Manipulation Motion Planning

Jane Li

Assistant Professor
Mechanical Engineering Department, Robotic Engineering Program
Worcester Polytechnic Institute
Quiz (10 pts)

- (3 pts) Compare the testing methods for testing path segment and finding first collision

- Compare the non-holonomic RRT with holonomic RRT: given a new node to connect to,
  - (3 pts) how to extend toward this node?
  - (3 pts) how to connect to this node for the last step?
Testing Path Segment vs. Finding First Collision

- PRM planning
  - Detect collision as quickly as possible → Bisection strategy

- Physical simulation, haptic interaction
  - Find first collision → Sequential strategy
RRTs for Non-Holonomic Systems

- Apply motion primitives (i.e. simple actions) at $q_{\text{near}}$

$$q' = f(q, u)$$

--- use action $u$ from $q$ to arrive at $q'$

$$\text{chose } u^* = \arg \min (d(q_{\text{rand}}, q'))$$

- You probably won’t reach $q_{\text{rand}}$ by doing this
  - Key point: No problem, you’re still exploring!

---

Holonomic RRT

Non-Holonomic RRT
BiDirectional Non-Holonomic RRT

How to bridge between the two points?
Shooting Method

• “Shoot” out trajectories in different directions until a trajectory of the desired boundary value is found.

  • System

\[ \frac{dy}{dx} + f(x, y) = 0. \]

  • Boundary condition

\[ y(0) = 0, \; y(1) = 1 \]
Manipulation motion planning
Recap

- We have learned the planning algorithms that can generalize across many types of robots
  - Discrete planning
  - Sampling-based planning

- Theoretically, we should be all set. However ...
  - When it comes to manipulator robots, we may have to handle an application-specific problem
Bimanual humanoid robot
Mobile manipulator robot
Kinematically redundant manipulators
Research Questions

• How to resolve the kinematic redundancy?

• How to coordinate macro- and micro-structures?
  • Arm-hand structure
  • Body-arm structure

• How to handle bimanual coordination?
Overview

- How to resolve the kinematic redundancy?
  - Solution to Inverse kinematics
  - Pseudo-inverse
  - Additional constraints and optimization criteria
Forward and inverse kinematics

Robot Joint Angles

Forward Kinematics

End Effector Pose

Inverse Kinematics
Kinematic Redundancy

- If $N > M$,
  - FK maps a *continuum* of configurations to one end-effector pose:

- If $N = M$,
  - FK maps a *finite number* of configurations to one end-effector pose:

- If $N < M$,
  - Target pose not reachable
Kinematics at different levels

- Direct kinematics
  \[ x = F K(q) \]

- First-order differential kinematics – Jacobian
  \[ \dot{x} = J(q)\dot{q} \]

- Second-order differential kinematics
  \[ \ddot{x} = J(q)\ddot{q} + \dot{J}(q, \dot{q})\dot{q} \]
Our primary concern is the **end-effector pose** in task space.

IK solver needs to compute a **C-space** motion that does the right thing in **task space**.
Inverse Kinematics at position levels

• Direct kinematics

\[ x = FK(q) \]

• IK solution
  • Analytical solution – robot geometry
  • Algebraic solution – homogeneous transformation matrices
Analytical solution
Algebraic Solution

\[ 0^N T = 0^1 T \ldots 0^{N-1} T = \begin{bmatrix}
    r_{11} & r_{12} & r_{13} & p_x \\
    r_{21} & r_{22} & r_{23} & p_y \\
    r_{31} & r_{32} & r_{33} & p_z \\
    0 & 0 & 0 & 1
\end{bmatrix} \]
Algebraic Solution

\[
^0T^6 = ^0T_1^1T_2^2T_3^3T_4^4T_5^5T_6^6 = \begin{bmatrix}
r_{11} & r_{12} & r_{13} & p_x \\
r_{21} & r_{22} & r_{23} & p_y \\
r_{31} & r_{32} & r_{33} & p_z \\
0 & 0 & 0 & 1
\end{bmatrix}
\]

