Ch3: Minimum-Time Trajectory Generation

Time-Optimal Time Scaling

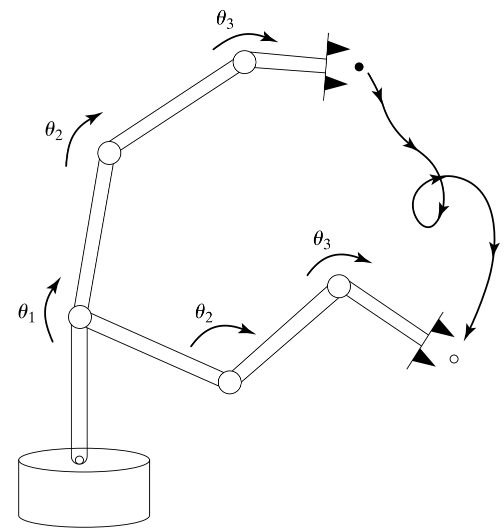
Path, Time Scaling, and Trajectory

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

$$\mathcal{C}: [0,1] \rightarrow \mathbb{C}$$

$s \in [0,1]$: 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.

Time Scaling $s(t)$ specifies the times when those robot configurations are reached:

$$s: [0, t_f] \rightarrow [0,1]$$

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.

Some Examples of Path Planning $c(s), s \in [0,1]$

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

$$\boldsymbol{\theta} \in \mathbb{R}^n$$

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

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

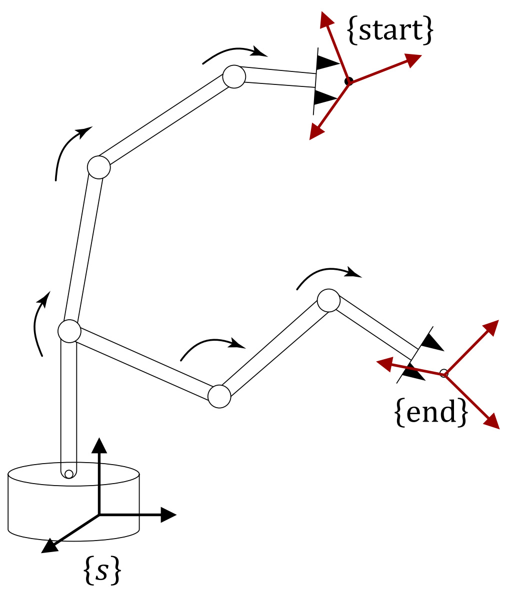
$\boldsymbol{x} \in \mathbb{R}^m$: minimum set of coordinates

or

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

$\boldsymbol{p} \in \mathbb{R}^3, \ \boldsymbol{R} \in SO(3)$

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



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

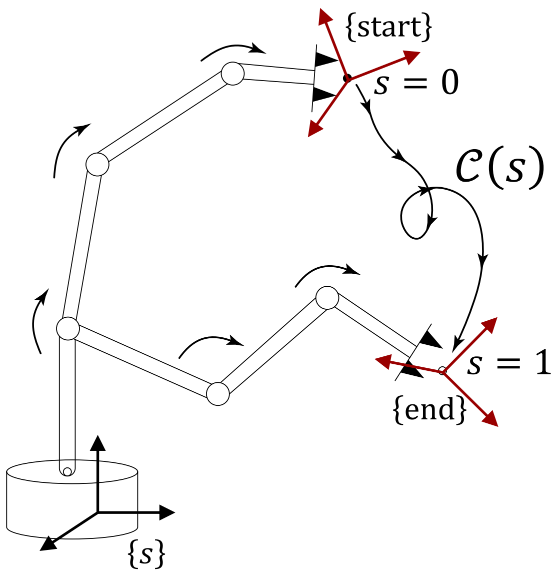
$$\boldsymbol{T}(s) = \boldsymbol{T}_{\text{start}} \exp\left(\log\left(\underbrace{\boldsymbol{T}_{\text{start}}^{-1} \boldsymbol{T}_{\text{end}}}_{\boldsymbol{T}_{\text{start,end}}}\right) s\right)$$

$\boldsymbol{T} = (\boldsymbol{R}, \boldsymbol{p}) \in SE(3)$

Time-Optimal Time Scaling

Consider a case where the **path** $\mathcal{C}(s)$, $s \in [0,1]$, is fully specified by the task or an obstacle-avoiding path planner. The **time-optimal time scaling** is finding a **time scaling** $s(t)$ that minimizes the time of motion along the path, subject to the robot's **actuator limits**.

