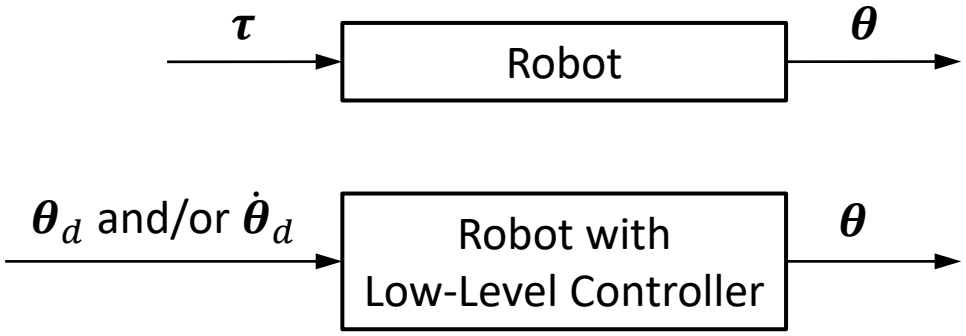
Ch9: Kinematic Control

Introduction

Kinematic Control of Robots

A robot is an electromechanical system driven by actuating torques $\boldsymbol{\tau}$ produced by its motors. However, if the robot (typically an industrial robot) is equipped with **low-level feedback control** at the joints—'ideally' allowing the imposing of commanded reference **positions $\boldsymbol{\theta}_d$ and/or velocities $\dot{\boldsymbol{\theta}}_d$**—it is possible to treat a kinematic command (most often a velocity) as the control input to the system.

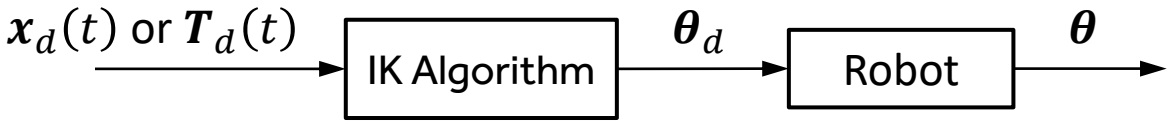
Note: The performance can be quite satisfactory only if the desired motion is not too fast and does not require large accelerations, thereby keeping the effect of inertial forces small. Otherwise, the position/velocity error ($\boldsymbol{e} = \boldsymbol{\theta}_d - \boldsymbol{\theta}$ or $\dot{\boldsymbol{e}} = \dot{\boldsymbol{\theta}}_d - \dot{\boldsymbol{\theta}}$) will be large.



In this case, if the input is $\dot{\boldsymbol{\theta}}_d$, the robot act as a simple integrator $\int$ (with a given $\boldsymbol{\theta}_0$).

Inverse Kinematics Scheme

If end-effector desired trajectory $\boldsymbol{x}_d(t)$ or $\boldsymbol{T}_d(t)$ are given (e.g., after motion planning), a typical approach is to use **Inverse Kinematics** at each time step to find $\boldsymbol{\theta}_d(t)$ and send them to the robot.



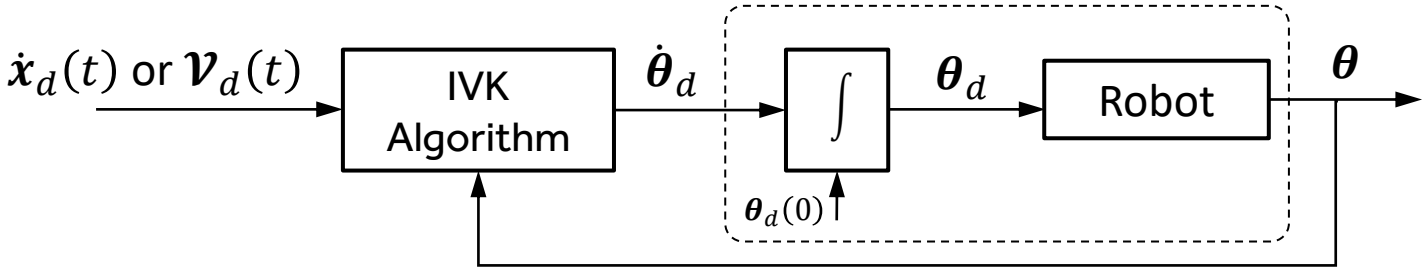
Disadvantages:

