Trajectory Planning for Robot Manipulators

Claudio Melchiorri

Dipartimento di Elettronica, Informatica e Sistemistica (DEIS)

Università di Bologna

email: claudio.melchiorri@unibo.it

Trajectory planning

Kinematics: geometrical relationships in terms of position/velocity between the joint- and work-space.

Dynamics: relationships between the torques applied to the joints and the consequent movements of the links.

Control: computation of the control actions (joint torques) necessary to execute a desired motion.

Trajectory planning: planning of the *desired* movements of the manipulator.

Usually, the user is requested to define some points and general features of the trajectory (e.g. initial/final points, duration, maximum velocity, etc.), and the real computation of the trajectory is demanded to the control system.

Trajectory planning

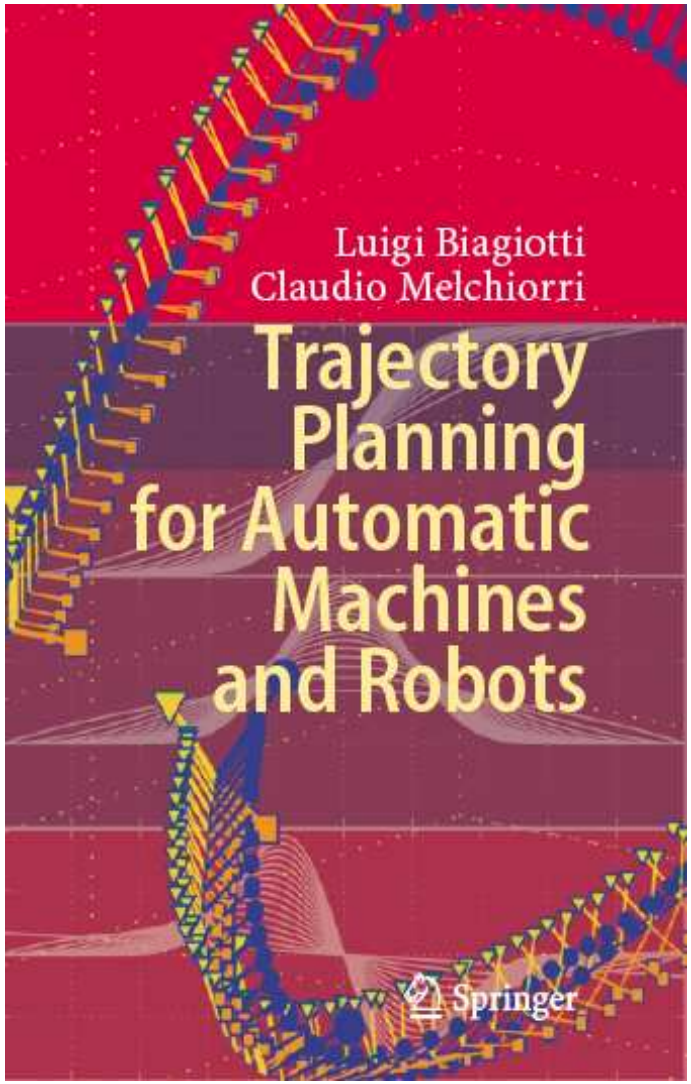
Trajectory planning: IMPORTANT aspect in robotics, VERY IMPORTANT for the dimensioning, control, and use of electric motors in automatic machines (e.g. packaging).

Original problem (in mechanics): substitution of mechanical cams with *electric cams*.

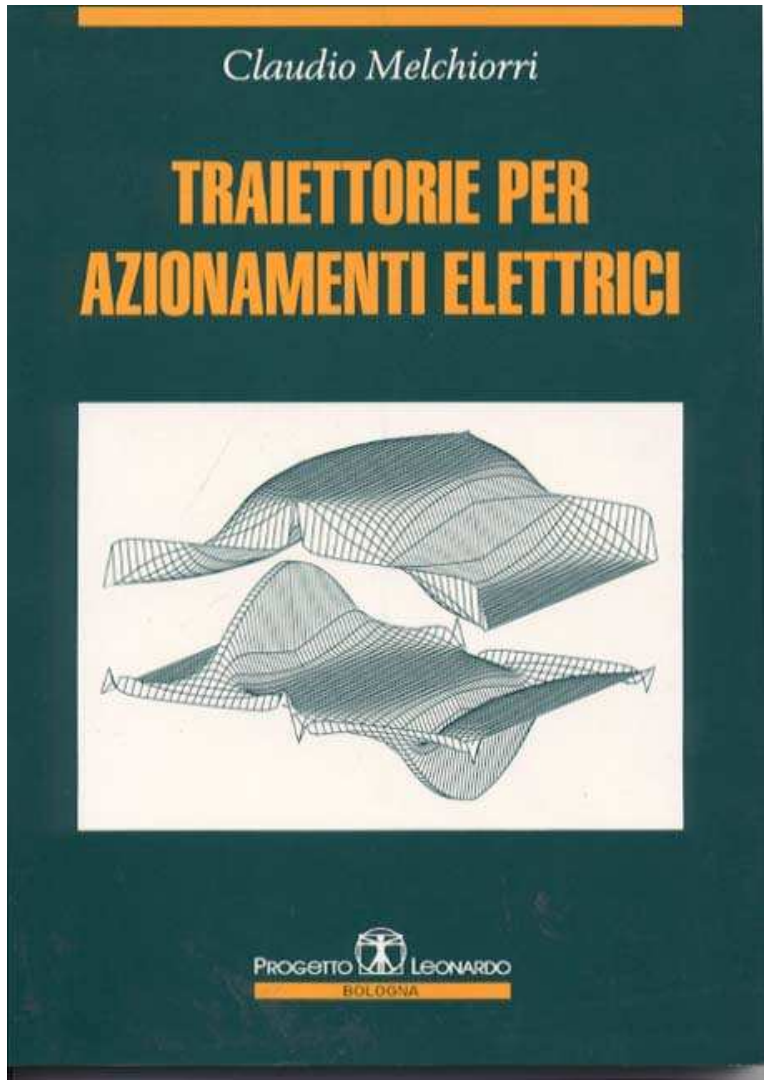
Some suggested references:

- C. Melchiorri, *Traiettorie per azionamenti elettrici*, Progetto Leonardo, Esculapio Ed., Bologna, Feb. 2000;
- G. Canini, C. Fantuzzi, *Controllo del moto per macchine automatiche*, Pitagora Ed., Bo, 2003;
- G. Legnani, M. Tiboni, R. Adamini, *Meccanica degli azionamenti: Vol. 1 - Azionamenti elettrici*, Progetto Leonardo, Esculapio Ed., Bologna, Feb. 2002.
- L. Biagiotti, C. Melchiorri, *Trajectory Planning for Automatic Machines and Robots*, Springer, 2008.

Trajectory planning



Springer, 2008



Esculapio, 2000

Trajectory planning

The planning modalities for trajectories may be quite different:

- point-to-point
- with pre-defined path

Or:

- in the joint space;
- in the work space, either defining some points of interest (initial and final points, *via points*) or the whole geometric path $\mathbf{x} = \mathbf{x}(t)$.

For planning a desired trajectory, it is necessary to specify two aspects:

- *geometric path*
- *motion law*

with constraints on the continuity (smoothness) of the trajectory and on its time-derivatives up to a given degree.

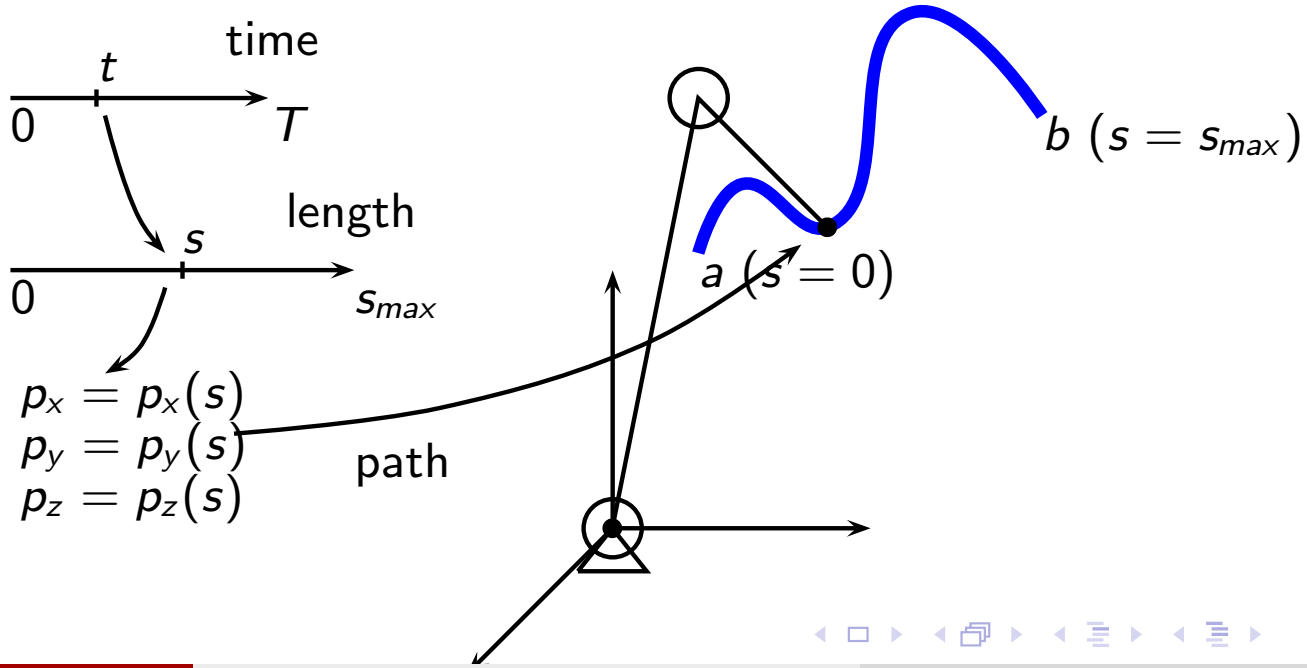
Geometric path and motion law

The geometric path can be defined in the work-space or in the joint-space. Usually, it is expressed in a parametric form as

$$p = p(s) \qquad \text{work-space}$$

$$q = q(\sigma) \qquad \text{joint-space}$$

The parameter s (σ) is defined as a function of time, and in this manner the motion law $s = s(t)$ ($\sigma = \sigma(t)$) is obtained.



Geometric path and motion law

Examples of geometric paths: (in the work space) linear, circular or parabolic segments or, more in general, tracts of analytical functions.

In the joint space, geometric paths are obtained by assigning initial/final (and, in case, also intermediate) values for the joint variables, along with the desired motion law.

Concerning the motion law, it is necessary to specify continuos functions up to a given order of derivations (often at least first and second order, i.e. velocity and acceleration).

Usually, polynomial functions a of proper degree n are employed:

$$s(t) = a_0 + a_1 t + a_2 t^2 + \ldots + a_n t^n$$

In this manner, a "smooth" interpolation of the points defining the geometric path is achieved.

Trajectory planning

Input data to an algorithm for trajectory planning are:

- data defining on the path (points),
- geometrical constraints on the path (e.g. obstacles),
- constraints on the mechanical dynamics
- constraints due to the actuation system

Output data is:

- the trajectory in the joint- or work-space, given as a sequence (in time) of the acceleration, velocity and position values:

$$a(kT), \qquad v(kT), \qquad p(kT) \qquad k = 0, \ldots, N$$

being T a proper time interval defining the instants in which the trajectory is computed (and converted in the joint space) and sent to each actuator.

Trajectory planning

Usually, the user has to specify only a minimum amount of information about the trajectory, such as initial and final points, duration of the motion, maximum velocity, and so on.

- *Work-space trajectories* allow to consider directly possible constraints on the path (obstacles, path geometry, . . .) that are more difficult to take into consideration in the joint space (because of the non linear kineamtics)
- *Joint space trajectories* are computationally simpler and allow to consider problems due to singular configurations, actuation redundancy, velocity/acceleration constraints.

Joint-space trajectories

Trajectories are specified by defining some characteristic points:

- directly assigned by some specifications
- assigned by defining desired configurations $\mathbf{x}$ in the work-space, which are then converted in the joint space using the inverse kinematic model.

The algorithm that computes a function $\mathbf{q}(t)$ *interpolating* the given points is characterized by the following features:

- trajectories must be computationally efficient
- the position and velocity profiles (at least) must be continuos functions of time
- undesired effects (such as non regular curvatures) must be minimized or completely avoided.

In the following discussion, a single joint is considered.

If more joints are present, a coordinated motion must be planned, e.g. considering for each of them the same initial and final time instant, or evaluating the most stressed joint (with the largest displacement) and then scaling suitably the motion of the remaining ones.

Polynomial trajectories

In the most simple cases, (a segment of) a trajectory is specified by assigning initial and final conditions on: time (duration), position, velocity, acceleration, ... Then, the problem is to determine a function

$$q = q(t) \qquad \text{or} \qquad q = q(\sigma), \qquad \sigma = \sigma(t)$$

so that those conditions are satisfied.

