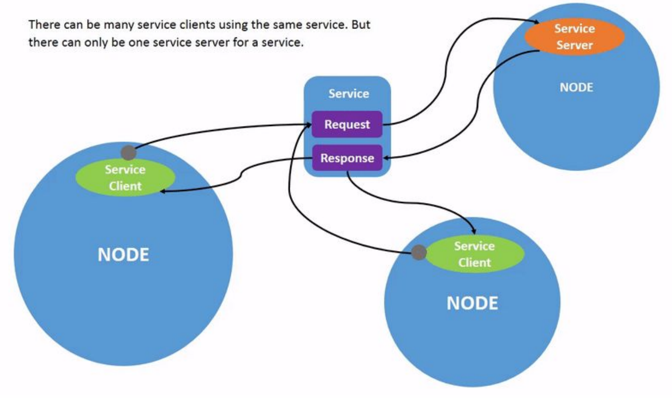
```python
def my_callback(recieved_message):
    print((recieved_message.message))

def my_sub():
    #                 Topic Name   Msg Type   Callback func
    rospy.Subscriber("/my_topic", String, my_callback)
    # Prevents node from exiting until this program is stopped
    rospy.spin()

if __name__ == '__main__':
    rospy.init_node('my_sub', anonymous=True)
    my_sub()
```

Services, Requests, and Clients

- Synchronous and One-to-one
- Server offers service
- Client calls that service
- Like synchronous function calls
    - Image you, a client, visits google.com
    - Client requests, waits for response
    - Server fulfills request and responds
    - Client processes response



More ROS Stuff to Know

- Things to know
  - General Python file structure
  - How to run Python files (rosrun pkg file)
  - Make requests from command line (rosservice call …)


SNL/UNIVERSAL TELEVISION
ANY QUESTIONS?