- Requires accurate low-level position controllers.
- Requires IK computation which may be time consuming for real-time motion tracking.
- Cannot efficiently exploit redundant DOFs. Redundancy resolution is much more naturally and efficiently handled in inverse velocity kinematics (IVK) via null-space projection or optimization.

Resolved-Rate Motion Control

Inverse Velocity Kinematics Scheme (Resolved-Rate Motion Control)

If end-effector desired velocity $\dot{\boldsymbol{x}}_d(t)$ or $\boldsymbol{\mathcal{V}}_d(t)$ are given (e.g., after motion planning), we can use **Velocity Kinematics** methods like the closed-form IVK (and exploiting redundancies) $\dot{\boldsymbol{\theta}}_d = \boldsymbol{J}(\boldsymbol{\theta})^+ \boldsymbol{\mathcal{V}}_d + \left(\boldsymbol{I}_n - \boldsymbol{J}(\boldsymbol{\theta})^+ \boldsymbol{J}(\boldsymbol{\theta})\right)\dot{\boldsymbol{\theta}}_0$ or solving an optimization problem (with constraints) to compute the corresponding desired $\dot{\boldsymbol{\theta}}_d$. If the robot accepts the joint velocities directly, we send the computed $\dot{\boldsymbol{\theta}}_d$ to the robot at each time step. However, if the robot only accepts joint positions, we need to use an integration method to find $\boldsymbol{\theta}_d$ at each time step by having the initial robot configuration $\boldsymbol{\theta}(0)$:



$$\boldsymbol{\theta}_d(t) = \int_0^t \dot{\boldsymbol{\theta}}_d(\varsigma)d\varsigma + \boldsymbol{\theta}_d(0) \xrightarrow[k = 0,1,\ldots]{\text{Using Euler integration method and an integration interval } \Delta t = t_{k+1} - t_k} \boldsymbol{\theta}_d(t_{k+1}) = \boldsymbol{\theta}_d(t_k) + \dot{\boldsymbol{\theta}}_d(t_k)\Delta t$$

Remarks

For example, $\dot{\boldsymbol{\theta}}_d(t_k) = \boldsymbol{J}^+\big(\boldsymbol{\theta}(t_k)\big)\boldsymbol{\mathcal{V}}_d(t_k)$

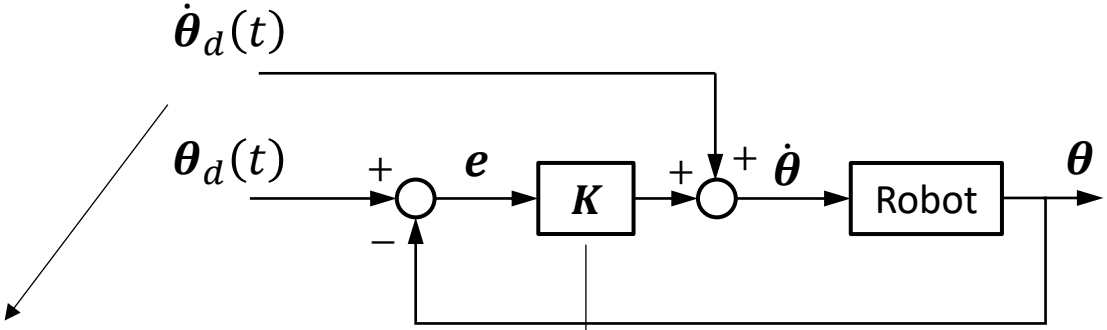
or $\boldsymbol{\theta}_d(t_{k+1}) = \boldsymbol{\theta}_d(t_k) + \dot{\boldsymbol{\theta}}_d(t_k)\Delta t = \boldsymbol{\theta}_d(t_k) + \boldsymbol{J}^+\big(\boldsymbol{\theta}(t_k)\big)\boldsymbol{\mathcal{V}}_d(t_k)\Delta t$

This formulation is called **Resolved-Rate Motion Control (RRMC)**.

