FOUNDATIONS OF ROBOTICS

Trajectory Planning

B. Siciliano, L. Villani, G. Oriolo, A. De Luca

These slides are distributed as companion material to the textbook
Foundations of Robotics by B. Siciliano, L. Villani, G. Oriolo, A. De Luca
published by Springer Nature.
Modification is not allowed without permission

Outline

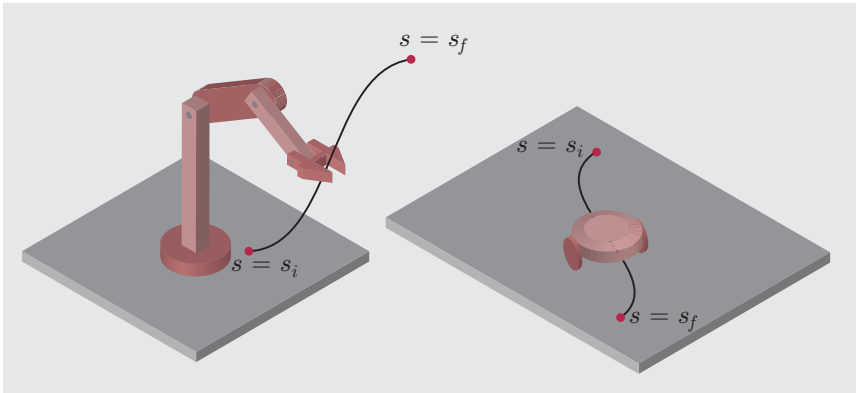
- The Trajectory Planning Problem
- Classification of Robot Trajectories
- Path Planning via Interpolation
- Choice of Timing Law
- Trajectories in Task Space

The Trajectory Planning Problem

- The trajectory planner is the functional unit of the robot that takes inputs from the task planner, and possibly also from the available sensors,
  and provides output reference trajectories for the controller which drives the robot through its actuators
- The task planner is a software component devoted to the high-level definition of the elementary actions that the robot should perform in order to complete a complex task
- It uses a priori information about the environment, data on the current state acquired from exteroceptive and proprioceptive sensors, and multi-modal user inputs

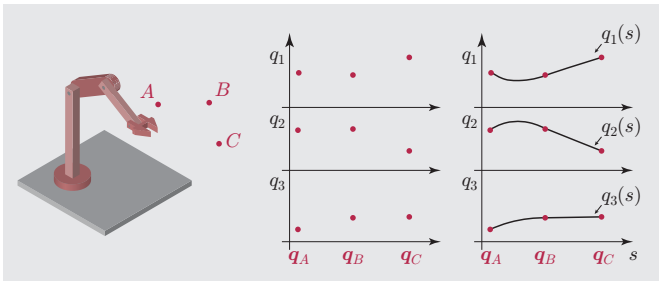
- In general, the trajectory planner can handle also the avoidance of few obstacles that are present in the robot workspace
- When the workspace is cluttered with obstacles, the search and definition of collision-free paths is referred to as the motion planning problem, which is treated in a different chapter
- The task planner produces a sequence of collision-free poses that have to be realized either by the end-effector of a manipulator or by the main body of a wheeled mobile robot
- The trajectory planner generates a continuous trajectory that interpolates the sequence of data for either $\boldsymbol{q} \in \mathcal{C}$ in configuration space or for $\boldsymbol{y} \in \mathcal{T}$ in task space

- Examples of parametrized paths between two points



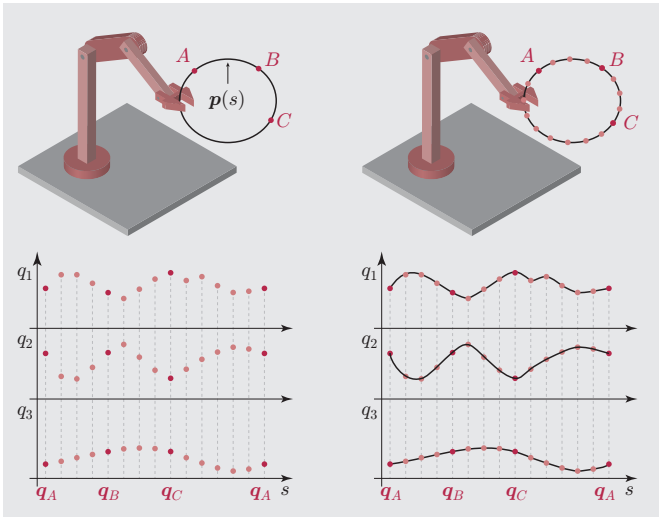
- A trajectory is specified by
  - a geometric path expressed in parametrized form,
  - a timing law that defines how the path is traced over time
- The geometric path satisfies boundary conditions and smoothness requirements
- The timing law is specified according to the task requirements and must be feasible with respect to the robot capabilities
- The scalar variable s (path parameter) used to describe the path can be chosen in the most convenient way
- From now on, only the case of robot manipulators is considered

- The decomposition of a robot trajectory in space and time allows keeping the path geometry unchanged when modifying the timing of motion execution
- Thus, in the trajectory planning problem one can talk about path planning separately from the definition of a timing law
- Example
  - the task planner provides three points A, B, C, with positions $\boldsymbol{p}_A$, $\boldsymbol{p}_B$, $\boldsymbol{p}_C$
  - by inverse kinematics, these are converted to three configurations $\boldsymbol{q}_A$, $\boldsymbol{q}_B$, $\boldsymbol{q}_C$
  - these are interpolated by a geometric path $\boldsymbol{q}(s)$ in joint space, for $s \in [s_i, s_f]$



- In general, one may prefer to have better control on the shape of the planned path
- In the previous example
  - the three points A, B, C (when not collinear) can be interpolated by a unique arc of a circle
  - if a cyclic task is desired, the complete circle is expressed as a parametrized circular path $\boldsymbol{p}(s)$ for the end-effector position
  - the circular path $\boldsymbol{p}(s)$ is sampled at a higher spatial rate and converted through inverse kinematics into a more dense sequence of configurations
  - these are interpolated at the joint level using a suitable geometric path $\boldsymbol{q}(s)$, conveniently using spline functions with continuity up to the second derivative with respect to the path parameter

- Path planning in task space with joint-space interpolation



- In this way, the joint path corresponds (through the direct kinematics) to a path which is close to the circle even outside the interpolated points
  - the closer, the denser the sampling of the original path in task space
- On the other hand, direct joint interpolation (without task data point interpolation and resampling) requires a smaller computational effort
  - outside the interpolated points A, B and C, the resulting path in task space wanders in a less predictable way

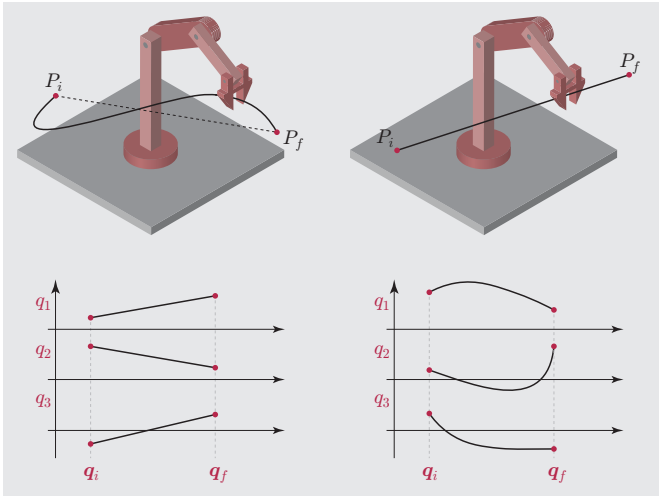
- Only task space position without orientation (i.e., $\boldsymbol{y} = \boldsymbol{p} \in \mathcal{T}$) has been considered
- Planning for the robot end-effector orientation often requires a separate treatment
- The robot may or may not be redundant with respect to the task
- In the presence of redundancy, generating inverse kinematic solutions in joint space requires the methods described for solving redundancy (e.g., task priorities)
- Constraints on path geometry, limits for the timing law, or performance criteria for both components of the trajectory can be considered
- Such problems are addressed in trajectory planning either in an integrated way, or separately in two stages

Classification of Robot Trajectories

- Robot trajectories can be classified in terms of
  - space in which planning is performed
  - type of motion problem addressed
  - class of interpolating functions used to define the geometric path or the timing law
- There are basic differences between planning trajectories in task or configuration space
- Planning in task space allows a more direct visualization of the generated path and requires no a priori choice among inverse kinematic solutions
- Motion control laws executed on a task-space trajectory require online inversion of the direct (or differential) kinematics, with the need of singularity and redundancy handling schemes
- Planning in configuration space may conveniently convert offline the task data
- Crossing of singularity causes no problems when controlling the robot along a joint-space trajectory because no Jacobian inversion is required at run time

- The trigonometric nature of direct and inverse kinematics modulates the corresponding trajectory in the other space
- A linear path planned in joint space between two configurations $\boldsymbol{q}_i$ and $\boldsymbol{q}_f$ corresponds to a nonlinear path profile in task space, moving out of the straight line between P_i and P_f
- A linear path planned in task space between P_i and P_f corresponds to a more complex path in joint space, with a profile that may not be confined between the extreme values of the components in $\boldsymbol{q}_i$ and $\boldsymbol{q}_f$

- Example of planning in the joint or task space



- A point-to-point (PTP) problem connects two points, e.g., two end-effector poses
  - the interpolating path is usually given by a single function defined for each component
  - boundary conditions and limitations on the derivatives can be imposed
  - multiple path segments may also be used
- A multi-point (MP) problem connects a sequence of via points that need to be connected
  - intermediate points, also called knots, are either exactly interpolated or approximated by the planned path
  - when the path does not pass exactly through a via point, there is an overfly of this point
  - planning requires a sufficiently rich class of functions and additional constraints on derivatives at knots and at path boundaries

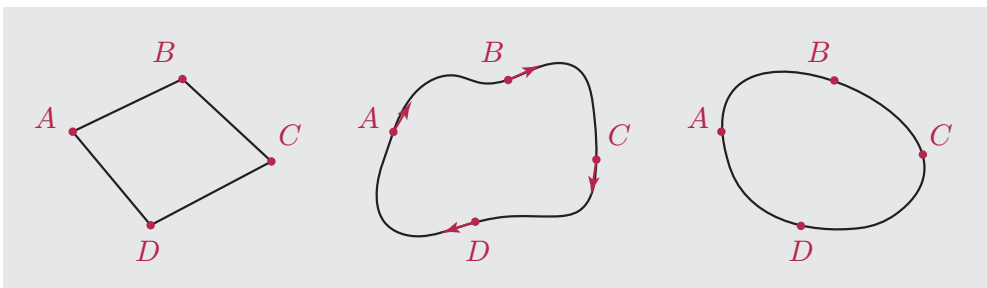
- The simplest choice for a PTP problem is a linear interpolating path, either as $\boldsymbol{q}(s)$ in configuration space or as $\boldsymbol{y}(s)$ in task space
- Other profiles include polynomial, harmonic, cycloidal, or exponential functions
- The available parameters in a given class of functions are used to satisfy boundary conditions on spatial derivatives
- In the MP case, interpolation of a large number of knots calls for concatenation of low-order polynomials (broadly called spline functions)
- Different classes of splines can be defined depending on polynomial degree, interpolating or approximating nature, continuity conditions at the knots, and boundary conditions at the initial and final path points

