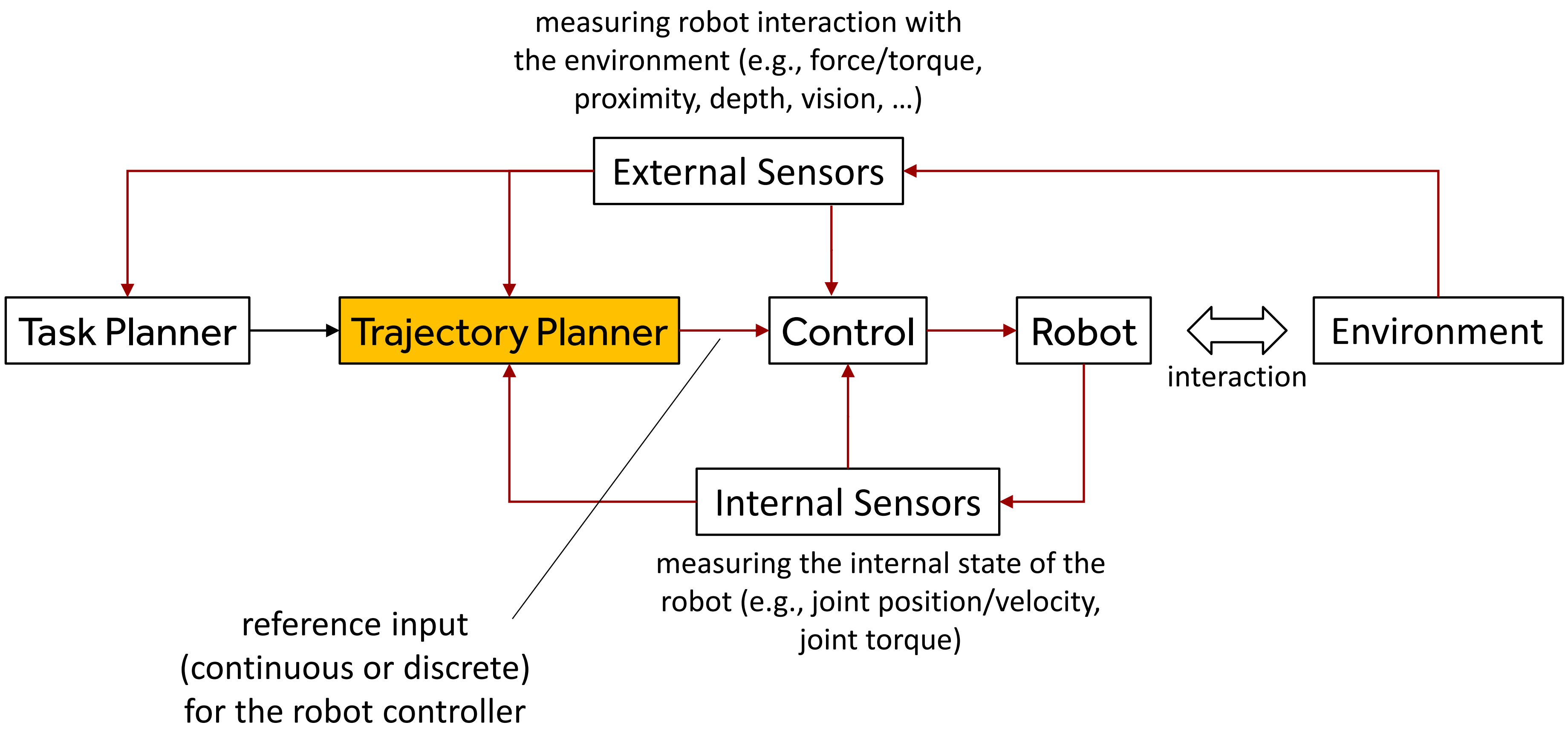
Ch8: Trajectory Planning

Path and Trajectory

Trajectory Planner Interfaces


measuring robot interaction with the environment (e.g., force/torque, proximity, depth, vision, …)
External Sensors
Task Planner
Trajectory Planner
Control
Robot
Environment
interaction
Internal Sensors
measuring the internal state of the robot (e.g., joint position/velocity, joint torque)
reference input (continuous or discrete) for the robot controller

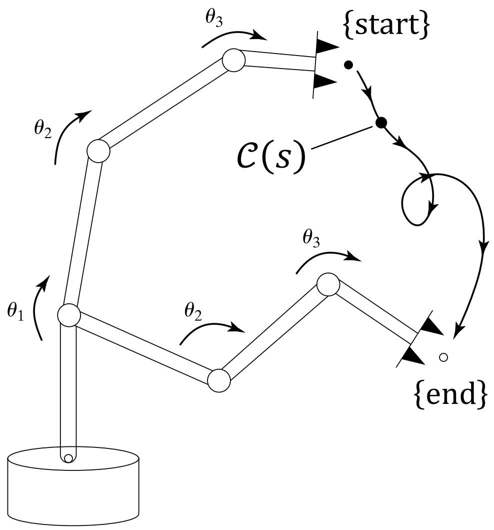
Path, Time Scaling, and Trajectory

Path $\mathcal{C}(s)$ is a purely geometric description of the sequence of configurations achieved by the robot:

$$\mathcal{C}\colon \underbrace{[0,1]}_{s \in [0,1]\text{: scalar path parameter}} \to \mathbb{C}$$

$s \in [0,1]$: scalar path parameter
(0 at the start and 1 at the end of the path)

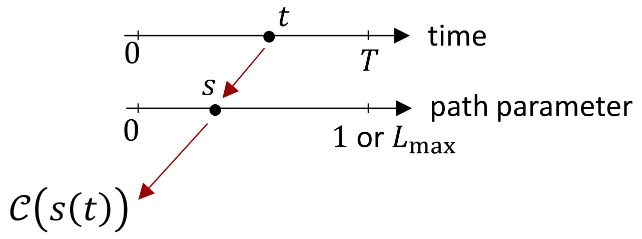
Robot's C-space



- As s increases from 0 to 1, the robot moves along the path.
- s can also be defined as arc length. In this case: $s \in [0, L_{\max}]$

Time Scaling $s(t)$ controls **how fast** the path is followed.
It specifies the times when those robot configurations are reached:

$$s\colon [0,T] \to [0,1]$$



Trajectory $\mathcal{C}(s(t))$ or $\mathcal{C}(t)$ specifies the robot configuration as a function of time, i.e., the combination of a **path** and a **time scaling**.

Trajectory Planning in Joint-Space vs Task-Space

- In the presence of **path constraints** (e.g., obstacles), **trajectory planning in the task space** may be advisable, because these constraints are typically better described in the task space.

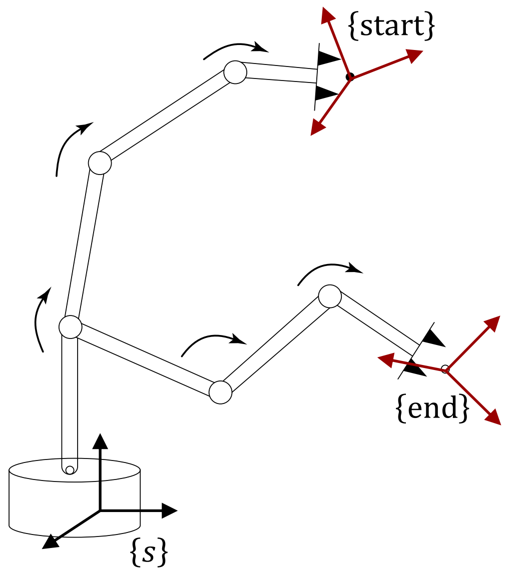
- In the presence of **joint limits**, **motion near singular configuration**, and **redundant DOFs**, trajectory planning in the task space may involve problems difficult to solve and it may be advisable to **plan trajectory in the joint space** to satisfy the constraints imposed on the trajectory.

- Since the control action on the robot is carried out in the joint space, if the trajectory planning is performed in the task space ($\boldsymbol{x}(t)$ or $\boldsymbol{T}(t)$), we have to use (online) **inverse kinematics** methods to reconstruct the corresponding time sequence of joint variables $\boldsymbol{\theta}(t)$ along the path.

Point-to-Point Path Planning

Point-to-Point Motion

Point-to-Point motion is the simplest type of motion which is from rest at one configuration (start) to rest at another configuration (end).