- Therefore, the velocity kinematics $\boldsymbol{\mathcal{V}} = \boldsymbol{J}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}}$ can be used as a simple solution to the **Inverse Kinematics** problem of finding $\boldsymbol{\theta}$, without direct computation of the inverse kinematics at each time step.
- Due to **drift phenomena** in numerical integration, small velocity errors are likely to accumulate over time, resulting in increasing position error $\boldsymbol{\theta}$. Therefore, the end-effector pose corresponding to the computed $\boldsymbol{\theta}$ (in the discrete time) differs from the desired one (in the continuous time).
- To increase accuracy, a feedback is required in the algorithm to ensure the end-effector follows the desired pose/motion. This falls under the topic of **Kinematic Control**.

Kinematic Motion Control

Kinematic Motion Control in Joint Space



- If $\dot{\boldsymbol{x}}_d(t)$ or $\boldsymbol{\mathcal{V}}_d(t)$ are given, we can use RRMC to compute $\boldsymbol{\theta}_d(t)$ and $\dot{\boldsymbol{\theta}}_d(t)$.

$\boldsymbol{K} \in \mathbb{R}^{n\times n}$ is a positive definite (usually diagonal) matrix.

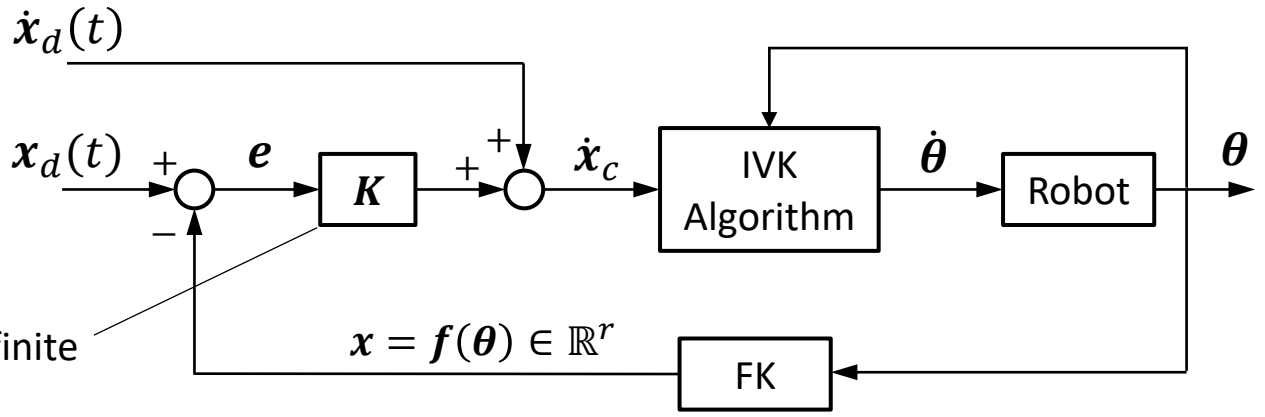
$$\boldsymbol{e} = \boldsymbol{\theta}_d - \boldsymbol{\theta} \longrightarrow \dot{\boldsymbol{e}} = \dot{\boldsymbol{\theta}}_d - \dot{\boldsymbol{\theta}} = \dot{\boldsymbol{\theta}}_d - \left(\dot{\boldsymbol{\theta}}_d + \boldsymbol{K}(\boldsymbol{\theta}_d - \boldsymbol{\theta})\right) = -\boldsymbol{K}\boldsymbol{e} \quad \rightarrow \quad \dot{\boldsymbol{e}} + \boldsymbol{K}\boldsymbol{e} = \boldsymbol{0}$$