A time-optimal trajectory maximizes the robot's productivity.



Actuation Constraints as a Function of s

In practice, the robot dynamics and joint actuator limits dependent on $(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$, thus, the maximum available velocities and accelerations change along the path.

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \dot{\boldsymbol{\theta}}^{\mathrm{T}}\boldsymbol{\Gamma}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}} + \boldsymbol{g}(\boldsymbol{\theta}) \qquad (1)$$

$$\tau_i^{\min}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) \le \tau_i \le \tau_i^{\max}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) \qquad i = 1, \dots, n \qquad \text{(actuation constraints)} \qquad (2)$$

A path $\mathcal{C}(s)$ can be always expressed in joint space $\boldsymbol{\theta}(s) \in \mathbb{R}^n$ using inverse kinematics. Thus,

$$\dot{\boldsymbol{\theta}} = \frac{d\boldsymbol{\theta}}{ds}\dot{s}, \qquad \ddot{\boldsymbol{\theta}} = \frac{d\boldsymbol{\theta}}{ds}\ddot{s} + \frac{d^2\boldsymbol{\theta}}{ds^2}\dot{s}^2$$

Dynamics along the path:

$$(1) \quad \rightarrow \qquad \boldsymbol{\tau} = \underbrace{\left(\boldsymbol{M}(\boldsymbol{\theta}(s))\frac{d\boldsymbol{\theta}}{ds}\right)}_{\boldsymbol{m}(s)\in\mathbb{R}^n}\ddot{s} + \underbrace{\left(\boldsymbol{M}(\boldsymbol{\theta}(s))\frac{d^2\boldsymbol{\theta}}{ds^2} + \left(\frac{d\boldsymbol{\theta}}{ds}\right)^{\mathrm{T}}\boldsymbol{\Gamma}(\boldsymbol{\theta}(s))\frac{d\boldsymbol{\theta}}{ds}\right)}_{\boldsymbol{c}(s)\in\mathbb{R}^n}\dot{s}^2 + \underbrace{\boldsymbol{g}(\boldsymbol{\theta}(s))}_{\boldsymbol{g}(s)\in\mathbb{R}^n}$$

$$= \boldsymbol{m}(s)\ddot{s} + \boldsymbol{c}(s)\dot{s}^2 + \boldsymbol{g}(s) \qquad (3)$$

Actuation Constraints as a Function of s

(2) $\rightarrow$ $\tau_i^{\min}(s,\dot{s}) \leq \tau_i \leq \tau_i^{\max}(s,\dot{s})$ (4)

(3), (4) $\rightarrow$ $\tau_i^{\min}(s,\dot{s}) \leq m_i(s)\ddot{s} + c_i(s)\dot{s}^2 + g_i(s) \leq \tau_i^{\max}(s,\dot{s})$ (5)

Let define $L_i(s,\dot{s})$ and $U_i(s,\dot{s})$ be the minimum and maximum $\ddot{s}$ satisfying (5):

- If $m_i(s) > 0$:

$$L_i(s,\dot{s}) = \frac{\tau_i^{\min}(s,\dot{s}) - c(s)\dot{s}^2 - g(s)}{m_i(s)}$$

$$U_i(s,\dot{s}) = \frac{\tau_i^{\max}(s,\dot{s}) - c(s)\dot{s}^2 - g(s)}{m_i(s)}$$

- If $m_i(s) < 0$:

$$L_i(s,\dot{s}) = \frac{\tau_i^{\max}(s,\dot{s}) - c(s)\dot{s}^2 - g(s)}{m_i(s)}$$

$$U_i(s,\dot{s}) = \frac{\tau_i^{\min}(s,\dot{s}) - c(s)\dot{s}^2 - g(s)}{m_i(s)}$$

By defining $L(s,\dot{s}) = \max_i L_i(s,\dot{s})$ and $U(s,\dot{s}) = \min_i U_i(s,\dot{s})$ as the lower and upper bounds on $\ddot{s}$ at $(s,\dot{s})$, (5) can be written as $L(s,\dot{s}) \leq \ddot{s} \leq U(s,\dot{s})$