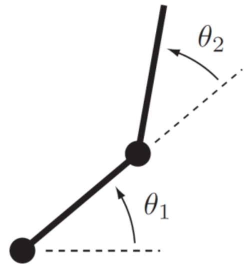
The path for point-to-point motion from a start configuration to an end configuration can be constructed in either **joint space** or **task space**.

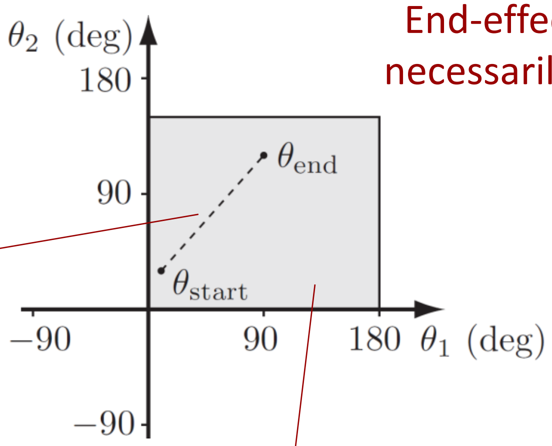
(1) Point-to-Point Straight-Line Path in Joint Space

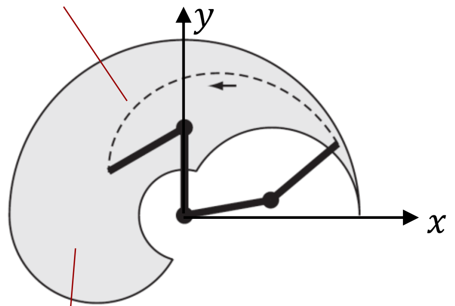
Straight-Line Path in Joint Space: $\boldsymbol{\theta}(s) = \boldsymbol{\theta}_{\text{start}} + s(\boldsymbol{\theta}_{\text{end}} - \boldsymbol{\theta}_{\text{start}})$

$s \in [0,1]$, $\boldsymbol{\theta} \in \mathbb{R}^n$ (n: number of joints)

Example: A 2R robot with joint limits: $0° \le \theta_1 \le 180°, 0° \le \theta_2 \le 150°$.







End-effector path is not necessarily a straight-line.

A straight line in **J-space** (joint limits and singularity avoidance are satisfied)

J-space is a convex set ($\boldsymbol{\theta}_{\min} \le \boldsymbol{\theta} \le \boldsymbol{\theta}_{\max}$)

Workspace is not necessarily a convex set.

(2.1) Point-to-Point Straight-Line Path in Task Space (in Cartesian Space $\mathbb{R}^3$)

- If the EE frame is represented by a minimum set of coordinates, i.e., $\boldsymbol{x} \in \mathbb{R}^r$:

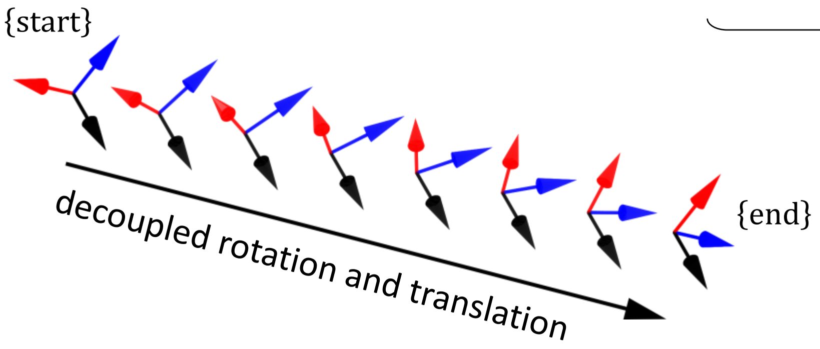
$$\boldsymbol{x}(s) = \boldsymbol{x}_{\text{start}} + s(\boldsymbol{x}_{\text{end}} - \boldsymbol{x}_{\text{start}})$$

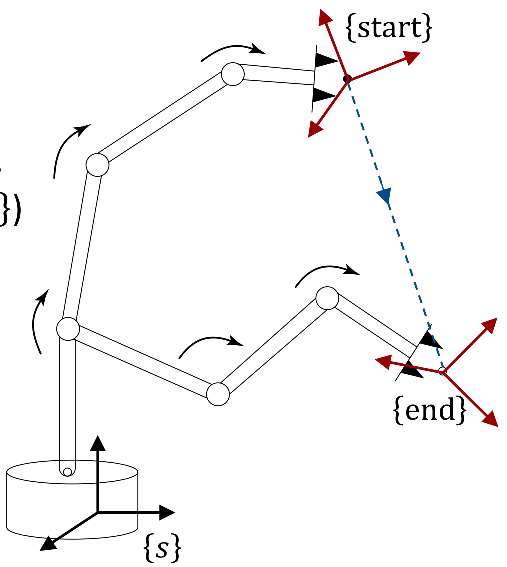
- If the EE frame is represented by position vector $\boldsymbol{p} \in \mathbb{R}^3$ and the rotation matrix $\boldsymbol{R} \in SO(3)$:

$$\begin{cases} \boldsymbol{p}(s) = \boldsymbol{p}_{\text{start}} + s(\boldsymbol{p}_{\text{end}} - \boldsymbol{p}_{\text{start}}) \\ \boldsymbol{R}(s) = \underbrace{\boldsymbol{R}_{\text{start}} \exp\Big(\underbrace{\log\big(\underbrace{\boldsymbol{R}_{\text{start}}^{\mathrm{T}}\boldsymbol{R}_{\text{end}}}_{\boldsymbol{R}_{\text{start,end}}}\big)}_{[\hat{\boldsymbol{\omega}}_{\text{start}}]\phi} s\Big)}_{\boldsymbol{R}_{\text{start,current}}(s)} \end{cases}$$

(axis of rotation is constant in {start})

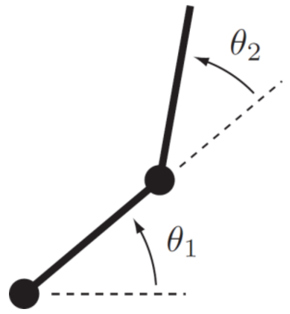
(A Straight-Line Path in $SO(3)$)



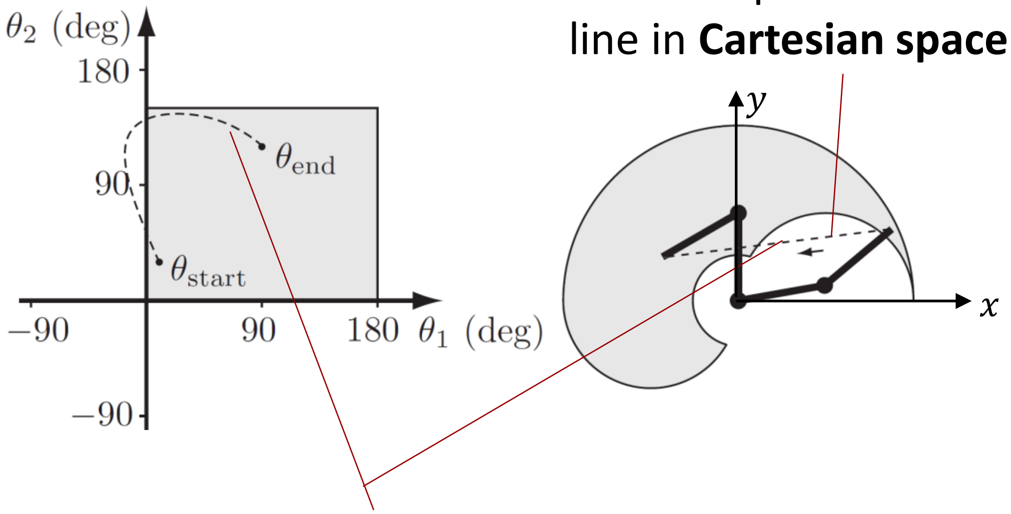


(2.1) Point-to-Point Straight-Line Path in Task Space (in Cartesian Space $\mathbb{R}^3$) (cont.)

Example: A 2R robot with joint limits: $0° \leq \theta_1 \leq 180°, 0° \leq \theta_2 \leq 150°$.



End-effector path is a straight-line in **Cartesian space**.