(joint position error)

- $\dot{\boldsymbol{e}} + \boldsymbol{K}\boldsymbol{e} = \boldsymbol{0}$ is Linear and, if $\boldsymbol{K}$ is diagonal, also Decoupled, for $i = 1, \cdots, n$.
- The system is Asymptotically/Exponentially Stable, i.e., $\boldsymbol{e} \to 0$ or $\boldsymbol{\theta} \to \boldsymbol{\theta}_d$ along the trajectory.
- The larger the eigenvalues of $\boldsymbol{K}$, the faster the convergence rate.

Note: This controller can be used to increase accuracy of tracking (e.g., when the robot low-level controller is not accurate).

Kinematic Motion Control in Task Space



$K \in \mathbb{R}^{r \times r}$ is a positive definite (usually diagonal) matrix.

If we use closed-form IVK based on Jacobian (pseudo-) inverse:

$$\dot{\theta} = J_a^+ \underbrace{(\dot{x}_d + Ke)}_{\dot{x}_c} + \underbrace{(I_n - J_a^+ J_a)\dot{\theta}_0}_{\text{For Exploiting Redundant DOFs}} \qquad \forall \dot{\theta}_0 \in \mathbb{R}^n$$

$$e = x_d - x \;\rightarrow\; \dot{e} = \dot{x}_d - \dot{x} = \dot{x}_d - J_a\dot{\theta} = \dot{x}_d - (\dot{x}_d + Ke) = -Ke \;\rightarrow\; \dot{e} + Ke = 0$$

(EE pose error)

- $\dot{e} + Ke = 0$ is Linear and, if K is diagonal, also Decoupled, for $i = 1, \cdots, r$.
- The system is Asymptotically/Exponentially Stable, i.e., $e \rightarrow 0$ or $x \rightarrow x_d$ along the trajectory.
- The larger the eigenvalues of K, the faster the convergence rate.

Remarks

- On the assumption that matrix $\boldsymbol{J}_a$ is square ($n = r$, nonredundant robot) and nonsingular:

$$\dot{\boldsymbol{\theta}} = \boldsymbol{J}_a^{-1}(\boldsymbol{\theta})(\dot{\boldsymbol{x}}_d + \boldsymbol{K}\boldsymbol{e})$$

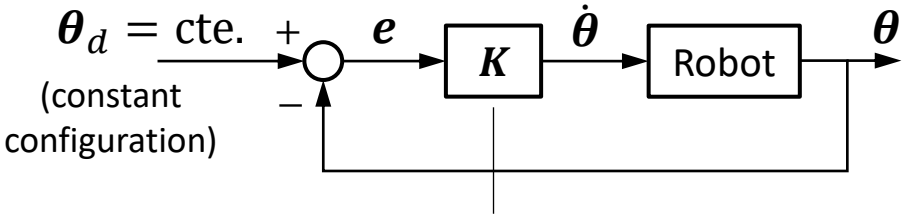
- If instead of $\boldsymbol{x}_d(t)$, matrix $\boldsymbol{T}_d(t) \in SE(3)$ is given at the input, in the diagram, we need to replace $\boldsymbol{x}$ with $\boldsymbol{T}$, $\dot{\boldsymbol{x}}_d(t)$ with $\boldsymbol{\mathcal{V}}_d(t)$, and the error is defined as

$$\boldsymbol{e}_b = \log\left(\boldsymbol{T}_{sb}^{-1}\boldsymbol{T}_d\right)^{\vee} \qquad \text{(expressed in EE body frame)}$$

$$\boldsymbol{e}_s = \left[\mathrm{Ad}_{\boldsymbol{T}_{sb}}\right]\log\left(\boldsymbol{T}_{sb}^{-1}\boldsymbol{T}_d\right)^{\vee} \qquad \text{(expressed in base frame)}$$

Kinematic Position Control

Kinematic Position Control in Joint Space



$\boldsymbol{K} \in \mathbb{R}^{n \times n}$ is a positive definite (usually diagonal) matrix.