Time-optimal Time-scaling Problem

Given a path $\boldsymbol{\theta}(s), s \in [0,1]$, an initial state $(s_0, \dot{s}_0) = (0,0)$, and a final state $\left(s_f, \dot{s}_f\right) = (1,0)$, find a monotonically increasing twice-differentiable time-scaling $s(t)$, s: $[0, t_f] \rightarrow [0,1]$ that

(a) satisfies: $s(0) = \dot{s}(0) = \dot{s}(t_f) = 0$ and $s(t_f) = 1$,

(b) minimizes the total travel time t_f along the path while respecting the actuator constraints:

$$L(s, \dot{s}) \leq \ddot{s} \leq U(s, \dot{s})$$

$\equiv \dot{s}(t) \geq 0$ (robot moves forward along the path)

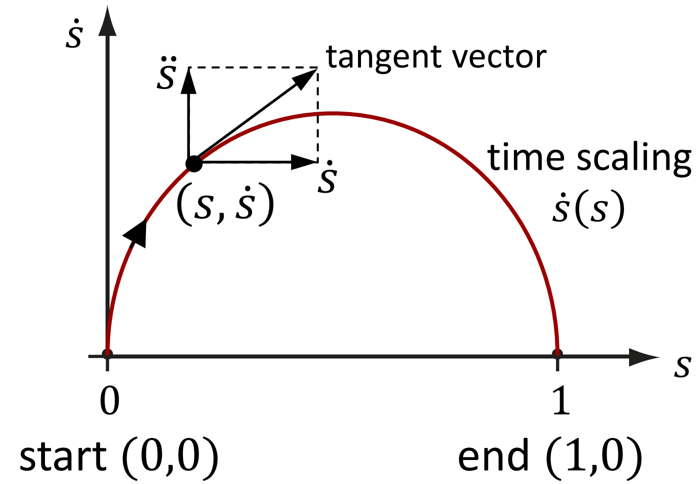
This problem is easily visualized in the $(s, \dot{s})$ **phase plane**.

$(s, \dot{s})$ Phase Plane

$(s, \dot{s})$ Phase Plane

$(s, \dot{s})$ **phase plane** is defined as a plane with s running from 0 to 1 on a horizontal axis and $\dot{s}$ on a vertical axis.

A time scaling $s(t)$ of the path is any curve $\dot{s}(s)$ in the phase plane that moves monotonically to the right from $(0,0)$ to $(1,0)$ in the top-right quadrant.

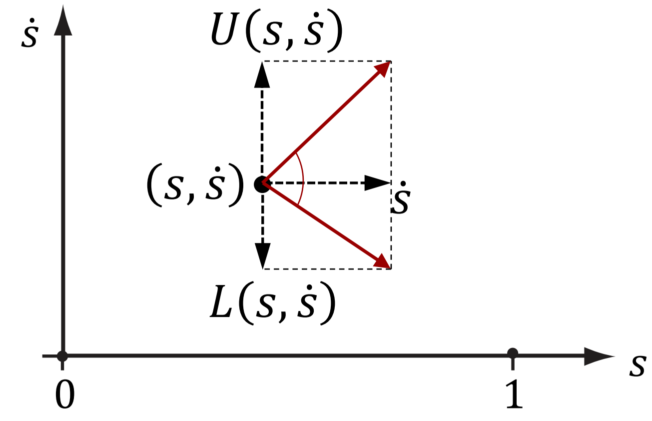


Among all these curves, we are looking for a time-optimal curve that satisfy the actuator/acceleration constraints $L(s, \dot{s}) \le \ddot{s} \le U(s, \dot{s})$.

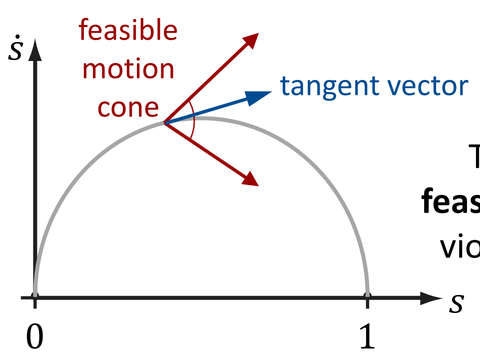
Feasible Motion Cone

By drawing the range of feasible accelerations $L(s,\dot{s}) \leq \ddot{s} \leq U(s,\dot{s})$ according to the dynamics at any state $(s,\dot{s})$, we find a cone called the **feasible motion cone**.