- The time evolution $s(t)$ of the path parameter can be specified using similar classes of functions as for the path components in a PTP problem
- A single polynomial of sufficiently high degree can satisfy $s(t_i) = s_i$ and $s(t_f) = s_f$ and additional boundary conditions on time derivatives
- Multiple polynomials of different degrees can comply with limitations imposed on time derivatives during motion
- The timing law may be used to handle transitions in MP problems between two linear path segments with different directions in order to guarantee continuity of the velocity
- A common timing law designed separately from path generation automatically results in a coordinated motion, with all scalar components starting and ending at the same time instants t_i and t_f

- When the robot task is fully known in advance, trajectories are planned offline and critical issues can be handled with due care
- When the robot task is defined from sensor information acquired in real time, reference trajectories can only be planned online, with an associated higher computational load
- In some cases, the geometric path is defined offline while the timing law has to be adapted online

- A sufficient degree of continuous differentiability in space and time is a common requirement in the generation of a trajectory
- Timing laws are designed to be at least functions of class C^1, but often continuous acceleration or even jerk are enforced
- Such degree of smoothness avoids excitation of resonances in the robot mechanical structure and helps the robot actuation and control systems in accurately reproducing the planned trajectory

- A MP path planned with interpolating linear segments introduces discontinuities of the path tangent at the knots
- Discontinuities of the path tangent require stopping motion at the knots to avoid physically unfeasible velocity discontinuities
- MP geometric paths are designed to have at least a continuous tangent (and sometimes continuous path curvature too)



- Numerical accuracy in passing through the given knots is a relevant choice criterion
- Predictability means that the planned path should not depart too much from the sequence of data points when away from interpolated knots (path wandering)
- In a PTP problem the planned motion should not go beyond the final end point value (overshooting)
- Flexibility expresses the ease by which primitive path elements can be concatenated while guaranteeing some smoothness in space
- Efficiency in computation of a trajectory is often traded off with use of memory space
  - one may store sampled values at control rate or store polynomial coefficients while evaluation is done at run time

Path Planning via Interpolation

- Consider planning in the n-dimensional configuration space $\mathcal{C}$
- A path is described in parametric form as

$$\boldsymbol{q}(s) = \begin{pmatrix} q_1(s) \\ q_2(s) \\ \vdots \\ q_n(s) \end{pmatrix} \qquad s \in [s_i, s_f]$$

- The function $\boldsymbol{q}(s)$ is (at least) continuously differentiable in s

- Consider the first two derivatives with respect to the parameter s

$$\boldsymbol{q}'(s) = \frac{d\boldsymbol{q}}{ds} \qquad \boldsymbol{q}''(s) = \frac{d^2\boldsymbol{q}}{ds^2}$$

- The first is the tangent vector to the path at s, while the second is related to the curvature of the path at s

$$\kappa(s) = \frac{\|\boldsymbol{q}'(s) \times \boldsymbol{q}''(s)\|}{\|\boldsymbol{q}'(s)\|^3}$$

- $\boldsymbol{q}''(s) = \mathbf{0}$ implies $\kappa(s) = 0$

- The regularity condition in the interior of the parametrization domain

$$\boldsymbol{q}'(s) \neq \mathbf{0} \qquad s \in (s_i, s_f)$$

  avoids singular points, called cusps, along the path and implies that the curvature is bounded everywhere
- Relaxing the regularity condition at the path extremities allows generating paths with cusps by patching regular subpaths

- The arc length σ along the path can be taken as path parameter

$$\sigma(s) = \int_{s_i}^{s} \left\| \boldsymbol{q}'(r) \right\| dr$$

- Along the path, σ goes from $\sigma_i = \sigma(s_i) = 0$ to $\sigma_f = \sigma(s_f) = L$
- With the arc length parametrization the tangent is a unit vector

$$\left\| \boldsymbol{q}'(\sigma) \right\| = \sqrt{\left(\frac{dq_1}{d\sigma} \right)^2 + \cdots + \left(\frac{dq_n}{d\sigma} \right)^2} = 1$$

and the curvature is simply

$$\kappa(\sigma) = \left\| \boldsymbol{q}''(\sigma) \right\|$$

- A smooth timing law $s(t)$ specifies the time evolution of the path parameter
- A trajectory $\boldsymbol{q}(s(t))$ has then a space-time decomposition with

$$\dot{\boldsymbol{q}} = \frac{d\boldsymbol{q}}{dt} = \frac{d\boldsymbol{q}}{ds}\frac{ds}{dt} = \boldsymbol{q}'\,\dot{s}$$

$$\ddot{\boldsymbol{q}} = \frac{d\dot{\boldsymbol{q}}}{dt} = \frac{d\boldsymbol{q}}{ds}\frac{d^2s}{dt^2} + \frac{d^2\boldsymbol{q}}{ds^2}\left(\frac{ds}{dt}\right)^2 = \boldsymbol{q}'\,\ddot{s} + \boldsymbol{q}''\,\dot{s}^2$$

- A trajectory may also be specified directly in time as $\boldsymbol{q}(t)$

Point-to-Point Path Planning

- Given an initial configuration $\boldsymbol{q}_i$ and a final configuration $\boldsymbol{q}_f$, let $\Delta\boldsymbol{q} = \boldsymbol{q}_f - \boldsymbol{q}_i$
- The simplest interpolating polynomial is a linear path with normalized parameter

$$\boldsymbol{q}(s) = \boldsymbol{q}_i + \Delta\boldsymbol{q}\, s \qquad s \in [0, 1]$$

- Using the arc length as parameter

$$\boldsymbol{q}(\sigma) = \boldsymbol{q}_i + \frac{\Delta\boldsymbol{q}}{\|\Delta\boldsymbol{q}\|}\,\sigma \qquad \sigma \in [0, L]$$

with $L = \|\Delta\boldsymbol{q}\|$

- Linear path derivatives

$$\boldsymbol{q}'(s) = \Delta\boldsymbol{q} \qquad \boldsymbol{q}'(\sigma) = \frac{\Delta\boldsymbol{q}}{\|\Delta\boldsymbol{q}\|}$$

$$\boldsymbol{q}''(s) = \boldsymbol{q}''(\sigma) = \mathbf{0}$$

- Both vectors are tangent to the linear path
- Only $\boldsymbol{q}'(\sigma)$ has unit norm

- If the initial and final path tangents are also assigned as $\boldsymbol{q}'_i$ and $\boldsymbol{q}'_f$, the interpolation problem is solved by a cubic path, defined with a normalized parameter as

$$\boldsymbol{q}(s) = \boldsymbol{a}_0 + \boldsymbol{a}_1 s + \boldsymbol{a}_2 s^2 + \boldsymbol{a}_3 s^3 \qquad s \in [0, 1]$$

- In the following, we consider the scalar case $q(s)$ without loss of generality
- The path curvature takes then the expression

$$\kappa(s) = \frac{|q''(s)|}{\sqrt{(1 + (q'(s))^2)^3}}$$

- Given $\Delta q = q_f - q_i$, a scalar cubic path is

$$q(s) = a_0 + a_1 s + a_2 s^2 + a_3 s^3 \qquad s \in [0, 1]$$

- Boundary conditions

$$q(0) = q_i \qquad q'(0) = q'_i \qquad q(1) = q_f \qquad q'(1) = q'_f$$

- Coefficients in the solution

$$a_0 = q_i \qquad a_1 = q'_i \qquad a_2 = 3\Delta q - (q'_f + 2q'_i) \qquad a_3 = -2\Delta q + (q'_f + q'_i)$$

Example

- PTP path planning problem in joint space for a 2R planar robot with unit length links
- End-effector task points $\boldsymbol{p}_i = (0.6, -0.4)$ [m] and $\boldsymbol{p}_f = (1, 1)$ [m]
- Initial and final tangents in task space $\boldsymbol{p}'_i = (-2, 0)$ and $\boldsymbol{p}'_f = (2, 2)$
- Using the inverse kinematics formulas and choosing the elbow-down solution

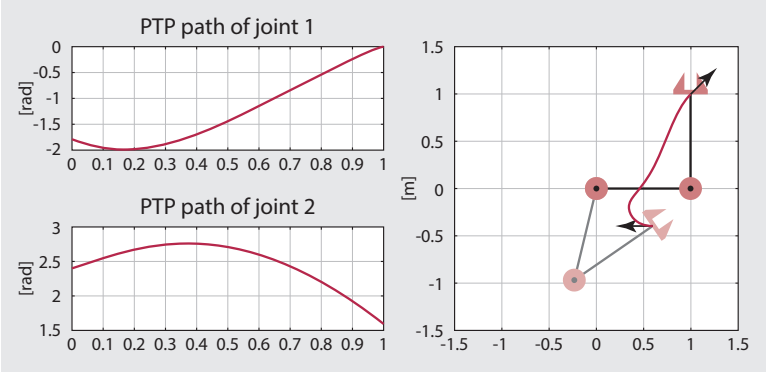
  $\boldsymbol{q}_i = (-1.7899, 2.4039) \quad \boldsymbol{q}_f = (0, 1.5708) \quad \Delta\boldsymbol{q} = \boldsymbol{q}_f - \boldsymbol{q}_i = (1.7899, -0.8331)$ [rad]
- The robot Jacobian is

$$\boldsymbol{J}(\boldsymbol{q}) = \begin{pmatrix} -(s_1 + s_{12}) & -s_{12} \\ c_1 + c_{12} & c_{12} \end{pmatrix}$$

$$\boldsymbol{J}(\boldsymbol{q}_i) = \begin{pmatrix} 0.4000 & -0.5761 \\ 0.6000 & 0.8174 \end{pmatrix} \qquad \boldsymbol{J}(\boldsymbol{q}_f) = \begin{pmatrix} -1 & -1 \\ 1 & 0 \end{pmatrix}$$

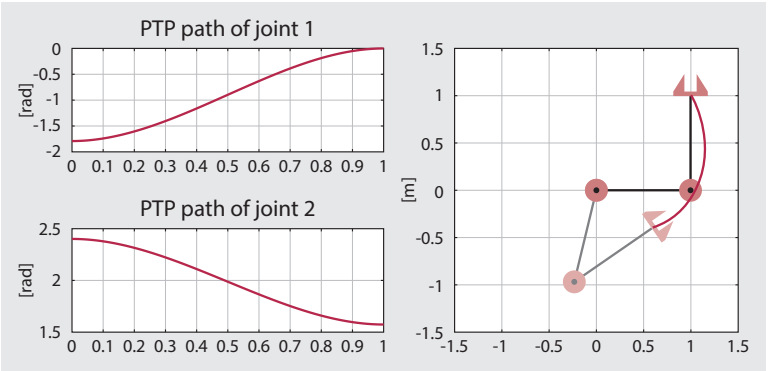
$$\boldsymbol{q}'_i = \boldsymbol{J}^{-1}(\boldsymbol{q}_i)\,\boldsymbol{p}'_i = (-2.4305, 1.7841) \qquad \boldsymbol{q}'_f = \boldsymbol{J}^{-1}(\boldsymbol{q}_f)\,\boldsymbol{p}'_f = (2, -4)$$

$$\boldsymbol{q}(s) = \begin{pmatrix} q_1(s) \\ q_2(s) \end{pmatrix} = \begin{pmatrix} -1.7899 - 2.4305\,s + 8.2308\,s^2 - 4.0104\,s^3 \\ 2.4039 + 1.7841\,s - 2.0674\,s^2 - 0.5498\,s^3 \end{pmatrix}$$