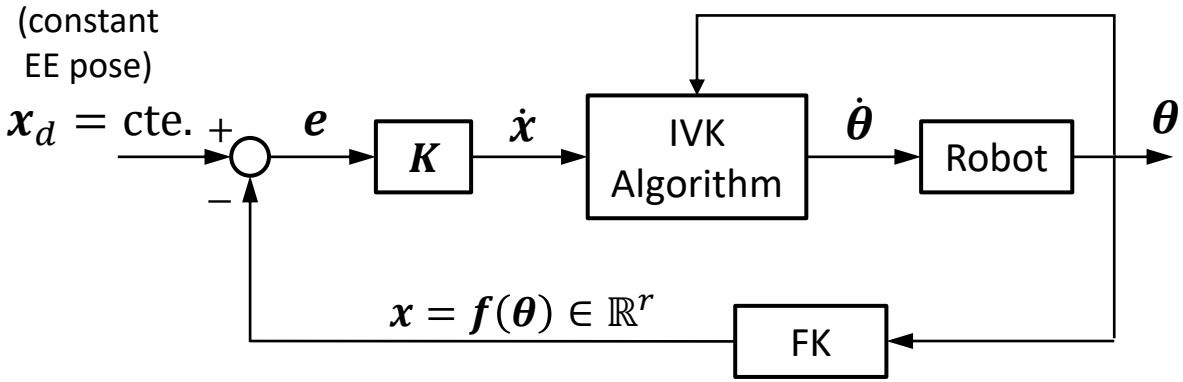
This controller is similar to the kinematic motion controller in the absence of feedforward.

$$\boldsymbol{e} = \boldsymbol{\theta}_d - \boldsymbol{\theta} \longrightarrow \dot{\boldsymbol{e}} = -\dot{\boldsymbol{\theta}} = -\boldsymbol{K}\boldsymbol{e} \quad \rightarrow \quad \dot{\boldsymbol{e}} + \boldsymbol{K}\boldsymbol{e} = \boldsymbol{0} \quad \rightarrow \quad \boldsymbol{e} \rightarrow 0 \text{ as } t \rightarrow \infty$$

- This controller can be interpreted as $\dot{\boldsymbol{\theta}}(k\Delta t) = \dfrac{\left(\boldsymbol{\theta}_d(k\Delta t) - \boldsymbol{\theta}\left((k-1)\Delta t\right)\right)}{\Delta t}$

  when $\boldsymbol{K} = (1/\Delta t)\boldsymbol{I}_n$. The desired new joint angles $\boldsymbol{\theta}_d(k\Delta t)$ at each discrete timestep k are being compared with the most recently measured actual joint angles $\boldsymbol{\theta}\left((k-1)\Delta t\right)$ to calculate the required joint velocities $\dot{\boldsymbol{\theta}}$ during each time interval $[(k-1)\Delta t, k\Delta t]$.

Kinematic Position Control in Task Space



This controller is similar to the kinematic motion controller in the absence of feedforward.

$\boldsymbol{K} \in \mathbb{R}^{r \times r}$ is a positive definite (usually diagonal) matrix.

