Configuration Space (Pre Lecture)

Dr. Neil T. Dantam

CSCI-534, Colorado School of Mines

Spring 2020

Introduction

Configuration Space

- Defines positions of all points in the system
- The space which we search in motion planning

Outcomes

- Know definitions of configuration space
- Identify the degrees-of-freedom (DoF) of a robot
- Compute transforms for various joint types
- Relate:
  - Physical mechanism
  - Joints
  - Frames and transforms
  - Configurations

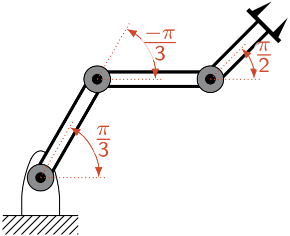
MINES

Configuration Space

Definition: Configuration

A specification for the position of all points in the system (robot).

Typically a real vector: $q \in \mathbb{R}^n$.



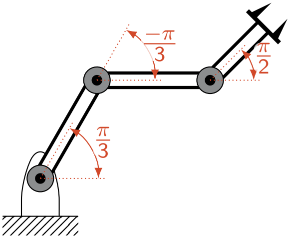
$$q = \begin{bmatrix} \frac{\pi}{3} \\ \frac{-\pi}{3} \\ \frac{\pi}{2} \end{bmatrix}$$

Degrees of Freedom

Definition: Degrees of Freedom

The smallest number of real-valued coordinates necessary to represent a configuration.

Typically a natural number: $n \in \mathbb{N}$



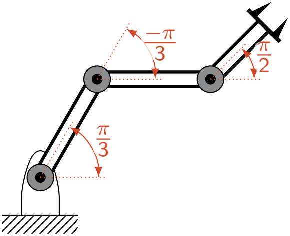
$n = 3$

Configuration Space

Definition: Configuration Space

The n-dimensional space containing all possible configurations of the robot.

Typically a real vector (sub)space: $\mathcal{Q} \subseteq \mathbb{R}^n$



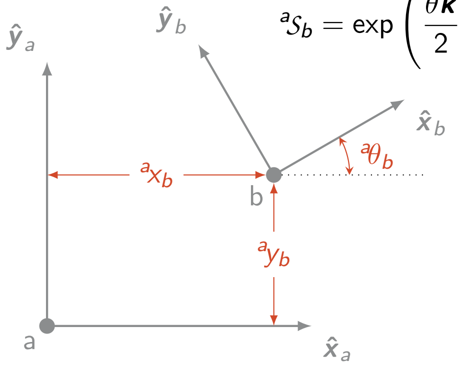
$\mathcal{Q} \subseteq \mathbb{R}^3$

Motion Planning: Search in the configuration space

Outline

Planar Rigid Bodies

$$
{}^{a}\mathcal{S}_{b} = \underbrace{\exp\left(\frac{\theta \hat{\mathbf{k}}}{2}\right)}_{\text{rotation}} + \overbrace{\frac{1}{2}\underbrace{\left({}^{a}x_{b}\hat{\boldsymbol{\imath}} + {}^{a}y_{b}\hat{\boldsymbol{\jmath}}\right)}_{\text{vector}} \otimes \underbrace{\exp\left(\frac{\theta \hat{\mathbf{k}}}{2}\right)}_{\text{rotation}} \varepsilon}^{\text{translation part}}
$$

