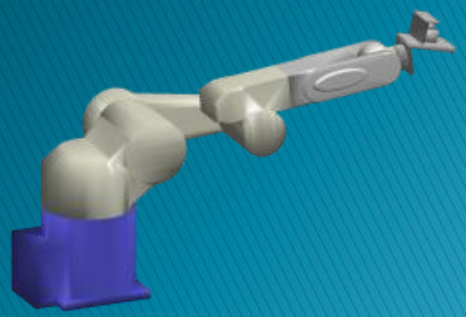
5.2 Inverse Kinematics via Optimization

Centre for Autonomous Systems
University of Technology Sydney

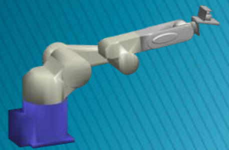


Inverse Kinematics via Optimization

- Closed-form solutions for inverse kinematics becomes difficult for complex robot geometry
- Furthermore, each robot requires a unique calculation.
- Mathematical optimization can be used to solve inverse kinematics for any generic robot arm:

Decision variable $\longrightarrow$ $$\min_{\mathbf{q}} \|\mathbf{x} - \mathbf{f}(\mathbf{q})\|$$ $\longleftarrow$ Objective function

$$\text{subject to: } \mathbf{q}_{\min} \leq \mathbf{q} \leq \mathbf{q}_{\max}$$ $\longleftarrow$ Inequality constraints



Gradient Descent Optimization

Suppose we want to minimise some cost function $g(\mathbf{q}) \in \mathbb{R}, \mathbf{q} \in \mathbb{R}^n$.

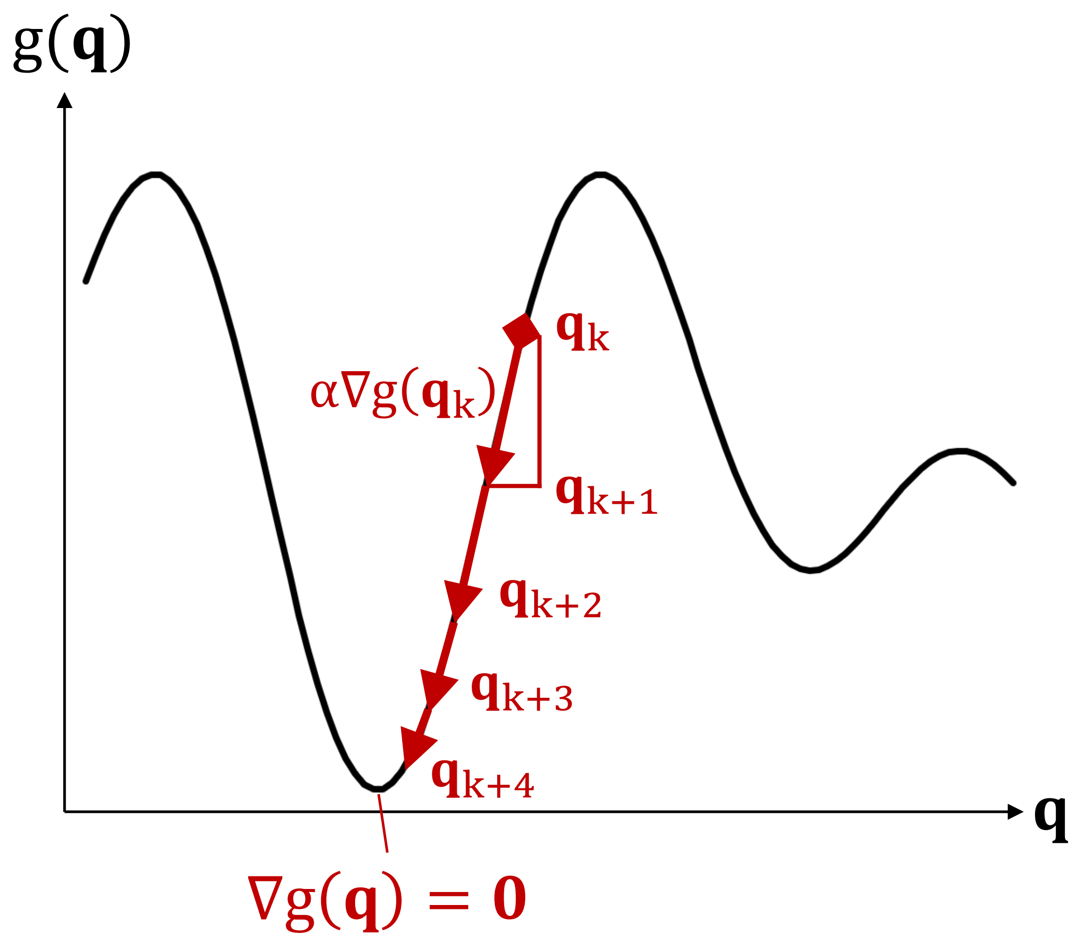
$g(\mathbf{q})$ is a minimum when its gradient $\nabla g(\mathbf{q}) = \mathbf{0}$.