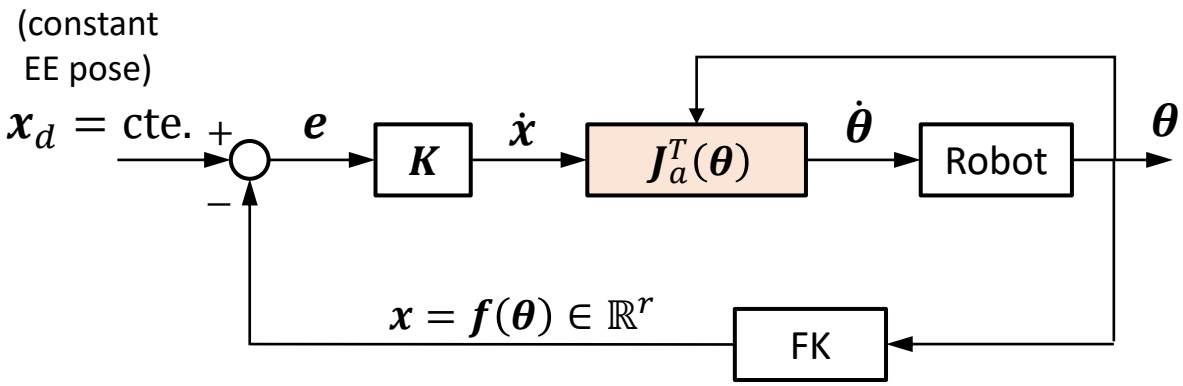
$$\boldsymbol{e} = \boldsymbol{x}_d - \boldsymbol{x} \;\rightarrow\; \dot{\boldsymbol{e}} = -\dot{\boldsymbol{x}} = -\boldsymbol{K}\boldsymbol{e} \;\rightarrow \dot{\boldsymbol{e}} + \boldsymbol{K}\boldsymbol{e} = \boldsymbol{0} \;\rightarrow\; \boldsymbol{e} \rightarrow 0 \text{ as } t \rightarrow \infty$$

Note: If we use closed-form IVK based on Jacobian (pseudo-) inverse, then $\dot{\boldsymbol{\theta}} = \boldsymbol{J}_a^{+}(\boldsymbol{K}\boldsymbol{e}) + (\boldsymbol{I}_n - \boldsymbol{J}_a^{+}\boldsymbol{J}_a)\dot{\boldsymbol{\theta}}_0$ ($\forall \dot{\boldsymbol{\theta}}_0 \in \mathbb{R}^n$) and this controller for $\dot{\boldsymbol{\theta}}_0 = \boldsymbol{0}$ and $\boldsymbol{K} = \lambda \boldsymbol{I}_n$ corresponds to the **Inverse Kinematic** based on Newton–Raphson method (Jacobian (pseudo-)inverse method).

Note: If we use this controller with time-variant input, the error $\boldsymbol{e}$ does not go to zero (due to lack of feedforward part).

Note: Since $\dot{\boldsymbol{x}} = \boldsymbol{K}\boldsymbol{e} = \boldsymbol{K}(\boldsymbol{x}_d - \boldsymbol{x})$, this controller move the robot end-effector from its initial pose to the desired pose on a straight line in $\mathbb{R}^3$ for position (or in $SE(3)$ if the input is $\boldsymbol{T}_d$).

Kinematic Position Control in Task Space



$\boldsymbol{K} \in \mathbb{R}^{r \times r}$ is a positive definite (usually diagonal) matrix.

Using Lyapunov direct method, we can show that by choosing $\dot{\boldsymbol{\theta}} = \boldsymbol{J}_a^T(\boldsymbol{\theta})\boldsymbol{K}\boldsymbol{e}$, the asymptotic stability of the error $\boldsymbol{e}$ will be ensured (i.e., $\boldsymbol{e} \to 0$ as $t \to \infty$).

Note: This controller for $\boldsymbol{K} = \lambda \boldsymbol{I}_n$ corresponds to the **Inverse Kinematic** based on Gradient Descent method (Jacobian transpose method).

$$\boldsymbol{\theta}^{k+1} = \boldsymbol{\theta}^k + \lambda \boldsymbol{J}_a^T(\boldsymbol{\theta}^k)\left(\boldsymbol{x}_d - \boldsymbol{f}(\boldsymbol{\theta}^k)\right), \qquad k = 0,1,2,\dots$$