- Instead, when $q_i' = q_f' = 0$, the cubic interpolation path is

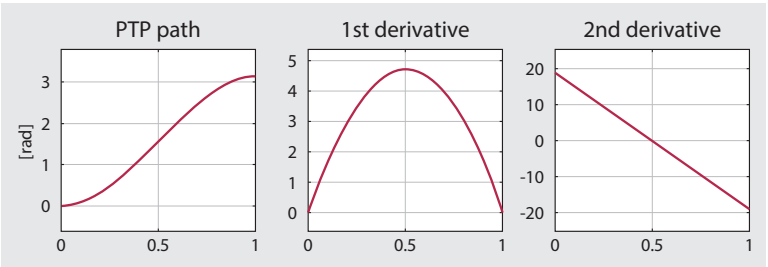
$$q(s) = q_i + \Delta q\,(3s^2 - 2s^3) \qquad s \in [0, 1]$$



- A similar result is obtained using a harmonic path

$$q(s) = q_i + \frac{\Delta q}{2}(1 - \cos \pi s) \qquad s \in [0, 1]$$

$$q'(s) = \frac{\pi \Delta q}{2} \sin \pi s \qquad q''(s) = \frac{\pi^2 \Delta q}{2} \cos \pi s$$



- If the initial and final second derivatives q_i'' and q_f'' are also assigned, the interpolation problem is solved by a quintic path with normalized parameter

$$q(s) = a_0 + a_1\, s + a_2\, s^2 + a_3\, s^3 + a_4\, s^4 + a_5\, s^5 \qquad s \in [0, 1]$$

with the boundary conditions

$$q(0) = q_i \quad q'(0) = q_i' \quad q''(0) = q_i'' \quad q(1) = q_f \quad q'(1) = q_f' \quad q''(1) = q_f''$$

- The low-order coefficients in the solution are easily obtained

$$a_0 = q_i \qquad a_1 = q_i' \qquad a_2 = \frac{1}{2}\, q_i''$$

- The higher-order coefficients are computed as

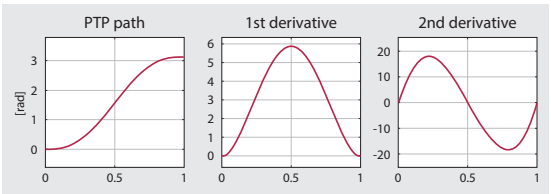
$$a_3 = 10\,\Delta q - \frac{1}{2}\,(8\,q'_f + 12\,q'_i + 3\,q''_i - q''_f)$$

$$a_4 = -15\,\Delta q + \frac{1}{2}\,(14\,q'_f + 16\,q'_i + 3\,q''_i - 2\,q''_f)$$

$$a_5 = 6\,\Delta q - \frac{1}{2}\,(6\,q'_f + 6\,q'_i + q''_i - q''_f)$$

- If $q'_i = q'_f = 0$ and $q''_i = q''_f = 0$

$$q(s) = q_i + \Delta q\,(10\,s^3 - 15\,s^4 + 6\,s^5) \qquad s \in [0, 1]$$



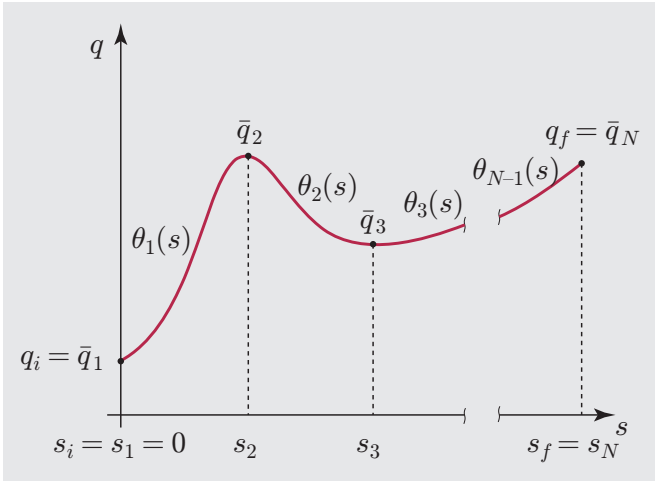
Multi-Point Path Planning

- Given $\{\bar{\boldsymbol{q}}_1, \ldots, \bar{\boldsymbol{q}}_N\}$ of $N \geq 3$ configurations (knots)
- Generate a path which connects $\boldsymbol{q}_i = \bar{\boldsymbol{q}}_1$ to $\boldsymbol{q}_f = \bar{\boldsymbol{q}}_N$ and passes through $\bar{\boldsymbol{q}}_2, \ldots, \bar{\boldsymbol{q}}_{N-1}$
- Using a single polynomial of degree $N-1$ is prone to wandering and motion reversal
- Use a sequence of $N-1$ low-order interpolating polynomials over subintervals

$$q(s) = \{\theta_k(s), \text{ for } s \in [s_k, s_{k+1}], \ k = 1, \ldots, N-1\}$$

$$\theta_{k-1}(s_k) = \theta_k(s_k) = \bar{q}_k \qquad k = 2, \ldots, N-1$$

- Sequence of interpolating polynomials



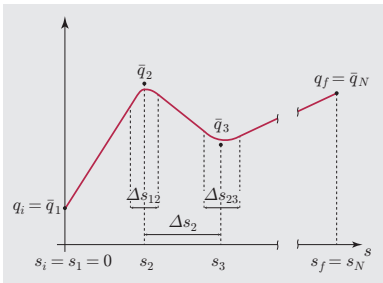
- Multi-point requirements
  - continuity of the tangent $q'(s)$ over the entire interval (s_1, s_N)
  - assigned tangents $q'(s_1)$ and $q'(s_N)$ at the first and last knot
  - continuity up to $q''(s)$ over the entire interval (s_1, s_N)
  - assigned $q''(s_1)$ and $q''(s_N)$ at the first and last knot

Linear Polynomials with Parabolic Blends

- Use segments to interpolate $\{\bar{q}_1, \ldots, \bar{q}_N\}$
- Let $\Delta s_k = s_{k+1} - s_k$

$$\theta_k(s) = \bar{q}_k + (\bar{q}_{k+1} - \bar{q}_k)\,\frac{s - s_k}{\Delta s_k} \qquad s \in [s_k, s_{k+1}]$$

- Parabolic blend to pass from one segment to the next



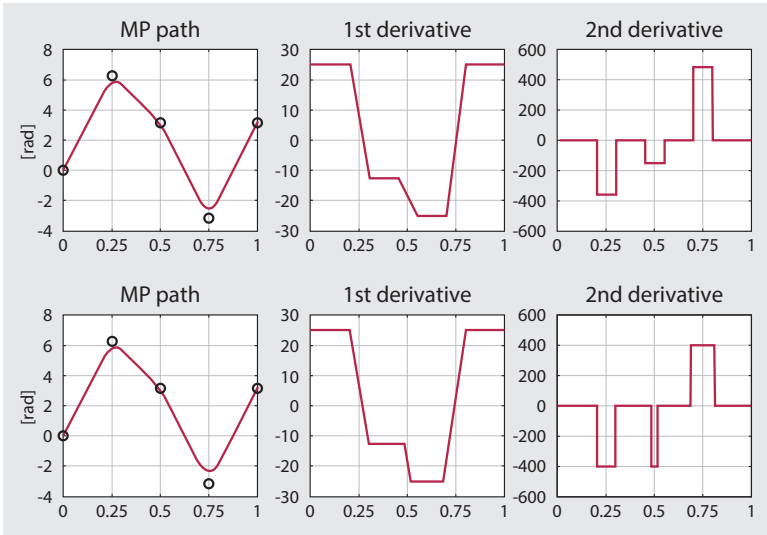
- Definition of tangents for the segments and second path derivative for the blendings

$$\theta'_{k-1} = \frac{\bar{q}_k - \bar{q}_{k-1}}{\Delta s_{k-1}} \qquad \theta'_k = \frac{\bar{q}_{k+1} - \bar{q}_k}{\Delta s_k}$$

$$\theta''_k = \frac{\theta'_k - \theta'_{k-1}}{\Delta s_{k-1,k}} \qquad s \in [s_k - \frac{\Delta s_{k-1,k}}{2}, s_k + \frac{\Delta s_{k-1,k}}{2}]$$

- The final path will not pass exactly through the via points $\bar{q}_k$
- The smaller the blending interval, the closer the path to the via point and the larger the path curvature around it

- Constant vs variable blending intervals



Cubic Splines

- A cubic spline is a piecewise cubic polynomial function
- Denote by $\theta_k(s)$ the cubic polynomial in $[s_k, s_{k+1}]$

$$\begin{aligned} \theta_k(s_k) = \bar{q}_k \qquad \theta_k(s_{k+1}) = \bar{q}_{k+1} \qquad & k = 1, \ldots, N-1 \\ \theta'_{k-1}(s_k) = \theta'_k(s_k) \qquad & k = 2, \ldots, N-1 \\ \theta''_{k-1}(s_k) = \theta''_k(s_k) \qquad & k = 2, \ldots, N-1 \end{aligned}$$

- Continuity of q' and q'' guarantees continuous curvature along the path
- Two additional conditions are usually specified as the initial and final path tangents

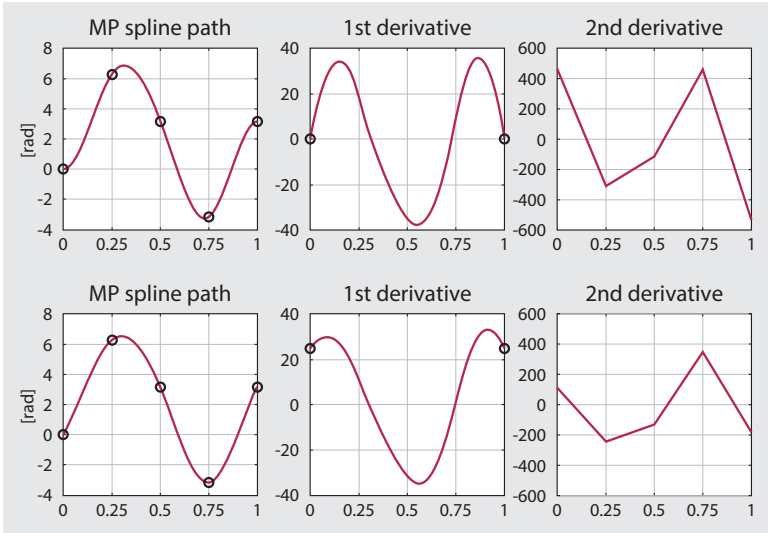
$$\theta'_1(s_1) = q'_i \qquad \theta'_{N-1}(s_N) = q'_f$$