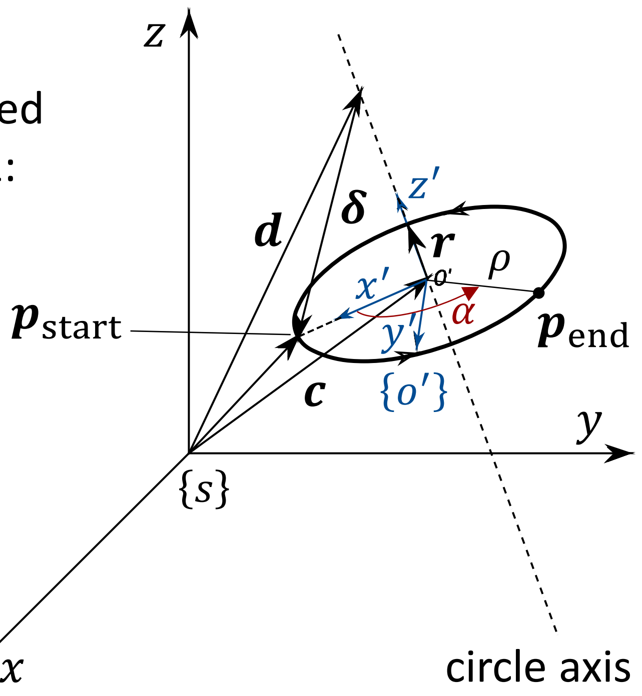


- The path in the J-space may violate the **joint limits**.
- The path may pass near a **kinematic singularity** (where joint velocities become large).

(2.1') Point-to-Point Circular Path in Task Space (in Cartesian Space $\mathbb{R}^3$)

Path corresponding motion of position vector $\boldsymbol{p} \in \mathbb{R}^3$ of EE along a circle from start point $\boldsymbol{p}_{\text{start}}$ by an angle α (measured from x'-axis of $\{o'\}$) to reach $\boldsymbol{p}_{\text{end}}$ when s goes from 0 to 1:

$$\boldsymbol{p}(s) = \boldsymbol{c} + \boldsymbol{R}_{so'} \begin{bmatrix} \rho\cos(\alpha s) \\ \rho\sin(\alpha s) \\ 0 \end{bmatrix}$$



$\boldsymbol{d}$: a point along the circle axis
$\boldsymbol{r}$: unit vector of the circle axis
$\boldsymbol{c}$: center of the circle
ρ: radius of the circle
$\{o'\}$: a frame at the center of the circle

$$\boldsymbol{\delta} = \boldsymbol{p}_{\text{start}} - \boldsymbol{d}$$

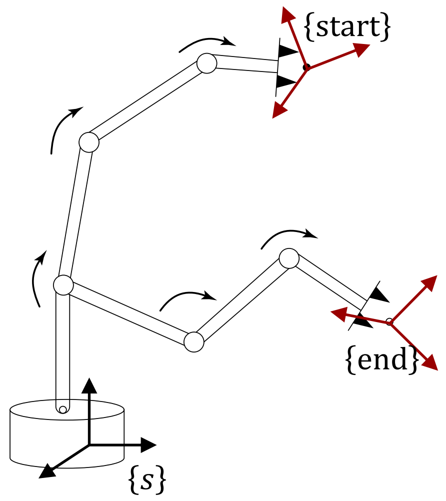
$$\boldsymbol{c} = \boldsymbol{d} + (\boldsymbol{\delta}^T \boldsymbol{r})\boldsymbol{r}$$

$$\rho = \|\boldsymbol{p}_{\text{start}} - \boldsymbol{c}\|$$

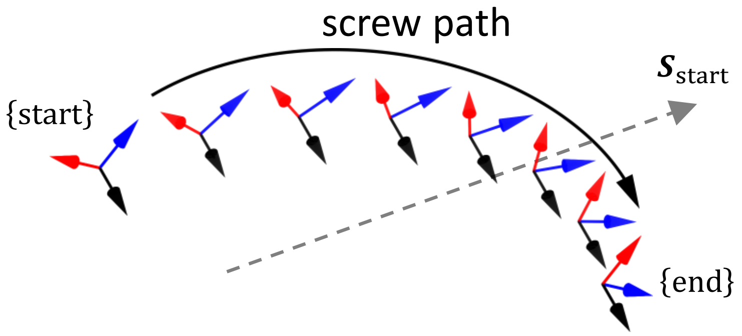
(2.2) Point-to-Point Straight-Line Path in Task Space (in SE(3))

If EE frame is represented by $\boldsymbol{T} = (\boldsymbol{R}, \boldsymbol{p}) \in SE(3)$:

$$\underbrace{\boldsymbol{T}(s)}_{\text{(A Straight-Line Path in } SE(3))} = \boldsymbol{T}_{\text{start}} \exp(\underbrace{\underbrace{\log(\underbrace{\boldsymbol{T}_{\text{start}}^{-1}\boldsymbol{T}_{\text{end}}}_{\boldsymbol{T}_{\text{start,end}}})}_{[\boldsymbol{S}_{\text{start}}]\phi}\, s}_{\boldsymbol{T}_{\text{start,current}}(s)})$$



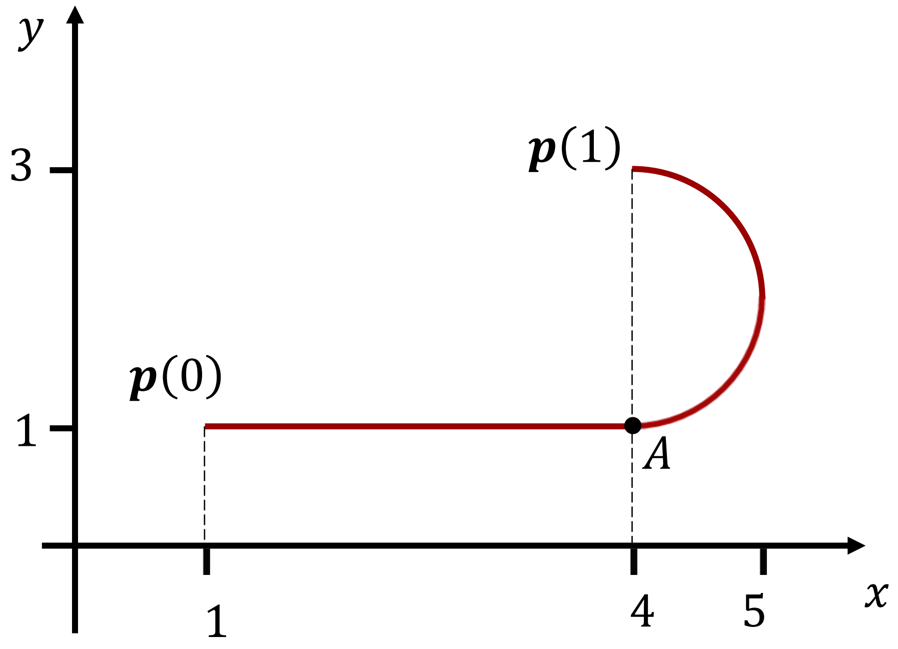
This path is equivalent to a constant screw motion of the EE frame (simultaneous rotation about and translation along a fixed screw axis $\boldsymbol{S}_{\text{start}}$) in Cartesian space $\mathbb{R}^3$.



Note: The origin of the EE frame does not generally follow a straight line in Cartesian space $\mathbb{R}^3$.

Example

Give an expression for the path $\boldsymbol{p}(s) = \big(x(s), y(s)\big) \in \mathbb{R}^2$, $s \in [0,1]$. Assume that point A is at $s = 0.5$.



Time Scaling a Path

Time Scaling a Path

A time scaling $s(t)$ of a path should ensure that the motion is appropriately smooth, and it should satisfy any constraints on joint velocities, accelerations, or torques or EE velocities and accelerations. It may also consider an optimality criteria (min total time, min energy,…).