\[
r_{11} = c_1 [c_23(c_4c_5c_6 - s_4s_6) - s_23s_5c_6] + s_1(s_4c_5c_6 + c_4s_6),
\]

\[
r_{21} = s_1 [c_23(c_4c_5c_6 - s_4s_6) - s_23s_5c_6] - c_1(s_4c_5c_6 + c_4s_6),
\]

\[
r_{31} = -s_23(c_4c_5c_6 - s_4s_6) - c_23s_5c_6,
\]

\[
r_{12} = c_1 [c_23(-c_4c_5s_6 - s_4c_6) + s_23s_5s_6] + s_1(c_4c_6 - s_4c_5s_6),
\]

\[
r_{22} = s_1 [c_23(-c_4c_5s_6 - s_4c_6) + s_23s_5s_6] - c_1(c_4c_6 - s_4c_5s_6),
\]

\[
r_{32} = -s_23(-c_4c_5s_6 - s_4c_6) + c_23s_5s_6,
\]

\[
r_{13} = -c_1(c_23c_4s_5 + s_23c_5) - s_1s_4s_5,
\]

\[
r_{23} = -s_1(c_23c_4s_5 + s_23c_5) + c_1s_4s_5,
\]

\[
r_{33} = s_23c_4s_5 - c_23c_5,
\]

\[
p_x = c_1 [a_2c_2 + a_3c_23 - d_4s_23] - d_3s_1,
\]

\[
p_y = s_1 [a_2c_2 + a_3c_23 - d_4s_23] + d_3c_1,
\]

\[
p_z = -a_3s_23 - a_2s_2 - d_4c_23.
\]
IK strategies

- Do not care about the redundant DOFs motion
  - Standard IK solvers, using pseudo-inverse

- Utilize redundant DOFs to handle additional constraints
  - Obstacle

- Utilize redundant DOFs to optimize performance
  - What are the performance indices?
Inverse Kinematics at velocity level

• First-order differential kinematics
  \[ \dot{x} = J(q)\dot{q} \]

• IK solution
  • Inverse the Jacobian (non-redundant manipulator)
  • Pseudo-inverse of Jacobian
• Start with Forward Kinematics function

\[ x = FK(q) \]

• Take the derivative with respect to time:

\[ \frac{dx}{dt} = \frac{d[FK(q)]}{dt} = \frac{dFK(q)}{dq} \frac{dq}{dt} \]

• Now we get the standard Jacobian equation:

\[ \frac{dx}{dt} = J(q) \frac{dq}{dt} \quad \Rightarrow \quad \frac{dFK(q)}{dq} = J(q) \]
Example

\[ T(\theta_1, \theta_2, \theta_3) = \begin{bmatrix} c_{123} & -s_{123} & L_1 c_1 + L_2 c_{12} + L_3 c_{123} \\ s_{123} & c_{123} & L_1 s_1 + L_2 s_{12} + L_3 s_{123} \end{bmatrix} \]

\[ J = \begin{bmatrix} \frac{\partial x}{d\theta_1} & \frac{\partial x}{d\theta_2} & \frac{\partial x}{d\theta_3} \\ \frac{\partial y}{d\theta_1} & \frac{\partial y}{d\theta_2} & \frac{\partial y}{d\theta_3} \end{bmatrix} \]
Inverting the Jacobian

- If \( N = M \),
  - Jacobian is square \( \rightarrow \) Standard matrix inverse

- If \( N > M \),
  - Pseudo-Inverse
  - Weighted Pseudo-Inverse
  - Damped least squares
Pseudo-Inverse

• Pseudo-inverse matrix
  • The unique matrix satisfying the Moore–Penrose conditions
    \[ JJ^\dagger J = J \quad (JJ^\dagger)^T = JJ^\dagger \]
    \[ J^\dagger JJ^\dagger = J^\dagger \quad (J^\dagger J)^T = J^\dagger J \]