This is a boundary condition problem, that can be easily solved by considering polynomial functions such as:

$$q(t) = a_0 + a_1 t + a_2 t^2 + \ldots + a_n t^n$$

The degree n (3, 5, ...) of the polynomial depends on the number of boundary conditions that must be verified and on the desired "smoothness" of the trajectory.

Polynomial trajectories

In general, besides the initial and final values, other constraints could be specified on the values of some time-derivatives (velocity, acceleration, jerk, ...) in generic instants t_j. In other terms, one could be interested in defining a polynomial function $q(t)$ whose k-th derivative has a specified value $q^k(t_j)$ at a given istant t_j.

Mathematically, these conditions may be expressed as:

$$k!a_k + (k+1)!a_{k+1}t_j + \ldots + \frac{n!}{(n-k)!}a_n t_j^{n-k} = q^k(t_j)$$

or, in matrix form:

$$\mathbf{M}\,\mathbf{a} = \mathbf{b}$$

where:

- $\mathbf{M}$ is a known $(n+1) \times (n+1)$ matrix,
- $\mathbf{b}$ is the vector with the $n+1$ constraints on the trajectory (known data),
- $\mathbf{a} = [a_0, a_1, \ldots, a_n]^T$ contains the unknown parameters to be computed

$$\mathbf{a} = \mathbf{M}^{-1}\mathbf{b}$$

Third-order polynomial trajectories

Given an initial and a final instant t_i, t_f, a (segment of a) trajectory may be specified by assigning initial and final conditions:

- initial position and velocity q_i, $\dot{q}_i$;
- final position and velocity q_f, $\dot{q}_f$

There are four boundary conditions, and therefore a polynomial of degree 3 (at least) must be considered

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{1}$$

where the four parameters a_0, a_1, a_2, a_3 must be defined so that the boundary conditions are satisfied.From the boundary conditions, it follows that

$$\begin{array}{rcrcl} q(t_i) & = & a_0 + a_1 t_i + a_2 t_i^2 + a_3 t_i^3 & = & q_i \\ \dot{q}(t_i) & = & a_1 + 2a_2 t_i + 3a_3 t_i^2 & = & \dot{q}_i \\ q(t_f) & = & a_0 + a_1 t_f + a_2 t_f^2 + a_3 t_f^3 & = & q_f \\ \dot{q}(t_f) & = & a_1 + 2a_2 t_f + 3a_3 t_f^2 & = & \dot{q}_f \end{array} \tag{2}$$

Third-order polynomial trajectories

In order to solve these equations, let us assume for the moment that $t_i = 0$. Therefore:

$$a_0 = q_i \tag{3}$$

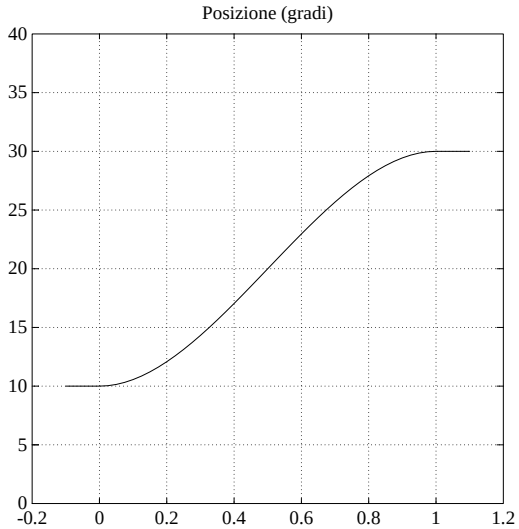
$$a_1 = \dot{q}_i \tag{4}$$

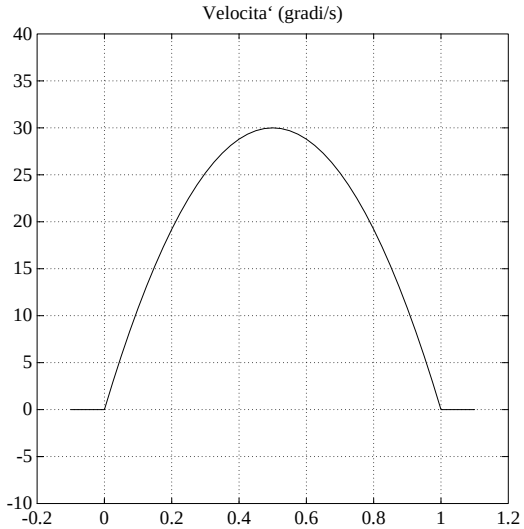
$$a_2 = \frac{-3(q_i - q_f) - (2\dot{q}_i + \dot{q}_f)t_f}{t_f^2} \tag{5}$$

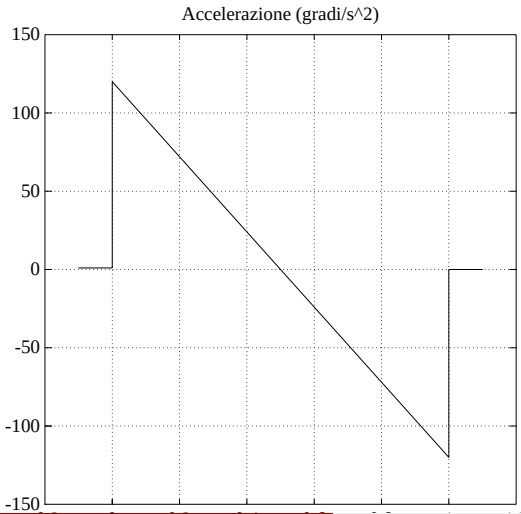
$$a_3 = \frac{2(q_i - q_f) + (\dot{q}_i + \dot{q}_f)t_f}{t_f^3} \tag{6}$$

Third-order polynomial trajectories

Position, velocity and acceleration profiles obtained with a cubic polynomial and boundary conditions: $q_i = 10^o, q_f = 30^o$, $\dot{q}_i = \dot{q}_f = 0\ ^o/s$, ßt$_i = 0, t_f = 1s$:





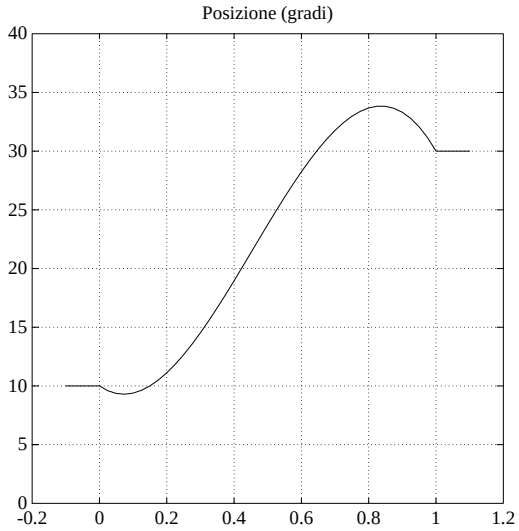


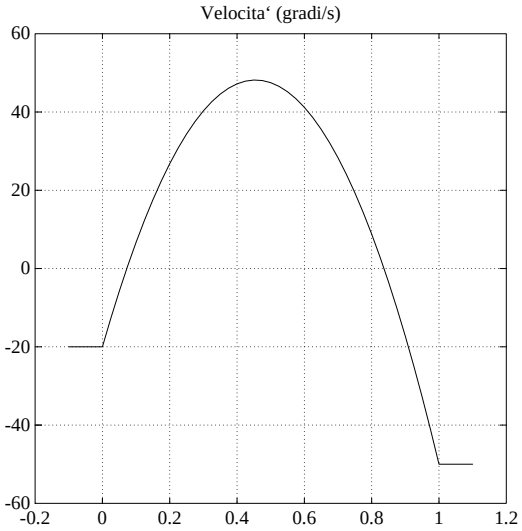
Obviously:

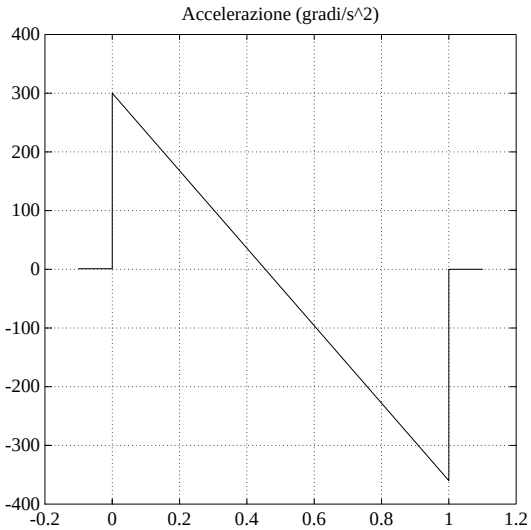
position	$\rightarrow$	cubic function
velocity	$\rightarrow$	parabolic function
acceleration	$\rightarrow$	linear function

Third-order polynomial trajectories

Non null initial velocity: $q_i = 10^o, q_f = 30^o, \dot{q}_i = -20\ ^o/s, \dot{q}_f = -50\ ^o/s,\ t_i = 0, t_f = 1s.$







Third-order polynomial trajectories

The results obtained with the polynomial (1) and the coefficients (3)-(6) can be generalized to the case in which $t_i \neq 0$. One obtains:

$$q(t) = a_0 + a_1(t - t_i) + a_2(t - t_i)^2 + a_3(t - t_i)^3 \qquad t_i \leq t \leq t_f$$

with coefficients

$$\begin{aligned} a_0 &= q_i \\ a_1 &= \dot{q}_i \\ a_2 &= \frac{-3(q_i - q_f) - (2\dot{q}_i + \dot{q}_f)(t_f - t_i)}{(t_f - t_i)^2} \\ a_3 &= \frac{2(q_i - q_f) + (\dot{q}_i + \dot{q}_f)(t_f - t_i)}{(t_f - t_i)^3} \end{aligned}$$

In this manner, it is very simple to plan a trajectory passing through a sequence of intermediate points.

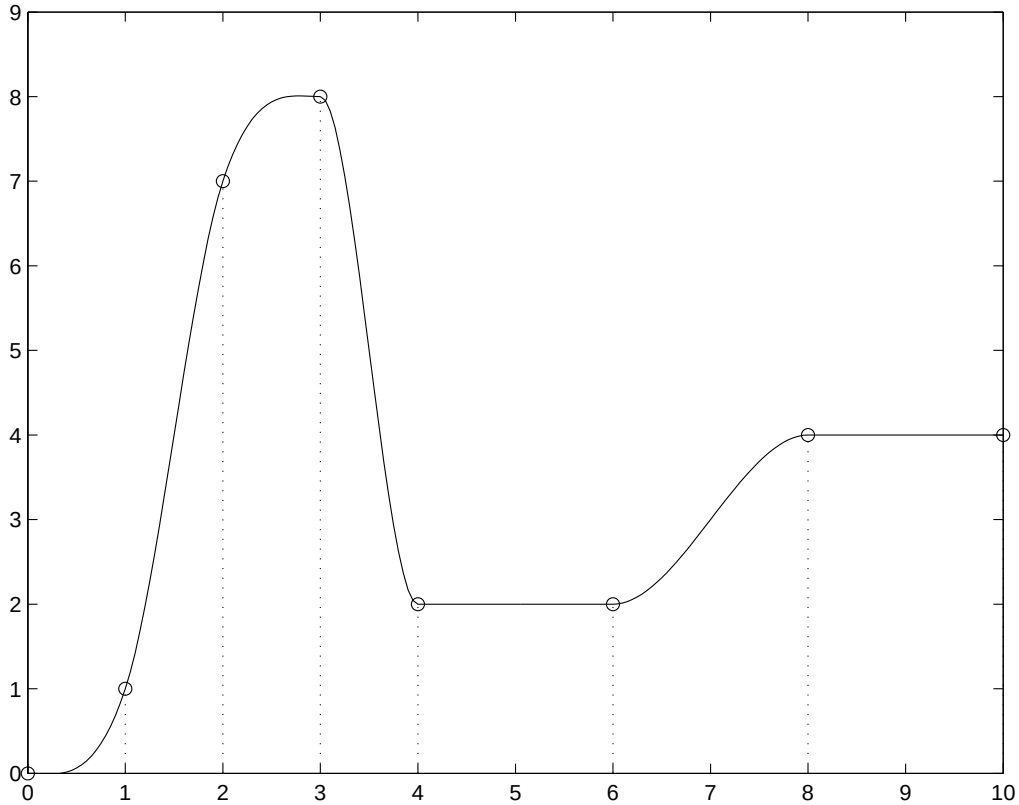
Third-order polynomial trajectories

The trajectory is divided in n segments, each of them defined by:

- initial and final point q_k e q_{k+1}
- initial and final instant t_k, t_{k+1}
- initial and final velocity $\dot{q}_k, \dot{q}_{k+1}$

$$k = 0, \ldots, n-1.$$

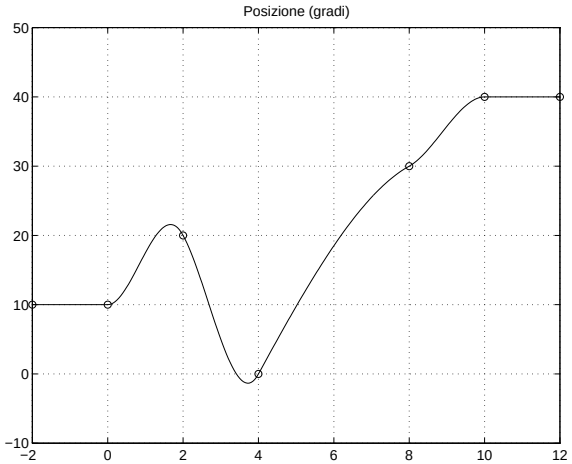
The above relationships are then adopted for each of these segments.