∗ Joint velocities and accelerations for straight-line path in joint space:

$$\dot{\boldsymbol{\theta}} = \frac{d\boldsymbol{\theta}}{ds}\dot{s} = \dot{s}(\boldsymbol{\theta}_{\text{end}} - \boldsymbol{\theta}_{\text{start}}) \qquad \ddot{\boldsymbol{\theta}} = \frac{d\boldsymbol{\theta}}{ds}\ddot{s} + \frac{d^2\boldsymbol{\theta}}{ds^2}\dot{s}^2 = \ddot{s}(\boldsymbol{\theta}_{\text{end}} - \boldsymbol{\theta}_{\text{start}})$$

∗ EE velocities and accelerations for straight-line path in task space parametrized by a minimum set of coordinates $\boldsymbol{x} \in \mathbb{R}^r$:

$$\dot{\boldsymbol{x}} = \frac{d\boldsymbol{x}}{ds}\dot{s} = \dot{s}(\boldsymbol{x}_{\text{end}} - \boldsymbol{x}_{\text{start}}) \qquad \ddot{\boldsymbol{x}} = \frac{d\boldsymbol{x}}{ds}\ddot{s} + \frac{d^2\boldsymbol{x}}{ds^2}\dot{s}^2 = \ddot{s}(\boldsymbol{x}_{\text{end}} - \boldsymbol{x}_{\text{start}})$$

The most common methods for time-scaling $s(t)$:

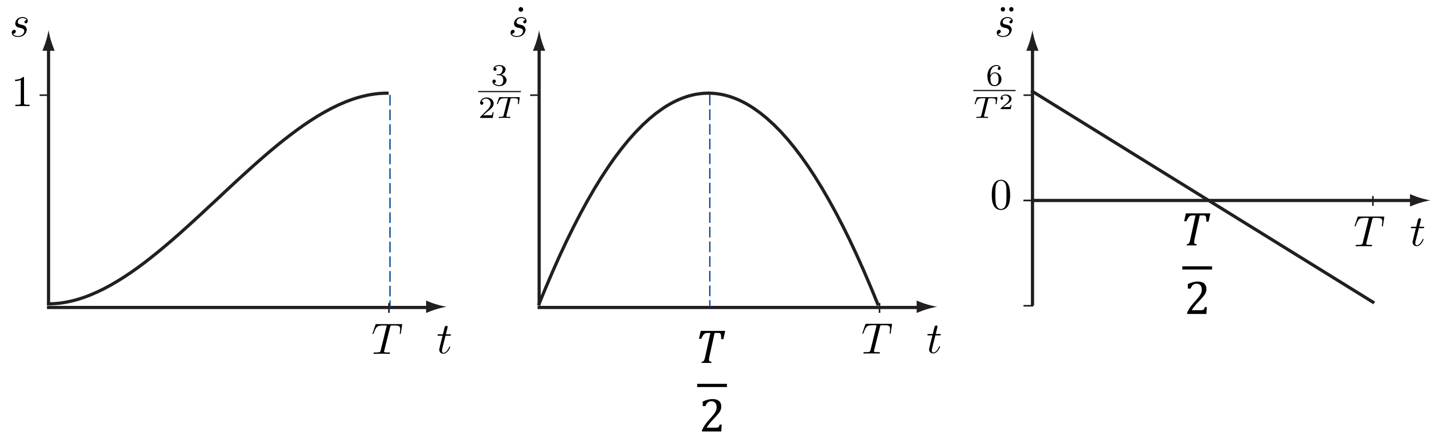
1. 3rd-Order (Cubic) Polynomial Position Profile
2. 5th-Order (Quintic) Polynomial Position Profile
3. Trapezoidal Velocity Profile
4. S-Curve Velocity Profile

1. 3rd-Order (Cubic) Polynomial Position Profile

Time scaling $s(t)$ using 3rd-order (cubic) polynomial position profile with the motion time T and rest-to-rest:

$$s(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \xrightarrow[\text{(constraints)}]{\begin{array}{c} s(0)=0 \\ \dot{s}(0)=0 \\ s(T)=1 \\ \dot{s}(T)=0 \end{array}} s(t) = \left(\frac{3}{T^2}\right) t^2 + \left(-\frac{2}{T^3}\right) t^3$$

$$t \in [0, T]$$



1. 3rd-Order (Cubic) Polynomial Position Profile (cont.)

For example, for a straight-line path in joint space (i.e., $\boldsymbol{\theta}(s) = \boldsymbol{\theta}_{\text{start}} + s(\boldsymbol{\theta}_{\text{end}} - \boldsymbol{\theta}_{\text{start}})$):

$$\boldsymbol{\theta}(t) = \boldsymbol{\theta}_{\text{start}} + \left(\frac{3t^2}{T^2} - \frac{2t^3}{T^3}\right)(\boldsymbol{\theta}_{\text{end}} - \boldsymbol{\theta}_{\text{start}}) \longrightarrow \begin{cases} \dot{\boldsymbol{\theta}}(t) = \left(\frac{6t}{T^2} - \frac{6t^2}{T^3}\right)(\boldsymbol{\theta}_{\text{end}} - \boldsymbol{\theta}_{\text{start}}) \\ \ddot{\boldsymbol{\theta}}(t) = \left(\frac{6}{T^2} - \frac{12t}{T^3}\right)(\boldsymbol{\theta}_{\text{end}} - \boldsymbol{\theta}_{\text{start}}) \end{cases}$$

$$\dot{\boldsymbol{\theta}}_{\max} = \dot{\boldsymbol{\theta}}\Big|_{t=T/2} = \frac{3}{2T}(\boldsymbol{\theta}_{\text{end}} - \boldsymbol{\theta}_{\text{start}})$$

(maximum joint velocities)

$$\ddot{\boldsymbol{\theta}}_{\max/\min} = \ddot{\boldsymbol{\theta}}\Big|_{t=0 \text{ or } T} = \pm\left|\frac{6}{T^2}(\boldsymbol{\theta}_{\text{end}} - \boldsymbol{\theta}_{\text{start}})\right|$$

(maximum joint accelerations and decelerations)

Note: If there are given limits on the maximum joint velocities and accelerations (i.e., $|\dot{\boldsymbol{\theta}}| \le \dot{\boldsymbol{\theta}}_{\text{limit}}$, $|\ddot{\boldsymbol{\theta}}| \le \ddot{\boldsymbol{\theta}}_{\text{limit}}$), we can solve for the minimum possible motion time T that satisfies both constraints.

Note: For a straight-line path in task space parametrized by a minimum set of coordinates $\boldsymbol{x} \in \mathbb{R}^r$, simply replace $\boldsymbol{\theta}$, $\dot{\boldsymbol{\theta}}$, and $\ddot{\boldsymbol{\theta}}$ by $\boldsymbol{x}$, $\dot{\boldsymbol{x}}$, and $\ddot{\boldsymbol{x}}$.

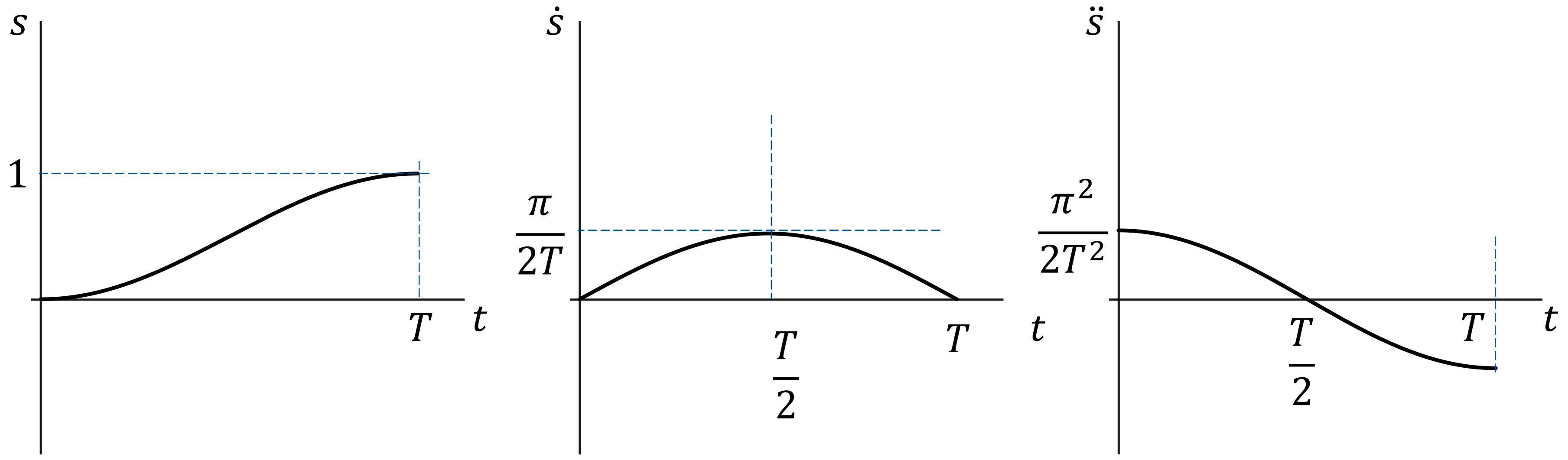
1’. A Trigonometric Alternative

This trigonometric function satisfy all the rest-to-rest constraints:

$$s(t) = \frac{1-\cos\frac{\pi t}{T}}{2}$$

$$t \in [0, T]$$

$$s(0) = 0$$
$$\dot{s}(0) = 0$$
$$s(T) = 1$$
$$\dot{s}(T) = 0$$



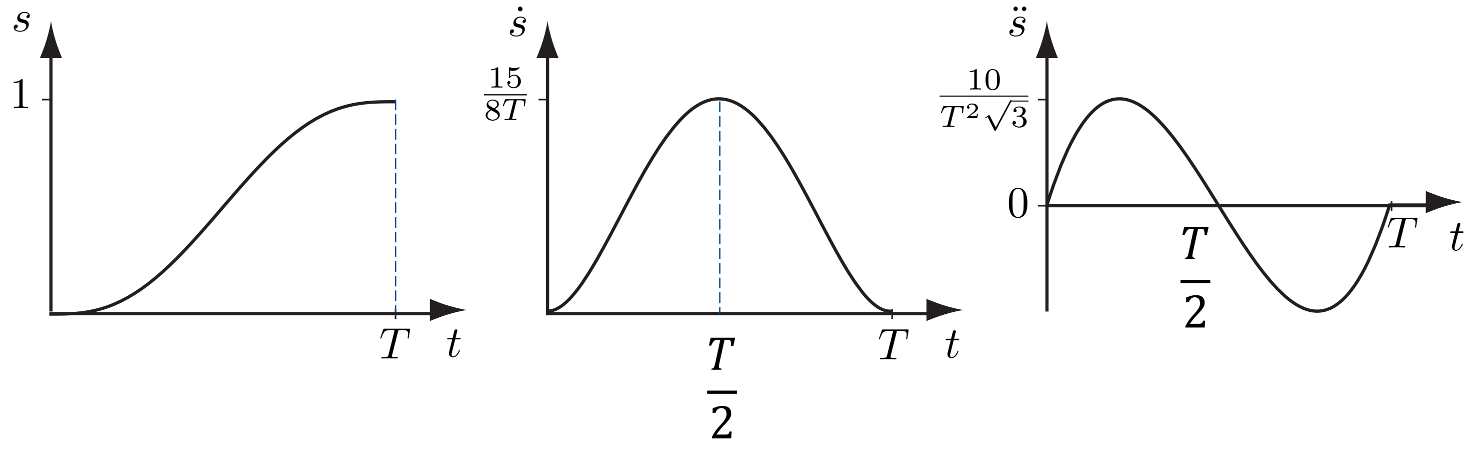
2. 5th-Order (Quintic) Polynomial Position Profile

The discontinuous jump in acceleration at both $t = 0$ and $t = T$ of the 3rd-order polynomial position profile (which implies an infinite jerk d^3s/dt^3) may cause <u>vibration of the robot</u>.

This problem can be solved by adding two constraints $\ddot{s}(0) = \ddot{s}(T) = 0$ and using a 5th-order (quintic) polynomial position profile for time scaling as

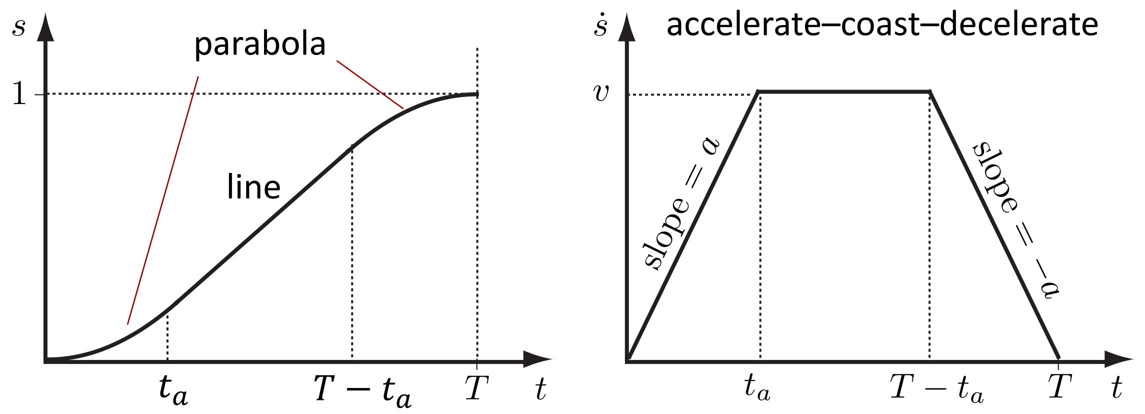
$$s(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5 \qquad t \in [0, T]$$

$$a_0 = 0, a_1 = 0, a_2 = 0, a_3 = \frac{10}{T^3}, a_4 = -\frac{15}{T^4}, a_5 = \frac{6}{T^5} \qquad \text{(rest-to-rest)}$$



3. Trapezoidal Velocity Profile

This motion consists of a constant acceleration phase $\ddot{s} = a$ of time t_a, followed by a constant velocity phase $\dot{s} = v$ of time $T - 2t_a$, followed by a constant deceleration phase $\ddot{s} = -a$ of time t_a.



$$\begin{cases} t_a = \dfrac{v}{a} \\ \displaystyle\int_0^T \dot{s}\, dt = s(T) = 1 \end{cases}$$

Disadvantage: It is not as smooth as the 3rd-order position profile time scaling.

Advantage: For example, in joint space, if there are known constant limits on the joint velocities $\dot{\boldsymbol{\theta}}_{\text{limit}} \in \mathbb{R}^n$ and joint accelerations $\ddot{\boldsymbol{\theta}}_{\text{limit}} \in \mathbb{R}^n$, this motion with the largest v and a satisfying

$$|(\boldsymbol{\theta}_{\text{end}} - \boldsymbol{\theta}_{\text{start}})v| \le \dot{\boldsymbol{\theta}}_{\text{limit}}$$
$$|(\boldsymbol{\theta}_{\text{end}} - \boldsymbol{\theta}_{\text{start}})a| \le \ddot{\boldsymbol{\theta}}_{\text{limit}}$$

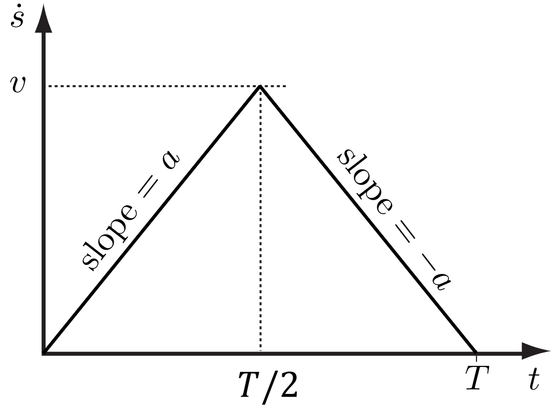
is the fastest straight-line motion possible (i.e., minimum total time T).