- Reduced linear system

$$\boldsymbol{A}\,\boldsymbol{x} = \boldsymbol{b} \qquad \boldsymbol{x} = \begin{pmatrix} q'_2 & q'_3 & \dots & q'_{N-1} \end{pmatrix}^T$$

$$\boldsymbol{A} = \begin{pmatrix} 2(\Delta s_1+\Delta s_2) & \Delta s_1 & & & & \\ \Delta s_3 & 2(\Delta s_2+\Delta s_3) & \Delta s_2 & & & \\ & & \dots & & & \\ & & & \Delta s_{N-2} & 2(\Delta s_{N-3}+\Delta s_{N-2}) & \Delta s_{N-3} \\ & & & & \Delta s_{N-1} & 2(\Delta s_{N-2}+\Delta s_{N-1}) \end{pmatrix}$$

$$\boldsymbol{b} = \begin{pmatrix} \frac{3}{\Delta s_1\,\Delta s_2}\left(\Delta s_1^2\,\Delta q_2+\Delta s_2^2\,\Delta q_1\right)-\Delta s_2\,q'_1 \\ \frac{3}{\Delta s_2\,\Delta s_3}\left(\Delta s_2^2\,\Delta q_3+\Delta s_3^2\,\Delta q_2\right) \\ \vdots \\ \frac{3}{\Delta s_{N-3}\,\Delta s_{N-2}}\left(\Delta s_{N-3}^2\,\Delta q_{N-2}+\Delta s_{N-2}^2\,\Delta q_{N-3}\right) \\ \frac{3}{\Delta s_{N-2}\,\Delta s_{N-1}}\left(\Delta s_{N-2}^2\,\Delta q_{N-1}+\Delta s_{N-1}^2\,\Delta q_{N-2}\right)-\Delta s_{N-2}\,q'_N \end{pmatrix}$$


MP spline path
1st derivative
2nd derivative
[rad]
MP spline path
1st derivative
2nd derivative
[rad]

- The previous plots compare a cubic spline having

$$q_i' = q_f' = 0$$

  with a cubic spline having the implicit assignment of initial and final path tangents

$$q_i' = \frac{\bar{q}_2 - \bar{q}_1}{s_2 - s_1} \qquad q_f' = \frac{\bar{q}_N - \bar{q}_{N-1}}{s_N - s_{N-1}}$$

- Once $(\bar{q}_k, s_k)$ and q_i', q_f' are given, the spline is uniquely determined
- If $\bar{q}_1 = \bar{q}_N$ and $q_i' = q_f'$, the path is cyclic
- To enforce continuity at loop closure, replace endpoint tangent conditions with

$$\theta_1'(s_1) = \theta_{N-1}'(s_N) \qquad \theta_1''(s_1) = \theta_{N-1}''(s_N)$$

- To impose also $q_i'' = q_f'' = 0$, we need to introduce two virtual knots at $s_a \in (s_1, s_2)$ and $s_b \in (s_{N-1}, s_N)$
- A modified linear system $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ can be defined as before by adding new equations for the first and last (new) subintervals
  - in particular, for the first cubic polynomial

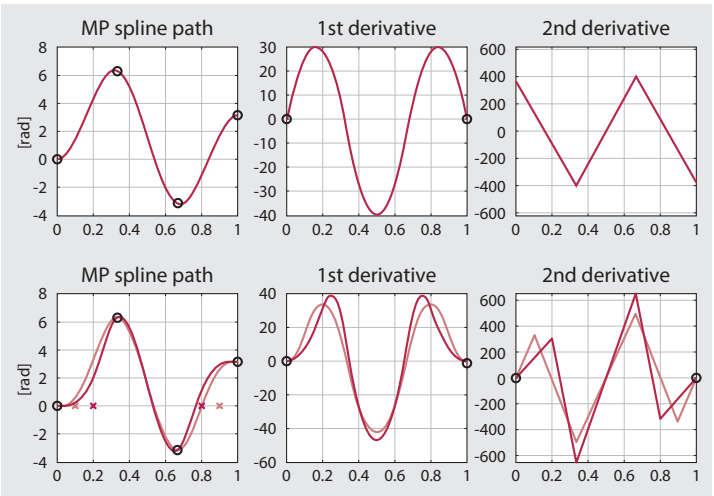
$$\theta_1(s) = \bar{q}_1 + q_1' \Delta s_1 \, s_n + a_{13} \, s_n^3 \qquad a_{13} = \frac{1}{3} \left(q_a' - q_1'\right) \Delta s_1$$

  - continuity conditions at s_a

$$\theta_1(s_a) = \bar{q}_1 + q_1' \Delta s_1 + a_{13} = \bar{q}_1 + \frac{1}{3}(q_a' + 2q_i') \, \Delta s_1 = \theta_a(s_a)$$

$$\theta_1''(s_a) = \frac{6a_{13}}{\Delta s_1^2} = \frac{2\left(q_a' - q_i'\right)}{\Delta s_1} = \theta_a''(s_a)$$

- Cubic splines without and with the boundary conditions $q_i'' = q_f'' = 0$ (and for different positions of the two added virtual knots)



Choice of Timing Law

- Once a path $\boldsymbol{q}(s)$, $s \in [s_i, s_f]$, has been planned, a timing law $s(t)$ completes the generation of a trajectory
- Timing laws can be defined by a single profile or by composite profiles
- Only monotonic profiles are acceptable timing laws, to avoid doubling back on already traversed portions of the path

Single Time Profile

- Motion time interval $t \in [t_i, t_f]$ with duration $T = t_f - t_i > 0$
- Rest-to-rest motion requires

$$\dot{s}(t_i) = 0 \qquad \dot{s}(t_f) = 0$$

- Normalized time variable

$$\tau = \frac{t - t_i}{T} \in [0, 1] \qquad \Delta s = s_f - s_i$$

- Cubic timing law

$$s(\tau) = s_i + \Delta s \left(3\tau^2 - 2\tau^3\right)$$

- Quintic timing law is needed to handle the additional requirements

$$\ddot{s}(t_i) = 0 \qquad \ddot{s}(t_f) = 0$$

Example

- Problem data
  - $q_i = 0$ at $t_i = 0$ and $q_f = \pi$ at $t_f = 4$
  - $q_i' = q_f' = 1$
  - $\dot{q}(t_i) = \dot{q}(t_f) = 0$
  - $\ddot{q}(t_i) = \ddot{q}(t_f) = 0$
- Cubic parametrized path

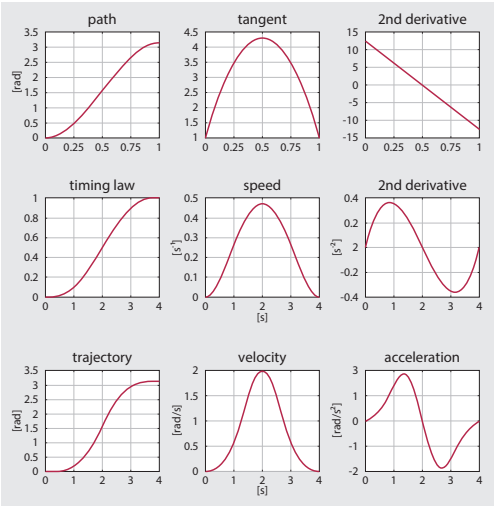
$$q(s) = s + 3(\pi - 1)s^2 - 2(\pi - 1)s^3 \qquad s \in [0, 1]$$

- Quintic timing law with $\tau = t/T = t/4$

$$s(t) = 10\,\tau^3 - 15\,\tau^4 + 6\,\tau^5 \qquad \tau \in [0, 1]$$

- Substituting $s(t)$ in $q(s)$ provides the trajectory $q(t)$ for $t \in [0, 4]$

- Cubic polynomial path with quintic polynomial timing law



- Rest-to-rest trajectories may also begin and end with zero acceleration, zero jerk, zero snap, and so on
- High-order doubly normalized polynomials can be used in this case as solutions, with $[0, 1]$ both as domain and codomain
- Since the number of boundary conditions is even, only polynomials of odd degree k are considered
- Example of order $k = 7$

$$s(t) = 35\tau^4 - 84\tau^5 + 70\tau^6 - 20\tau^7 \qquad \tau = t/T \in [0, 1]$$

- Example of order $k = 9$

$$s(t) = 126\tau^5 - 420\tau^6 + 540\tau^7 - 315\tau^8 + 70\tau^9 \qquad \tau = t/T \in [0, 1]$$

- Doubly normalized boundary conditions

$$\left.\frac{d^i s}{d\tau^i}\right|_{s=0} = \left.\frac{d^i s}{d\tau^i}\right|_{s=1} = 0 \qquad i = 1, \ldots, \frac{k-1}{2}$$

- Properties
  - all coefficients of terms up to degree $(k-1)/2$ are zero
  - nonzero coefficients are integers with alternating signs whose sum is 1
  - an explicit formula exists for computing nonzero coefficients for any odd $k \geq 3$
- Pro: no overshooting or wandering, even for very large k
- Con: motion tends to occur mostly around the center of the time interval, with velocity peaks increasing with k

Composite Time Profiles

- Composite time profiles satisfy boundary conditions and additional requests such as reducing transfer time under kinetic limits
- Two common methods are discussed
  - trapezoidal speed profile (with possible blending for multiple profiles)
  - sequence of 4-3-4 polynomials

Trapezoidal Speed Profile

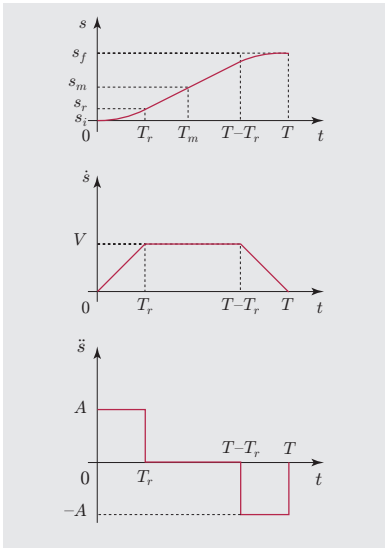
- Seek a rest-to-rest timing law under the bounds

$$|\dot{s}(t)| \leq V \qquad |\ddot{s}(t)| \leq A \qquad \text{for all } t \in [0, T]$$

- Min time solution: $\ddot{s} = A$ for rise time T_r, then $\dot{s} = V$ in cruise, then $\ddot{s} = -A$ for T_r
- Resulting $s(t)$ is a linear segment connected to s_i and s_f by two quadratic profiles

$$s(t) = \begin{cases} s_i + \dfrac{1}{2} A\, t^2 & t \in [0, T_r] \\ s_i + V(t - T_r) + \dfrac{1}{2} A\, T_r^2 & t \in [T_r, T - T_r] \\ s_f - \dfrac{1}{2} A\, (T - t)^2 & t \in [T - T_r, T] \end{cases}$$

- Bang-coast-bang (in acceleration) composite timing law



- From $T_r A = V$ and $(T - T_r)V = \Delta s = s_f - s_i$

$$T_r = \frac{V}{A} \qquad T = \frac{A\,\Delta s + V^2}{A\,V} = \frac{\Delta s}{V} + \frac{V}{A}$$

- Cruise phase exists if and only if $\Delta s > V^2/A$
- When $\Delta s \leq V^2/A$, the speed profile becomes triangular (bang-bang in acceleration) with

$$T = \sqrt{\frac{4\,\Delta s}{A}} \qquad \bar{V} = \frac{A\,T}{2} \leq V$$

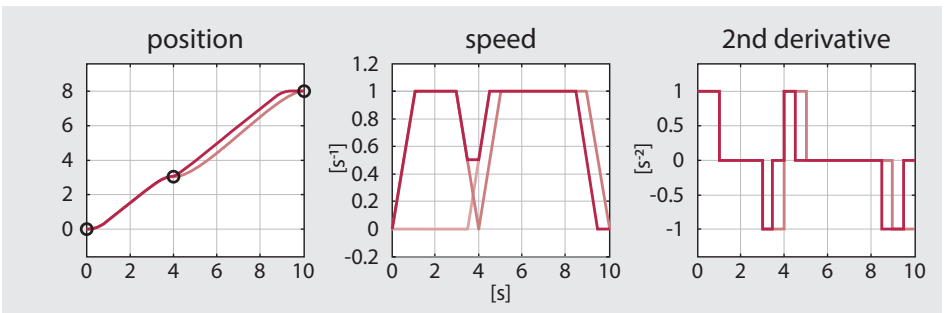
- If we start with the input data Δs, T, V assigned, then

$$A = \frac{V^2}{VT - \Delta s} \qquad T_r = \frac{V}{A} = \frac{VT - \Delta s}{V}$$

