Lecture 12:
inverse kinematics II

Prof. Katie Driggs-Campbell
March 7, 2023
Administrivia

• HW 6 is due Friday at 8pm
  • See discord for code to use
• First Guest Lecture on Th 3/9
• Exam 2 is on 3/27 – 3/29
  • Review session will be on Th 3/23
  • No class on Tu 3/28
  • Topics: Forward, Velocity, and Inverse Kinematics
• Second Guest Lecture on Th 3/30
Who is Joseph Raphson?

- Joseph Raphson (1648 – 1715) was an English mathematician known best for the **Newton–Raphson method**
- Raphson's most notable work is *Analysis Aequationum Universalis* (1690), which approximates the roots of an equation
- Isaac Newton had developed a very similar formula in his Method of Fluxions, written in 1671, but published in 1736
- Raphson's method is simpler than Newton’s, so Raphson’s version is generally used in textbooks today

Credit: Wikipedia
Inverse Kinematics

- **Forward Kinematics**: computes the end-effector position from joint angles:
  \[
  \text{joint space} \rightarrow SE(3) \\
  \theta \rightarrow T(\theta)
  \]

- **Inverse Kinematics**: computes the possible joint angles from the pose of the end-effector
  \[
  SE(3) \rightarrow \text{joint space} \\
  x \rightarrow \theta
  \]
Inverse Kinematics

- **Forward Kinematics**: computes the end-effector position from joint angles:

- **Inverse Kinematics**: computes the possible joint angles from the pose of the end-effector

- Given $T(\theta)$, find solutions $\theta$ that satisfy $T(\theta) = X$

- When the analytic solution is hard or impossible to come by, we numerically solve $T(\theta) - X = 0$

$g(\theta) = f(\theta) - x = 0$

goal: find roots of $g$
Newton-Raphson Methods (1)

solve: \( g(\theta) = 0 \), \( g \) is differentiable

1. start w/ initial guess \( \theta^0 \) for \( \theta_d \)
2. write Taylor expansion of \( g(\theta) \) at \( \theta^0 \) up to the first order

\[
g(\theta) = g(\theta^0) + \frac{\partial g}{\partial \theta}(\theta^0) \cdot (\theta - \theta^0) + \text{h.o.t.} = 0
\]

\[
\frac{\partial g}{\partial \theta} \bigg|_{\theta^0}
\]
Newton-Raphson Method (1.5)

3. Set approx \( g \) equal to zero and solve for \( \theta \):

\[
g(\theta^0) + \frac{\partial g}{\partial \theta}|_{\theta^0} (\theta - \theta^0) = 0
\]

\[
\theta' = \theta^0 - \left(\frac{\partial g}{\partial \theta}|_{\theta^0}\right)^{-1} g(\theta^0)
\]

4. Iterate with new estimate of \( \theta \):

\[
\theta^{k+1} = \theta^k - \left(\frac{\partial g}{\partial \theta}|_{\theta^k}\right)^{-1} g(\theta^k)
\]
Newton-Raphson Method (2)

\[ \Theta^{k+1} = \Theta^k - \left( \frac{\partial g}{\partial \Theta} \bigg|_{\Theta^k} \right)^{-1} g(\Theta^k) \]

\[ \frac{\partial g}{\partial \Theta} (\Theta) = \begin{bmatrix} \frac{\partial g_1}{\partial \Theta_1} (\Theta) & \ldots & \frac{\partial g_1}{\partial \Theta_n} (\Theta) \\ \vdots & \ddots & \vdots \\ \frac{\partial g_m}{\partial \Theta_1} (\Theta) & \ldots & \frac{\partial g_m}{\partial \Theta_n} (\Theta) \end{bmatrix} \in \mathbb{R}^{m \times n} \]

we execute this iteration until some stopping condition is met:

\[ \frac{|g(\Theta^k) - g(\Theta^{k+1})|}{|g(\Theta^k)|} \leq \varepsilon \]
Numerical IK

Given $\mathbf{x}_d$: end-effector position

$f(\theta)$: forward kinematics

$\theta^0$: initial guess

$g(\theta_d) = x_d - f(\theta_d) = 0$

find $\theta_d$

$$\mathbf{x}_d = f(\theta_d) = f(\theta^0) + \frac{\partial f}{\partial \theta}(\theta^0)(\theta_d - \theta^0) + \text{h.o.t.}$$

$$\mathbf{x}_d - f(\theta^0) = \mathbf{J}(\theta^0)\Delta\theta$$

if $\mathbf{J}(\theta^0)$ is invertible:

$$\Delta\theta = \mathbf{J}^{-1}(\theta^0)(\mathbf{x}_d - f(\theta^0))$$

$$\theta' = \theta^0 + \Delta\theta$$

then reset guess $\theta^0$ to $\theta'$ and iterate\

$s(\theta^0, \theta', \theta^2, \ldots)$ converge to desired $\theta_d$
Numerical Inverse Kinematics

- If the forward kinematics is linear in $\theta$ (i.e., h.o.t. are zero), then $\theta^1$ exactly satisfies $x_d = f(\theta^1)$
- If the forward kinematics is not linear in $\theta$ (as is usually the case), then $\theta^1$ should be closer to the root than $\theta^0$, but will require more iterations
Non-invertible Jacobians and the Pseudo-Inverse