$$\nabla g(\mathbf{q}) = \frac{\partial g}{\partial \mathbf{q}} = \begin{bmatrix} \partial g/\partial q_1 \\ \vdots \\ \partial g/\partial q_n \end{bmatrix} \in \mathbb{R}^n$$

Basic idea: Take a step where the function slopes downwards.

$$\begin{aligned} \mathbf{q}_{k+1} &= \mathbf{q}_k + \Delta\mathbf{q} \\ &= \mathbf{q}_k - \alpha\nabla g(\mathbf{q}_k) \end{aligned}$$

Repeat until $\|\Delta\mathbf{q}\|$ is very small, i.e. gradient is practically zero $\nabla g(\mathbf{q}) \approx \mathbf{0}$.



Local and Global Minimums

Optimization involves finding the minimum of a function:

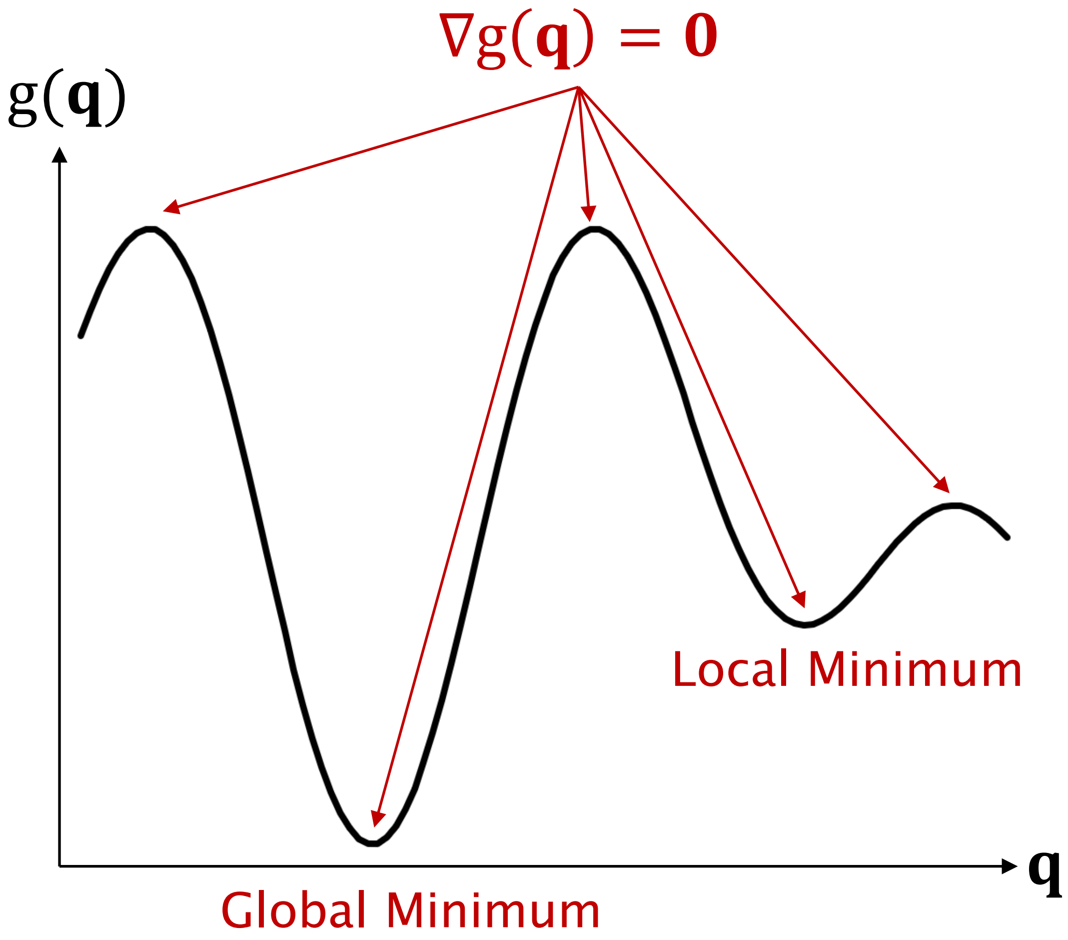
$$\min_{\mathbf{q}} \mathrm{g}(\mathbf{q}) = \|\mathbf{x} - \mathbf{f}(\mathbf{q})\|$$