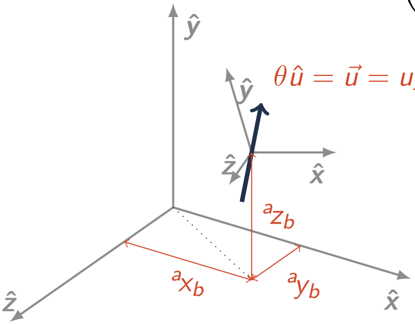


Three parameters: x, y, θ

3D Rigid Bodies

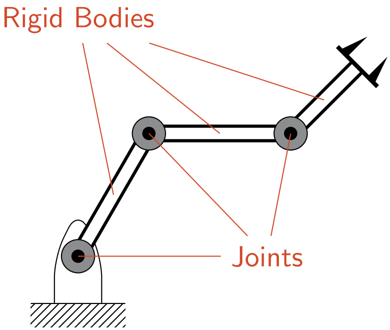
$$\overbrace{{}^{a}\mathcal{S}_b = \exp\left(\frac{\vec{u}}{2}\right)}^{\text{rotation}} + \overbrace{\frac{1}{2}\underbrace{\left({}^{a}x_b\hat{\imath} + {}^{a}y_b\hat{\jmath} + {}^{a}z_b\hat{k}\right)}_{\text{vector } {}^{a}v_b} \otimes \underbrace{\exp\left(\frac{\vec{u}}{2}\right)}_{\text{rotation}}}^{\text{translation part}}$$

$$\theta\hat{u} = \vec{u} = u_x\hat{\imath} + u_x\hat{\jmath} + u_x\hat{k}$$



Six parameters: (x, y, z) and (u_x, u_y, u_z)

Robot DoF



Joint Types

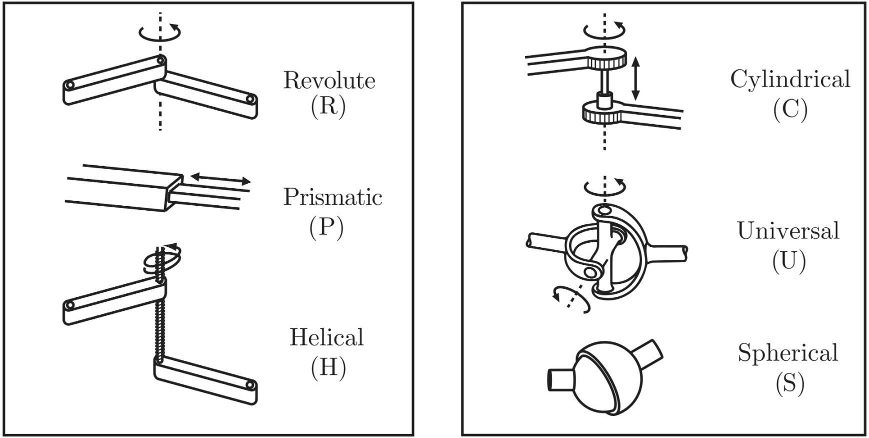


Figure 2.3: Typical robot joints.

(Lynch and Park. Modern Robotics.)

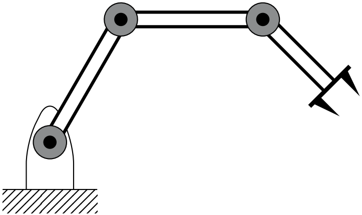
Joint Constraints DoF

Joint Type	Constraints (2D)	Constraints (3D)	Net DoF
Revolute	2	5	1
Prismatic	2	5	1
Helical	N/A	5	1
Cylindrical	N/A	4	2
Universal	N/A	4	2
Spherical	N/A	3	3

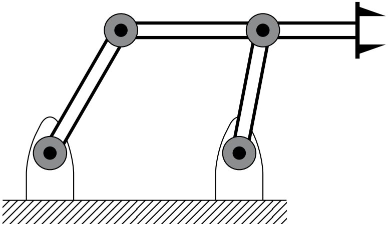
Open vs. Closed Chains

Serial vs. Parallel Manipulators

Serial



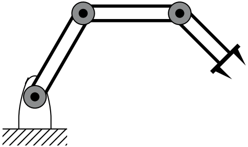
Parallel



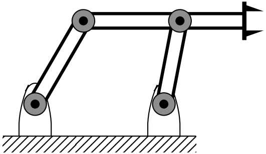
Grübler's Formula

Mechanism DoF

$$\text{dof} = \overbrace{m(\underbrace{N}_{\text{links}} - 1)}^{\text{rigid body dofs}} - \overbrace{\sum_{i=1}^{J} c_i}^{\text{joint constraints}}$$



