INFO0948
Time and Motion

Bernard Boigelot
boigelot@montefiore.ulg.ac.be

February 16th, 2017

These slides are partly based on Chapters 3 and 4 of the book *Robotics, Vision and Control: Fundamental Algorithms in MATLAB* by Peter Corke, published by Springer in 2011, on course material prepared by Renaud Detry in 2016, and on Stéphane Lens's PhD thesis (ULg, 2015).

Basic notions

- Pose: the position and orientation ξ of an object.
- Path: a varying pose $\xi(s)$, for some parameter $s \in [s_0, s_T]$.
- Trajectory: a path with specified timing: $\xi(s(t))$, for $t \in [0, T]$.

Basic notions

- Pose: the position and orientation ξ of an object.
- Path: a varying pose $\xi(s)$, for some parameter $s \in [s_0, s_T]$.
- Trajectory: a path with specified timing: $\xi(s(t))$, for $t \in [0, T]$.

Note: The notions of path and trajectory can also be generalized to motion in the *configuration space* or *joint space* of a robot.

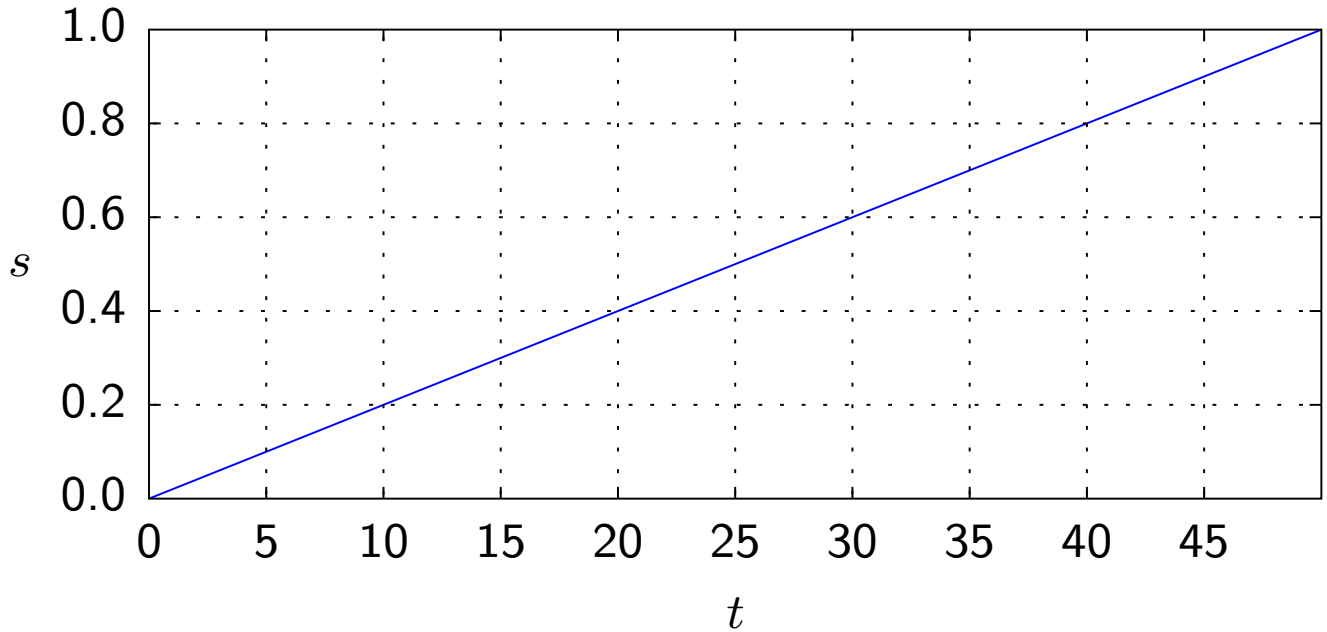
Generating trajectories: 1D case

Illustration:

- The path is imposed by the geometry of the track.
- At $t = 0$, the train must leave the departure station: $s(0) = s_0$.
- At $t = T$, it must reach the destination station: $s(T) = s_T$.

