

Lecture 12: inverse kinematics I

Prof. Katie Driggs-Campbell

March 12, 2026

Inverse Kinematics in Animation



Forward Kinematics



Inverse Kinematics

What is going on in this class?

- Rigid Body Motion
- Forward Kinematics
 - Calculate the position of the end-effector of an open chain given joint angles

$$x(t) = f(\theta(t))$$

- Velocity Kinematics
 - Calculate the velocity (twist!) of the end-effector

$$\frac{d}{dt}x(t) = \frac{\partial f}{\partial \theta}(\theta) \cdot \dot{\theta}$$

- Inverse Kinematics
 - Computes the possible joint angles from the pose of the end-effector
 - Given $T(\theta)$, find solutions θ that satisfy $T(\theta) = X$

A few geometric tricks

Law of Cosines

$$c^2 = a^2 + b^2 - 2ab \cos C$$

Two argument arctan function atan2

$$\text{atan2}(y, x) = \begin{cases} \text{atan}\left(\frac{y}{x}\right) & \text{if } x > 0 \\ \text{atan}\left(\frac{y}{x}\right) + \pi & \text{if } x < 0 \text{ and } y \geq 0 \\ \text{atan}\left(\frac{y}{x}\right) - \pi & \text{if } x < 0 \text{ and } y < 0 \\ \frac{\pi}{2} & \text{if } x = 0 \text{ and } y > 0 \\ -\frac{\pi}{2} & \text{if } x = 0 \text{ and } y < 0 \\ \text{undefined} & \text{if } x = 0 \text{ and } y = 0 \end{cases}$$

Euler Angles

Given some arbitrary rotation, how to find the associated angles?

ZYX Euler angles represent a rotation as follows:

$$R(\alpha, \beta, \gamma) = \text{Rot}(\hat{z}, \alpha) \text{Rot}(\hat{y}, \beta) \text{Rot}(\hat{x}, \gamma)$$

$$\alpha, \gamma \in (-\pi, \pi] \text{ and } \beta \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right)$$

where

$$\text{Rot}(\hat{z}, \alpha) = \begin{bmatrix} \cos \alpha & -\sin \alpha & 0 \\ \sin \alpha & \cos \alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Euler Angles (see Appendix B)

$$R(\alpha, \beta, \gamma) = \begin{bmatrix} c_\alpha c_\beta & c_\alpha s_\beta s_\gamma - s_\alpha c_\gamma & c_\alpha s_\beta s_\gamma + s_\alpha s_\gamma \\ s_\alpha c_\beta & s_\alpha s_\beta s_\gamma + c_\alpha c_\gamma & s_\alpha s_\beta s_\gamma - c_\alpha s_\gamma \\ -s_\beta & c_\beta s_\gamma & c_\beta c_\gamma \end{bmatrix}$$

If r_{ij} denotes the ij^{th} element in R , then

1. If $r_{31} \neq \pm 1$, set $\beta = \text{atan2}(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2})$, $\alpha = \text{atan2}(r_{21}, r_{11})$, $\gamma = \text{atan2}(r_{32}, r_{33})$
2. If $r_{31} = -1$, then $\beta = \frac{\pi}{2}$ and there are infinite solutions for α and γ . One solution is: $\alpha = 0$, $\gamma = \text{atan2}(r_{12}, r_{22})$
3. If $r_{31} = 1$, then $\beta = -\frac{\pi}{2}$ and there are infinite solutions for α and γ . One solution is: $\alpha = 0$, $\gamma = -\text{atan2}(r_{12}, r_{22})$

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 θ that satisfy $T(\theta) = X$