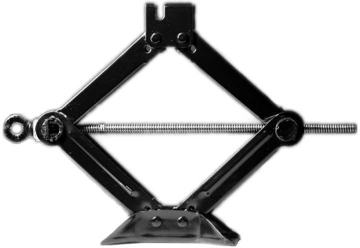
- Planar: $m = 3$
- 4 links: $N = 4$
- $\text{dof} = 3(4-1) - (2+2+2)$
  $= 3$

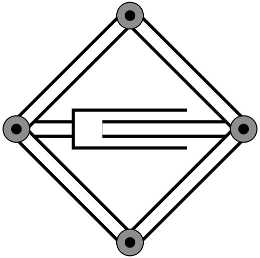


- Planar: $m = 3$
- 4 links: $N = 4$
- $\text{dof} = 3(4-1) - (2+2+2+2)$
  $= 1$

Exercise: Scissor Jack Mechanism

Simplified Planar Model





Hint: look for constraints between pairs of links

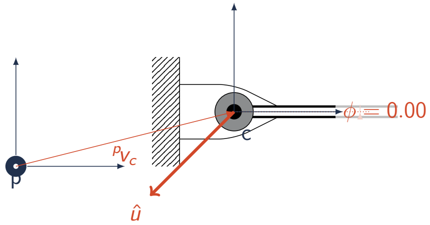
- Links:
- Joints:

Exercise: Human Arm

Kinematic Model

Outline

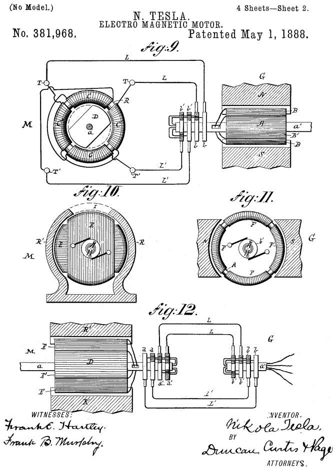
Revolute Joint Animation

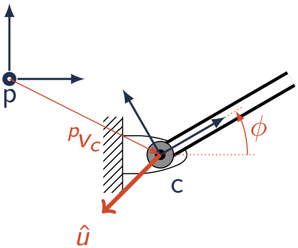


$${}^{p}S_c = \exp\left(\frac{0.00}{2}\hat{u}\right) + \frac{1}{2}\,{}^{p}v_c \otimes \exp\left(\frac{0.00}{2}\hat{u}\right)\varepsilon$$

Revolute Joints

Rotating Motion

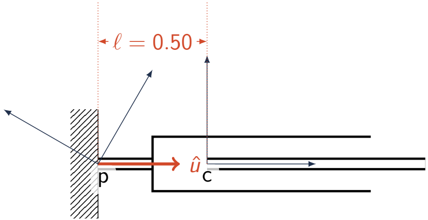




- $^{p}v_{c}$: Translation from parent p to child c (fixed)
- $\hat{u}$: Axis of rotation (fixed)
- ϕ: Rotation angle (varying)
- $${}^{p}S_{c}(\phi) = \underbrace{\exp\left(\frac{\phi}{2}\hat{u}\right)}_{\text{rotation}} + \underbrace{\frac{1}{2}\,{}^{p}v_{c} \otimes \exp\left(\frac{\phi}{2}\hat{u}\right)\varepsilon}_{\text{translation}}$$

MINES

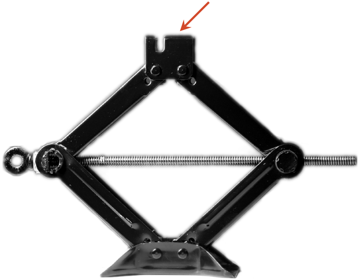
Prismatic Joint Animation

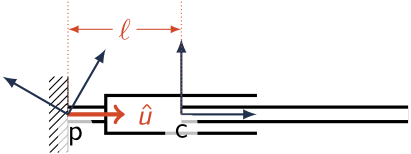


$$ {}^{p}S_{c} = {}^{p}\hbar_{c} + \tfrac{1}{2}(\ 0.50\ )\hat{u} \otimes {}^{p}\hbar_{c}\varepsilon $$