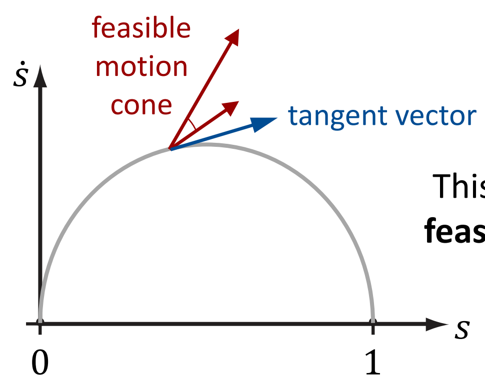
Note: Vector $\dot{s}$ is proportional to the height of the point along the $\dot{s}$ axis.



At each state $(s,\dot{s})$, the tangent vector to the time scaling must lie **inside feasible motion cone** to satisfy the actuator limits (or the acceleration constraints).



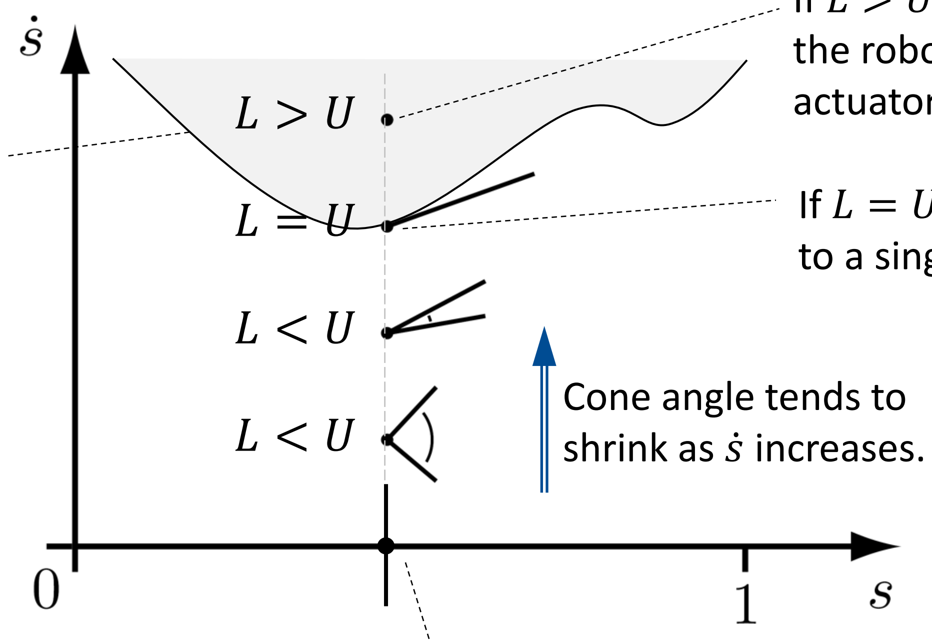
This time scaling is **feasible** since it does not violate actuator limits.



This time scaling is **not feasible** since it violates actuator limits.

Velocity Limit Curve

Let keep s constant but increase $\dot{s}$:



At states on velocity limit curve, only <u>a single acceleration</u> is possible; at states above this curve (inadmissible states), the robot leaves the path immediately; and at states below the curve, there is a cone of possible tangent vectors.

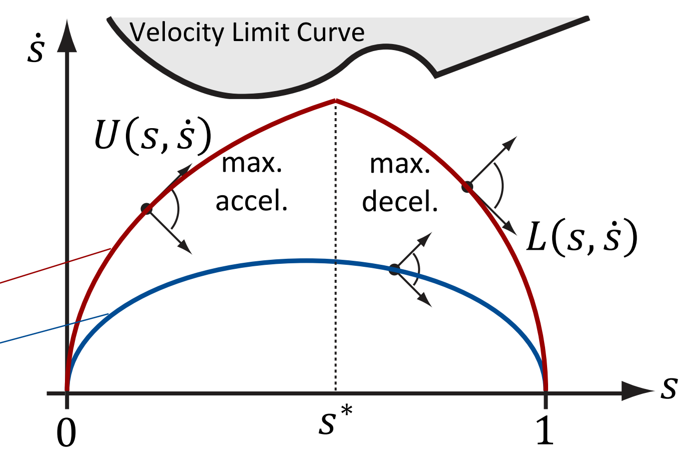
Bang-Bang Time Scaling

The total time of motion t_f can be written as

$$t_f = \int_0^{t_f} 1dt = \int_0^{t_f} \frac{ds}{ds}dt = \int_0^1 \frac{dt}{ds}ds = \int_0^1 \dot{s}^{-1}(s)ds$$