Minimums (and maximums) exist where the gradient is zero:

$$\nabla \mathrm{g}(\mathbf{q}) = \mathbf{0}$$

Nonlinear functions can have many different minimums

- Local minimums
- Global minimums

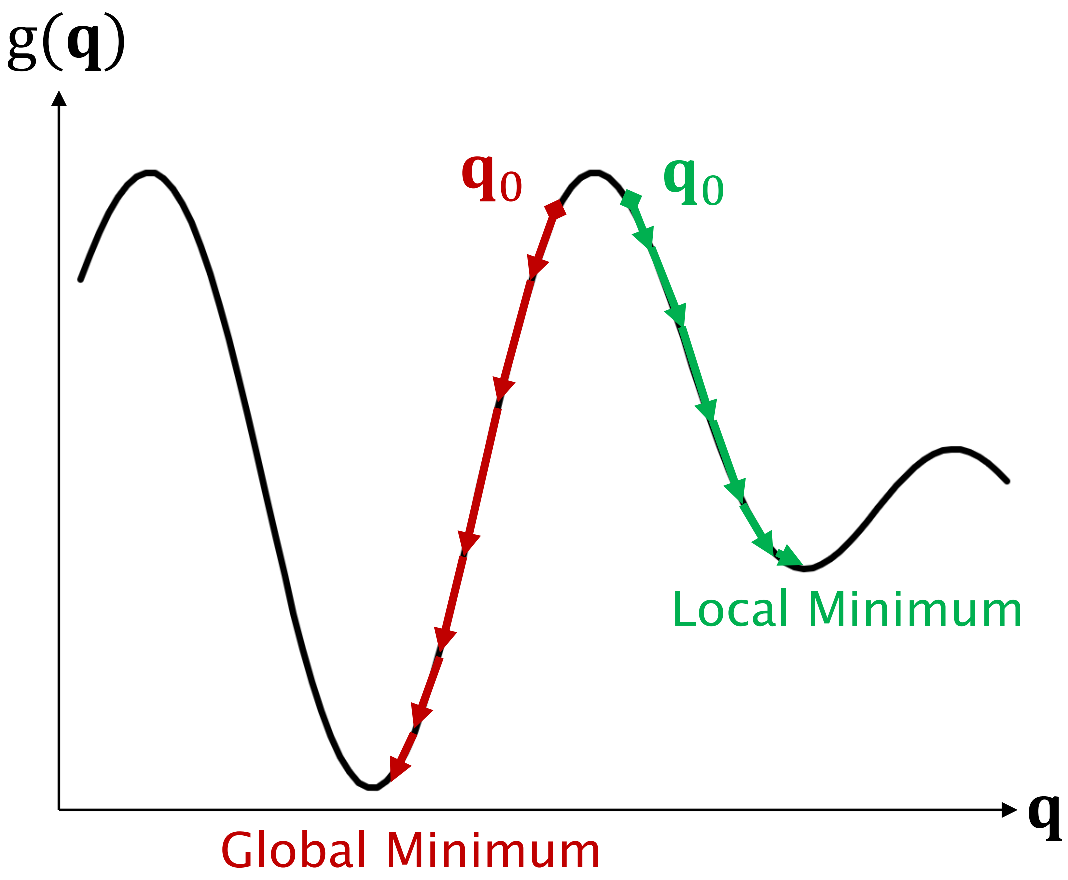


Initial Guess $\mathbf{q}_0$

Gradient descent involves the cumulative sum of gradients from the initial guess $\mathbf{q}_0$:

$$
\begin{aligned}
\mathbf{q}_1 &= \mathbf{q}_0 - \alpha\nabla g(\mathbf{q}_0) \\
\mathbf{q}_2 &= \mathbf{q}_1 - \alpha\nabla g(\mathbf{q}_1) \\
&= \mathbf{q}_0 - \alpha\nabla g(\mathbf{q}_0) - \alpha\nabla g(\mathbf{q}_1) \\
&\vdots \\
\mathbf{q}_k &= \mathbf{q}_0 - \alpha \sum_{j=0}^{k-1} \nabla g(\mathbf{q}_j)
\end{aligned}
$$

