Some topics in Kinematics

EE366/CE366/CS380: Introduction to Robotics

Dr. Basit Memon

Electrical and Computer Engineering
Habib University

March 18, 20, 25, 2024

Table of Contents

Table of Contents

Solution Strategies: Numerical vs Closed-form

- **Numerical Approach**
- Computing numerical solutions is slow.
- Numerical strategies are universal.

- **Closed-form solution**
- Closed-form expression allow us to set rules for selecting one solution out of multiple.
- No single strategy applicable to every manipulator for finding closed-form expression.
- Closed-form solutions don't always exist.

Numerical Inverse Kinematics [2, Section 5.5]

- Say we're given $x^d \in \mathbb{R}^m$, the desired position (m=3) or the desired position and orientation of end-effector in minimal representation (m=6).

- Say we're given $x^d \in \mathbb{R}^m$, the desired position (m=3) or the desired position and orientation of end-effector in minimal representation (m=6).
- If $f : \mathbb{R}^n \to \mathbb{R}^m$ is a function describing the forward kinematics, then we want to find q^d such that

$$x^d = f(q^d).$$

- Expanding f as a Taylor series about q^d,

$$f(q) = f(q^d) + J(q^d)(q - q^d) + h.o.t.$$

where J is analytic Jacobian.

- Expanding f as a Taylor series about q^d,

$$f(q) = f(q^d) + J(q^d)(q - q^d) + h.o.t.$$

where J is analytic Jacobian.

- Neglecting higher order terms

$$q^d - q = J^{-1}(q)\left[f(q^d) - f(q)\right] = J^{-1}(q)\left[x^d - f(q)\right].$$

Numerical Inverse Kinematics: Jacobian Inverse Method

- **Algorithm:** Start with initial guess q_0. For successive estimates,

$$q_k = q_{k-1} + \alpha_k J^{-1}(q_{k-1}) \left[x^d - f(q_{k-1})\right]$$

- **Algorithm:** Start with initial guess q_0. For successive estimates,

$$q_k = q_{k-1} + \alpha_k J^{-1}(q_{k-1}) \left[x^d - f(q_{k-1})\right]$$

- Step-size α_k can be scalar or matrix, constant or function of k

- **Algorithm:** Start with initial guess q_0. For successive estimates,

$$q_k = q_{k-1} + \alpha_k J^{-1}(q_{k-1}) \left[x^d - f(q_{k-1})\right]$$

- Step-size α_k can be scalar or matrix, constant or function of k
- If J^{-1} doesn't exist, use pseudoinverse.

Numerical Inverse Kinematics: Jacobian Transpose Method

- Set it up as an optimization problem:

$$\min_q F(q) = \min_q \frac{1}{2} \left[f(q) - x^d \right]^T \left[f(q) - x^d \right]$$

- Set it up as an optimization problem:

$$\min_q F(q) = \min_q \frac{1}{2} \left[f(q) - x^d\right]^T \left[f(q) - x^d\right]$$

- Gradient of the above is:

$$\nabla F(q) = J^T(q) \left[f(q) - x^d\right]$$

Numerical Inverse Kinematics: Jacobian Transpose Method

- Set it up as an optimization problem:

$$\min_q F(q) = \min_q \frac{1}{2}\left[f(q) - x^d\right]^T \left[f(q) - x^d\right]$$

- Gradient of the above is:

$$\nabla F(q) = J^T(q)\left[f(q) - x^d\right]$$

- **Gradient Descent Algorithm:**

$$\begin{aligned} q_k &= q_{k-1} - \alpha_k \nabla F(q_{k-1}) \\ &= q_{k-1} - \alpha_k J^T(q_{k-1})\left[f(q_{k-1}) - x^d\right] \end{aligned}$$

- Set it up as an optimization problem:

$$\min_q F(q) = \min_q \frac{1}{2} \left[f(q) - x^d\right]^T \left[f(q) - x^d\right]$$

- Gradient of the above is:

$$\nabla F(q) = J^T(q) \left[f(q) - x^d\right]$$

- **Gradient Descent Algorithm:**

$$\begin{aligned} q_k &= q_{k-1} - \alpha_k \nabla F(q_{k-1}) \\ &= q_{k-1} - \alpha_k J^T(q_{k-1}) \left[f(q_{k-1}) - x^d\right] \end{aligned}$$

- Computing J^T is convenient than J^{-1}, but slower convergence than previous.