For a minimum-time motion, $\dot{s}^{-1}$ should be as small as possible, and therefore, $\dot{s}$ must be as large as possible, at all s, while still satisfying the acceleration constraints $L(s,\dot{s}) \le \ddot{s} \le U(s,\dot{s})$ and the boundary constraints $s(0) = \dot{s}(0) = \dot{s}(t_f) = 0, s(t_f) = 1$.

This implies that the time scaling must always operate either at the limit $U(s,\dot{s})$ (the upper edge of the motion cones) or at the limit $L(s,\dot{s})$ (the lower edge of the motion cones), and we should determine switching point s^* between these limits.

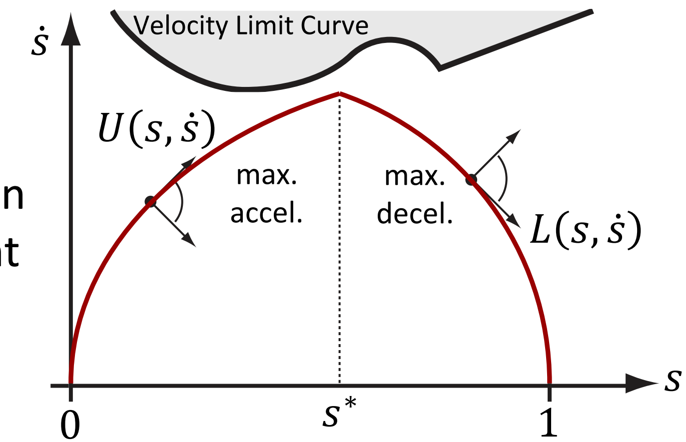


Time-optimal "bang-bang" time scaling

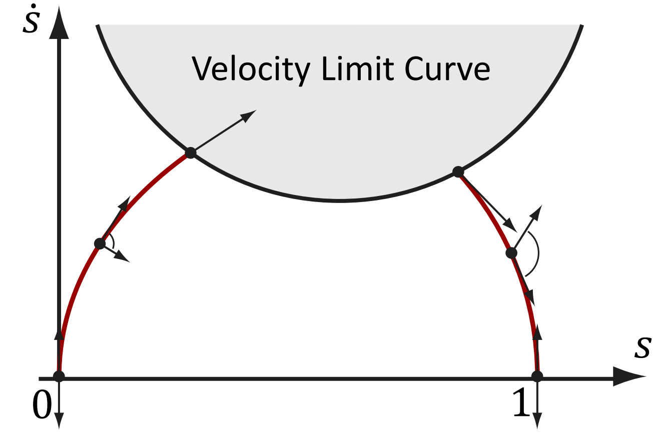
A non-optimal time scaling
(The tangent to the time scaling is always inside the motion cones, not on the edges.)

Bang-Bang Time Scaling

In general, the time scaling is calculated by numerically integrating $\ddot{s} = U(s, \dot{s})$ (the maximum possible accelerations) forward in s from $(0,0)$, integrating $\ddot{s} = L(s, \dot{s})$ (the maximum possible decelerations) backward in s from $(1,0)$, and finding the intersection (switching point s^*) of these curves.



However, in some cases, the existence of a velocity limit curve prevents a single-switch solution (two curves do not intersect and run into the velocity limit curve). In these cases, bang-bang control is not possible, and it requires an algorithm to find multiple switching points.