3. Trapezoidal Velocity Profile (cont.)

After choosing v and a:

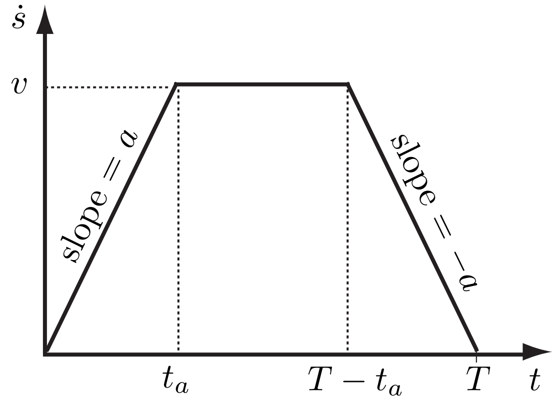
- If $v^2/a \geq 1$, the motion never reaches the velocity v (or reaches only at $T/2$) and the velocity profile is triangular.
- If $v^2/a < 1$, the motion reaches the velocity v and the velocity profile is trapezoidal:

accelerate–decelerate "bang-bang" motion



$$s(t) = \begin{cases} at^2/2 & 0 \leq t \leq v/a \\ vt - \dfrac{v^2}{2a} & v/a < t \leq T - v/a \\ \dfrac{2avT - 2v^2 - a^2(t-T)^2}{2a} & T - v/a < t \leq T \end{cases}$$

$$\dot{s}(t) = \begin{cases} at & 0 \leq t \leq v/a \\ v & v/a < t \leq T - v/a \\ -a(t-T) & T - v/a < t \leq T \end{cases} \qquad \ddot{s}(t) = \begin{cases} a & 0 \leq t \leq v/a \\ 0 & \dfrac{v}{a} < t \leq T - v/a \\ -a & T - v/a < t \leq T \end{cases}$$



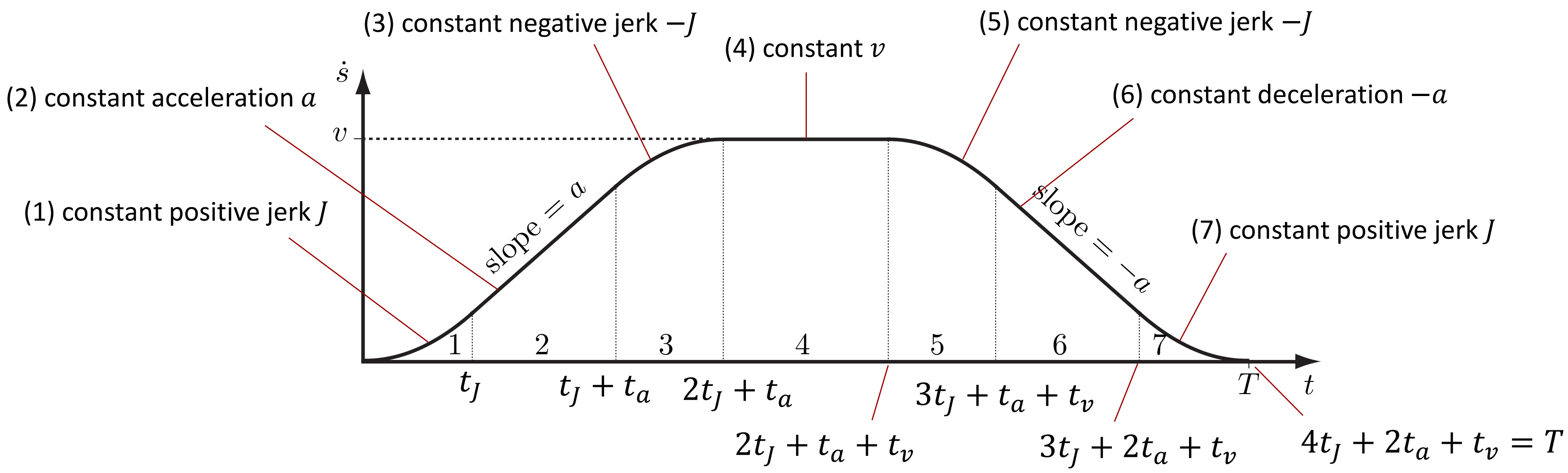
Note: Only two of v, a, and T can be chosen independently, since they must satisfy $s(T) = 1$.

* If v, a are given, T is derived from $\int_0^T \dot{s}\, dt = s(T) = 1$ as: $T = (a + v^2)/(va)$

4. S-Curve Velocity Profile

The discontinuous jump in acceleration at $t \in \{0, t_a, T - t_a, T\}$ of the trapezoidal velocity profile (which implies an infinite jerk d^3s/dt^3) may cause vibration of the robot.

This problem can be solved by using a smoother S-curve velocity profile for time scaling.

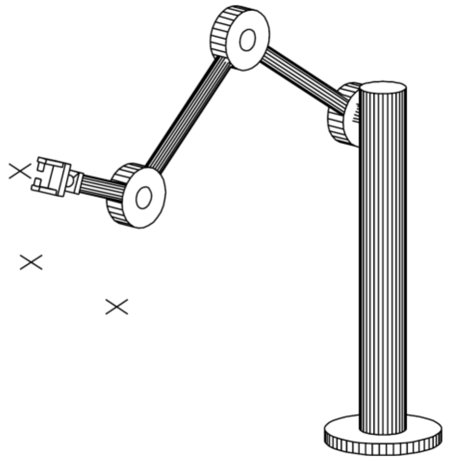

(3) constant negative jerk $-J$
(4) constant v
(5) constant negative jerk $-J$
(2) constant acceleration a
(6) constant deceleration $-a$
(1) constant positive jerk J
(7) constant positive jerk J
$\dot{s}$
v
slope = a
slope = $-a$
1
2
3
4
5
6
7
t_J
$t_J + t_a$
$2t_J + t_a$
$2t_J + t_a + t_v$
$3t_J + t_a + t_v$
$3t_J + 2t_a + t_v$
T
t
$4t_J + 2t_a + t_v = T$

It is fully specified by 7 quantities: $t_J, t_a, t_v, T, J, a, v$. The constraints are $t_J J = a$, $J t_J^2 + a t_a = v$, $4t_J + 2t_a + t_v = T$, $s(T) = 1$. Therefore, 3 of the 7 quantities can be specified independently.

Polynomial Via Point Trajectories

Polynomial Via Point Trajectories

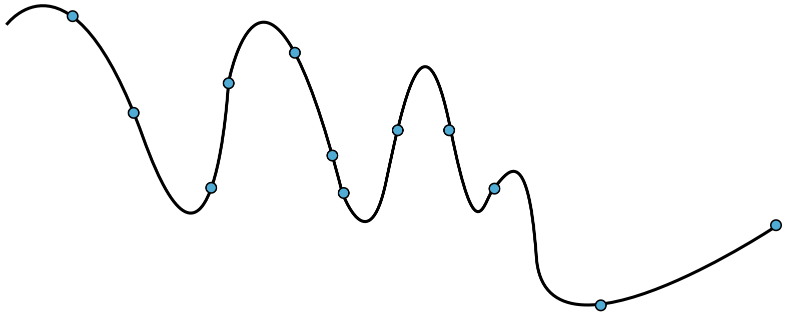
If the goal is that the trajectories pass through a sequence of via points at specified times (e.g., to avoid obstacles or perform a specific task) without a strict specification on the shape of path between consecutive points, a simple solution is to use polynomial interpolation to directly find the trajectories in the joint space $\boldsymbol{\theta}(t)$ or in the task space $\boldsymbol{x}(t)$ without first specifying a path $\boldsymbol{c}(s)$ and then a time scaling $s(t)$.



Polynomial Via Point Trajectories

Using a single polynomial $\boldsymbol{\theta}(t)$ or $\boldsymbol{x}(t)$ which passes through all N via points has the following disadvantages (N points $\rightarrow$ N constraints $\rightarrow$ $(N-1)$-order polynomial):

- As the order of a polynomial increases, its oscillatory behavior increases, and this may lead to trajectories which are not natural for the manipulator.
- Polynomial coefficients depend on all the assigned points; thus, if it is desired to change a point, all of them have to be recomputed.