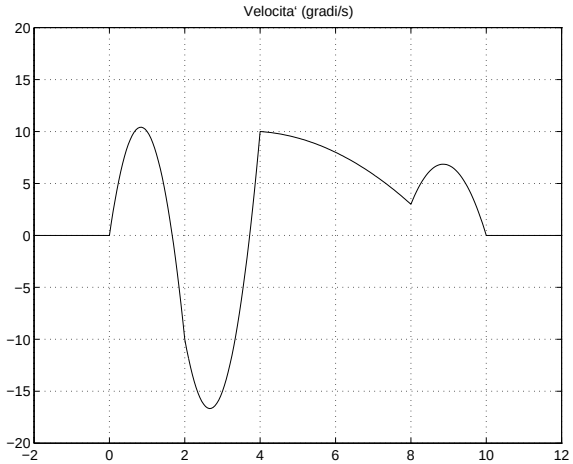


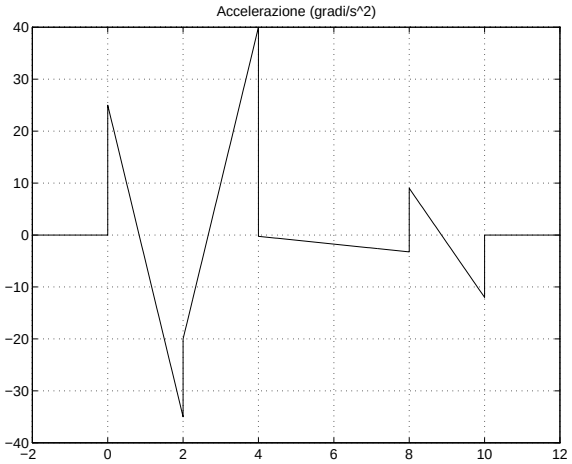
Third-order polynomial trajectories

Position, velocity and acceleration profiles with:

$t_0 = 0$	$t_1 = 2$	$t_2 = 4$	$t_3 = 8$	$t_4 = 10$
$q_0 = 10^o$	$q_1 = 20^o$	$q_2 = 0^o$	$q_3 = 30^o$	$q_4 = 40^o$
$\dot{q}_0 = 0^o/s$	$\dot{q}_1 = -10^o/s$	$\dot{q}_2 = 20^o/s$	$\dot{q}_3 = 3^o/s$	$\dot{q}_4 = 0^o/s$







Third-order polynomial trajectories

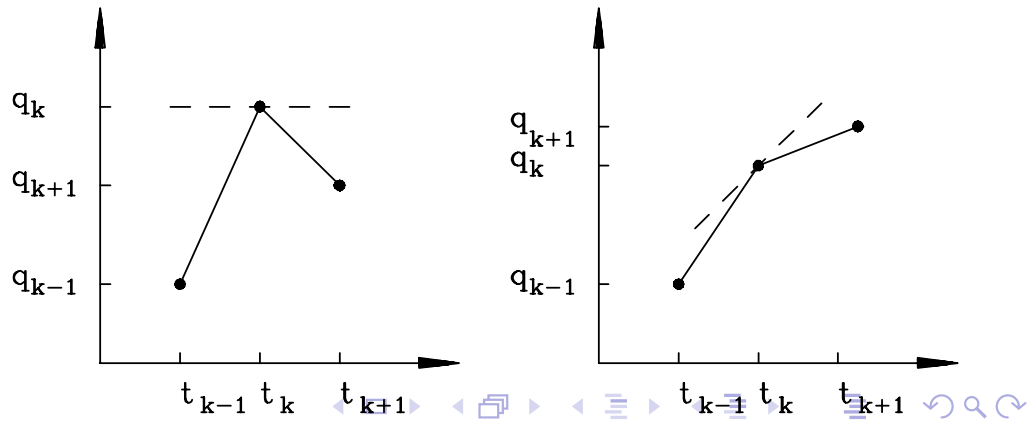
Often, a trajectory is assigned by specifying a sequence of desired points (*via-points*) without indication on the velocity in these points.
In these cases, the "most suitable" values for the velocities must be automatically computed.
This assignment is quite simple with heuristic rules such as:

$$\begin{aligned} \dot{q}_1 &= 0; \\ \dot{q}_k &= \begin{cases} 0 & \text{sign}(v_k) \neq \text{sign}(v_{k+1}) \\ \frac{1}{2}(v_k + v_{k+1}) & \text{sign}(v_k) = \text{sign}(v_{k+1}) \end{cases} \\ \dot{q}_n &= 0 \end{aligned}$$

being

$$v_k = \frac{q_k - q_{k-1}}{t_k - t_{k-1}}$$

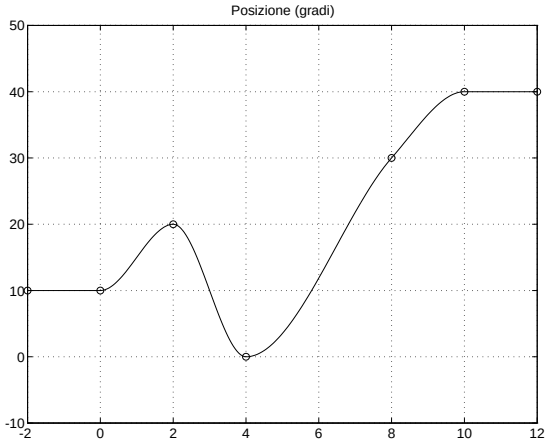
the 'slope' of the tract $[t_{k-1} - t_k]$.

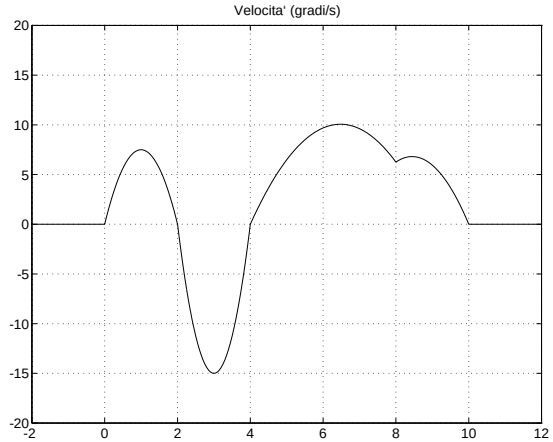


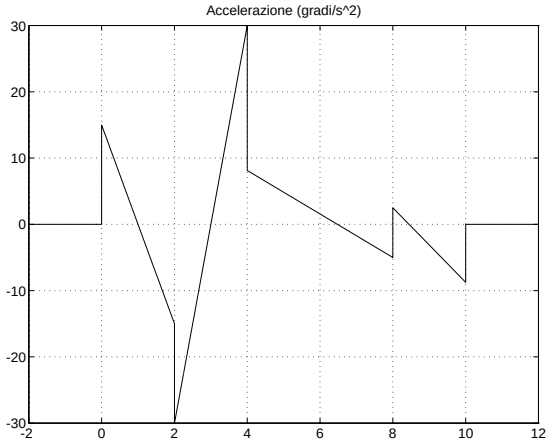
Third-order polynomial trajectories

Automatic computation of the intermediate velocities (data as in the previous example)

$t_0 = 0$ $q_0 = 10^o$ $t_1 = 2$ $q_1 = 20^o$ $t_2 = 4$ $q_2 = 0^o$ $t_3 = 8$ $q_3 = 30^o$ $t_4 = 10$ $q_4 = 40^o$







Fifth-order polynomial trajectories

From the above examples, it may be noticed that both the position and velocity profiles are continuous functions of time.

This is not true for the acceleration, that presents therefore discontinuities among different segments. Moreover, it is not possible to specify for this signal suitable initial/final values in each segment.

In many applications, these aspects do not constitute a problem, being the trajectories "smooth" enough.

On the other hand, if it is requested to specify initial and final values for the acceleration (e.g. for obtaining acceleration profiles), then (at least) fifth-order polynomial functions should be considered

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5$$

with the six boundary conditions:

$$\begin{aligned} q(t_i) &= q_i & \qquad q(t_f) &= q_f \\ \dot{q}(t_i) &= \dot{q}_i & \qquad \dot{q}(t_f) &= \dot{q}_f \\ \ddot{q}(t_i) &= \ddot{q}_i & \qquad \ddot{q}(t_f) &= \ddot{q}_f \end{aligned}$$

Fifth-order polynomial trajectories

In this case, (if $T = t_f - t_i$) the coefficients of the polynomial are

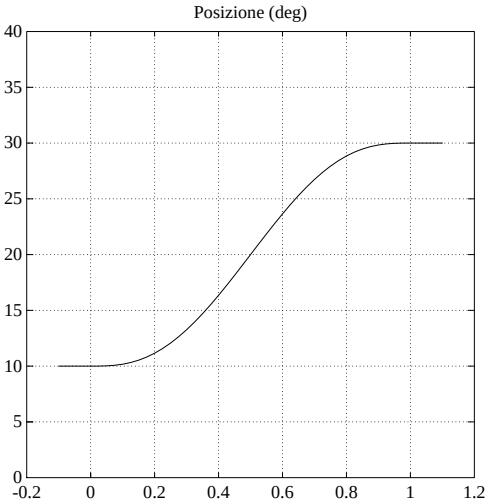
$$
\begin{aligned}
a_0 &= q_i \\
a_1 &= \dot{q}_i \\
a_2 &= \frac{1}{2}\ddot{q}_i \\
a_3 &= \frac{1}{2T^3}[20(q_f - q_i) - (8\dot{q}_f + 12\dot{q}_i)T - (3\ddot{q}_f - \ddot{q}_i)T^2] \\
a_4 &= \frac{1}{2T^4}[30(q_i - q_f) + (14\dot{q}_f + 16\dot{q}_i)T + (3\ddot{q}_f - 2\ddot{q}_i)T^2] \\
a_5 &= \frac{1}{2T^5}[12(q_f - q_i) - 6(\dot{q}_f + \dot{q}_i)T - (\ddot{q}_f - \ddot{q}_i)T^2]
\end{aligned}
$$

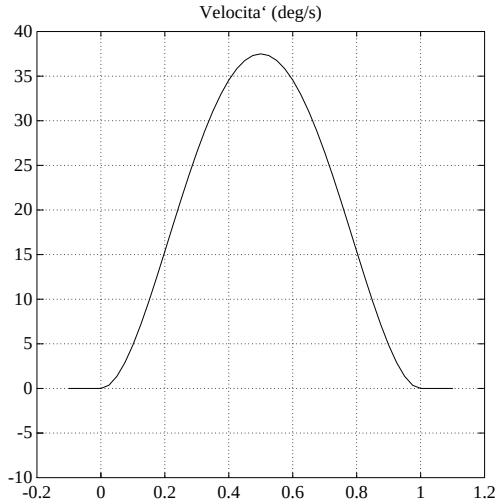
If a sequence of points is given, the same considerations made for third-order polynomials trajectories can be made in computing the intermediate velocity values.