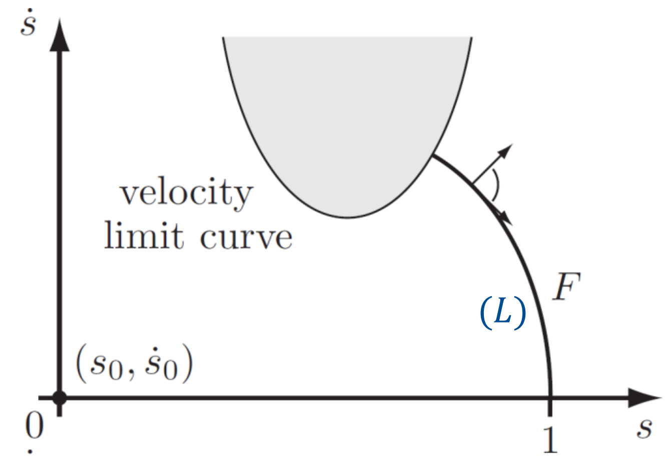
Note: When $\dot{s} = 0$ the system must accelerate before s can change, therefore, the curve must be normal to the s-axis when $\dot{s} = 0$.

Time-Scaling Algorithm

Time-Scaling Algorithm

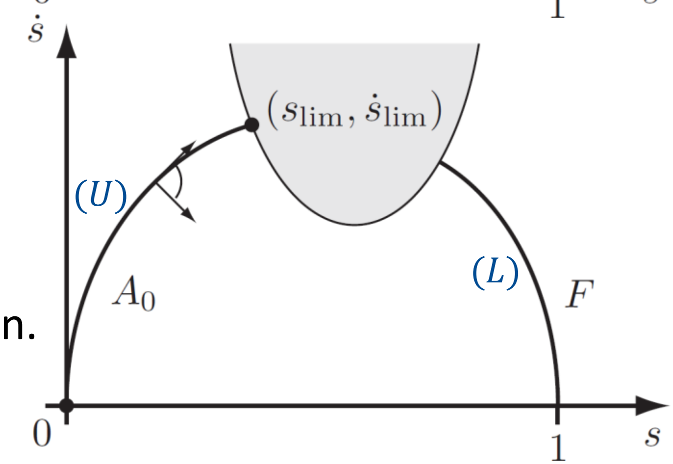
Since time-optimal trajectories consist of only maximum acceleration $U(s,\dot{s})$ and minimum acceleration $L(s,\dot{s})$, we need to find the switches between U and L:

Step 1: Integrate $\ddot{s} = L(s,\dot{s})$ backward in time from $(1{,}0)$ until (i) the velocity limit curve is penetrated $(L(s,\dot{s}) > U(s,\dot{s}))$ or (ii) $s = 0$. Call this curve F.



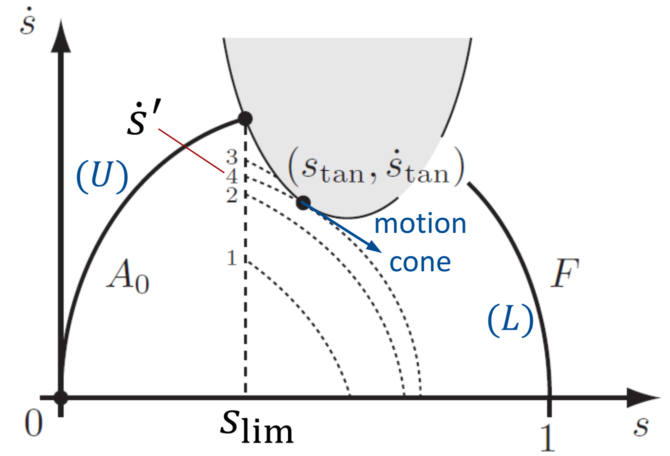
Step 2: Integrate $\ddot{s} = U(s,\dot{s})$ forward in time from $(0{,}0)$ until (i) it intersects F or (ii) until the velocity limit curve is penetrated $(L(s,\dot{s}) > U(s,\dot{s}))$. Call this curve A_0.

- If (i) happens, the problem is solved.
- If (ii) happens, let $(s_{\text{lim}}, \dot{s}_{\text{lim}})$ be the point of penetration.