Prismatic Joints

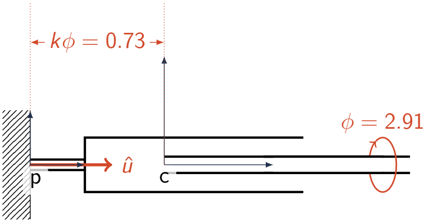
Linear / Sliding Motion





- ${}^{p}h_c$: Rotation from parent p to child c (fixed)
- $\hat{u}$: Axis of translation (fixed)
- ℓ: Translation length (varying)
- $${}^{p}\mathcal{S}_c(\ell) = \underbrace{{}^{p}h_c}_{\text{rotation}} + \underbrace{\frac{1}{2}\ell\hat{u}\otimes{}^{p}h_c\varepsilon}_{\text{translation}}$$

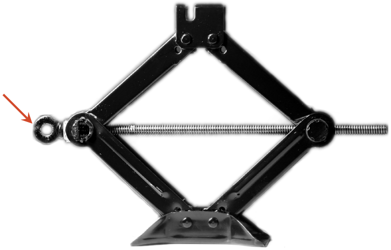
Helical Joint Animation

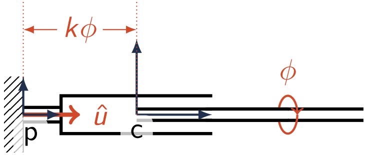


$$^{p}S_c = \exp\left(\frac{2.91}{2}\hat{u}\right) + \frac{1}{2}(0.73)\hat{u} \otimes \exp\left(\frac{2.91}{2}\hat{u}\right)$$

Helical Joints

Coupled Rotation and Linear Motion





- k: thread pitch
- $\hat{u}$: Axis (fixed)
- ϕ: Rotation angle (varying)
- $$ {}^{p}\mathcal{S}_{c}(\phi) = \underbrace{\exp\left(\frac{\phi}{2}\hat{u}\right)}_{\text{rotation}} + \underbrace{\frac{1}{2}(k\phi\hat{u}) \otimes \exp\left(\frac{\phi}{2}\hat{u}\right)\varepsilon}_{\text{translation}} $$

Multi-DoF Joints

Cylindrical: Revolute $\otimes$ Prismatic

$$^{p}\mathcal{S}_c\left(\ell, \phi\right) = {^{p}\mathcal{S}_{c1}}\left(\ell\right) \otimes {^{c1}\mathcal{S}_c}\left(\phi\right)$$

Universal: Revolute $\otimes$ Revolute

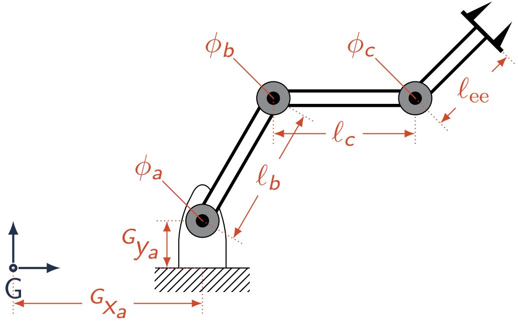
$$^{p}\mathcal{S}_c\left(\phi_0, \phi_1\right) = {^{p}\mathcal{S}_{c0}}\left(\phi_0\right) \otimes {^{c0}\mathcal{S}_c}\left(\phi_1\right)$$

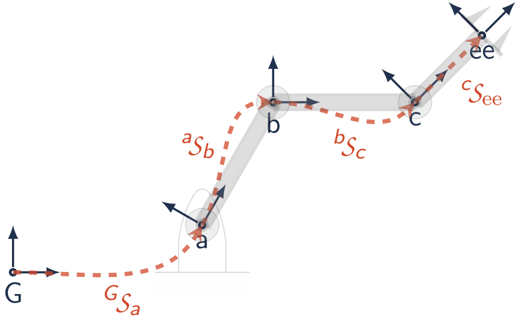
Spherical: Revolute $\otimes$ Revolute $\otimes$ Revolute

$$^{p}\mathcal{S}_c\left(\phi_0, \phi_1, \phi_2\right) = {^{p}\mathcal{S}_{c0}}\left(\phi_0\right) \otimes {^{c0}\mathcal{S}_{c_1}}\left(\phi_1\right) \otimes {^{c1}\mathcal{S}_c}\left(\phi_2\right)$$