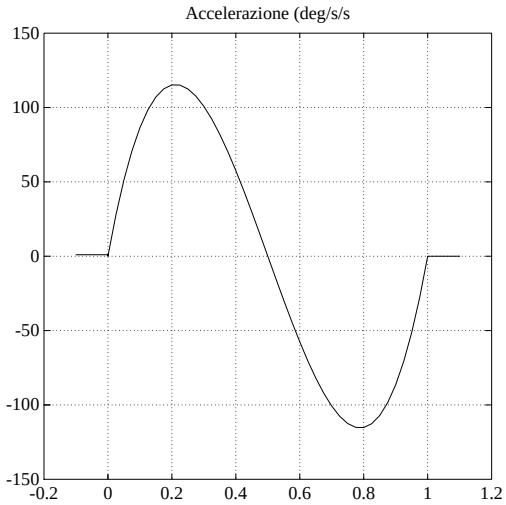
Fifth-order polynomial trajectories

Fifth-order trajectory with the boundary conditions:

$q_i = 10^o, q_f = 30^o, \dot{q}_i = \dot{q}_f = 0\ ^o/s, \ddot{q}_i = \ddot{q}_f = 0\ ^o/s^2, t_i = 0s, t_f = 1s.$





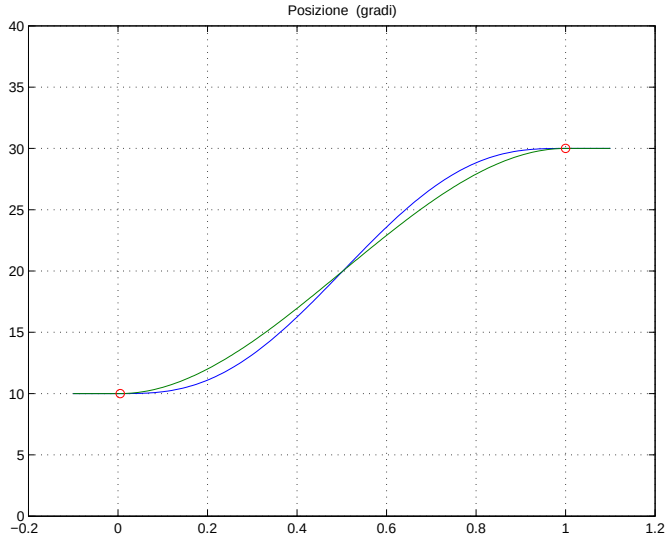


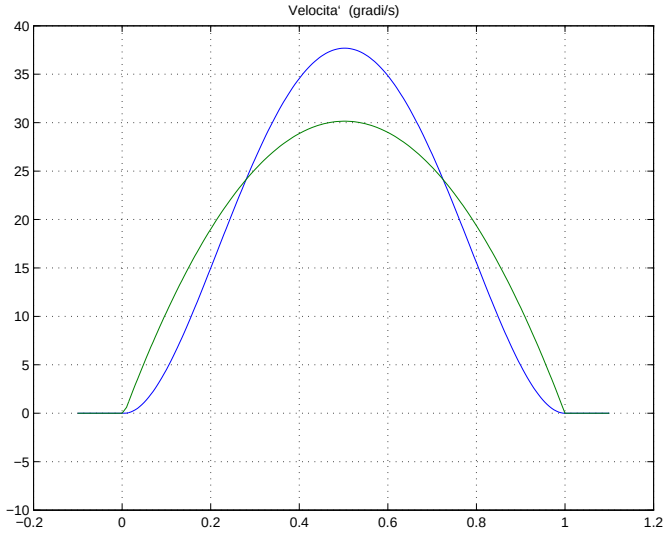
Obviously:

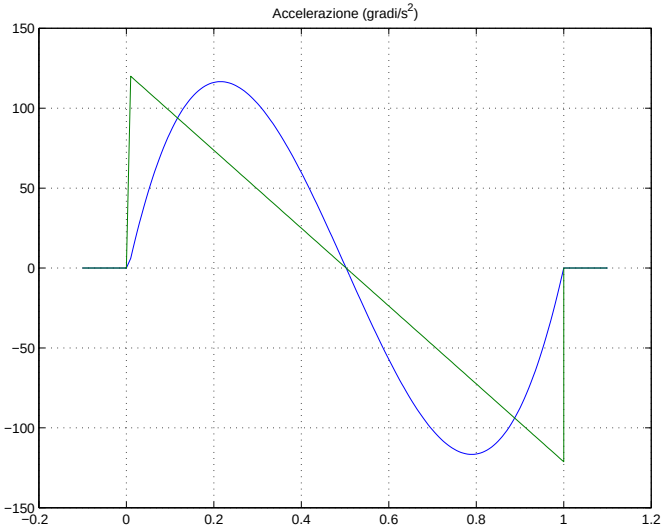
position	$\rightarrow$	5-th order function
velocity	$\rightarrow$	4-th order function
acceleration	$\rightarrow$	3-rd order function

Fifth-order polynomial trajectories

Comparison of fifth- and third-order trajectories with the boundary conditions:
$q_i = 10^o, q_f = 30^o, \dot{q}_i = \dot{q}_f = 0\ ^o/s, \ddot{q}_i = \ddot{q}_f = 0\ ^o/s^2, t_i = 0s, t_f = 1s.$



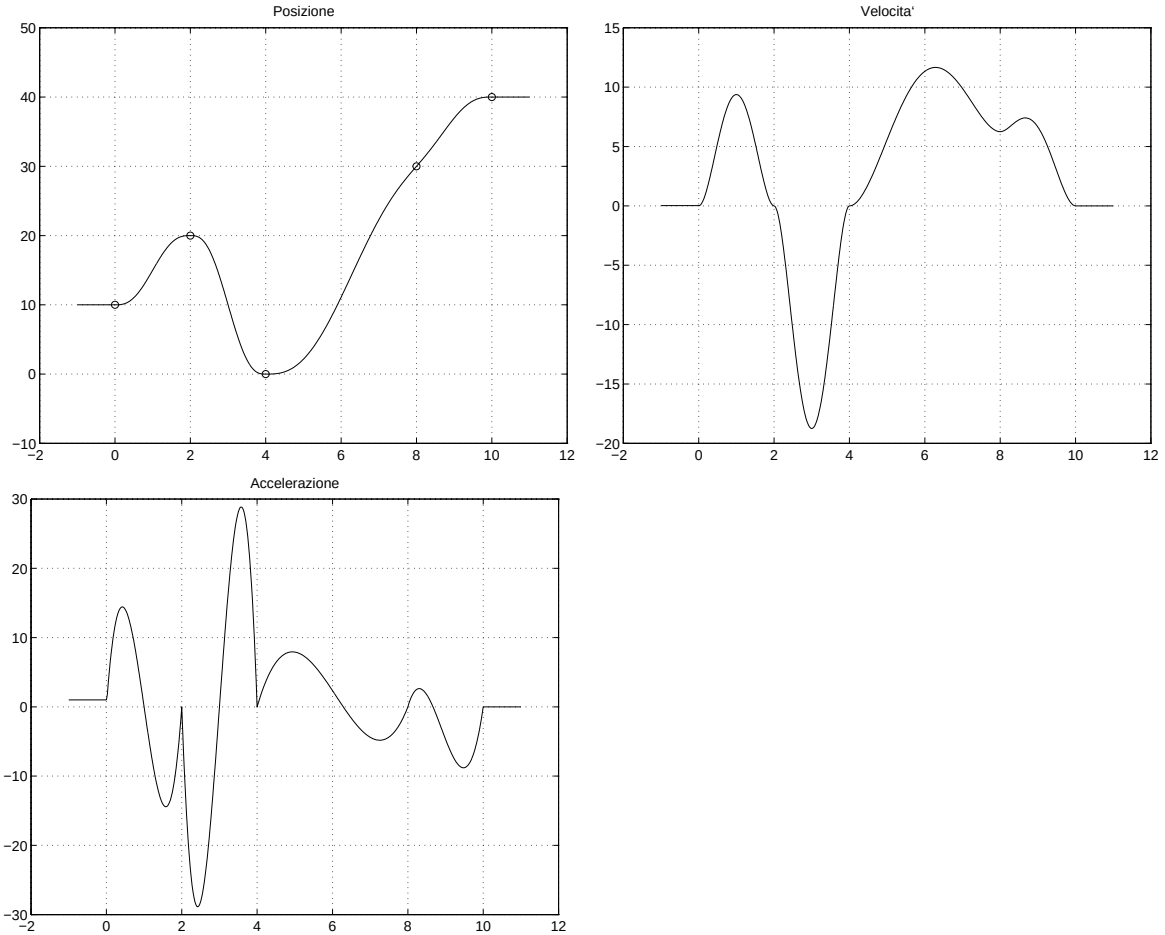




blue $\rightarrow$ fifth-order
green $\rightarrow$ third-order

Fifth-order polynomial trajectories

Position, velocity, acceleration profiles with automatic assignment of the intermediate velocities and null accelerations.



Note that the resulting motion has smoother profiles.

Trapezoidal trajectories

A different approach for planning a trajectory is to compute linear segments joined with parabolic blends.

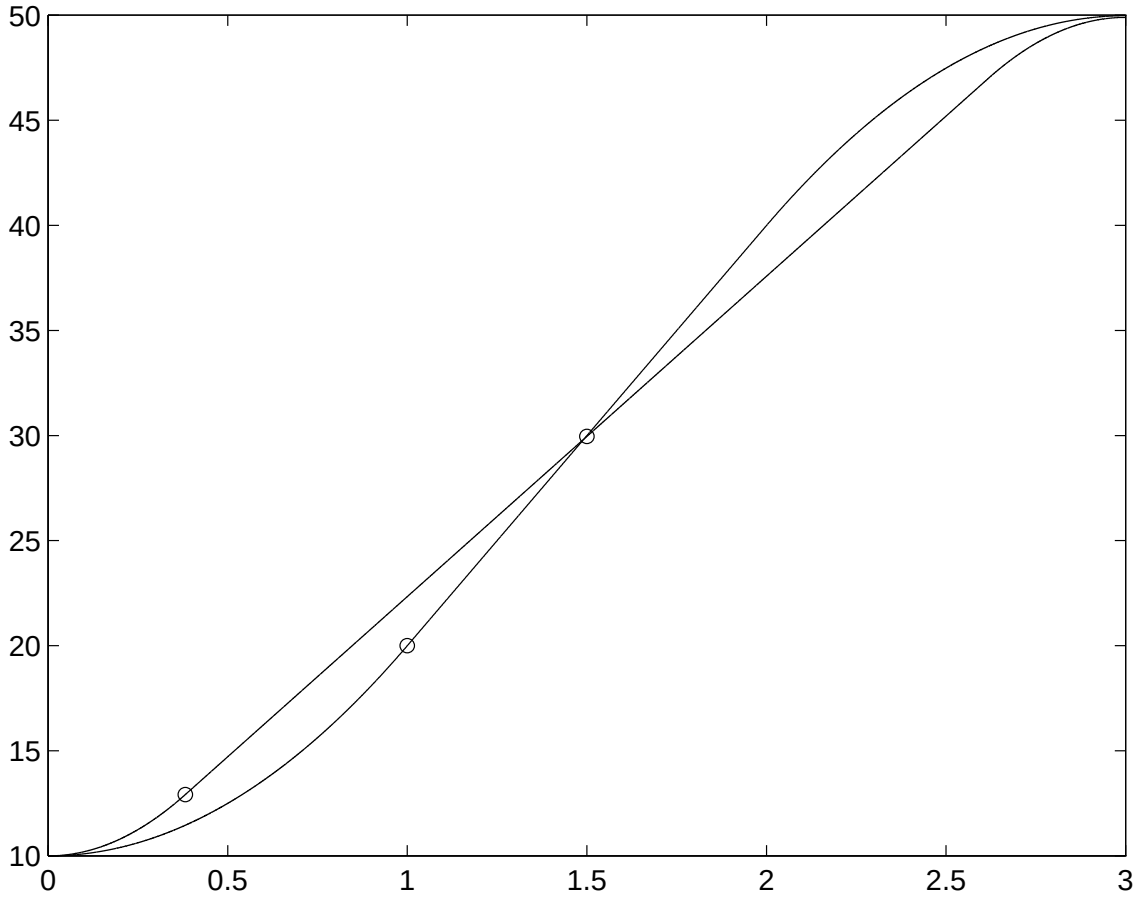
In the linear tract, the velocity is constant while, in the parabolic blends, it is a linear function of time: trapezoidal profiles, typical of this type of trajectory, are then obtained.

In trapezoidal trajectories, the duration is divided into three parts:

1. in the first part, a constant acceleration is applied, then the velocity is linear and the position a parabolic function of time
2. in the second, the acceleration is null, the velocity is constant and the position is linear in time
3. in the last part a (negative) acceleration is applied, then the velocity is a negative ramp and the position a parabolic function.

Trapezoidal trajectories

Usually, the acceleration and the deceleration phases have the same duration ($t_a = t_d$). Therefore, symmetric profiles, with respect to a central instant $(t_f - t_i)/2$, are obtained.



Trapezoidal trajectories

The trajectory is computed according to the following equations.

1) **Acceleration phase, $t \in [0 \div t_a]$.**
The position, velocity and acceleration are described by

$$\begin{aligned} q(t) &= a_0 + a_1 t + a_2 t^2 \\ \dot{q}(t) &= a_1 + 2a_2 t \\ \ddot{q}(t) &= 2a_2 \end{aligned}$$

The parameters are defined by constraints on the initial position q_i and the velocity $\dot{q}_i$, and on the desired constant velocity $\dot{q}_v$ that must be obtained at the end of the acceleration period. Assuming a null initial velocity and considering $t_i = 0$ one obtains

$$\begin{aligned} a_0 &= q_i \\ a_1 &= 0 \\ a_2 &= \frac{\dot{q}_v}{2t_a} \end{aligned}$$

In this phase, the acceleration is constant and equal to $\dot{q}_v / t_a$.