• For redundant manipulator
  \[ J^\dagger = J^T (JJ^T)^{-1} \]
Pseudo-Inverse

- Pseudo-Inverse specifies a **unique solution** for inverse kinematics

- Implicitly, it performs the following optimization

  - Minimize $\frac{1}{2} \dot{\theta}^T \dot{\theta}$, given $\dot{x} = J(q) \dot{q}$
Weighted Pseudo-Inverse

- Multiply weighting coefficient matrix to Pseudo-Inverse Jacobian
  \[ \dot{q} = J_w^\dagger(q) \dot{x} \quad \text{where} \quad J^\dagger = W^{-1}J^T(JW^{-1}J^T)^{-1} \]

- Optimization?
  - Minimize \( \frac{1}{2} \dot{\theta}^T W \dot{\theta} \), given \( \dot{x} = J(q) \dot{q} \)
How to choose the weighting coefficient matrix?

- $W > 0$ and symmetric

- Large weight $\rightarrow$ small joint velocity

- Weights $\sim$ inverse of the joint angle range
Singularity

- Singular Value Decomposition (SVD)

\[ \mathbf{J}_{M \times N} = \mathbf{U}_{M \times M} \mathbf{\Sigma}_{M \times N} \mathbf{V}_{N \times N}^T \]

\[ \mathbf{\Sigma} = \begin{pmatrix} \sigma_1 \\ \sigma_2 \\ \vdots \\ \sigma_M \end{pmatrix} \]

\[ \sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_p > 0, \quad \sigma_{p+1} = \ldots = \sigma_M = 0 \]

Singular values of \( \mathbf{J} \)
Singularity

\[
\dot{q} \xrightarrow{V^T} V^T \dot{q} \xrightarrow{\Sigma} \Sigma V^T \dot{q} \xrightarrow{U} U \Sigma V^T \dot{q} = J \dot{q}
\]

\[
\begin{align*}
\Sigma_w & \quad \Sigma & \quad V^T \left(J J^T \right)^+ V & \leq 1
\end{align*}
\]
Singularity

\[ J = U \Sigma V^T \quad \text{where} \quad \Sigma = \begin{pmatrix} \sigma_1 & \cdots & 0_{M \times (N-M)} \\ \vdots & & \vdots \\ \sigma_M & \cdots & 0_{M \times (N-M)} \end{pmatrix} \]

\[ J^\dagger = V \Sigma^\dagger U^T \quad \text{where} \quad \Sigma^\dagger = \begin{pmatrix} \frac{1}{\sigma_1} & \cdots & 0_{(N-M) \times M} \\ \vdots & \ddots & \vdots \\ \frac{1}{\sigma_M} & \cdots & 0_{(N-M) \times M} \end{pmatrix} \]
Distance to singularity

• Manipulability index – Jacobian matrix determinant

\[ \mu = \sqrt{|JJ^T|} \]

• Which is indeed

\[ \mu = \prod_{i=1}^{M} \sigma_i \]

• Is it a good measurement?
Distance to singularity

• Manipulability index – condition number
  \[ \kappa = \frac{\sigma_{\text{max}}}{\sigma_{\text{min}}} \]

• Alternatively, can use isotropy
  \[ \text{Isotropy} = \frac{\sigma_{\text{min}}}{\sigma_{\text{max}}} \]

• Is it good enough?
Distance to singularity

• Manipulability index – the smallest singular value

\[ \sigma_{\text{min}} \]

• Direction of velocity disadvantage

• Is it good enough?
Distance to singularity

• Manipulability index

\[ \mu' = \sum_{i=1}^{M} \sqrt{|J_iJ_i^T|} \]

• What does it imply?
  • Manipulability of every sub-manipulator (non-redundant)
To render robust behavior when crossing the singularity, we can add a small constant along the diagonal of \((J(q)^T J(q))\) to make it invertible when it is singular.
The matrix will be invertible but this technique introduces a small inaccuracy.
Damped Least Squares