- Input triplet must satisfy $\Delta s/T < V$ to guarantee $A > 0$
- Cruise phase exists if and only if $V < 2\,\Delta s/T$

Blending of Trapezoidal Speed Profiles

- Concatenating rest-to-rest trapezoidal profiles enforces intermediate stops
- Blend by starting the second primitive before the end of the first
- Add the speed profiles and integrate to obtain a timing law with no intermediate stop



Sequence of 4-3-4 Polynomials

- Pick-and-place motion with four poses at t_i, t_a, t_b, t_f (three intermediate phases: lift-off, travel, set-down)
- Poses converted through inverse kinematics to $\boldsymbol{q}_i$, $\boldsymbol{q}_a$, $\boldsymbol{q}_b$, $\boldsymbol{q}_f$
- Passage through $\boldsymbol{q}_a$ and $\boldsymbol{q}_b$ is mandatory while stopping there is not required
- Rest-to-rest trajectory with continuous acceleration planned directly in time t
- 14 boundary and continuity conditions for the three phases (l,t,s)

$$q_l(t_i) = q_i \quad q_l(t_a) = q_t(t_a) = q_a \quad q_t(t_b) = q_s(t_b) = q_b \quad q_s(t_f) = q_f$$

$$\dot{q}_l(t_i) = \ddot{q}_l(t_i) = 0 \qquad \dot{q}_s(t_f) = \ddot{q}_s(t_f) = 0$$

$$\dot{q}_l(t_a) = \dot{q}_t(t_a) \quad \dot{q}_t(t_b) = \dot{q}_s(t_b) \quad \ddot{q}_l(t_a) = \ddot{q}_t(t_a) \quad \ddot{q}_t(t_b) = \ddot{q}_s(t_b)$$

- Choose three polynomials of degree 4, 3, and 4 (with reversed time)

$$
\begin{aligned}
q_l(\tau) &= a_{l0}+a_{l1}\tau+a_{l2}\tau^2+a_{l3}\tau^3+a_{l4}\tau^4 && \tau=(t-t_i)/T_l \\
q_t(\tau) &= a_{t0}+a_{t1}\tau+a_{t2}\tau^2+a_{t3}\tau^3 && \tau=(t-t_a)/T_t \\
q_s(\tau) &= a_{s0}+a_{s1}(\tau-1)+a_{s2}(\tau-1)^2+a_{s3}(\tau-1)^3+a_{s4}(\tau-1)^4 && \tau=(t-t_b)/T_s
\end{aligned}
$$

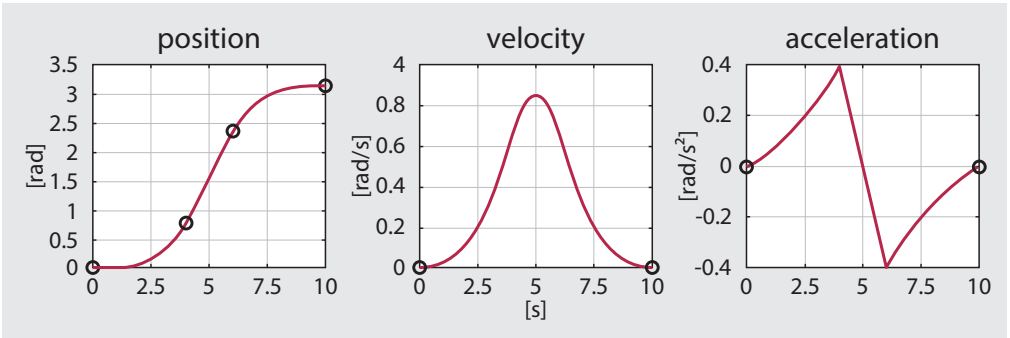
with

$$T_l = t_a - t_i \qquad T_t = t_b - t_a \qquad T_s = t_f - t_b$$

$$a_{l0} = q_i \qquad a_{l1} = a_{l2} = 0 \qquad a_{t0} = q_a \qquad a_{s0} = q_f \qquad a_{s1} = a_{s2} = 0$$

- The remaining conditions lead to a linear system whose unique solution provides the remaining 7 coefficients

- 4-3-4 trajectory



- With these data, lift-off and set-down phases take 80% of the total time, while 50% of the displacement occurs during the travel phase

Uniform Time Sscaling

- If the timing law is chosen without considering velocity and acceleration limits, time scaling on the same geometric path can recover feasibility or achieve better performance
- Uniform time scaling from $t \in [0, T]$ to $r = k\,t \in [0, T_s]$, with $k > 0$ and $T_s = k\,T$
- $k < 1$ speeds up the motion, $k > 1$ slows it down

$$\boldsymbol{q}_s(r) = \boldsymbol{q}_s(k\,t) = \boldsymbol{q}(t)$$

$$\dot{\boldsymbol{q}}_s(r) = \frac{d\boldsymbol{q}_s}{dr} = \frac{d\boldsymbol{q}}{dt}\,\frac{dt}{dr} = \frac{\dot{\boldsymbol{q}}(t)}{k}$$

$$\ddot{\boldsymbol{q}}_s(r) = \frac{d\dot{\boldsymbol{q}}_s}{dr} = \frac{d\dot{\boldsymbol{q}}_s}{dt}\,\frac{dt}{dr} = \frac{\ddot{\boldsymbol{q}}(t)}{k^2}$$

- Limits in joint space

$$|\dot{q}_j(t)| \leq v_{j,\max} \qquad |\ddot{q}_j(t)| \leq a_{j,\max} \qquad j = 1, \ldots, n$$

- For a given trajectory $\boldsymbol{q}(t)$, compute

$$v_{j,\text{peak}} = \max_{t\in[0,T]} |\dot{q}_j(t)| \qquad a_{j,\text{peak}} = \max_{t\in[0,T]} |\ddot{q}_j(t)| \qquad j = 1, \ldots, n$$

$$k = \max_{j=1,\ldots,n} \left\{ \frac{v_{j,\text{peak}}}{v_{j,\max}}, \sqrt{\frac{a_{j,\text{peak}}}{a_{j,\max}}} \right\}$$

- If one or more bounds are violated, $k > 1$ yields a slowdown
- If all bounds are strictly satisfied, $k < 1$ yields a speedup
- The scaled trajectory is feasible and saturates at least one velocity or acceleration component at one instant in the scaled time interval $[0, kT]$

Trajectories in Task Space

- Trajectory planning is performed in task space $\mathcal{T}$ by defining paths for task vector $\boldsymbol{y}$
  - a position path can be planned for $\boldsymbol{p}(s) = (x(s), y(s), z(s))$
  - an orientation path can be planned for $\boldsymbol{\phi}(s) = (\varphi(s), \vartheta(s), \psi(s))$
- Task-space planning often targets a position path with a specific shape and treats orientation as a whole
- A single timing law can be chosen for the path parameter s
- Timing laws can use quantities depending on the whole task vector

Position Paths

- Position $\boldsymbol{p} \in I\!R^3$ is planned as a parametrized path $\boldsymbol{p}(s)$, $s \in [s_i, s_f]$
  - tangent is $\boldsymbol{p}' = d\boldsymbol{p}/ds$
  - curvature is

$$\kappa(s) = \frac{\|\boldsymbol{p}'(s) \times \boldsymbol{p}''(s)\|}{\|\boldsymbol{p}'(s)\|^3}$$

- When using the arc length σ as parameter, the tangent is a unit vector and the curvature is $\|\boldsymbol{p}''(\sigma)\|$
- Regularity condition

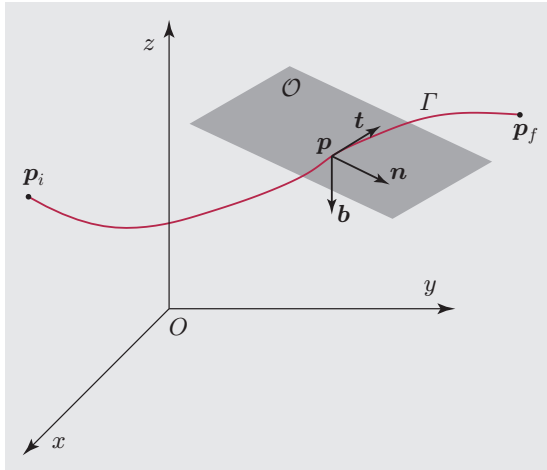
$$\boldsymbol{p}'(s) \neq \mathbf{0} \qquad s \in [s_i, s_f]$$

- Frenet frame unit vectors associated to a path $\boldsymbol{p}(s)$

$$\boldsymbol{t}(s) = \frac{\boldsymbol{p}'(s)}{\|\boldsymbol{p}'(s)\|}$$
$$\boldsymbol{n}(s) = \frac{\boldsymbol{t}'(s)}{\|\boldsymbol{t}'(s)\|} = \frac{\boldsymbol{p}'(s) \times (\boldsymbol{p}''(s) \times \boldsymbol{p}'(s))}{\|\boldsymbol{p}'(s) \times (\boldsymbol{p}''(s) \times \boldsymbol{p}'(s))\|}$$
$$\boldsymbol{b}(s) = \boldsymbol{t}(s) \times \boldsymbol{n}(s) = \frac{\boldsymbol{p}'(s) \times \boldsymbol{p}''(s)}{\|\boldsymbol{p}'(s) \times \boldsymbol{p}''(s)\|}$$

- Normal vector $\boldsymbol{n}$ is undefined when $\boldsymbol{t}'(s) = \boldsymbol{p}''(s) = \boldsymbol{0}$
- Frenet frame provides a natural evolution of orientation along a position path

- Path with Frenet frame
  - tangent unit vector $\boldsymbol{t}(s)$
  - mormal unit vector $\boldsymbol{n}(s)$
  - binormal unit vector $\boldsymbol{b}(s)$
  - osculating plane $\mathcal{O}$



Linear Paths

- Initial and final positions $\boldsymbol{p}_i$ and $\boldsymbol{p}_f$, with $\Delta\boldsymbol{p} = \boldsymbol{p}_f - \boldsymbol{p}_i \in I\!R^3$, $L = \|\Delta\boldsymbol{p}\|$
- Linear path with normalized parameter

$$\boldsymbol{p}(s) = \boldsymbol{p}_i + \Delta\boldsymbol{p}\, s \qquad s \in [0, 1]$$

- Linear path with arc length parameter

$$\boldsymbol{p}(\sigma) = \boldsymbol{p}_i + \sigma\, \frac{\Delta\boldsymbol{p}}{\|\Delta\boldsymbol{p}\|} = \boldsymbol{p}_i + \sigma\, \boldsymbol{k} \qquad \sigma \in [0, L]$$