Trapezoidal trajectories

2) **Constant velocity phase,** $t \in [t_a \div t_f - t_a]$.
Position, velocity and acceleration are now defined as

$$\begin{aligned} q(t) &= b_0 + b_1 t \\ \dot{q}(t) &= b_1 \\ \ddot{q}(t) &= 0 \end{aligned}$$

where, because of continuity,

$$b_1 = \dot{q}_v$$

Moreover, the following equation must hold

$$q(t_a) = q_i + \dot{q}_v \frac{t_a}{2} = b_0 + \dot{q}_v t_a$$

and then

$$b_0 = q_i - \dot{q}_v \frac{t_a}{2}$$

Trapezoidal trajectories

3) **Deceleration phase,** $t \in [t_f - t_a \div t_f]$.
The position, velocity and acceleration are given by

$$\begin{aligned} q(t) &= c_0 + c_1 t + c_2 t^2 \\ \dot{q}(t) &= c_1 + 2c_2 t \\ \ddot{q}(t) &= 2c_2 \end{aligned}$$

The parameters are now defined with constrains on the final position q_f and velocity $\dot{q}_f$, and on the velocity $\dot{q}_v$ at the beginning of the deceleration period. If the final velocity is null, then:

$$\begin{aligned} c_0 &= q_f - \frac{\dot{q}_v}{2}\frac{t_f^2}{t_a} \\ c_1 &= \dot{q}_v \frac{t_f}{t_a} \\ c_2 &= -\frac{\dot{q}_v}{2t_a} \end{aligned}$$

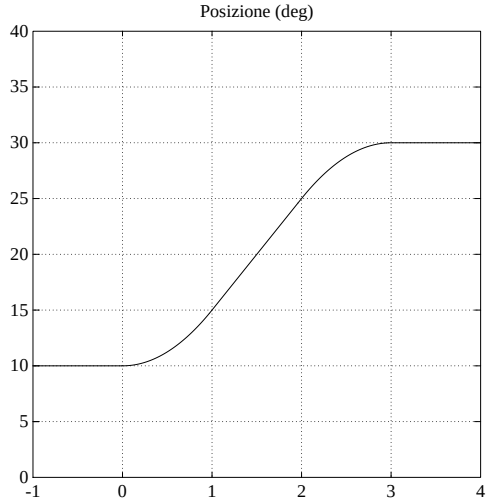
Trapezoidal trajectories

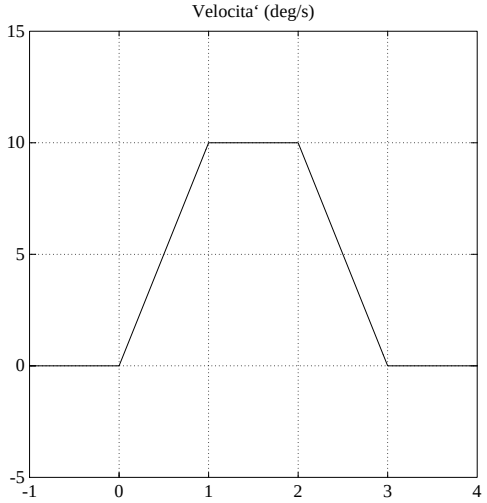
Summarizing, the trajectory is computed as

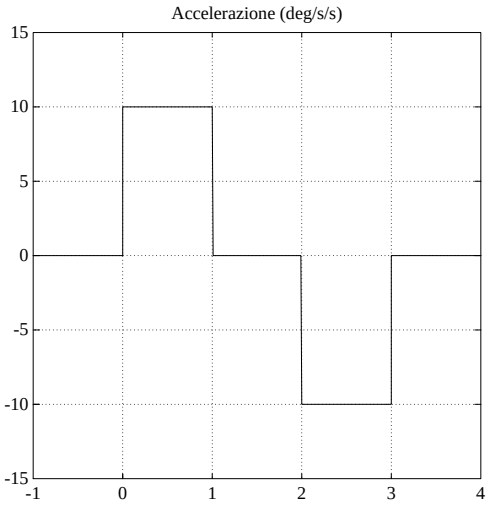
$$q(t) = \begin{cases} q_i + \frac{\dot{q}_v}{2t_a} t^2 & 0 \leq t < t_a \\ q_i + \dot{q}_v (t - \frac{t_a}{2}) & t_a \leq t < t_f - t_a \\ q_f - \frac{\dot{q}_v}{t_a} \frac{(t_f - t)^2}{2} & t_f - t_a \leq t \leq t_f \end{cases}$$

Trapezoidal trajectories

Typical position, velocity and acceleration profiles of a trapezoidal trajectory.







Trapezoidal trajectories

Some additional constraints must be specified in order to solve the previous equations.
A typical constraint concerns the duration of the acceleration/deceleration periods t_a which, for symmetry, must satisfy the condition

$$t_a \leq t_f/2$$

Moreover, the following condition must be verified (for the sake of simplicity, consider $t_i = 0$):

$$\ddot{q}t_a = \frac{q_m - q_a}{t_m - t_a} \qquad \begin{cases} q_a = q(t_a) \\ q_m = (q_i + q_f)/2 \\ t_m = t_f/2 \end{cases}$$

$$q_a = q_i + \frac{1}{2}\ddot{q}t_a^2$$

from which

$$\ddot{q}t_a^2 - \ddot{q}t_f t_a + (q_f - q_i) = 0 \tag{7}$$

Finally:

$$\dot{q}_v = \frac{q_f - q_i}{t_f - t_a}$$

Trapezoidal trajectories

Any pair of values $(\ddot{q}, t_a)$ verifying (7) can be considered.

Given the acceleration $\ddot{q}$ (for example $\ddot{q}_{max}$), then

$$t_a = \frac{t_f}{2} - \frac{\sqrt{\ddot{q}^2 t_f^2 - 4\ddot{q}(q_f - q_i)}}{2\ddot{q}}$$

from which we have also that the minimum value for the acceleration is

$$|\ddot{q}| \geq \frac{4|q_f - q_i|}{t_f^2}$$

if the value $|\ddot{q}| = \frac{4|q_f - q_i|}{t_f^2}$ is assigned, then $t_a = t_f/2$ and the constant velocity tract does not exist.

If the value $t_a = t_f/3$ is specified, the following velocity and acceleration values are obtained

$$\dot{q}_v = \frac{3(q_f - q_i)}{2t_f} \qquad \ddot{q} = \frac{9(q_f - q_i)}{2t_f^2}$$

Trapezoidal trajectories

Another way to compute this type of trajectory is to define a maximum value $\ddot{q}_a$ for the desired acceleration and then compute the relative duration t_a of the acceleration and deceleration periods.

If the maximum values ($\ddot{q}_{max}$ and $\dot{q}_{max}$, known) for the acceleration and velocity must be reached, it is possible to assign

$$\begin{cases} t_a = \dfrac{\dot{q}_{max}}{\ddot{q}_{max}} & \text{acceleration time} \\ \dot{q}_{max}(T - t_a) = q_f - q_i = L & \text{displacement} \\ T = \dfrac{L\ddot{q}_{max} + \dot{q}_{max}^2}{\ddot{q}_{max}\dot{q}_{max}} & \text{time duration} \end{cases}$$

and then $(t_f = t_i + T)$

$$q(t) = \begin{cases} q_i + \frac{1}{2}\ddot{q}_{max}(t - t_i)^2 & t_i \le t \le t_i + t_a \\ q_i + \ddot{q}_{max} t_a (t - t_i - \frac{t_a}{2}) & t_i + t_a < t \le t_f - t_a \\ q_f - \frac{1}{2}\ddot{q}_{max}(t_f - t - t_i)^2 & t_f - t_a < t \le t_f \end{cases} \quad (8)$$

Trapezoidal trajectories

In this case, the linear tract exists if and only if

$$L \geq \frac{\dot{q}_{max}^2}{\ddot{q}_{max}}$$

Otherwise

$$\begin{cases} t_a &= \sqrt{\frac{L}{\ddot{q}_{max}}} & \text{acceleration time} \\ T &= 2t_a & \text{total time duration} \end{cases}$$

and (still $t_f = t_i + T$)

$$q(t) = \begin{cases} q_i + \frac{1}{2}\ddot{q}_{max}(t - t_i)^2 & t_i \leq t \leq t_i + t_a \\ q_f - \frac{1}{2}\ddot{q}_{max}(t_f - t)^2 & t_f - t_a < t \leq t_f \end{cases} \tag{9}$$

Trapezoidal trajectories

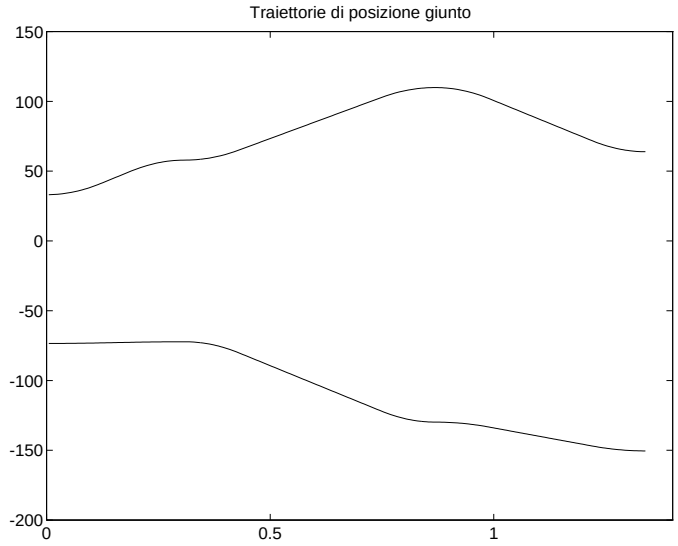
With this modality for computing the trajectory, the time duration of the motion from q_i to q_f is not specified. In fact, the period T is computed on the basis of the maximum acceleration and velocity values.

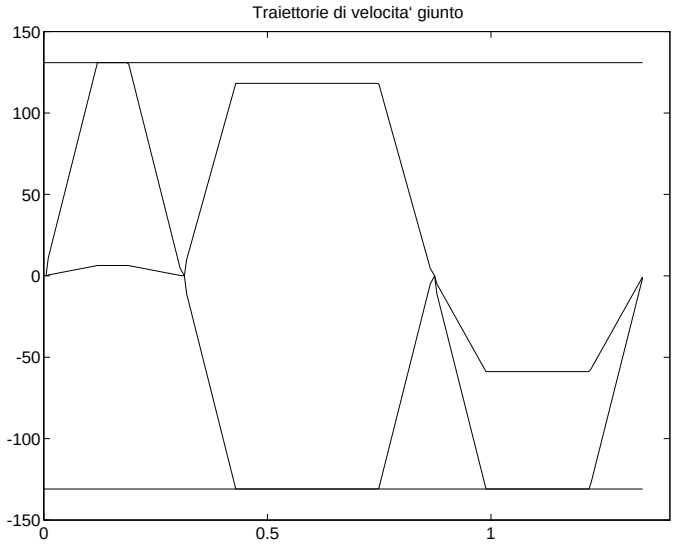
If more joints have to be co-ordinated with the same constraints on the maximum acceleration and velocity, the joint with the largest displacement must be individuated. For this joint, the maximum value $\ddot{q}_{max}$ for the acceleration is assigned and then the corresponding values t_a and T are computed.

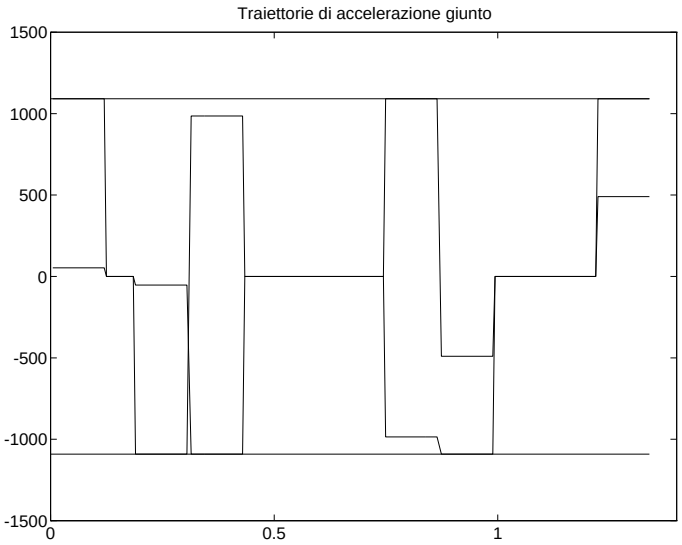
For the remaining joints, the acceleration and velocity values must be computed on the basis of these values of t_a and T, and on the basis of the given displacement L_i:

$$\ddot{q}_i = \frac{L_i}{t_a(T - t_a)}, \qquad \dot{q}_i = \frac{L_i}{T - t_a}$$