Products of revolute and prismatic joints

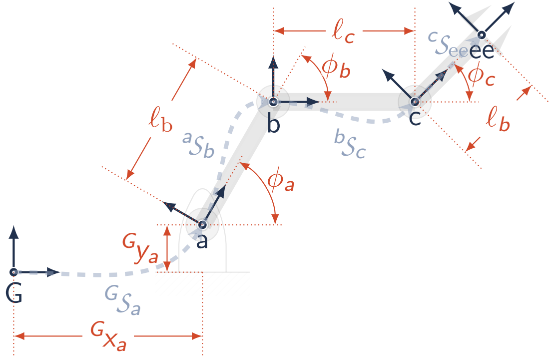
Serial Manipulator





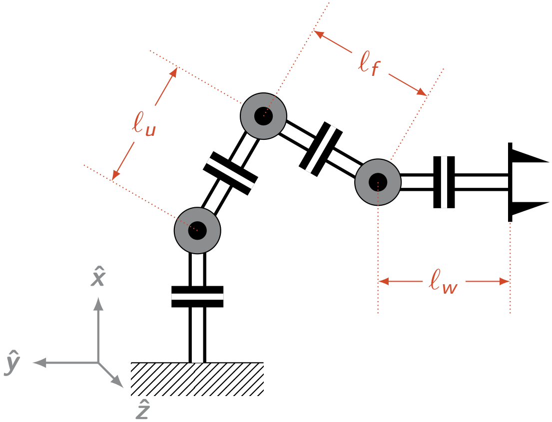
Serial Manipulator

Transforms



- **Relative:** $\mathcal{S} = h + \frac{1}{2}\vec{v} \otimes h\varepsilon$
  - $^{G}\mathcal{S}_a = \exp\left(\frac{\phi_a}{2}\hat{\boldsymbol{k}}\right) + \frac{1}{2}{}^{G}v_a \otimes \exp\left(\frac{\phi_a}{2}\hat{\boldsymbol{k}}\right)\varepsilon$
  - $^{a}\mathcal{S}_b = \exp\left(\frac{\phi_b}{2}\hat{\boldsymbol{k}}\right) + \frac{1}{2}\ell_b\hat{\boldsymbol{\imath}} \otimes \exp\left(\frac{\phi_b}{2}\hat{\boldsymbol{k}}\right)\varepsilon$
  - $^{b}\mathcal{S}_c = \exp\left(\frac{\phi_c}{2}\hat{\boldsymbol{k}}\right) + \frac{1}{2}\ell_c\hat{\boldsymbol{\imath}} \otimes \exp\left(\frac{\phi_c}{2}\hat{\boldsymbol{k}}\right)\varepsilon$
  - $^{c}\mathcal{S}_{ee} = 1 + \frac{1}{2}\ell_{ee}\hat{\boldsymbol{\imath}}\varepsilon$
- **Absolute:** $^{G}\mathcal{S}_n = {}^{G}\mathcal{S}_m \otimes {}^{m}\mathcal{S}_n$
  - $^{G}\mathcal{S}_b = {}^{G}\mathcal{S}_a \otimes {}^{a}\mathcal{S}_b$
  - $^{G}\mathcal{S}_c = {}^{G}\mathcal{S}_b \otimes {}^{b}\mathcal{S}_c$
  - $^{G}\mathcal{S}_{ee} = {}^{G}\mathcal{S}_c \otimes {}^{c}\mathcal{S}_{ee}$

Anthropomorphic arm

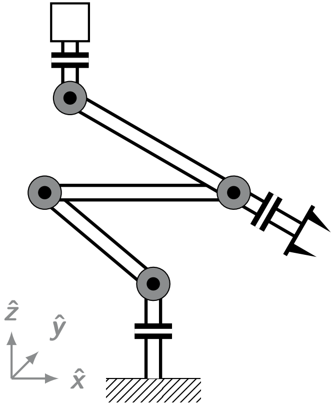


How do LWA4 joints map to human joints?