- Unit vector of directional cosine $\boldsymbol{k} = (k_x, k_y, k_z)$

$$\dot{\boldsymbol{p}} = \dot{\sigma}\, \boldsymbol{k} \qquad \ddot{\boldsymbol{p}} = \ddot{\sigma}\, \boldsymbol{k}$$

- Zero curvature and no unique Frenet frame along linear paths

- A series of kinetic limits that can be considered
  - bounds on norms
    $$\|\dot{\boldsymbol{p}}\| \leq v_{\max} \qquad \|\ddot{\boldsymbol{p}}\| \leq a_{\max}$$
  - corresponding bounds on σ
    $$|\dot{\sigma}| \leq v_{\max} = V \qquad |\ddot{\sigma}| \leq a_{\max} = A$$
  - componentwise velocity bounds $v_{x,\max}$, $v_{y,\max}$, $v_{z,\max}$
    $$|\dot{\sigma}| \leq \min\left\{\frac{v_{x,\max}}{|k_x|}, \frac{v_{y,\max}}{|k_y|}, \frac{v_{z,\max}}{|k_z|}\right\} = V$$
  - componentwise acceleration bounds $a_{x,\max}$, $a_{y,\max}$, $a_{z,\max}$
    $$|\ddot{\sigma}| \leq \min\left\{\frac{a_{x,\max}}{|k_x|}, \frac{a_{y,\max}}{|k_y|}, \frac{a_{z,\max}}{|k_z|}\right\} = A$$
  - with normalized $s \in [0, 1]$, bounds are divided by L
- V and A can be used for a trapezoidal speed profile

Blending Linear Paths

- Sequence of points $\{\boldsymbol{p}_i = \bar{\boldsymbol{p}}_1, \ldots, \bar{\boldsymbol{p}}_i, \ldots, \bar{\boldsymbol{p}}_N = \boldsymbol{p}_f\}$
- Constant speeds $\dot{\sigma}_i = V_i$ on each linear segment
- Example with three points $\{\boldsymbol{p}_i, \boldsymbol{p}_m, \boldsymbol{p}_f\}$ connected by two segments

$$\boldsymbol{p}_1(\sigma_1) = \boldsymbol{p}_i + \sigma_1 \frac{\boldsymbol{p}_m - \boldsymbol{p}_i}{\|\boldsymbol{p}_m - \boldsymbol{p}_i\|} = \boldsymbol{p}_i + \sigma_1 \boldsymbol{k}_1 \qquad \sigma_1 \in [0, L_1]$$

$$\boldsymbol{p}_2(\sigma_2) = \boldsymbol{p}_m + \sigma_2 \frac{\boldsymbol{p}_f - \boldsymbol{p}_m}{\|\boldsymbol{p}_f - \boldsymbol{p}_m\|} = \boldsymbol{p}_m + \sigma_2 \boldsymbol{k}_2 \qquad \sigma_2 \in [0, L_2]$$

- Direct connection yields a discontinuity in Cartesian velocity at

$$T_m = \|\boldsymbol{p}_m - \boldsymbol{p}_i\| / V_1$$

- Blending interval of duration ΔT centered at T_m
- Constant acceleration in the blending interval

$$\ddot{\boldsymbol{p}}(t) = \frac{\boldsymbol{k}_2 V_2 - \boldsymbol{k}_1 V_1}{\Delta T} \qquad t \in [T_m - \Delta T/2, T_m + \Delta T/2] = [t_1, t_2]$$

- Interpolating trajectory (with leaving and landing distances d_1 and d_2 from $\boldsymbol{p}_m$

$$\boldsymbol{p}(t) = \boldsymbol{p}(t_1) + \boldsymbol{k}_1 V_1 (t - t_1) + (\boldsymbol{k}_2 V_2 - \boldsymbol{k}_1 V_1) \frac{(t - t_1)^2}{2\,\Delta T} \qquad t \in [t_1, t_2]$$

$$\boldsymbol{p}(t_1) = \boldsymbol{p}(T_m - \Delta T/2) = \boldsymbol{p}_m - V_1 \frac{\Delta T}{2} \boldsymbol{k}_1 = \boldsymbol{p}_m - d_1 \boldsymbol{k}_1 \qquad d_1 = V_1 \Delta T/2$$

$$\boldsymbol{p}(t_2) = \boldsymbol{p}(T_m + \Delta T/2) = \boldsymbol{p}_m + V_2 \frac{\Delta T}{2} \boldsymbol{k}_2 = \boldsymbol{p}_m + d_2\, \boldsymbol{k}_2 \qquad d_2 = V_2 \Delta T/2$$

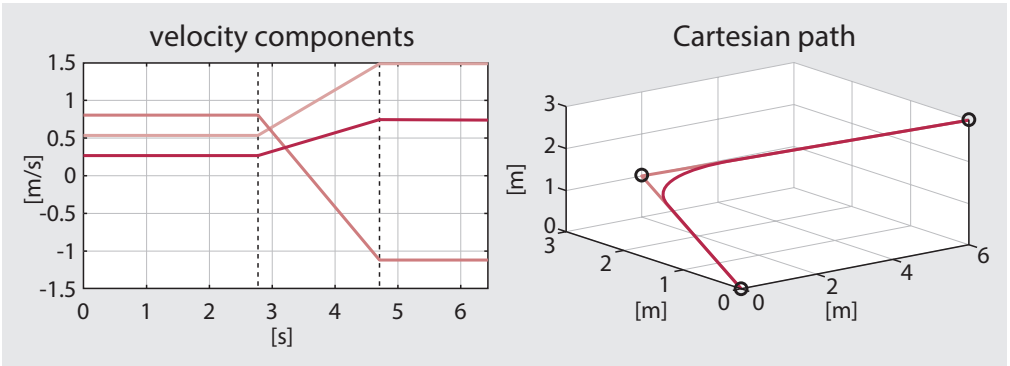
- Arc of parabola joining $\boldsymbol{p}(t_1)$ to $\boldsymbol{p}(t_2)$ with overfly of $\boldsymbol{p}_m$ under acceleration limits

$$|\ddot{p}_i| = \frac{|k_{2,i}V_2 - k_{1,i}V_1|}{\Delta T} \leq a_{i,\max} \qquad i = x, y, z$$

$$\Delta T_{\min} = \max\left\{\frac{|k_{2,i}V_2 - k_{1,i}V_1|}{a_{i,\max}},\ i = x, y, z\right\}$$

- Example
  - $\boldsymbol{p}_i = (0, 0, 0)$, $\boldsymbol{p}_m = (2, 3, 1)$, $\boldsymbol{p}_f = (6, 0, 3)$
  - $V_1 = 1$, $V_2 = 2$
  - choose $\Delta T = 1.93$
  - obtain $d_1 = 0.97$, $d_2 = 1.93$, and $T = 6.43$

- Blending two linear paths in Cartesian space



Circular Paths

- Circle on (x, y) plane with center $C_0 = (x_0, y_0, 0)$ and radius r, in normalized form

$$\boldsymbol{p}(s) = C_0 + r \begin{pmatrix} \cos(2\pi s + \phi_i) \\ \sin(2\pi s + \phi_i) \\ 0 \end{pmatrix} \qquad s \in [0, 1]$$

- For an initial (and final) point $P_i = (x_i, y_i, 0)$ at $s = 0$ (and $s = 1$)

$$\phi_i = \operatorname{atan2}(y_i - y_0, x_i - x_0)$$

- Arc length parameterization

$$\boldsymbol{p}(\sigma) = C_0 + r \begin{pmatrix} \cos(\sigma/r + \phi_i) \\ \sin(\sigma/r + \phi_i) \\ 0 \end{pmatrix} \qquad \sigma \in [0, 2\pi r]$$

- Generic circle in $\mathbb{R}^3$ with associated Frenet frame
- Start with the formulation in the (x, y) plane

$$\boldsymbol{p}'(s) = 2\pi r \begin{pmatrix} -\sin(2\pi s + \phi_i) \\ \cos(2\pi s + \phi_i) \\ 0 \end{pmatrix} \qquad \boldsymbol{p}''(s) = -4\pi^2 r \begin{pmatrix} \cos(2\pi s + \phi_i) \\ \sin(2\pi s + \phi_i) \\ 0 \end{pmatrix}$$

$$\boldsymbol{t}(s) = \begin{pmatrix} -\sin(2\pi s + \phi_i) \\ \cos(2\pi s + \phi_i) \\ 0 \end{pmatrix} \qquad \boldsymbol{n}(s) = -\begin{pmatrix} \cos(2\pi s + \phi_i) \\ \sin(2\pi s + \phi_i) \\ 0 \end{pmatrix} \qquad \boldsymbol{b} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

- Curvature is $\kappa = 1/r$
- Move the path to 3D space, with a displacement $\boldsymbol{r}_t \in \mathbb{R}^3$ and a rotation $\boldsymbol{R}_t \in SO(3)$
- Parametric equations

$$\tilde{\boldsymbol{p}}(s) = \boldsymbol{r}_t + \boldsymbol{R}_t\, \boldsymbol{p}(s) \qquad s \in [0, 1]$$

Example

- Three nonaligned points A, B, C with positions $\boldsymbol{p}_A$, $\boldsymbol{p}_B$, $\boldsymbol{p}_C$
- Unit vectors

$$\boldsymbol{n}_1 = \frac{\boldsymbol{p}_B - \boldsymbol{p}_A}{\|\boldsymbol{p}_B - \boldsymbol{p}_A\|} \qquad \boldsymbol{n}_2 = \frac{\boldsymbol{p}_C - \boldsymbol{p}_A}{\|\boldsymbol{p}_C - \boldsymbol{p}_A\|}$$

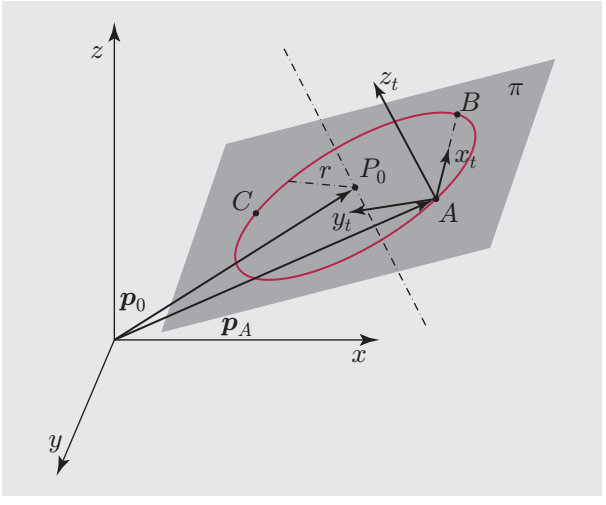
- Target frame with $\boldsymbol{r}_t = \boldsymbol{p}_A$ and $\boldsymbol{R}_t = (\boldsymbol{x}_t\, \boldsymbol{y}_t\, \boldsymbol{z}_t)$

$$\boldsymbol{x}_t = \boldsymbol{n}_1 \qquad \boldsymbol{z}_t = \pm \frac{\boldsymbol{n}_1 \times \boldsymbol{n}_2}{\|\boldsymbol{n}_1 \times \boldsymbol{n}_2\|} \qquad \boldsymbol{y}_t = \boldsymbol{z}_t \times \boldsymbol{x}_t$$