These drawbacks can be overcome if a suitable number of low-order interpolating polynomials, continuous at the path points, are considered in place of a single high-order polynomial. The most common methods:

1. Using Cubic Polynomials
2. Using Linear and Quadratic Polynomials (B-Spline)

1. Polynomial Via Point Trajectories Using Cubic Polynomials

Let's focus on finding a single trajectory $\beta(t)$:
Note: $\beta(t)$ can be $\theta_i(t)$ in J-space or $x_i(t)$ in T-space.

$$\beta(t) = \begin{cases} \pi_1(t) & t_1 = 0 \le t \le t_2 \\ \vdots & \vdots \\ \pi_{N-1}(t) & t_{N-1} \le t \le t_N = T \end{cases}$$

$$\pi_k(t) = a_{k,0} + a_{k,1}t + a_{k,2}t^2 + a_{k,3}t^3$$
$$t_k \le t \le t_{k+1}$$

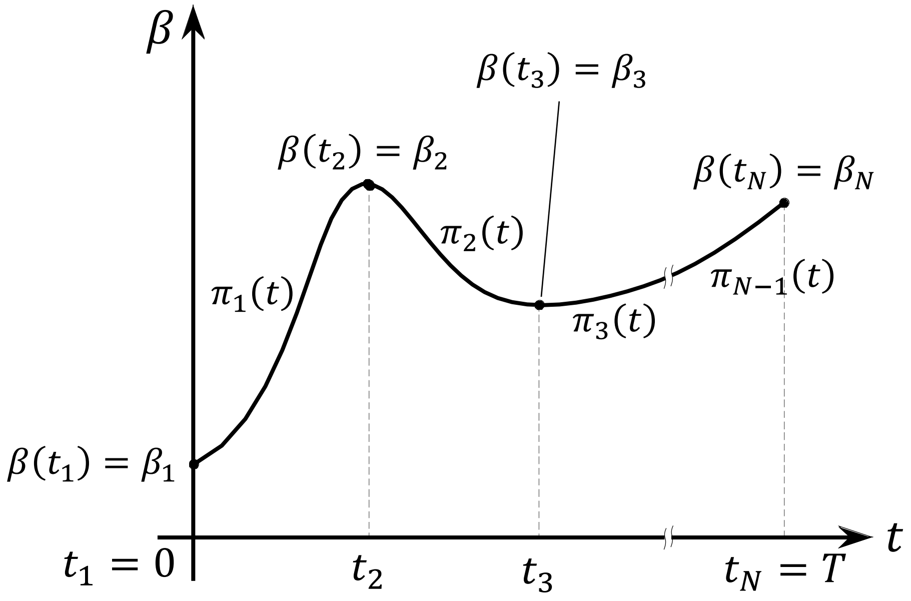
# of via points: N

# of segments: $N - 1$

Position constraints:

$$\beta(t_k) = \beta_k \qquad k \in \{1, \dots, N\}$$

where $t_1 = 0$ and $t_N = T$.



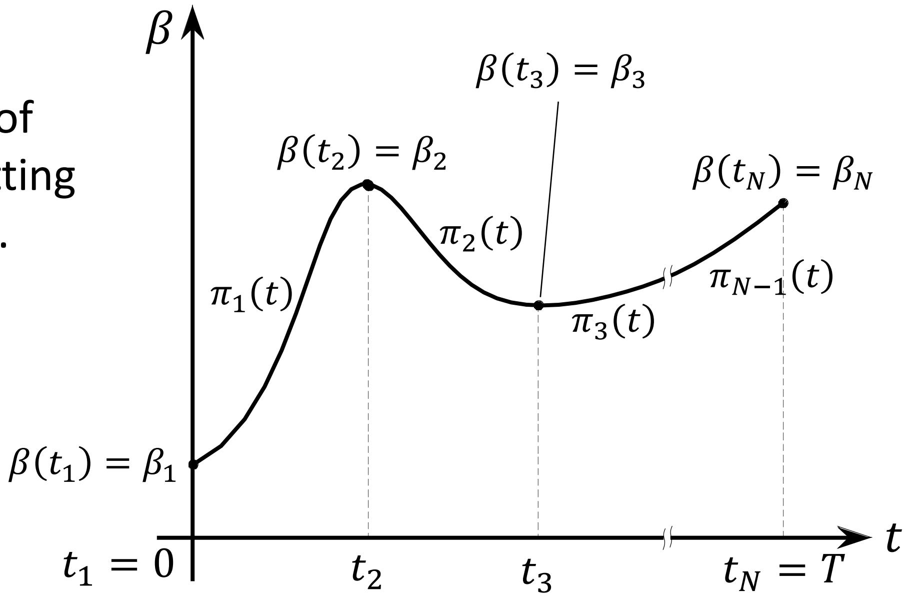
(1.a) Cubic Polynomials with Imposed Velocities at Via Points

The desired velocity at each via point is given: $\dot{\beta}(t_k) = \dot{\beta}_k \qquad k \in \{1, \dots, N\}$

$$\begin{cases} \pi_k(t_k) = \beta_k \\ \pi_k(t_{k+1}) = \beta_{k+1} \\ \dot{\pi}_k(t_k) = \dot{\beta}_k \\ \dot{\pi}_k(t_{k+1}) = \dot{\beta}_{k+1} \end{cases} \qquad k \in \{1, \dots, N-1\} \quad \Rightarrow$$

$N - 1$ systems of four equations that can be solved <u>independently</u> or <u>simultaneously</u> by forming $\boldsymbol{Ax} = \boldsymbol{b}$.

- Typically, $\dot{\beta}(0) = \dot{\beta}(T) = 0$ and continuity of velocity at the path points is ensured by setting $\dot{\pi}_k(t_{k+1}) = \dot{\pi}_{k+1}(t_{k+1}), k \in \{1, \dots, N-2\}$.



Note: The approach is easily generalized to the use of 5th-order polynomials by having accelerations at the via points.

(1.b) Cubic Polynomials with Computed Velocities at Via Points

If the desired velocity at each via point is not given, the velocity at each point can be computed by the assumption that the trajectory between two consecutive points is a linear segment. Thus, the velocities $\dot{\beta}_k$ can be computed as:

$$\dot{\beta}_1 = 0$$

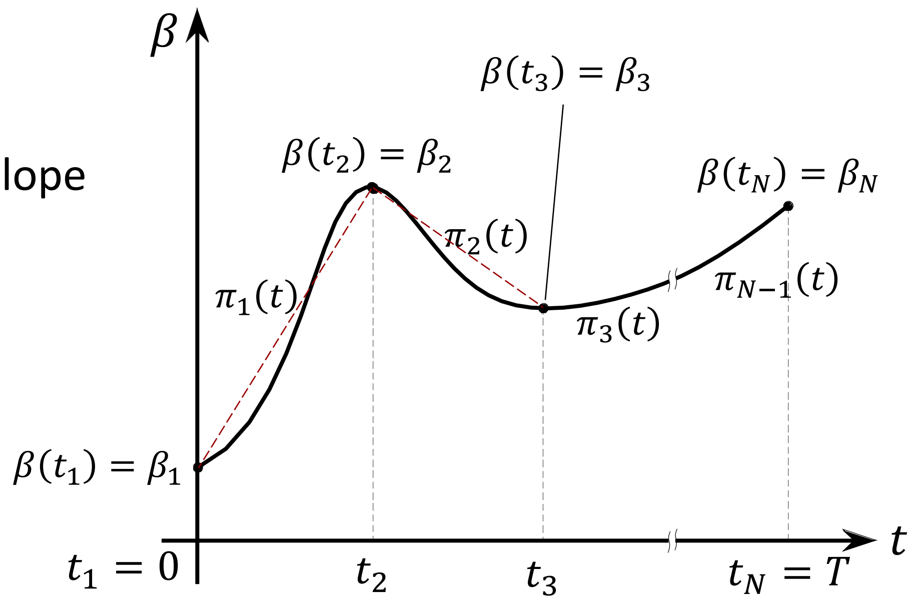
$$\dot{\beta}_k = \begin{cases} 0 & \operatorname{sgn}(v_k) \neq \operatorname{sgn}(v_{k+1}) \\ \frac{1}{2}(v_k + v_{k+1}) & \operatorname{sgn}(v_k) = \operatorname{sgn}(v_{k+1}) \end{cases} \qquad k \in \{2, \ldots, N-1\}$$

$$\dot{\beta}_N = 0,$$

where $v_k = (\beta_k - \beta_{k-1})/(t_k - t_{k-1})$ is the slope of the segment in the time interval $[t_{k-1}, t_k]$.

Then, after computing velocity at each via point, we use the method in (1.a).

Drawback: At some via points the velocity is 0.