- Induced error by damped least squares

\[ \dot{e} = \mu^2 \left( JJ^T + \mu^2 I_M \right)^{-1} \dot{X} \quad (\text{as in } N=M \text{ case}) \]

using SVD of \( J=U \Sigma V^T \Rightarrow J_{DLS} = V \Sigma_{DLS} U^T \) with \( \Sigma_{DLS} = \begin{pmatrix} \frac{\sigma_i}{\sigma_i^2 + \mu^2} \\ \rho \times \rho \\ 0_{(N-M)\times \rho} \\ 0_{(N-M)\times(N-\rho)} \end{pmatrix} \)

- Choice of the damping factor \( \mu^2(q) \geq 0 \)
  - As a function the minimum singular value \( \rightarrow \) measure of distance to singularity
  - Induce the damping only/mostly in the non-feasible direction of the task
Augmented Jacobian

- Project a task space velocity vector into the null-space
  - Primary task
    \[ \dot{x} = J(q)\dot{q} \]
  - Additional constraint
    \[ x_c = FK_c(q) \]
  - Secondary task
    \[ \dot{x}_c = J_c(q)\dot{q} \quad \text{where} \quad J_c(q) = \frac{\partial FK_c}{\partial q} \]

\[ J_a(q) = \begin{bmatrix} J(q) \\ J_c(q) \end{bmatrix} \]

Full rank square Jacobian Invertible!
The Null-space of Jacobian

- Secondary tasks is satisfied in the **null-space** of the Jacobian pseudo-inverse

- In linear algebra, the **null-space** of a matrix $A$ is the set of vectors $V$ such that, for any $v$ in $V$, $0 = A^T v$.

- $V$ is orthogonal to the range of $A$

![Diagram](image)
The Null-space of Jacobian

- Given the null space of Jacobian, the secondary task will not disturb the primary task

- The **null-space projection matrix** for the Jacobian pseudo-inverse is:

\[
N(q) = I - J(q)^\dagger J(q)
\]

\[
J^\dagger JJ^\dagger = J^\dagger
\]
The Null-space of Jacobian

- Project a **task space velocity vector** into the null-space

\[
\dot{q} = J(q)^\dag \dot{x} + (I - J(q)^\dag J(q)) J_c(q)^\dag \dot{x}_c
\]

Primary task

Secondary task
• The null-space is often used to “push” IK solvers away from
  • Joint limits, obstacles
  • How to define the secondary task for the constraints in both task and joint space?

\[ \dot{q} = J(q)^\dagger \dot{x} + (I - J(q)^\dagger J(q)) J_c(q)^\dagger \dot{x}_c \]
For non-linear systems, magnitude differences in primary and secondary can cause numerical problems
- One can overwhelm the other when you normalize later
- Introduce a normalization factor

\[
\dot{q} = J(q)^\dagger \dot{x} + \beta (I - J(q)^\dagger J(q)) \dot{q}_c
\]

Conflicts with primary?

Primary task

Secondary task
Recursive Null-space Projection

• What if you have three or more tasks?
  • The $i$-th task is:
    \[ T_i = J_i^\dagger(q) \dot{x}_i \]
  • The $i$-th null-space is:
    \[ N_i(q) = I - J_i^\dagger(q) J_i(q) \]
  • The recursive null-space formula is then:
    \[ \dot{q} = T_1 + N_1(T_2 + N_2(T_3 + N_3(T_4 + \cdots N_{n-1}T_n))) \]
Inverse Kinematics at acceleration level

• Second-order differential kinematics

\[ \ddot{x} = J(q)\dot{q} + \dot{J}(q, \dot{q})\dot{q} \]

• IK solution

\[ \ddot{q} = J^\dagger(q) (\ddot{x} - \dot{J}\dot{q}) + (I - J^\dagger J)\ddot{q}_0 \]

• \( \ddot{q}_0 = 0 \) is an arbitrary joint-space acceleration
Inverse Kinematics at acceleration level

• Choose \( \ddot{q}_0 = 0 \)

• We have \( \ddot{q} = J^+(q)(\ddot{x} - \dot{J}\dot{q}) \)

Minimum-norm acceleration solution
Reference

- Chapter 10 Redundant Robots in *Handbook of Robotics, 2nd ed*
End