Trapezoidal trajectories

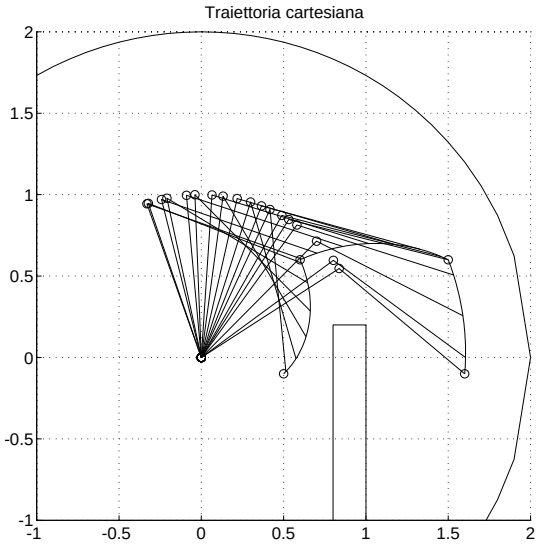


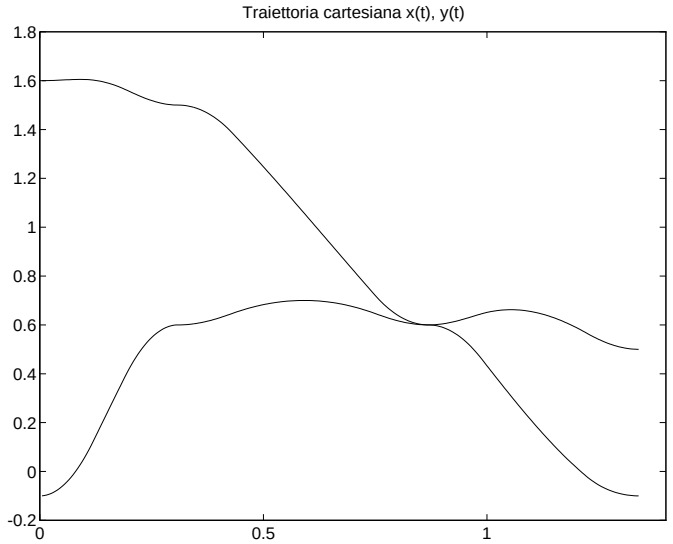


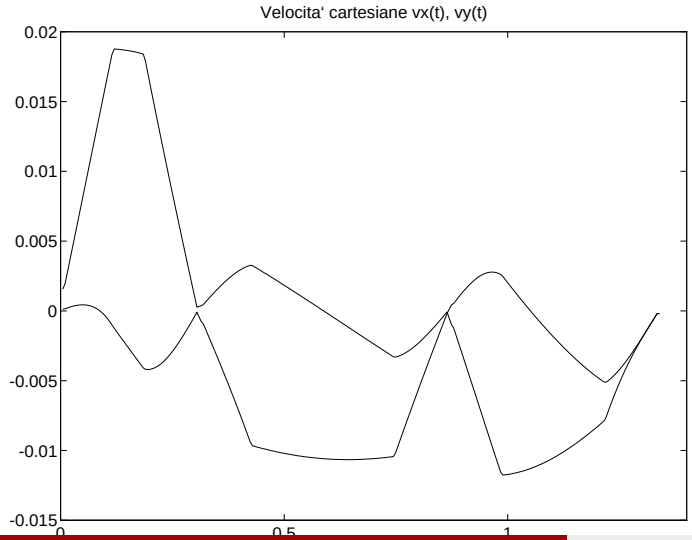


Trapezoidal trajectories

The trajectories in the workspace are:





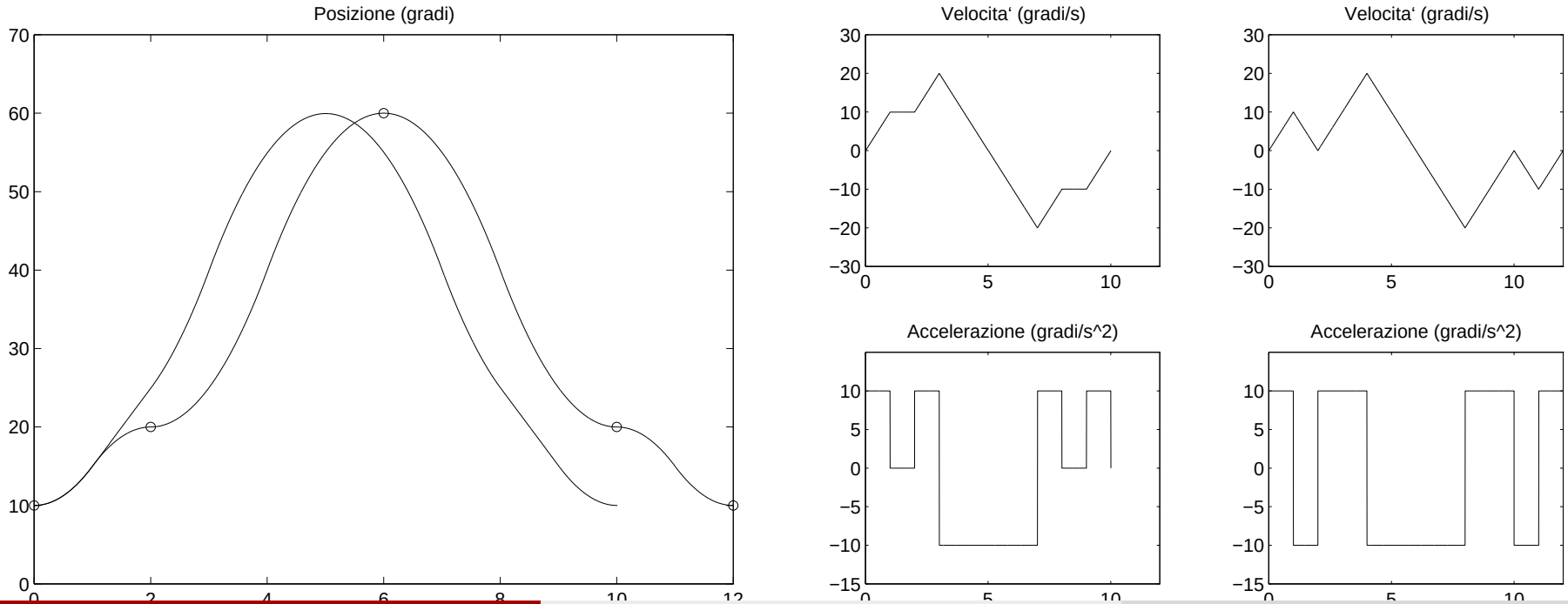


Trapezoidal trajectories

If a trajectory interpolating more consecutive points is computed with the above technique, a motion with null velocities in the via-points is obtained.

Since this behavior may be undesirable, it is possible to "anticipate" the actuation of a tract of the trajectory between points q_k and q_{k+q} *before* the motion from q_{k-1} to q_k is terminated. This is possible by adding (starting at an instant $t_k - t'_a$) the velocity and acceleration contributions of the two segments $[q_{k-1} - q_k]$ and $[q_k - q_{k+1}]$.

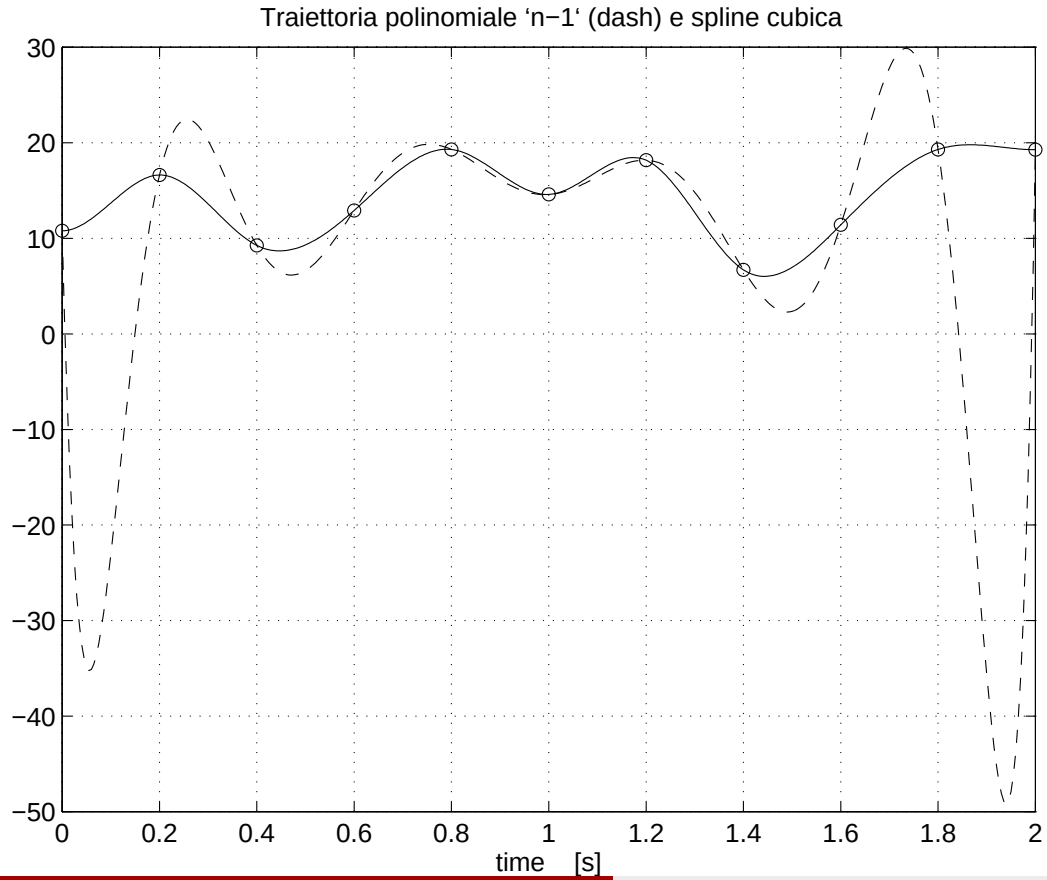
Obviously, another possibility is to compute the parameters of the functions defining the trapezoidal trajectory in order to have desired boundary conditions (i.e. velocities) for each segment.



Spline

In general, the problem of defining a function interpolating a set of n points can be solved with a polynomial function of degree $n-1$.

In planning a trajectory, this approach does not give good results since the resulting motions in general present large oscillations.



In general, given:

2 points $\Longrightarrow$ unique line

3 points $\Longrightarrow$ unique quadric

...

n points $\Longrightarrow$ unique polynomial with degree $n-1$

Spline

The (unique) polynomial $p(x)$ with degree $n-1$ interpolating n points (x_i, y_i) can be computed by the *Lagrange* expression:

$$p(x) = \frac{(x-x_2)(x-x_3)\cdots(x-x_n)}{(x_1-x_2)(x_1-x_3)\cdots(x_1-x_n)}y_1 + \frac{(x-x_1)(x-x_3)\cdots(x-x_n)}{(x_2-x_1)(x_2-x_3)\cdots(x_2-x_n)}y_2 + \cdots + \\ + \cdots + \frac{(x-x_1)(x-x_2)\cdots(x-x_{n-1})}{(x_n-x_1)(x_n-x_2)\cdots(x_n-x_{n-1})}y_n$$

Other (recursive) expressions have been defined, more efficient from a computational point of view (*Neville formulation*).

Spline

Another (less efficient) approach for the computation of the coefficients of the polynomial $p(x)$ is based on the following procedure:

$$y_i = p(x_i) = a_{n-1}x_i^{n-1} + \cdots + a_1 x_i + a_0 \qquad i = 1, \ldots, n$$

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix} = \begin{bmatrix} x_1^{n-1} & x_1^{n-2} & \cdots & x_1 & 1 \\ x_2^{n-1} & x_2^{n-2} & \cdots & x_2 & 1 \\ & & \vdots & & \\ x_{n-1}^{n-1} & x_{n-1}^{n-2} & \cdots & x_{n-1} & 1 \\ x_n^{n-1} & x_n^{n-2} & \cdots & x_n & 1 \end{bmatrix} \begin{bmatrix} a_{n-1} \\ a_{n-2} \\ \vdots \\ a_1 \\ a_0 \end{bmatrix} = \mathbf{X}\mathbf{a}$$

and then, by inverting matrix $\mathbf{X}$, the parameters are obtained

$$\mathbf{a} = \mathbf{X}^{-1}\mathbf{y}$$

$\Longrightarrow$ **Numerical problems in computing $\mathbf{X}^{-1}$ for high values of n!!!**

Spline

Given n points, in order to avoid the problem of high 'oscillations' (and also of the numerical precision):

NO: one polynomial of degree $n-1$

$\Longrightarrow$ **YES: $n-1$ polynomials with lower degree p $(p < n-1)$: each polynomial interpolates a segment of the trajectory.**

Usually, the degree p of the $n-1$ polynomials is chosen so that continuity of the velocity and acceleration profile is achieved. In this case, the choice $p = 3$ is made (cubic polynomials):

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

There are 4 coefficients for each polynomial, and therefore it is necessary to compute $4(n-1)$ coefficients.

Obviously, it is possible to choose higher values for p (e.g. $p = 5, 7, \ldots$).

Spline

$4(n-1)$ coefficients

On the other hand, there are:

- $2(n-1)$ conditions on the position (each cubic function interpolates the initial/final points);
- $n-2$ conditions on the continuity of velocity in the intermediate points;
- $n-2$ conditions on the continuity of acceleration in the intermediate points.