EOD 510 Packbot

Endeavor Robotics

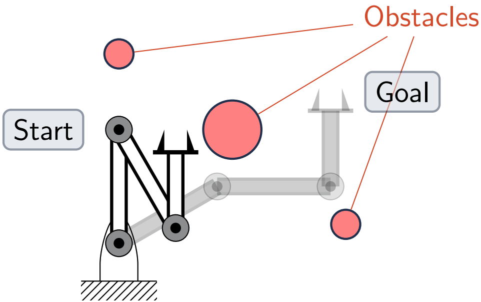
http://endeavorrobotics.com/products



What's wrong (kinematically) with this design?

Outline

Motion Planning Illustration



Find collision free path from start to goal.

Paths

Definition: Path (Mathematical)

A path through configuration space $\mathcal{Q}$ is a continuous function defining the configuration along a time-independent parameter:

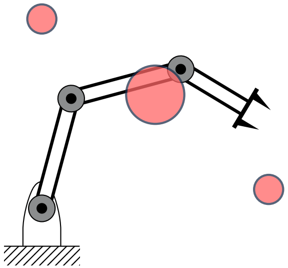
$$\tau : [0, 1] \mapsto \mathcal{Q}$$

Definition: Path (Programming Implementation)

A path is a sequence of points in configuration space $\mathcal{Q}$ such that valid transitions exist between subsequent points.

$$\tau \in \mathcal{Q}^*$$

Collisions and Free Configuration Space



- Want paths in *Collision Free Space*
- Usually no explicit representation / parameterization
- Blackbox collision checking: $\texttt{is-valid} : \mathcal{Q} \mapsto \mathbb{B}$
- Obstacle Region: $\mathcal{Q}_{\text{obs}} = \{ q \in \mathcal{Q} \mid \neg\texttt{is-valid}(q) \}$
- Free Space: $\mathcal{Q}_{\text{free}} = \{ q \in \mathcal{Q} \mid \texttt{is-valid}(q) \}$
- Total Space:
  - $\mathcal{Q}_{\text{obs}} \cap \mathcal{Q}_{\text{free}} = \emptyset$
  - $\mathcal{Q}_{\text{obs}} \cup \mathcal{Q}_{\text{free}} = \mathcal{Q}$

Dynamics vs. Combinatorics

- What is the challenge?
  - Stability?
  - Dimensionality?
- Trade-offs:
  - Algorithmic completeness
  - Computational efficiency
  - Plan optimality
- Different perspectives:
  - Solving differential equations
  - State-space search

Understand the problem to solve.

Piano Mover's Problem

Given: Environment, Robot, Start and Goal configurations

- World $\mathcal{W}$ in $\mathbb{R}^2$ or $\mathbb{R}^3$
- A robot in $\mathcal{W}$: either a single or collection of rigid bodies
- Configuration space $\mathcal{Q}$ for the robot,
  from which $\mathcal{Q}_{\text{obs}}$ and $\mathcal{Q}_{\text{free}}$ are derived.
- Initial configuration $q_0 \in \mathcal{Q}_{\text{free}}$
- Goal configuration $q_G \in \mathcal{Q}_{\text{free}}$

Find: A valid path from start q_0 to goal q_G.

Overview of Motion Planning Approaches

High-dimensional / Manipulation Problems

Local Search

1. Start from initial configuration or (possibly invalid) path
2. Progressively "improve" the configuration/path, via:
   - Gradient descent
   - Hill-Climbing
   - Sequential Optimization
   - Etc.
3. Terminate when:
   - We reach the goal/ collision free path
   - Reach a local minimum

Sampling-Based Planning

1. Start from initial configuration
2. Sample new states in the configuration space
3. Add valid states to tree or graph
4. Terminate when:
   - We find a path to the goal in the tree/graph
   - We exhaust a timeout