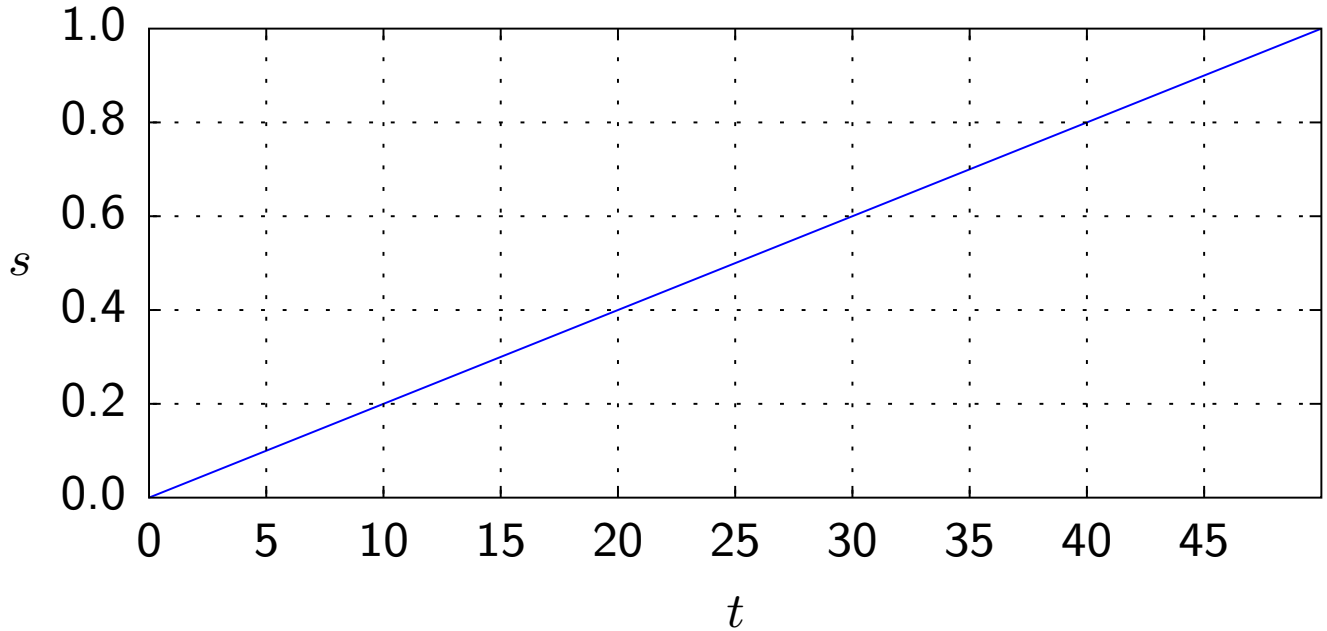
Solution 1: Linear interpolation

$$s(t) = \left(1 - \frac{t}{T}\right)s_0 + \frac{t}{T}s_T.$$



Solution 1: Linear interpolation

$$s(t) = \left(1 - \frac{t}{T}\right)s_0 + \frac{t}{T}s_T.$$



Problem: Not physically feasible, since the train would experience infinite acceleration or deceleration at $t = 0$ and $t = T$.

Solution 2: Polynomial interpolation

Requirements:

- at $t = 0$: $s(0) = s_0$, $\dot{s}(0) = \dot{s}_0$, $\ddot{s}(0) = \ddot{s}_0$.
- at $t = T$: $s(T) = s_T$, $\dot{s}(T) = \dot{s}_T$, $\ddot{s}(T) = \ddot{s}_T$.
- the trajectory must be smooth: the first few derivatives of $s(t)$ have to be continuous.

Solution 2: Polynomial interpolation

Requirements:

- at $t = 0$: $s(0) = s_0$, $\dot{s}(0) = \dot{s}_0$, $\ddot{s}(0) = \ddot{s}_0$.
- at $t = T$: $s(T) = s_T$, $\dot{s}(T) = \dot{s}_T$, $\ddot{s}(T) = \ddot{s}_T$.
- the trajectory must be smooth: the first few derivatives of $s(t)$ have to be continuous.

Solution: Since there are 6 constraints, use a 5-th order polynomial:

$$s(t) = At^5 + Bt^4 + Ct^3 + Dt^2 + Et + F$$

Solution 2: Polynomial interpolation

Requirements:

- at $t = 0$: $s(0) = s_0$, $\dot{s}(0) = \dot{s}_0$, $\ddot{s}(0) = \ddot{s}_0$.
- at $t = T$: $s(T) = s_T$, $\dot{s}(T) = \dot{s}_T$, $\ddot{s}(T) = \ddot{s}_T$.
- the trajectory must be smooth: the first few derivatives of $s(t)$ have to be continuous.