Therefore, there are

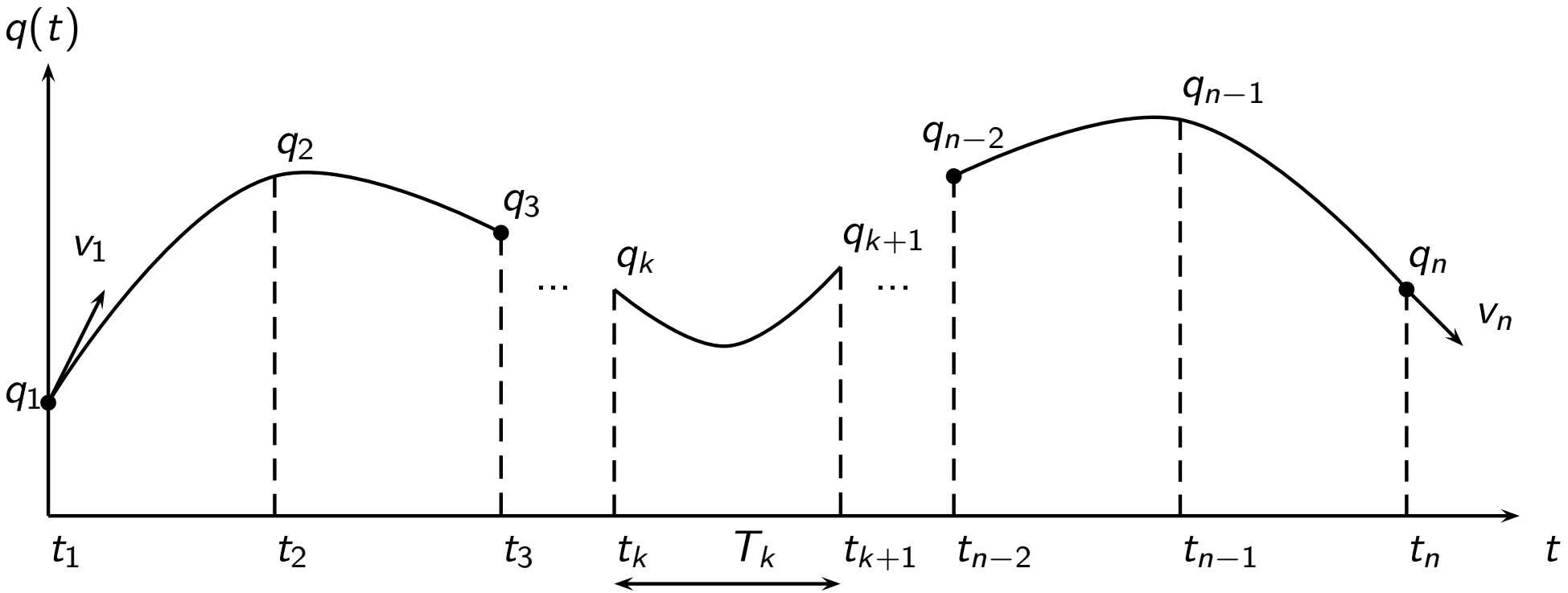
$$4(n-1) - 2(n-1) - 2(n-2) = 2$$

degrees of freedom left, that can be used *for example* for imposing proper conditions on the initial and final velocity.

Spline

The function obtained in this manner is a *spline*.

Among all the interpolating functions of n points with the same degree of continuity of derivation, the spline has the smallest *curvature*.



Spline

Mathematically, it is necessary to compute a function

$$\begin{cases} q(t) = \{q_k(t), \quad t \in [t_k,\ t_{k+1}], \quad k = 1, \ldots, n-1\} \\ q_k(\tau) = a_{k0} + a_{k1}\tau + a_{k2}\tau^2 + a_{k3}\tau^3, \quad \tau \in [0, T_k], \\ \qquad\qquad\qquad\qquad\qquad\qquad (\tau = t - t_k, \quad T_k = t_{k+1} - t_k) \end{cases}$$

with the conditions

$$q_k(0) = q_k, \quad q_k(T_k) = q_{k+1} \qquad k = 1, \ldots, n-1$$

$$\dot{q}_k(T_k) = \dot{q}_{k+1}(0) = v_k \qquad k = 1, \ldots, n-2$$

$$\ddot{q}_k(T_k) = \ddot{q}_{k+1}(0) \qquad k = 1, \ldots, n-2$$

Spline - Computation

The parameters a_{ki} are computed according to the following algorithm.

Let assume that the velocities $v_k,\ k = 2, \ldots, n-1$ in the intermediate points are known.

In this case, we can impose for each cubic polynomial the four boundary conditions on position and velocity:

$$\begin{cases} q_k(0) &= a_{k0} &= q_k \\ \dot{q}_k(0) &= a_{k1} &= v_k \\ q_k(T_k) &= a_{k0} + a_{k1}\tau + a_{k2}\tau^2 + a_{k3}\tau^3 &= q_{k+1} \\ \dot{q}_k(T_k) &= a_{k1} + 2a_{k2}\tau + 3a_{k3}\tau^2 &= v_{k+1} \end{cases}$$

and then

$$\begin{cases} a_{k0} &= q_k \\ a_{k1} &= v_k \\ a_{k2} &= \dfrac{1}{T_k}\left[\dfrac{3(q_{k+1}-q_k)}{T_k} - 2v_k - v_{k+1}\right] \\ a_{k3} &= \dfrac{1}{T_k^2}\left[\dfrac{2(q_k-q_{k+1})}{T_k} + v_k + v_{k+1}\right] \end{cases}$$

... but the velocities v_k are not known...

Spline - Computation

By using the conditions on continuity of the accelerations in the intermediate points, one obtains

$$\ddot{q}_k(T_k) = 2a_{k2} + 6a_{k3}\,T_k = 2a_{k+1,2} = \ddot{q}_{k+1}(0) \qquad k = 1, \ldots, n-2$$

form which, by substituting the expressions of $a_{k2}, a_{k3}, a_{k+1,2}$ and multiplying by $(T_k\ T_{k+1})/2$, one obtains

$$T_{k+1}v_k + 2(T_{k+1} + T_k)v_{k+1} + T_k v_{k+2} = \frac{3}{T_k\,T_{k+1}}\left[T_k^2(q_{k+2} - q_{k+1}) + T_{k+1}^2(q_{k+1} - q_k)\right]$$

These equations may be written in matrix form as

$$\begin{bmatrix} T_2 & 2(T_1+T_2) & T_1 & & & & & \\ 0 & T_3 & 2(T_2+T_3) & T_2 & & & & \\ & & & \ddots & & & & \\ & & & & T_{n-2} & 2(T_{n-3}+T_{n-2}) & T_{n-3} & 0 \\ & & & & & T_{n-1} & 2(T_{n-2}+T_{n-1}) & T_{n-2} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_{n-1} \\ v_n \end{bmatrix} = \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_{n-1} \\ c_n \end{bmatrix}$$

where the c_k are (known) constant terms depending on the intermediate positions and the duration of each segments.

Spline - Computation

Since the velocities v_1 and v_n are known, the corresponding columns can be eliminated form the left-hand side matrix, and then

$$\begin{bmatrix} 2(T_1+T_2) & T_1 & & & \\ T_3 & 2(T_2+T_3) & T_2 & & \\ & \ddots & \ddots & \ddots & \\ & & T_{n-2} & 2(T_{n-3}+T_{n-2}) & T_{n-3} \\ & & & T_{n-1} & 2(T_{n-2}+T_{n-1}) \end{bmatrix} \begin{bmatrix} v_2 \\ \vdots \\ v_{n-1} \end{bmatrix} =$$

$$\begin{bmatrix} \frac{3}{T_1 T_2}\left[T_1^2(q_3-q_2)+T_2^2(q_2-q_1)\right] - \mathbf{T_2 v_1} \\ \frac{3}{T_2 T_3}\left[T_2^2(q_4-q_3)+T_3^2(q_3-q_2)\right] \\ \vdots \\ \frac{3}{T_{n-3} T_{n-2}}\left[T_{n-3}^2(q_{n-1}-q_{n-2})+T_{n-2}^2(q_{n-2}-q_{n-3})\right] \\ \frac{3}{T_{n-2} T_{n-1}}\left[T_{n-2}^2(q_n-q_{n-1})+T_{n-1}^2(q_n-q_{n-2})\right] - \mathbf{T_{n-2} v_n} \end{bmatrix}$$

that is $$\mathbf{A(T)\,v = c(T, q}, v_1, v_n)$$

or $$\mathbf{Av = c}$$

Spline - Computation

- The matrix $\mathbf{A}$ is tridiagonal, and is always invertible if $T_k > 0$ ($|a_{kk}| > \sum_{j \neq k} |a_{kj}|$).
- Being $\mathbf{A}$ tridiagonal, its inverse is computed by efficient numerical algorithms (based on the Gauss-Jordan method).
- Once $\mathbf{A}^{-1}$ is known, the velocities $v_2, \ldots, v_{n-1}$ are computed as

$$\mathbf{v} = \mathbf{A}^{-1}\mathbf{c}$$

and the problem is solved.

Spline

The total duration of a spline is

$$T = \sum_{k=1}^{n-1} T_k = t_n - t_1$$

It is possible to define an optimality problem aiming at minimizing T. The values of T_k must be computed so that T is minimized and the constraints on the velocity and acceleration are satisfied.

Formally the problem is formulated as

$$\begin{cases} \min_{T_k} \quad T = \sum_{k=1}^{n-1} T_k \\ \\ \text{tale che} \quad \begin{aligned} |\dot{q}(\tau, T_k)| &< v_{max} \qquad \tau \in [0\, T] \\ |\ddot{q}(\tau, T_k)| &< a_{max} \qquad \tau \in [0\, T] \end{aligned} \end{cases}$$

Non linear optimization problem with linear objective function, solvable with classical techniques from the operational research field.

Spline - Example

A spline trough the points $q_1 = 0$, $q_2 = 2$, $q_3 = 12$, $q_4 = 5$ must be defined, minimizing the total duration T and with the constraints: $v_{max} = 3$, $a_{max} = 2$.

The non linear optimization problem

$$min \quad T = T_1 + T_2 + T_3$$

is defined, with the constraints reported in the following slide.

By solving this problem (e.g. with the Matlab *Optimization Toolbox*) the following values are obtained:

$$T_1 = 1.5549, \qquad T_2 = 4.4451, \qquad T_3 = 4.5826, \qquad \Rightarrow \qquad T = 10.5826 \; sec$$

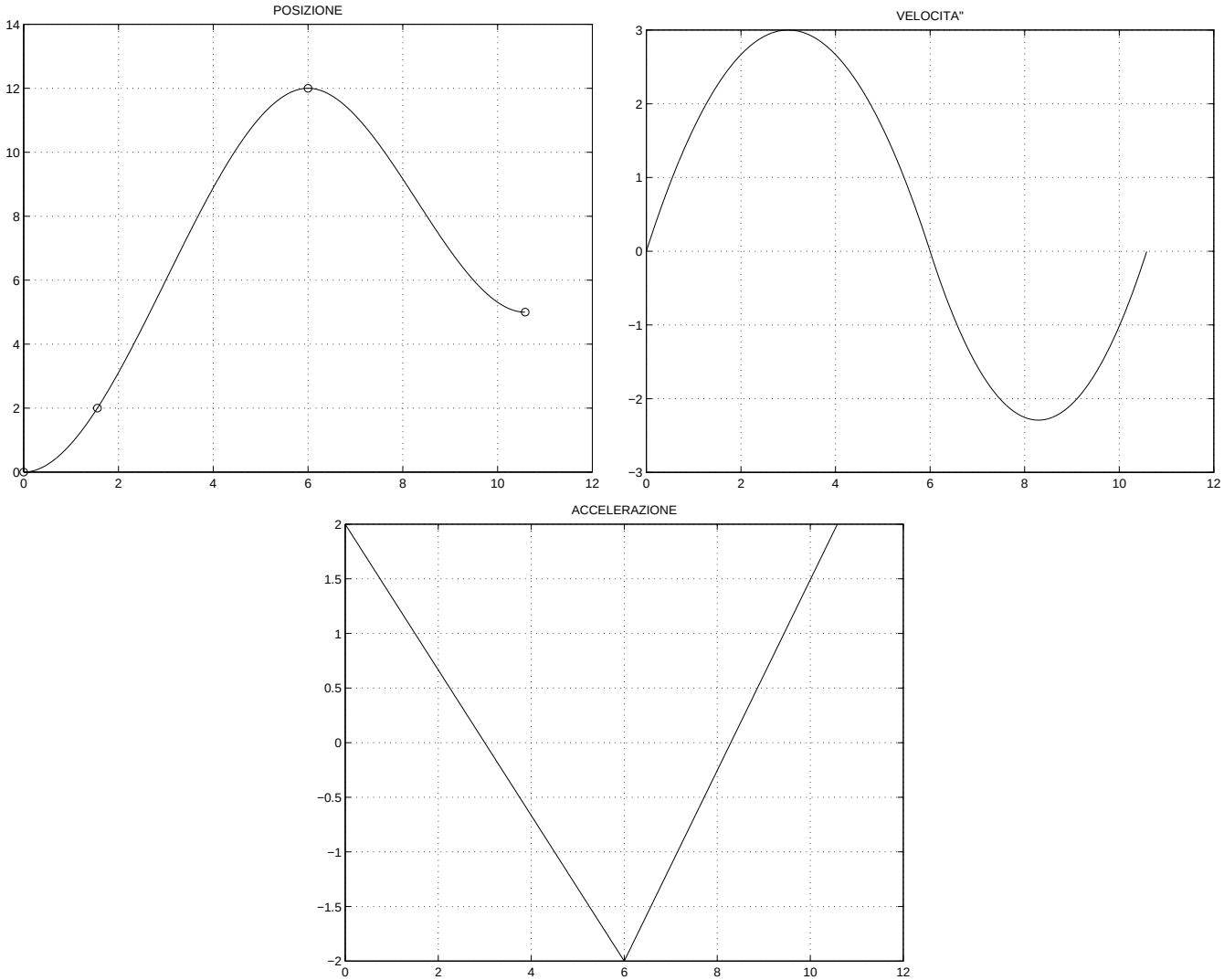
Spline - Esempio

Constraints on the optimization problem:

$$
\begin{cases}
a_{11} & & & & & & \leq & v_{max} & \text{(velocità iniziale del I tratto} \leq v_{max}) \\
a_{21} & & & & & & \leq & v_{max} & \text{(velocità iniziale del II tratto} \leq v_{max}) \\
a_{31} & & & & & & \leq & v_{max} & \text{(velocità iniziale del III tratto} \leq v_{max}) \\
a_{11} & + & 2a_{12}T_1 & + & 3a_{13}T_1^2 & & \leq & v_{max} & \text{(velocità finale del I tratto} \leq v_{max}) \\
a_{21} & + & 2a_{22}T_2 & + & 3a_{23}T_2^2 & & \leq & v_{max} & \text{(velocità finale del II tratto} \leq v_{max}) \\
a_{31} & + & 2a_{32}T_3 & + & 3a_{33}T_3^2 & & \leq & v_{max} & \text{(velocità finale del III tratto} \leq v_{max}) \\
a_{11} & + & 2a_{12}\left(-\frac{a_{12}}{3a_{13}}\right) & + & 3a_{13}\left(-\frac{a_{12}}{3a_{13}}\right)^2 & & \leq & v_{max} & \text{(velocità all'interno del I tratto} \leq v_{max}) \\
a_{21} & + & 2a_{22}\left(-\frac{a_{22}}{3a_{23}}\right) & + & 3a_{23}\left(-\frac{a_{22}}{3a_{23}}\right)^2 & & \leq & v_{max} & \text{(velocità all'interno del II tratto} \leq v_{max}) \\
a_{31} & + & 2a_{32}\left(-\frac{a_{32}}{3a_{33}}\right) & + & 3a_{33}\left(-\frac{a_{32}}{3a_{33}}\right)^2 & & \leq & v_{max} & \text{(velocità all'interno del III tratto} \leq v_{max}) \\
2a_{12} & & & & & & \leq & a_{max} & \text{(accelerazione iniziale del I tratto} \leq a_{max}) \\
2a_{22} & & & & & & \leq & a_{max} & \text{(accelerazione iniziale del II tratto} \leq a_{max}) \\
2a_{32} & & & & & & \leq & a_{max} & \text{(accelerazione iniziale del III tratto} \leq a_{max}) \\
2a_{12} & & & + & 6a_{13}T_1 & & \leq & a_{max} & \text{(accelerazione finale del I tratto} \leq a_{max}) \\
2a_{22} & & & + & 6a_{23}T_2 & & \leq & a_{max} & \text{(accelerazione finale del II tratto} \leq a_{max}) \\
2a_{32} & & & + & 6a_{33}T_3 & & \leq & a_{max} & \text{(accelerazione finale del III tratto} \leq a_{max})
\end{cases}
$$

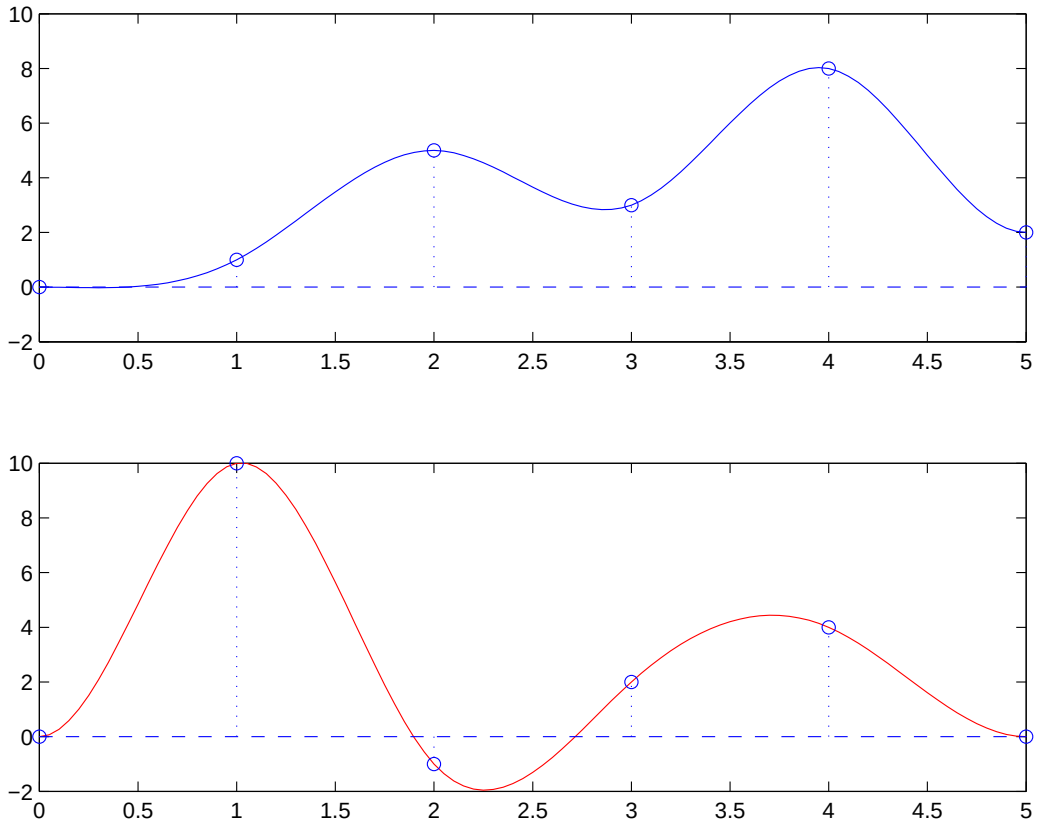
Spline - Esempio

Position, velocity and acceleration profiles of the optimal trajectory.



Spline

The above procedure for computing the spline is adopted also for more motion axes (joints). Notice that the matrix $\mathbf{A}_j(\mathbf{T}) = \mathbf{A}(\mathbf{T})$ is the same for all the joints (depends only on the parameters T_k), while the vector $\mathbf{c}(\mathbf{T}, \mathbf{q}_j, v_{i1}, v_{in})$ depends on the specific i-th joint.



Spline

From the expressions of matrix $\mathbf{A}$ and the vector $\mathbf{c}$ ($\mathbf{Av} = \mathbf{c}$)

$$\mathbf{A} = \begin{bmatrix} 2(T_1+T_2) & T_1 & & & \\ T_3 & 2(T_2+T_3) & T_2 & & \\ & & \ddots & & \\ & & T_{n-2} & 2(T_{n-3}+T_{n-2}) & T_{n-3} \\ & & & T_{n-1} & 2(T_{n-2}+T_{n-1}) \end{bmatrix}$$

$$\mathbf{c} = \begin{bmatrix} \frac{3}{T_1 T_2}\left[T_1^2(q_3-q_2)+T_2^2(q_2-q_1)\right] - T_2 v_1 \\ \frac{3}{T_2 T_3}\left[T_2^2(q_4-q_3)+T_3^2(q_3-q_2)\right] \\ \vdots \\ \frac{3}{T_{n-3} T_{n-2}}\left[T_{n-3}^2(q_{n-1}-q_{n-2})+T_{n-2}^2(q_{n-2}-q_{n-3})\right] \\ \frac{3}{T_{n-2} T_{n-1}}\left[T_{n-2}^2(q_n-q_{n-1})+T_{n-1}^2(q_n-q_{n-2})\right] - T_{n-2} v_n \end{bmatrix}$$

Spline

If

- the duration T_k of each interval is multiplied by a constant λ (linear scaling)
- the initial and final velocities are null

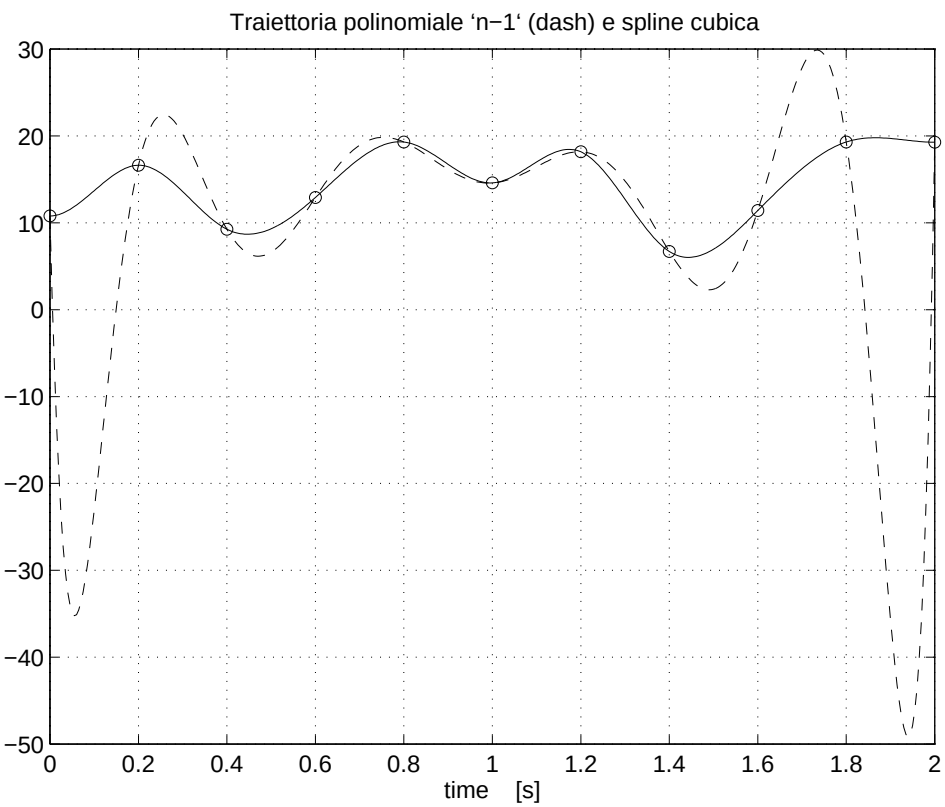
one obtains that the new duration T', the velocities and accelerations of the new trajectory are:

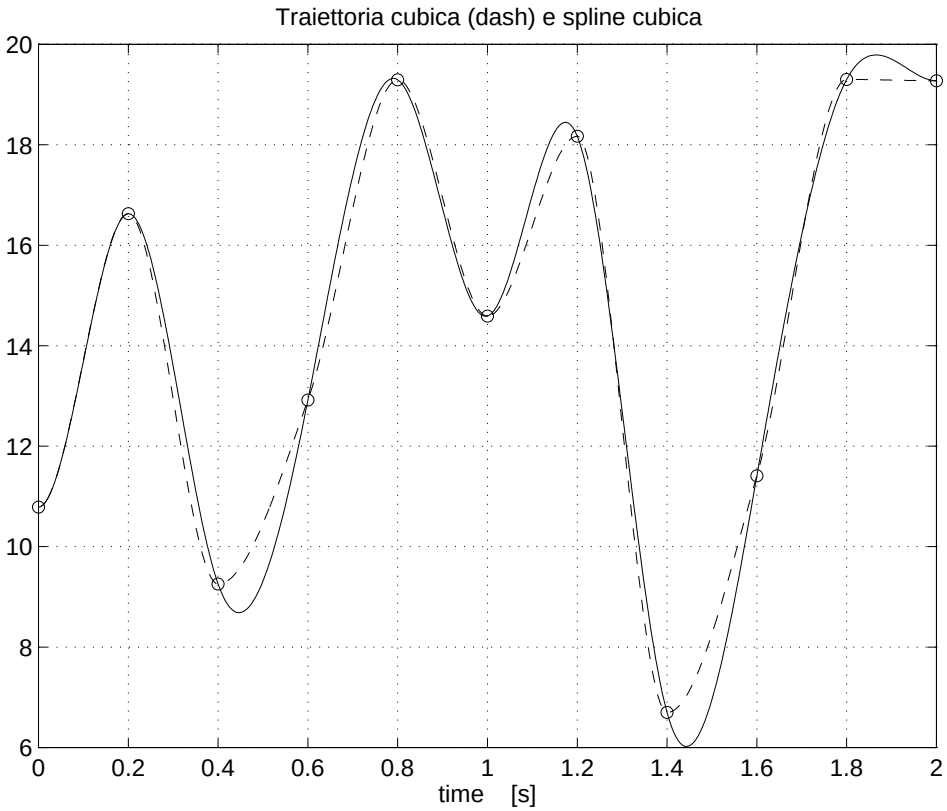
$$
\begin{aligned}
T' &= = \lambda T \\
v'_k &= \frac{1}{\lambda} v_k \\
a_k &= \frac{1}{\lambda^2} a_k
\end{aligned}
$$

Spline - Example

Comparison of a $n-1$ polynomial, a spline, and a composition of cubic polynomials.

11 points, $v_{in} = v_{fin} = 0 \;\; m/s$





Spline - Example