(1.c) Cubic Polynomials with Continuous Velocities and Accelerations at Via Points (Splines)

Both (1.a) and (1.b) do not ensure continuity of accelerations at the via points. In this method:

$$\begin{cases} \pi_k(t_k) = \beta_k \\ \pi_k(t_k) = \pi_{k-1}(t_k) \\ \dot{\pi}_k(t_k) = \dot{\pi}_{k-1}(t_k) \\ \ddot{\pi}_k(t_k) = \ddot{\pi}_{k-1}(t_k) \end{cases} \qquad k \in \{2, \dots, N-1\}$$

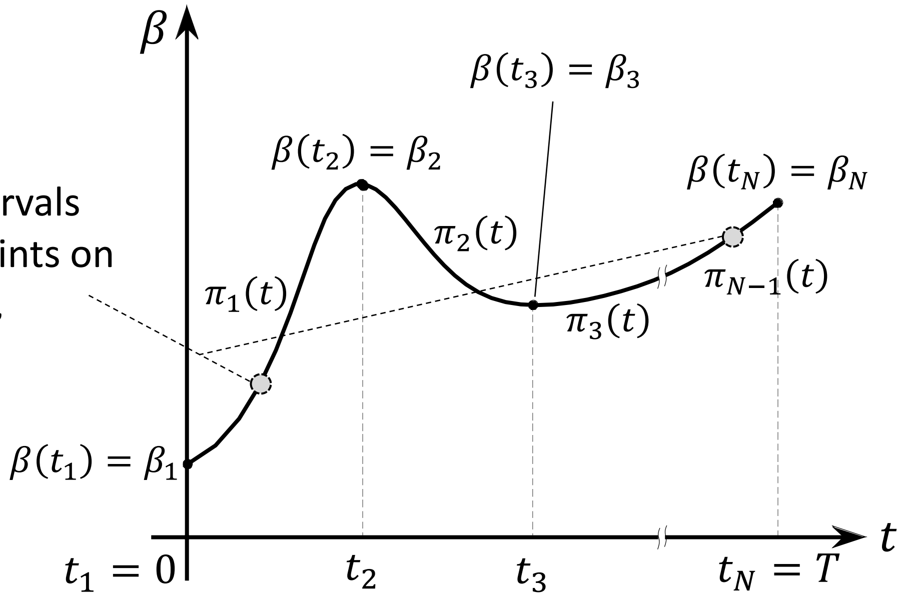
$$\pi_1(0) = \beta_1 \qquad \pi_{N-1}(T) = \beta_N$$
$$\dot{\pi}_1(0) = \dot{\beta}_1 \qquad \dot{\pi}_{N-1}(T) = \dot{\beta}_N$$
$$\ddot{\pi}_1(0) = \ddot{\beta}_1 \qquad \ddot{\pi}_{N-1}(T) = \ddot{\beta}_N$$

# of Cubic Polynomials: $N-1$, # of Unknowns: $4(N-1)$ $\neq$ # of Equations: $4(N-2)+6$!

3 Possible Solutions:

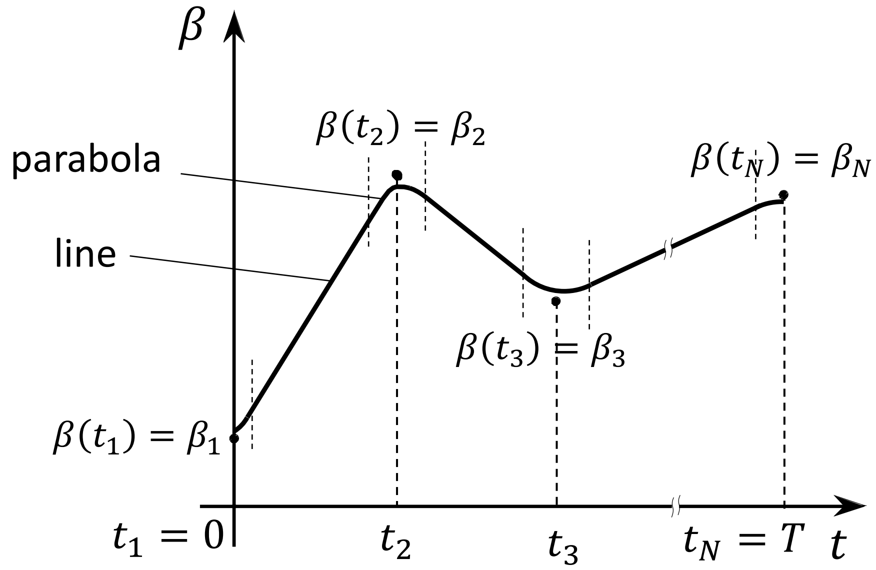
1. Eliminating $\ddot{\pi}_1(0) = \ddot{\beta}_1$ and $\ddot{\pi}_{N-1}(T) = \ddot{\beta}_N$.
2. Using 4th-order polynomials only for π_1 and π_{N-1}.
3. Introducing <u>two virtual points</u> arbitrarily in the intervals $[t_1, t_2]$ and $[t_{N-1}, t_N]$, for which continuity constraints on position, velocity and acceleration can be imposed, without specifying the actual positions:

   # of cubic polynomials: $N+1$
   # of unknowns: $4(N+1)$
   # of Equations: $4(N-2)+6+3+3$



2. Polynomial Via Point Trajectories Using Linear and Quadratic Polynomials (B-Spline)

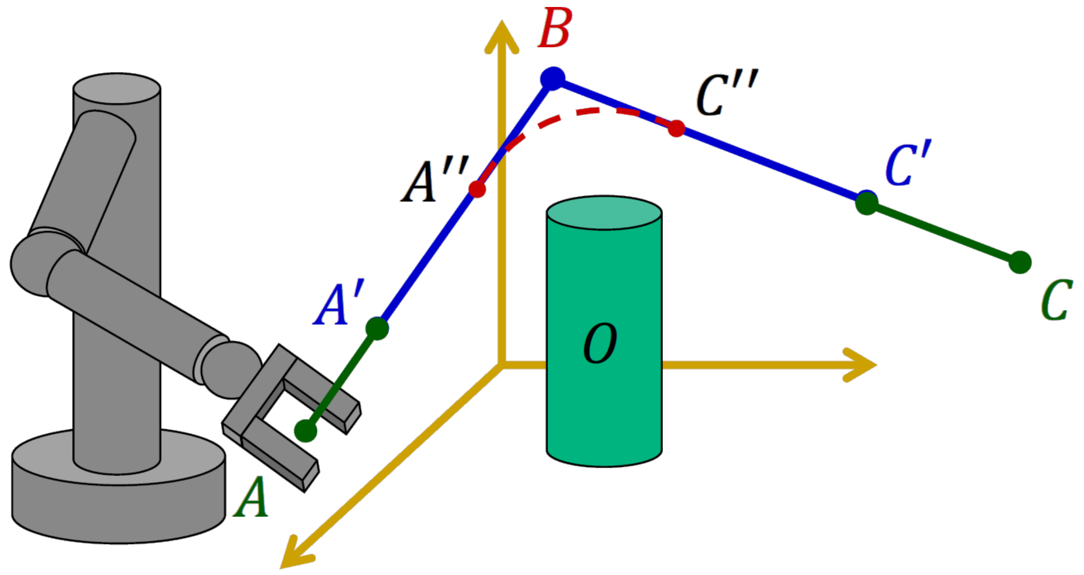
The entire trajectory is composed of a sequence of linear and quadratic polynomials.



- This is an application of the trapezoidal velocity profile law to the via points problem.
- Trajectory does not ensure continuity of accelerations at the via points.
- The path does not pass exactly through the via points. However, it stays within convex hull of the via points (this can be important to ensure that joint limits or workspace obstacles are respected).

2. Polynomial Via Point Trajectories Using Linear and Quadratic Polynomials (B-Spline) (cont.)

For example, to plan a trajectory in task space from A to C (rest-to-rest) that avoids the obstacle O, with $a \leq a_{\max}$ and $v \leq v_{\max}$, we can use this method to add a via point B sufficiently far from O and the designed trajectory continuously switch from line AB to BC without passing exactly through the via point B.



Remarks

Note: In this section, all the equations can be rewritten in the form $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ where unknown $\boldsymbol{x}$ is the vector of all coefficients. The solution to this system always exists and is unique.

Note: In general, the trajectory planning methods proposed in this section, can be applied either in the joint space or the task space by considering independently each component β of the vector $\boldsymbol{\theta}$ or $\boldsymbol{x}$.

- However, in the task space, when planning a trajectory for the three/four components of orientation, the resulting rotational motion may not be intuitive. Therefore, it is preferred to plan task space trajectories separately for position and orientation with the same time scaling and, if possible, use a straight-line path in $SO(3)$ for orientation as

$$\boldsymbol{R}(s) = \boldsymbol{R}_{\text{start}} \exp\left(\log\left(\boldsymbol{R}_{\text{start}}^{\mathrm{T}} \boldsymbol{R}_{\text{end}}\right) s\right)$$