- $J(\theta^0)$ may be singular ($\det(J(\theta^0)) = 0$) or may not be square
  - Non-invertible! Instead use Pseudo-Inverse!
- The Moore-Penrose pseudo-inverse for $J \in \mathbb{R}^{m \times n}$ is denoted $J^\dagger \in \mathbb{R}^{n \times n}$
- Consider equation: $Jy = z$. We want to find the solution: $y^* = J^\dagger z$, such that:
  - One solution if $n = m$ and $J$ is full rank
    - $y^* = J^{-1}z$
  - Many solutions if $n > m$
    - $y^*$ exactly satisfies $Jy^* = z$, and gives the minimal norm solution ($\|y^*\| \leq \|y\|$)
  - No solutions if $n < m$ and $z$ is not in the span of $J$
    - If no solution, $y^*$ minimizes the two-norm of the error: $\|Jy^* - z\| \leq \|J\tilde{y} - z\|$ for all $\tilde{y}$
The Pseudo Inverse

• When \( J \) is full column rank (\( n<m, \) tall):
  \[
  J^+ = (J^T J)^{-1} J^T
  \]
  (left inverse: \( J^+ J = I \))

• When \( J \) is full row rank (\( n>m, \) wide):
  \[
  J^+ = J^T (JJ^T)^{-1}
  \]
  (right inverse: \( JJ^+ = I \))

• When \( n=m \) and full rank:
  \[
  J^+ = J^{-1}
  \]

\[
\Delta \Theta = J^+ (\Theta^0) (x_a - f(\Theta^0))
\]
Using the Newton-Raphson Method for IK

- The Newton-Raphson algorithm needs to be modified given that $X \in SE(3)$, which is not a general matrix in $\mathbb{R}^{4 \times 4}$ or a coordinate vector.
Using the Newton-Raphson Method for IK

• The Newton-Raphson algorithm needs to be modified given that \( X \in SE(3) \), which is not a general matrix in \( \mathbb{R}^{4 \times 4} \) or a coordinate vector.

• The error vector \( e = x_d - f(\theta^i) \), represents the update needed to go from the current guess to the desired end-effector configuration (after being multiplied by the inverse Jacobian).

• Said otherwise, following the direction \( e \) for one second, starting from \( f(\theta^i) \), should send us (approximately) to \( x_d \).
Using the Newton-Raphson Method for IK

- The Newton-Raphson algorithm needs to be modified given that \( X \in SE(3) \), which is not a general matrix in \( \mathbb{R}^{4\times4} \) or a coordinate vector.

- The error vector \( e = x_d - f(\theta^i) \), represents the update needed to go from the current guess to the desired end-effector configuration (after being multiplied by the inverse Jacobian).

- Said otherwise, following the direction \( e \) for one second, starting from \( f(\theta^i) \), should send us (approximately) to \( x_d \).

- In our case, we are given \( X \in SE(3) \), and instead of computing \( X - T(\theta^i) \), we should compute the twist \( \mathbf{\nu}_b \) which, if followed for one second, sends us from \( T(\theta^i) \) to \( X \).
Numerical Inverse Kinematics with a Twist

want to find body twist $Y_b$ to move from $T_{sb}(\Theta')$ to desired $T_{sd} = X$

the twist that sends us from $T_{sb}(\Theta')$ to $T_{sd}$ satisfies:

$T_{sd} := X = T_{sb}(\Theta') e^{[Y_b]}$

$e^{[Y_b]} = T_{sb}^{-1}(\Theta') T_{sd}$

recall matrix log:

$[Y_b] = \log \left( T_{sb}^{-1}(\Theta') T_{sd} \right)$
Numerical Inverse Kinematics: Algorithm

0. Given $X = T_{sd}$ and the forward kinematics map $T_{sb}(\theta)$
   
   Given tolerances $\epsilon_\omega$ and $\epsilon_v$

1. Initialize $\theta^0$ and set $i = 0$

2. While $\|\omega_b\| > \epsilon_\omega$ or $\|\omega_v\| > \epsilon_v$
   
   1. Set $[\mathcal{V}_b] = \log T_{sb}^{-1}(\theta^i)T_{sd}$
   
   2. Set $\theta^{i+1} = \theta^i + J_b^\dagger(\theta^i)\mathcal{V}_b$
   
   3. Increment $i$
A few comments

• Re: algorithm on last slide:
  • An equivalent form can be derived in the space frame, using the spatial Jacobian and the spatial twist

• You can also extend this to find sequence of joint angles that will allow the end-effector to follow a trajectory $T_{sb}(t)$
  • Discretize the trajectory $T_{sb}(\theta_k)$ and compute $\theta_k$ such that $T_{sb}(\theta_k) = T_{sb}(t_k)$
  • Initialization is really important here!
Summary

• **Inverse Kinematics** gives us a method for finding the joint angles given an end-effector configuration

• This can sometimes be done analytically (geometrically), but this is often difficult so **numerical techniques** are more common in practice

• Introduced the **Newton-Raphson method**, which can be modified to solve inverse kinematics numerically
  • In this week’s HW, you will be given code for the Newton-Raphson method