- Difference vectors in target frame

$${}^t\boldsymbol{p}_{AB} = \boldsymbol{R}_t^T (\boldsymbol{p}_B - \boldsymbol{p}_A) \qquad {}^t\boldsymbol{p}_{AC} = \boldsymbol{R}_t^T (\boldsymbol{p}_C - \boldsymbol{p}_A)$$

- Circular path through three non-aligned points



- Circle equation form

$$x^2 + y^2 + a\,x + b\,y + c = 0$$

$$a = -2x_0 \qquad b = -2y_0 \qquad c = (x_0^2 + y_0^2) - r^2$$

- Linear system for (a, b, c)

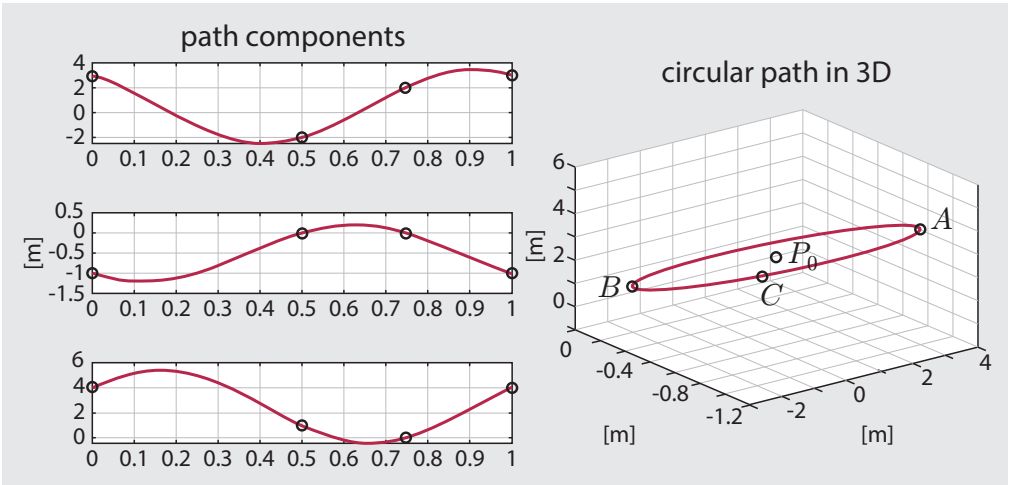
$$\begin{pmatrix} 0 & 0 & 1 \\ {}^tp_{AB,x} & {}^tp_{AB,y} & 1 \\ {}^tp_{AC,x} & {}^tp_{AC,y} & 1 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} 0 \\ -({}^tp_{AB,x}^2 + {}^tp_{AB,y}^2) \\ -({}^tp_{AC,x}^2 + {}^tp_{AC,y}^2) \end{pmatrix}$$

$$x_0 = -a/2 \qquad y_0 = -b/2 \qquad r = \sqrt{(x_0^2 + y_0^2) - c}$$

- Center P_0 in Cartesian space

$$\boldsymbol{p}_0 = \boldsymbol{p}_A + \boldsymbol{R}_t\,(x_0 \;\; y_0 \;\; 0)^T$$

- Circular path



- To define a timing law on the circular path

$$\dot{\boldsymbol{p}} = \boldsymbol{p}'\dot{s} = 2\pi r\dot{s} \begin{pmatrix} -\sin(2\pi s + \phi_i) \\ \cos(2\pi s + \phi_i) \\ 0 \end{pmatrix}$$

$$\ddot{\boldsymbol{p}} = \boldsymbol{p}'\ddot{s} + \boldsymbol{p}''\dot{s}^2 = 2\pi r\ddot{s} \begin{pmatrix} -\sin(2\pi s + \phi_i) \\ \cos(2\pi s + \phi_i) \\ 0 \end{pmatrix} - 4\pi^2 r\dot{s}^2 \begin{pmatrix} \cos(2\pi s + \phi_i) \\ \sin(2\pi s + \phi_i) \\ 0 \end{pmatrix}$$

- Bounds are derived from velocity and tangential/normal acceleration limits

$$|\dot{s}| \leq \min\left\{ \frac{v_{\max}}{2\pi r}, \frac{1}{2\pi}\sqrt{\frac{a_{n,\max}}{r}} \right\} \qquad |\ddot{s}| \leq \frac{a_{t,\max}}{2\pi r}$$

Helical Paths

- Helix with axis through $C_0 = (x_0, y_0, 0)$ parallel to z axis, radius r and pitch h

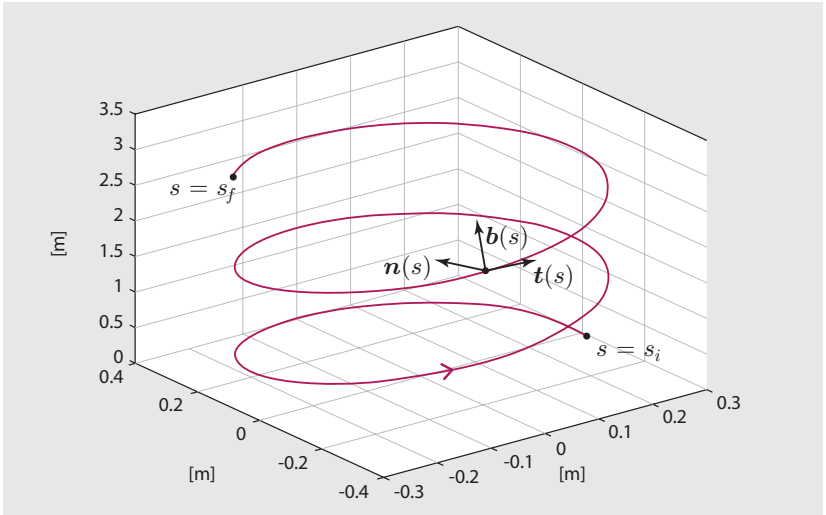
$$\boldsymbol{p}(s) = C_0 + \begin{pmatrix} r\cos(2\pi s + \phi_i) \\ r\sin(2\pi s + \phi_i) \\ hs \end{pmatrix} \qquad s \in [0,1]$$

$$\boldsymbol{p}'(s) = \begin{pmatrix} -2\pi r\sin(2\pi s + \phi_i) \\ 2\pi r\cos(2\pi s + \phi_i) \\ h \end{pmatrix} \qquad \boldsymbol{p}''(s) = -4\pi^2 r \begin{pmatrix} \cos(2\pi s + \phi_i) \\ \sin(2\pi s + \phi_i) \\ 0 \end{pmatrix}$$

$$\boldsymbol{t}(s) = \frac{1}{\sqrt{4\pi^2 r^2 + h^2}} \begin{pmatrix} -2\pi r\sin(2\pi s + \phi_i) \\ 2\pi r\cos(2\pi s + \phi_i) \\ h \end{pmatrix} \quad \boldsymbol{n}(s) = -\begin{pmatrix} \cos(2\pi s + \phi_i) \\ \sin(2\pi s + \phi_i) \\ 0 \end{pmatrix}$$

$$\boldsymbol{b}(s) = \frac{1}{\sqrt{4\pi^2 r^2 + h^2}} \begin{pmatrix} h\sin(2\pi s + \phi_i) \\ -h\cos(2\pi s + \phi_i) \\ 2\pi r \end{pmatrix} \qquad \kappa = \frac{4\pi^2 r}{4\pi^2 r^2 + h^2}$$

- Helical path



Orientation Paths

- Orientation is described by the rotation matrix $\boldsymbol{R}$
- A path $\boldsymbol{R}(s)$ must preserve orthonormality along the motion
  - direct interpolation from $\boldsymbol{R}_i$ to $\boldsymbol{R}_f$ cannot be used in general
- Three methods are presented next that use different representations for planning orientation paths and timing laws
  - absolute Euler angles
  - relative Euler angles
  - axis-angle

Absolute Euler Angles

- Use a generic sequence of Euler angles $\boldsymbol{\phi} = (\varphi, \vartheta, \psi)$
- Compute $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_f$ such that

$$\boldsymbol{R}(\boldsymbol{\phi}_i) = \boldsymbol{R}_i \qquad \boldsymbol{R}(\boldsymbol{\phi}_f) = \boldsymbol{R}_f$$

- Linear interpolation

$$\boldsymbol{\phi}(s) = \boldsymbol{\phi}_i + s\left(\boldsymbol{\phi}_f - \boldsymbol{\phi}_i\right) \qquad s \in [0, 1]$$

- Orientation path as $\boldsymbol{R}(\boldsymbol{\phi}(s))$ and timing law $s(t)$

- Related quantities to be computed

$$\boldsymbol{\omega} = \boldsymbol{T}(\boldsymbol{\phi})\,\boldsymbol{\phi}'\,\dot{s}$$

$$\dot{\boldsymbol{R}}(\boldsymbol{\phi}) = \left(\frac{\partial \boldsymbol{R}}{\partial \varphi}\,\varphi' + \frac{\partial \boldsymbol{R}}{\partial \vartheta}\,\vartheta' + \frac{\partial \boldsymbol{R}}{\partial \psi}\,\psi'\right)\dot{s} = \boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{R}(\boldsymbol{\phi})$$

$$\boldsymbol{S}(\boldsymbol{\omega}) = \dot{\boldsymbol{R}}(\boldsymbol{\phi})\boldsymbol{R}^T(\boldsymbol{\phi})$$

$$\ddot{\boldsymbol{R}}(\boldsymbol{\phi}) = \boldsymbol{S}(\dot{\boldsymbol{\omega}})\boldsymbol{R}(\boldsymbol{\phi}) + \boldsymbol{S}^2(\boldsymbol{\omega})\boldsymbol{R}(\boldsymbol{\phi})$$

$$\boldsymbol{S}(\dot{\boldsymbol{\omega}}) = (\ddot{\boldsymbol{R}}(\boldsymbol{\phi}) - \boldsymbol{S}^2(\boldsymbol{\omega})\boldsymbol{R}(\boldsymbol{\phi}))\,\boldsymbol{R}^T(\boldsymbol{\phi}) = \ddot{\boldsymbol{R}}(\boldsymbol{\phi})\boldsymbol{R}^T(\boldsymbol{\phi}) + \|\boldsymbol{\omega}\|^2\,\boldsymbol{I} - \boldsymbol{\omega}\,\boldsymbol{\omega}^T$$

- Bound on $\dot{s}$ for the timing law

$$|\dot{s}| \leq \frac{\omega_{\max}}{\max_{s\in[0,1]} \|\boldsymbol{T}(\boldsymbol{\phi}(s))\,\boldsymbol{\phi}'(s)\|}$$

Relative Euler Angles

- Relative rotation matrix between initial and final orientations matched with a generic sequence of Euler angles $\boldsymbol{\phi}_{if} = (\varphi_{if}, \vartheta_{if}, \psi_{if})$

$$ {}^{i}\boldsymbol{R}_f = \boldsymbol{R}_i^T \boldsymbol{R}_f \qquad \Rightarrow \qquad \boldsymbol{R}(\boldsymbol{\phi}_{if}) = {}^{i}\boldsymbol{R}_f $$

- Linear interpolation

$$ {}^{i}\boldsymbol{\phi}(s) = s\,\boldsymbol{\phi}_{if} \qquad s \in [0,1] $$

$$ \boldsymbol{R}(s) = \boldsymbol{R}_i\,\boldsymbol{R}({}^{i}\boldsymbol{\phi}(s)) \qquad (\boldsymbol{R}(1) = \boldsymbol{R}_f) $$