Table of Contents

Relationship between forces and torques - Statics [2]

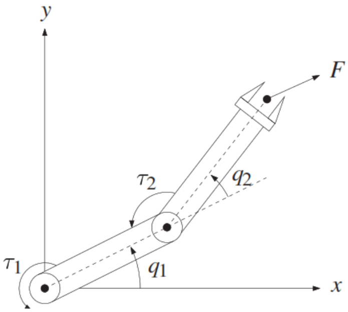


Figure: Forces at end-effector and torques at joints

- $F = (F_x, F_y, F_z, n_x, n_y, n_z)$ is vector of forces and torques at end-effector.
- τ is vector of joint torques.

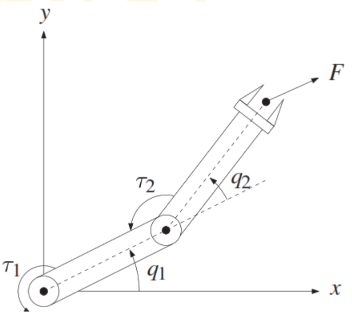


Figure: Forces at end-effector and torques at joints

- $F = (F_x, F_y, F_z, n_x, n_y, n_z)$ is vector of forces and torques at end-effector.
- τ is vector of joint torques.
- At static equilibrium,

$$\tau = J^T(q)\,F$$

Relationship between forces and torques - Statics [2]

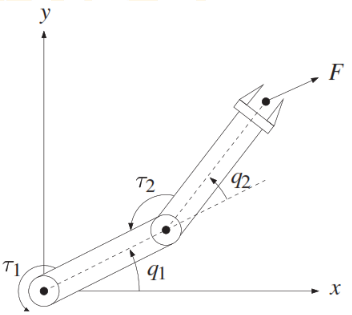


Figure: Forces at end-effector and torques at joints

- $F = (F_x, F_y, F_z, n_x, n_y, n_z)$ is vector of forces and torques at end-effector.
- τ is vector of joint torques.
- At static equilibrium,

$$\tau = J^T(q)\,F$$

- Simple proof based on principle of virtual work.

Kinematic Redundancy [1, Section 2.10]

A manipulator is **kinematically redundant** when it has a number of DOFs that is greater than number of variables necessary to describe a given task.

Kinematic Redundancy [1, Section 2.10]

A manipulator is **kinematically redundant** when it has a number of DOFs that is greater than number of variables necessary to describe a given task.

- Intrinsically redundant: $(m < n)$, where m is dimension of task space and n of joint space.

Kinematic Redundancy [1, Section 2.10]

A manipulator is **kinematically redundant** when it has a number of DOFs that is greater than number of variables necessary to describe a given task.

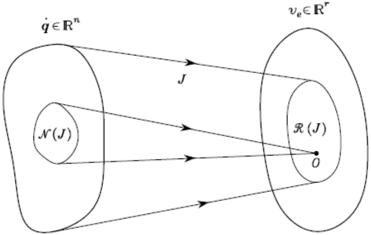
- Intrinsically redundant: $(m < n)$, where m is dimension of task space and n of joint space.
- Functionally redundant: $m = n$, but task is concerned with only $r < m$ components of task space.

A manipulator is **kinematically redundant** when it has a number of DOFs that is greater than number of variables necessary to describe a given task.

- Intrinsically redundant: $(m < n)$, where m is dimension of task space and n of joint space.
- Functionally redundant: $m = n$, but task is concerned with only $r < m$ components of task space.
- Why do we care about redundancy?

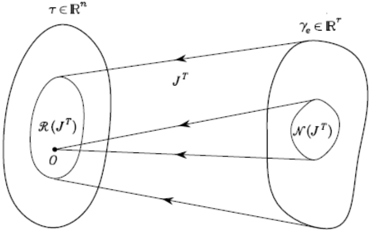
Decomposition into Jacobian subspaces at configuration q

Space of Joint Velocities



Space of Task Velocities

$\mathcal{N}(J) + \mathcal{R}(J^T) = \mathbb{R}^n$ $\qquad$ $\mathcal{N}(J^T) + \mathcal{R}(J) = \mathbb{R}^r$



Space of Joint Torques

Space of Task Forces

Inverse Velocity and Acceleration [2]

$$\xi = J\,\dot{q}$$

$$\xi = J\,\dot{q}$$

- Find joint velocities given end-effector velocities

$$\xi = J\,\dot{q}$$

- Find joint velocities given end-effector velocities

$$\dot{q} = J^{-1}\xi$$

$$\xi = J\,\dot{q}$$

- Find joint velocities given end-effector velocities

$$\dot{q} = J^{-1}\xi$$

- When would J^{-1} not exist?

Case: $n < 6$ [2]

Case: $n < 6$ [2]

- A solution may not exist.

Case: $n < 6$ [2]

- A solution may not exist.
- Vector $\xi \in$ range(J) iff

$$rank\ J(q) = rank\ [J(q) \,|\, \xi]$$

Case: $n < 6$ [2]

- A solution may not exist.
- Vector $\xi \in$ range(J) iff

$$rank\ J(q) = rank\left[J(q) \mid \xi\right]$$

- In this case, use Gaussian elimination to find $\dot{q}$.