- Our initial guess $\mathbf{q}_0$ determines the search direction
- Can lead us to a local minimum, or a global minimum
- Can determine whether optimisation succeeds or fails



Multiple Inverse Kinematics Solutions

Use the result of a previous IK solution to optimize for the next pose:

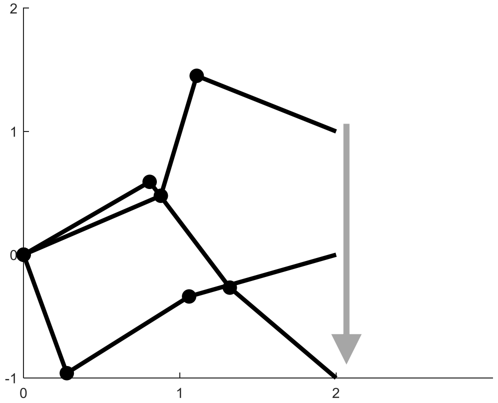
$$\mathbf{q}_1 = \mathbf{f}^{-1}(\mathbf{x}_1, \mathbf{q}_0)$$

$$\mathbf{q}_2 = \mathbf{f}^{-1}(\mathbf{x}_2, \mathbf{q}_1)$$

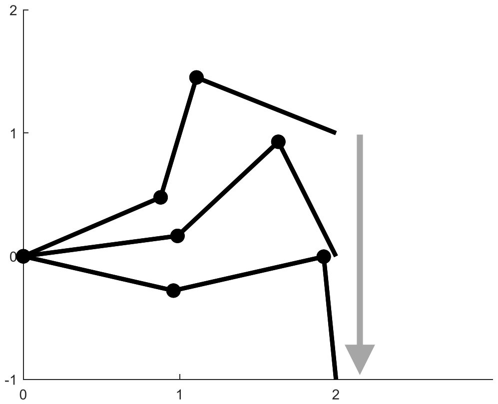
$$\mathbf{q}_3 = \mathbf{f}^{-1}(\mathbf{x}_3, \mathbf{q}_2)$$

This ensures congruous joint motion, but also solves the optimization faster.

Without previous solution:



With previous solution:



Inverse Kinematics via Gradient Descent

Start with a desired end-effector transform $\mathbf{T}_\mathrm{d}$, and actual end-effector transform $\mathbf{T}_\mathrm{k}$ calculated from forward kinematics at step k:

$$\mathbf{T}_\mathrm{d} = \begin{bmatrix} \mathbf{R}_\mathrm{d} & \mathbf{p}_\mathrm{d} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}, \qquad \mathbf{T}_\mathrm{k}(\mathbf{q}) = \begin{bmatrix} \mathbf{R}_\mathrm{k} & \mathbf{p}_\mathrm{k} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}$$

The difference in the desired and actual pose is:

$$\Delta\mathbf{x} = \begin{bmatrix} \mathbf{e}_\mathrm{p} \\ \mathbf{e}_\mathrm{o} \end{bmatrix}$$

Where:

$$\mathbf{e}_\mathrm{p} = \mathbf{p}_\mathrm{d} - \mathbf{p}_\mathrm{k}$$

$$\mathbf{e}_\mathrm{o} \leftarrow \mathbf{R}_\mathrm{e} = \mathbf{R}_\mathrm{d}\mathbf{R}_\mathrm{k}^\mathrm{T}$$

The orientation error $\mathbf{e}_\mathrm{o}$ is expressed in RPY angles, extracted from rotation error $\mathbf{R}_\mathrm{e}$.

From the forward kinematics:

$$\mathbf{x} = \mathbf{f}(\mathbf{q})$$

$$\Delta\mathbf{x} = (\partial\mathbf{f}/\partial\mathbf{q})\Delta\mathbf{q}$$

$$\Delta\mathbf{q} = (\partial\mathbf{f}/\partial\mathbf{q})^{-1}\Delta\mathbf{x}$$

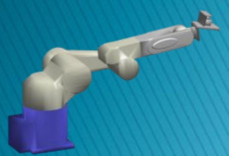
Then increment the joint configuration $\mathbf{q}$ to cause the actual end-effector pose to converge on the desired pose:

$$\begin{aligned} \mathbf{q}_{\mathrm{k}+1} &= \mathbf{q}_\mathrm{k} + \alpha\Delta\mathbf{q}_\mathrm{k} \\ &= \mathbf{q}_\mathrm{k} + \alpha(\partial\mathbf{f}/\partial\mathbf{q})^{-1}\Delta\mathbf{x}_\mathrm{k} \end{aligned}$$