- Transform angular velocity and acceleration from frame $\boldsymbol{R}_i$ to the base frame

$$ \boldsymbol{\omega} = \boldsymbol{R}_i\,{}^{i}\boldsymbol{\omega} \qquad\qquad \dot{\boldsymbol{\omega}} = \boldsymbol{R}_i\,{}^{i}\dot{\boldsymbol{\omega}} $$

- Use again ${}^{i}\boldsymbol{R}_f = \boldsymbol{R}_i^T \boldsymbol{R}_f$
- Relative rotation matrix matched with the inverse axis–angle solution given by unit axis ${}^{i}\boldsymbol{r}$ and angle $\vartheta_{if} \neq 0$

$$\boldsymbol{R}({}^{i}\boldsymbol{r}, \vartheta_{if}) = {}^{i}\boldsymbol{R}_f$$

- Linear interpolation on ϑ

$$\vartheta(s) = s\,\vartheta_{if} \qquad s \in [0, 1]$$

$$\boldsymbol{R}(s) = \boldsymbol{R}_i \boldsymbol{R}({}^{i}\boldsymbol{r}, \vartheta(s)) \qquad\qquad \boldsymbol{R}(1) = \boldsymbol{R}_f$$

- Choice of the timing law for the axis-angle method

$$s(t) = 3\tau^2 - 2\tau^3 \qquad \tau = t/T \in [0,1]$$

$$\vartheta(t) = \vartheta_{if}\left(3(t/T)^2 - 2(t/T)^3\right) \qquad t \in [0,T]$$

$${}^i\boldsymbol{\omega} = \dot{\vartheta}\ {}^i\boldsymbol{r} \qquad {}^i\dot{\boldsymbol{\omega}} = \ddot{\vartheta}\ {}^i\boldsymbol{r}$$

- Satisfying kinetic bounds

$$\|\boldsymbol{\omega}\| \le \omega_{\max} \quad \Rightarrow \quad |\dot{s}| \le \omega_{\max}/\vartheta_{if} \qquad \|\dot{\boldsymbol{\omega}}\| \le \dot{\omega}_{\max} \quad \Rightarrow \quad |\ddot{s}| \le \dot{\omega}_{\max}/\vartheta_{if}$$

$$|\boldsymbol{\omega}_i| \le \omega_{i,\max} \quad i = x,y,z \quad \Rightarrow \quad |\dot{s}| \le \min\left\{\frac{\omega_{i,\max}}{r_i\,\vartheta_{if}},\ i = x,y,z\right\}$$

$$|\dot{\boldsymbol{\omega}}_i| \le \dot{\omega}_{i,\max} \quad i = x,y,z \quad \Rightarrow \quad |\ddot{s}| \le \min\left\{\frac{\dot{\omega}_{i,\max}}{r_i\,\vartheta_{if}},\ i = x,y,z\right\}$$

where r_i is the i-th component of $\boldsymbol{r} = \boldsymbol{R}_i{}^i\boldsymbol{r}$

Example

- Rest-to-rest end-effector motion between

$$\boldsymbol{p}_i = (1, 1, 1)\ [\mathrm{m}] \qquad \boldsymbol{p}_f = (3, 4, 5)\ [\mathrm{m}] \qquad L = \|\boldsymbol{p}_f - \boldsymbol{p}_i\| = 5.3852\ \mathrm{m}$$

- Initial and final orientations

$$\boldsymbol{R}_i = \begin{pmatrix} -1 & 0 & 0 \\ 0 & -\sqrt{2}/2 & -\sqrt{2}/2 \\ 0 & -\sqrt{2}/2 & \sqrt{2}/2 \end{pmatrix} \qquad \boldsymbol{R}_f = \begin{pmatrix} -\sqrt{2}/2 & -\sqrt{2}/2 & 0 \\ 0 & 0 & 1 \\ -\sqrt{2}/2 & \sqrt{2}/2 & 0 \end{pmatrix}$$

- Relative rotation

$${}^{i}\boldsymbol{R}_f = \begin{pmatrix} \sqrt{2}/2 & \sqrt{2}/2 & 0 \\ 1/2 & -1/2 & -\sqrt{2}/2 \\ -1/2 & 1/2 & -\sqrt{2}/2 \end{pmatrix}$$

- Absolute ZYZ Euler angles

$$\boldsymbol{\phi}_i = (-\pi/2, \pi/4, -\pi/2) \qquad \boldsymbol{\phi}_f = (\pi/2, \pi/2, \pi/4) \qquad \text{[rad]}$$

- Relative ZYZ Euler angles

$$\boldsymbol{\phi}_{if} = (-\pi/2, 3\,\pi/4, \pi/4) \qquad \text{[rad]}$$

- Axis–angle

$${}^{i}\boldsymbol{r} = (0.9125, 0.3780, -0.1566) \qquad \vartheta_{if} = 2.4189 \quad \text{rad}$$

- Linear paths for position and orientation, with $s \in [0, 1]$

$$
\begin{aligned}
\boldsymbol{p}(s) &= \boldsymbol{p}_i + s\,(\boldsymbol{p}_f - \boldsymbol{p}_i) && \text{position} \\
\boldsymbol{\phi}(s) &= \boldsymbol{\phi}_i + s\,(\boldsymbol{\phi}_f - \boldsymbol{\phi}_i) && \text{orientation using absolute Euler angles} \\
{}^{i}\boldsymbol{\phi}(s) &= s\,\boldsymbol{\phi}_{if} && \text{orientation using relative Euler angles} \\
\vartheta(s) &= s\,\vartheta_{if} && \text{orientation using axis–angle}
\end{aligned}
$$

- Timing law with trapezoidal speed profile to comply with the limits

$$
\|\dot{\boldsymbol{p}}\| \leq v_{\max} = 1.5 \qquad \|\ddot{\boldsymbol{p}}\| \leq a_{\max} = 3
$$

$$
\|\boldsymbol{\omega}\| \leq \omega_{\max} = \frac{\pi}{4} \qquad \|\dot{\boldsymbol{\omega}}\| \leq \dot{\omega}_{\max} = \frac{\pi}{8}
$$

- Timing laws are designed first separately for the position and for each of representation of orientation, then in a coordinated way so to satisfy all bounds

- Bounds for position timing law

$$V = v_{\max}/L \qquad A = a_{\max}/L \qquad \Rightarrow \qquad T_{r,\mathrm{pos}} = 0.5 \qquad T_{\mathrm{pos}} = 4.0901 \ \text{[s]}$$

- Bounds on absolute Euler angles

$$|\dot{s}| \leq 0.1961 \ \mathrm{s}^{-1} \quad |\ddot{s}| \leq 0.1306 \ \mathrm{s}^{-2} \ \Rightarrow \ T_{r,\mathrm{Eabs}} = 1.5015 \ \ T_{\mathrm{Eabs}} = 6.6009 \ \text{[s]}$$

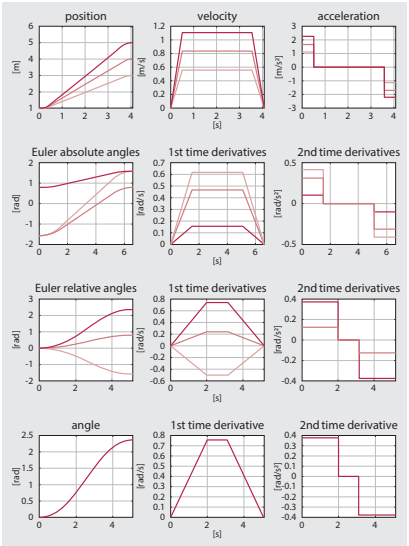
- Bounds on relative Euler angles

$$|\dot{s}| \leq 0.3162 \ \mathrm{s}^{-1} \quad |\ddot{s}| \leq 0.1581 \ \mathrm{s}^{-2} \quad \Rightarrow \quad T_{r,\mathrm{Erel}} = 2 \quad T_{\mathrm{Erel}} = 5.1625 \ \text{[s]}$$

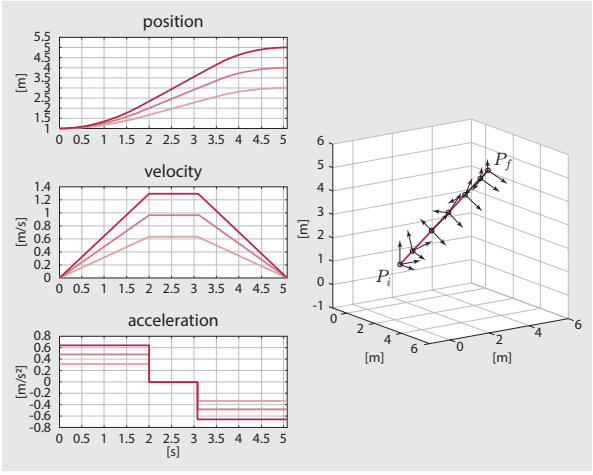
- Bounds on axis–angle method (fastest method, retained for coordinated motion)

$$V = \omega_{\max}/\vartheta_{if} \qquad A = \dot{\omega}_{\max}/\vartheta_{if} \qquad \Rightarrow \qquad T_{r,\mathrm{a\&a}} = 2 \qquad T_{\mathrm{a\&a}} = 5.0798 \ \text{[s]}$$

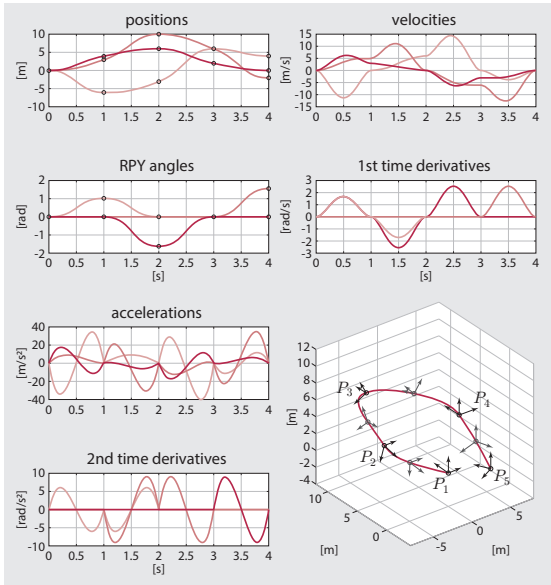
- Separate trajectory design for the position and each of the three orientation methods



- Scaled coordinated trajectory for position and axis–angle ($k = T_{\mathrm{a\&a}}/T_{\mathrm{pos}} = 1.2420$) with fixed axis in base frame $\boldsymbol{r} = \boldsymbol{R}_i\, {}^i\boldsymbol{r} = -(0.9125, 0.1566, 0.3780)$



- Complete example of MP task-space trajectory planning
  - rest-to-rest end-effector trajectory through $N = 5$ knots
  - specified positions and orientations at the knots
  - orientation represented by the ZYX sequence of RPY-type angles ϕ around fixed axes
- At the knots
  - different values are assigned to $\dot{\boldsymbol{p}}$
  - second derivatives $\ddot{\boldsymbol{p}}$ are set to zero
  - first and second derivatives $\dot{\phi}$ and $\ddot{\phi}$ are set to zero
- Each task coordinate follows a sequence of quintic polynomials in time


positions
velocities
RPY angles
1st time derivatives
accelerations
2nd time derivatives
[m]
[m/s]
[rad]
[rad/s]
[m/s²]
[rad/s²]
[s]
P_1
P_2
P_3
P_4
P_5