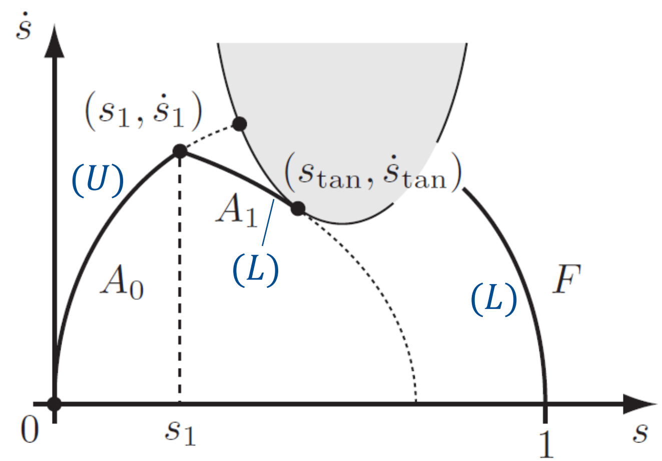


Time-Scaling Algorithm

Step 3: Perform a binary search (or half-interval search) on the velocity in the range $[0, \dot{s}_{\text{lim}}]$ at s_{lim} to find the velocity $\dot{s}'$ such that the curve integrating $\ddot{s} = L(s, \dot{s})$ forward in time from $(s_{\text{lim}}, \dot{s}')$ touches the velocity limit curve tangentially (or comes closest to the curve within a specified tolerance without hitting it) at $(s_{\text{tan}}, \dot{s}_{\text{tan}})$.

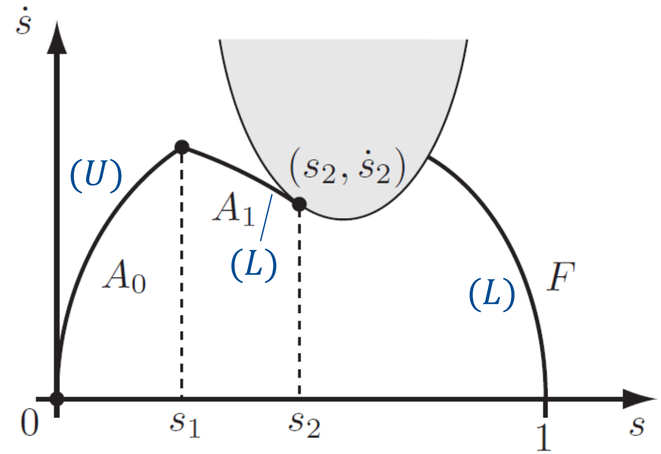


Step 4: Integrate $\ddot{s} = L(s, \dot{s})$ backward in time from $(s_{\text{tan}}, \dot{s}_{\text{tan}})$ until it intersects A_0 at $(s_1, \dot{s}_1)$. Call this curve A_1. $(s_1, \dot{s}_1)$ is the first switch point from maximum acceleration to maximum deceleration.



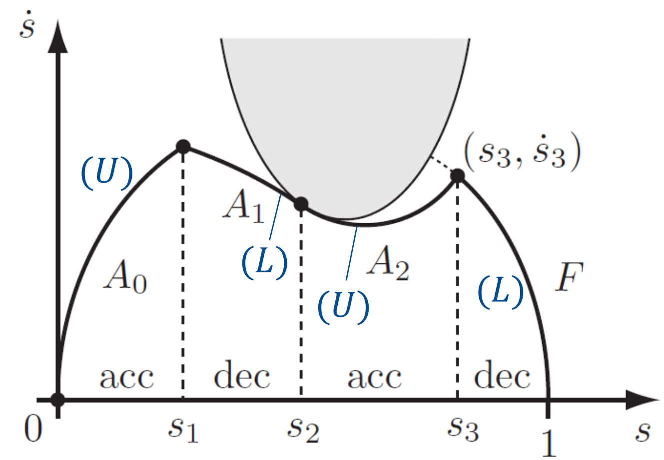
Time-Scaling Algorithm

Step 5: Mark the tangent point $(s_{\text{tan}}, \dot{s}_{\text{tan}})$ as the switch point $(s_2, \dot{s}_2)$ from maximum deceleration to maximum acceleration.