Where:

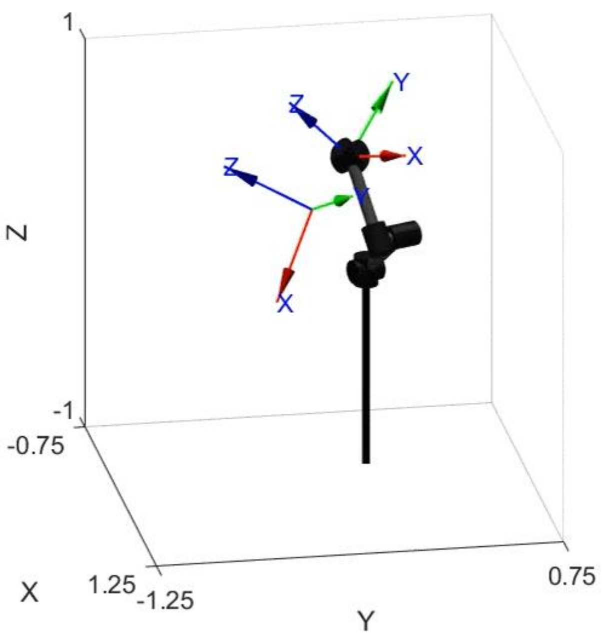
- $\alpha \in \mathbb{R}$ is a scalar
- $(\partial\mathbf{f}/\partial\mathbf{q})^{-1} \in \mathbb{R}^{\mathrm{n}\times\mathrm{m}}$ maps the pose error $\Delta\mathbf{x}$ from Cartesian space to joint space $\mathbf{q}$.

Repeat the process until $\|\Delta\mathbf{q}\| \approx 0$.

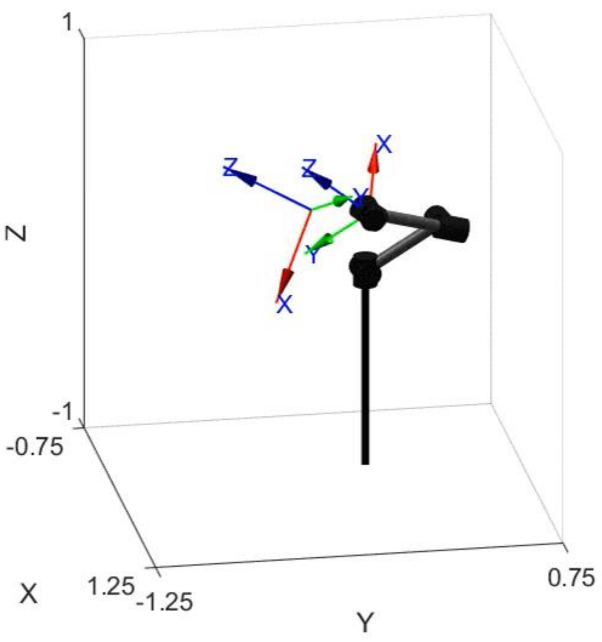


Demonstration of Inverse Kinematics via Gradient Descent

- 48 steps
- Position Error: 2.4081 (mm).
- Orientation Error: 0.013403 (rad).



- 63 steps
- Position Error: 0.72233 (mm).
- Orientation Error: 0.0056693 (rad).



Summary of Inverse Kinematics via Optimization

$$\min_{\mathbf{q}} g(\mathbf{q}) = \|\mathbf{x} - \mathbf{f}(\mathbf{q})\|$$

Inverse kinematics (IK) can be solved with optimization

$$\mathbf{q}_{k+1} = \mathbf{q}_k - \alpha \nabla g(\mathbf{q}_k)$$

Gradient descent minimizes a cost function by incrementing the state variable down the gradient of said function

$$\mathbf{q}_k = \mathbf{q}_0 - \alpha \sum_{i=0}^{k-1} \nabla g(\mathbf{q}_i)$$

The solution is dependent upon the initial guess

$$\mathbf{q}_k = \mathbf{f}^{-1}(\mathbf{x}_k, \mathbf{q}_{k-1})$$

Use previous solutions for IK to optimize the next configuration

$$\mathbf{q}_{k+1} = \mathbf{q}_k + \alpha(\partial \mathbf{f}/\partial \mathbf{x})^{-1} \Delta \mathbf{x}$$

IK via gradient descent optimization increments the joint state to reduce pose error