Inverse Kinematics (IK) Solutions

Analytic Inverse Kinematics

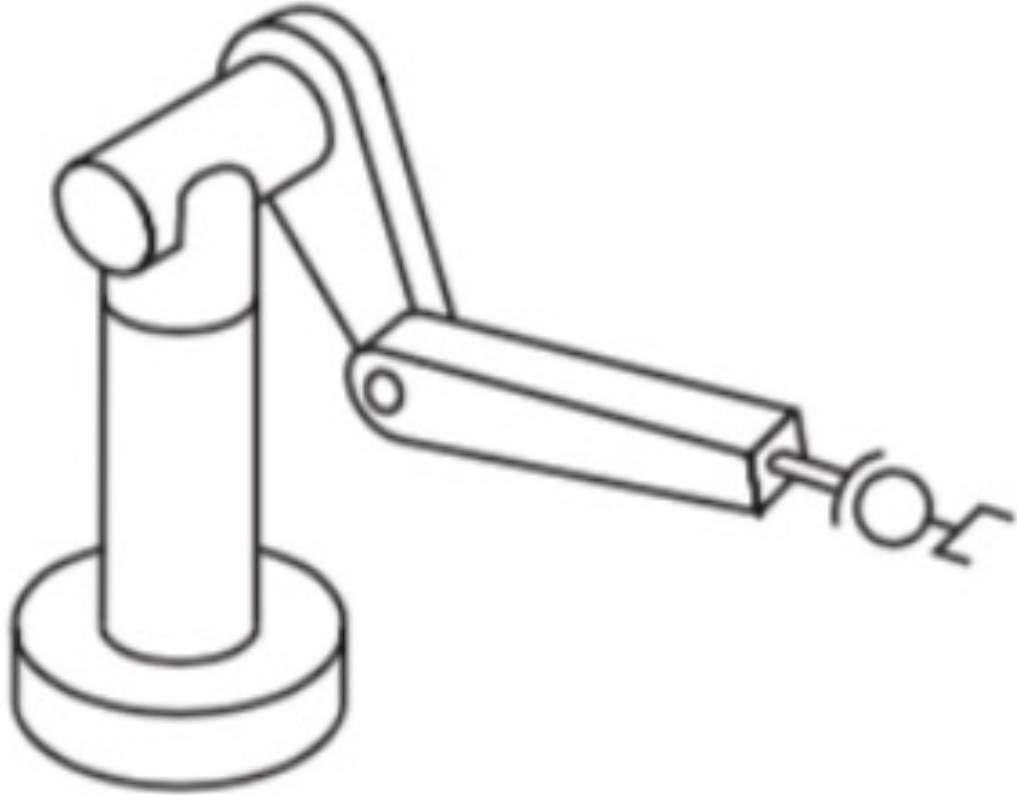
Analytic inverse kinematics

- In the 2R example, there were 2 DOFs for the end-effector and 2 joint angles. This implied that there was a **finite number of solutions**.
- If there are more joint angles than DOFs of the end-effector, there may be **an infinite number of solutions**.
- In practice, we will assume that #DOFs of end-effector = #joint angles.
- For now, assume in general:

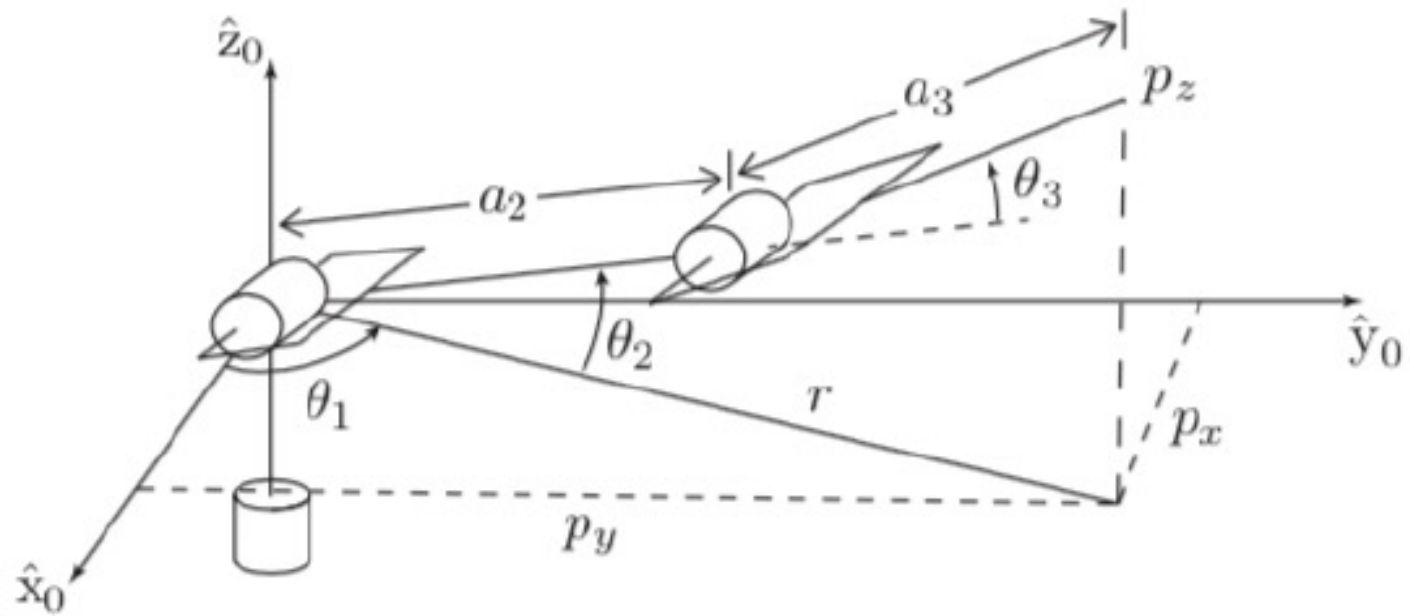
$$T(\theta) = e^{[S_1]\theta_1} e^{[S_2]\theta_2} \dots e^{[S_6]\theta_6} M$$

and that we are given an end-effector pose $X \in SE(3)$.

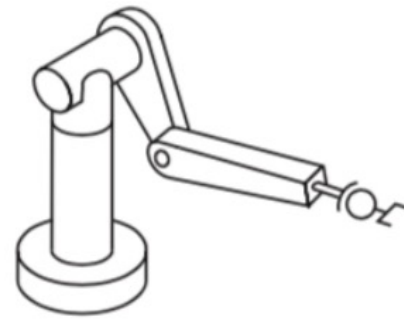
6R PUMA arm



6R PUMA arm



6R PUMA arm – orientation



Numerical Inverse Kinematics

- **Forward Kinematics:** $\theta \rightarrow T(\theta)$
- **Inverse Kinematics:** $X \rightarrow \theta$
- When the analytic solution is hard or impossible to come by, we **numerically** solve:

Summary

- Wrapped up **velocity kinematics** by showing example of computing the Jacobian by definition using adjoint maps
- Introduced analytic **inverse kinematics**, which helps us find the joint angles that will produce a desired end-effector pose
 - We can do this with the help of **atan2** and **ZYX Euler Angles**
 - However, this only works for relatively simple robot chains