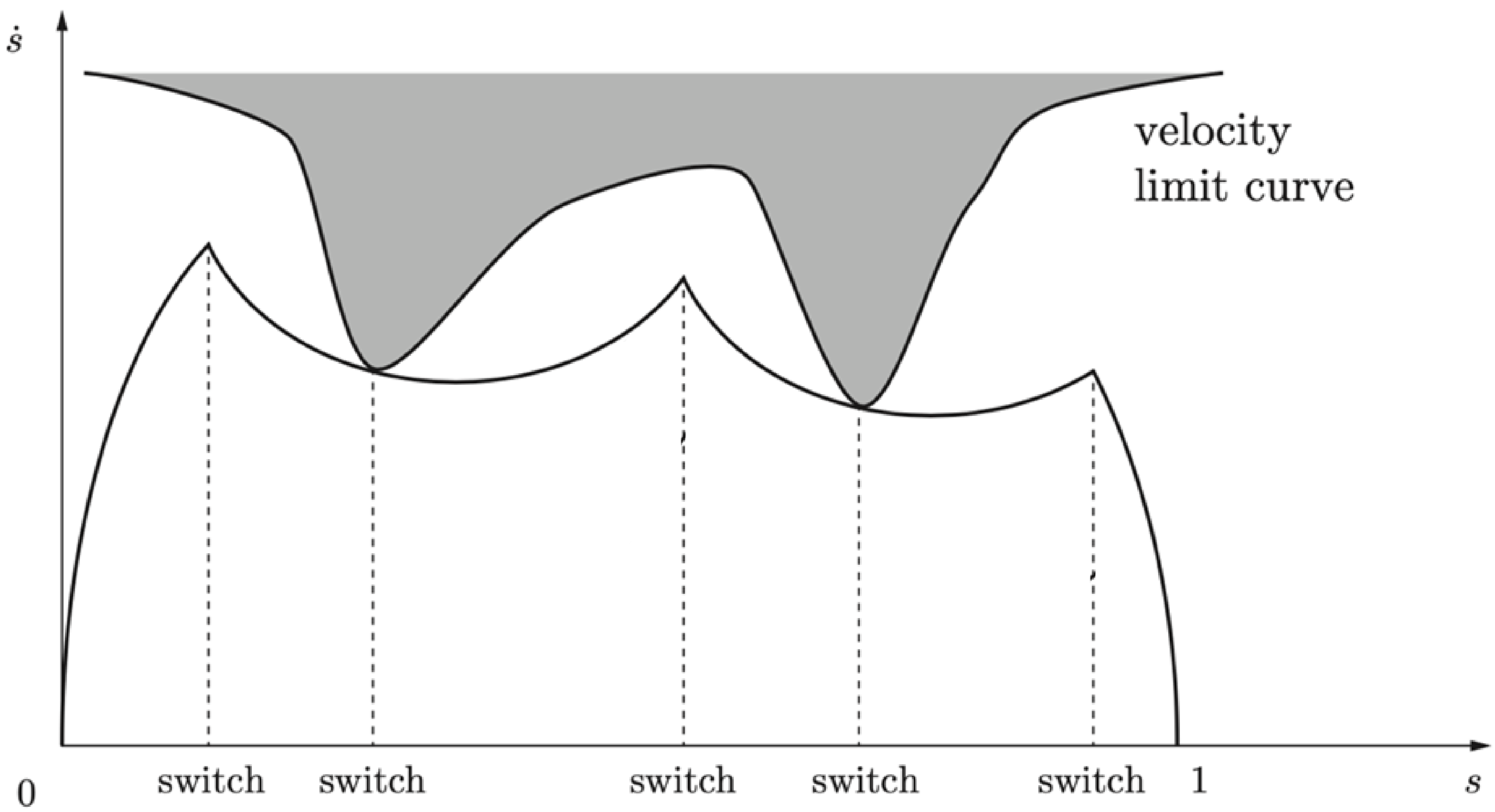
Step 6: Go back to **Step 2**, i.e., integrate $\ddot{s} = U(s, \dot{s})$ forward in time from $(s_2, \dot{s}_2)$ until (i) it intersects F or (ii) until the velocity limit curve is penetrated again ($L(s, \dot{s}) > U(s, \dot{s})$). Call this curve A_2.

- If (i) happens, the intersection point $(s_3, \dot{s}_3)$ is the final switch point from maximum acceleration to maximum deceleration and the algorithm is complete.
- If (ii) happens, let $(s_{\text{lim}}, \dot{s}_{\text{lim}})$ be the new point of penetration and repeat the process from **Step 3**.



Note: Now, by having $\dot{s}(s)$ and integrating $\dot{\boldsymbol{\theta}} = \frac{d\boldsymbol{\theta}}{ds}\dot{s}$, we can find $\boldsymbol{\theta}(t)$.

An Example of Multi-Switch Time-Optimal Time Scaling



Example

Draw the feasible time-optimal time-scaling for a driver rushing home with the max braking and max acceleration integral curves shown.