Case: $n > 6$ [1]

Case: $n > 6$ [1]

- Infinite solutions

- Infinite solutions
- If $\dot{q}$ satisfies $\xi = J\dot{q}$, then $\dot{q} + P\dot{q}_0$ also satisfies it.
  - $\dot{q}_0$ is an arbitrary $n \times 1$ vector
  - P is an $n \times n$ matrix such that $\mathcal{R}(P) = \mathcal{N}(J)$

Least norm solution for $n > 6$ case [1]

- Minimize speed

Least norm solution for $n > 6$ case [1]

- Minimize speed

- Set it up as optimization problem

$$\min_{\dot{q}} g(\dot{q}) = \min_{\dot{q}} \frac{1}{2}\dot{q}^T\dot{q}$$

Least norm solution for $n > 6$ case [1]

- Minimize speed
- Set it up as optimization problem

$$\min_{\dot{q}} g(\dot{q}) = \min_{\dot{q}} \frac{1}{2}\dot{q}^T\dot{q}$$

$$\dot{q} = J^{+}\xi,$$

where J^{+} is right pseudo-inverse of J.

Least norm solution for $n > 6$ case [1]

$$
\begin{aligned}
\xi &= J\dot{q} \\
&= JJ^T\lambda \\
\Rightarrow \lambda &= \left(JJ^T\right)^{-1}\xi
\end{aligned}
$$

$$
\begin{aligned}
\min_{\dot{q}} g(\dot{q}) &= \min_{\dot{q}} \frac{1}{2}\dot{q}^T\dot{q} + \lambda^T(\xi - J\dot{q}) \\
\Rightarrow \left(\frac{\partial g}{\partial \dot{q}}\right)^T &= 0 \\
\left(\frac{\partial g}{\partial \lambda}\right)^T &= 0 \\
\Rightarrow \dot{q} - J^T\lambda &= 0 \\
\dot{q} &= J^T\left(JJ^T\right)^{-1}\xi \\
&= J^+\xi
\end{aligned}
$$

Another objective for $n > 6$ case [1]

Another objective for $n > 6$ case [1]

- Leverage infinite solutions to add a secondary objective

Another objective for $n > 6$ case [1]

- Leverage infinite solutions to add a secondary objective
- Say $\dot{q}_0$ is chosen to achieve this secondary objective. Then, we can minimize

$$\min_{\dot{q}} g'(\dot{q}) = \min_{\dot{q}} \frac{1}{2} (\dot{q} - \dot{q}_0)^T (\dot{q} - \dot{q}_0)$$

Another objective for $n > 6$ case [1]

- Leverage infinite solutions to add a secondary objective
- Say $\dot{q}_0$ is chosen to achieve this secondary objective. Then, we can minimize

$$\min_{\dot{q}} g'(\dot{q}) = \min_{\dot{q}} \frac{1}{2} \left(\dot{q} - \dot{q}_0\right)^T \left(\dot{q} - \dot{q}_0\right)$$

- Find a solution that gets as close as possible to $\dot{q}_0$ and satisfies $\xi = J\dot{q}$.

Another objective for $n > 6$ case [1]

$$\dot{q} = J^{+}\xi + \left(I - J^{+}J\right)\dot{q}_0$$

$$\dot{q} = J^{+}\xi + \left(I - J^{+}J\right)\dot{q}_0$$

- Choose $\dot{q}_0 = k_0 \left(\frac{\partial w(q)}{\partial q}\right)^T$, or in direction of gradient of w, so that it is maximized.

Another objective for $n > 6$ case [1]

$$\dot{q} = J^{+}\xi + \left(I - J^{+}J\right)\dot{q}_0$$

- Choose $\dot{q}_0 = k_0 \left(\frac{\partial w(q)}{\partial q}\right)^T$, or in direction of gradient of w, so that it is maximized.
- What could be w?

Another objective for $n > 6$ case [1]

$$\dot{q} = J^{+}\xi + \left(I - J^{+}J\right)\dot{q}_0$$

- Choose $\dot{q}_0 = k_0 \left(\frac{\partial w(q)}{\partial q}\right)^T$, or in direction of gradient of w, so that it is maximized.
- What could be w?
- $w(q) = \sqrt{det(JJ^T)}$ keeps manipulator away from singularities.

Table of Contents

[1] **Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo.**
Robotics: modelling, planning and control.
Springer Science & Business Media, 2010.

[2] **Mark W Spong, Seth Hutchinson, and Mathukumalli Vidyasagar.**
Robot modeling and control.
John Wiley & Sons, 2020.