Solution: Since there are 6 constraints, use a 5-th order polynomial:

$$s(t) = At^5 + Bt^4 + Ct^3 + Dt^2 + Et + F$$

We thus have:

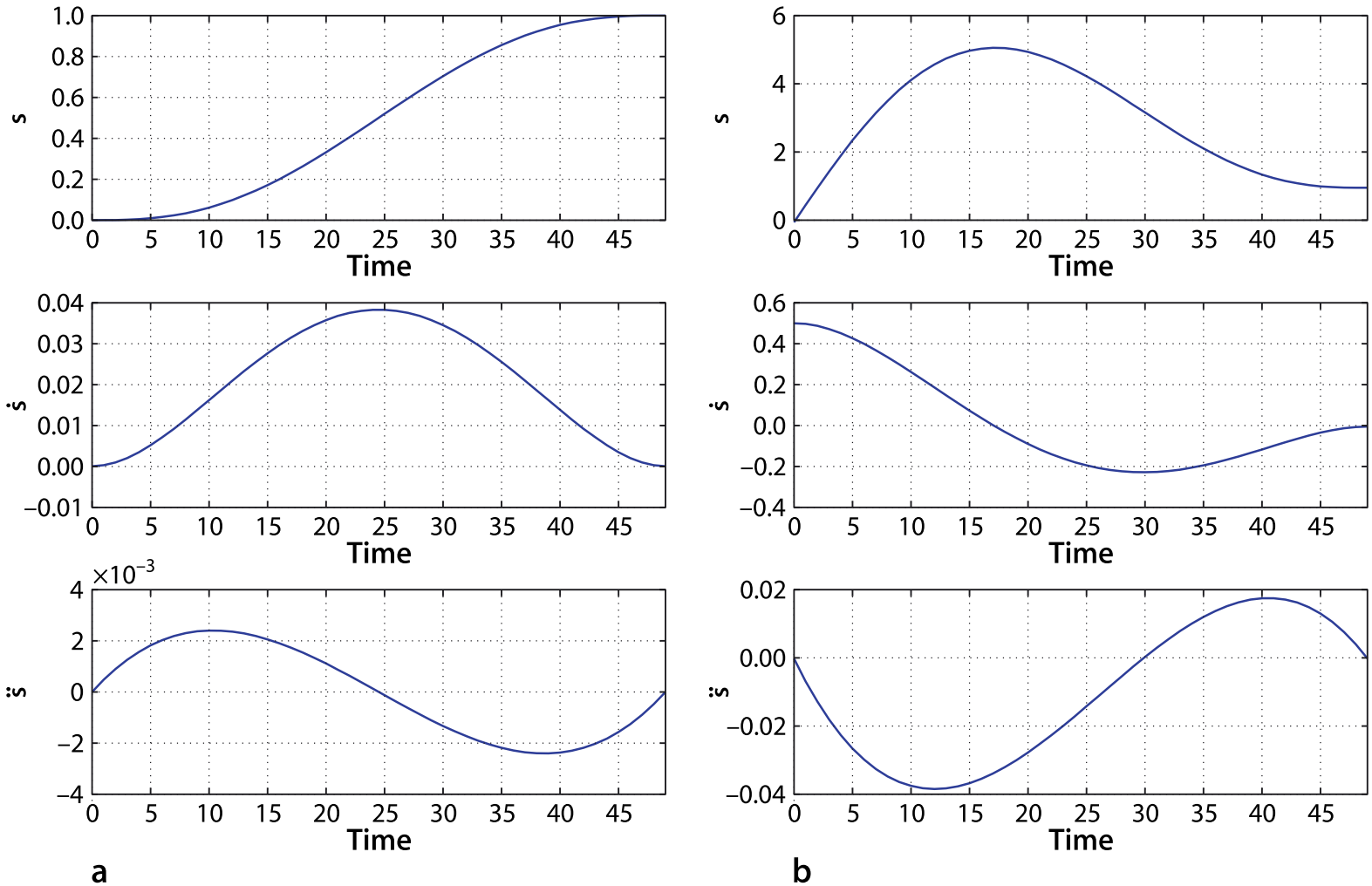
$$\begin{aligned} \dot{s}(t) &= 5At^4 + 4Bt^3 + 3Ct^2 + 2Dt + E \\ \ddot{s}(t) &= 20At^3 + 12Bt^2 + 6Ct + 2D \end{aligned}$$

The values of A, B, C, D, E and F can thus be obtained by solving the system:

$$\begin{bmatrix} s_0 \\ s_T \\ \dot{s}_0 \\ \dot{s}_T \\ \ddot{s}_0 \\ \ddot{s}_T \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ T^5 & T^4 & T^3 & T^2 & T & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 5T^4 & 4T^3 & 3T^2 & 2T & 1 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 20T^3 & 12T^2 & 6T & 2 & 0 & 0 \end{bmatrix} \begin{bmatrix} A \\ B \\ C \\ D \\ E \\ F \end{bmatrix}$$

(Toolbox function: `tpoly`.)

Examples: (**a**: $\dot{s}_0 = \dot{s}_T = 0$, **b**: $\dot{s}_0 = 0.5$ and $\dot{s}_T = 0$).



Drawbacks:

- The train may overshoot the destination (Example **b**).
- The maximum velocity is much higher than the average one (Example **a**).

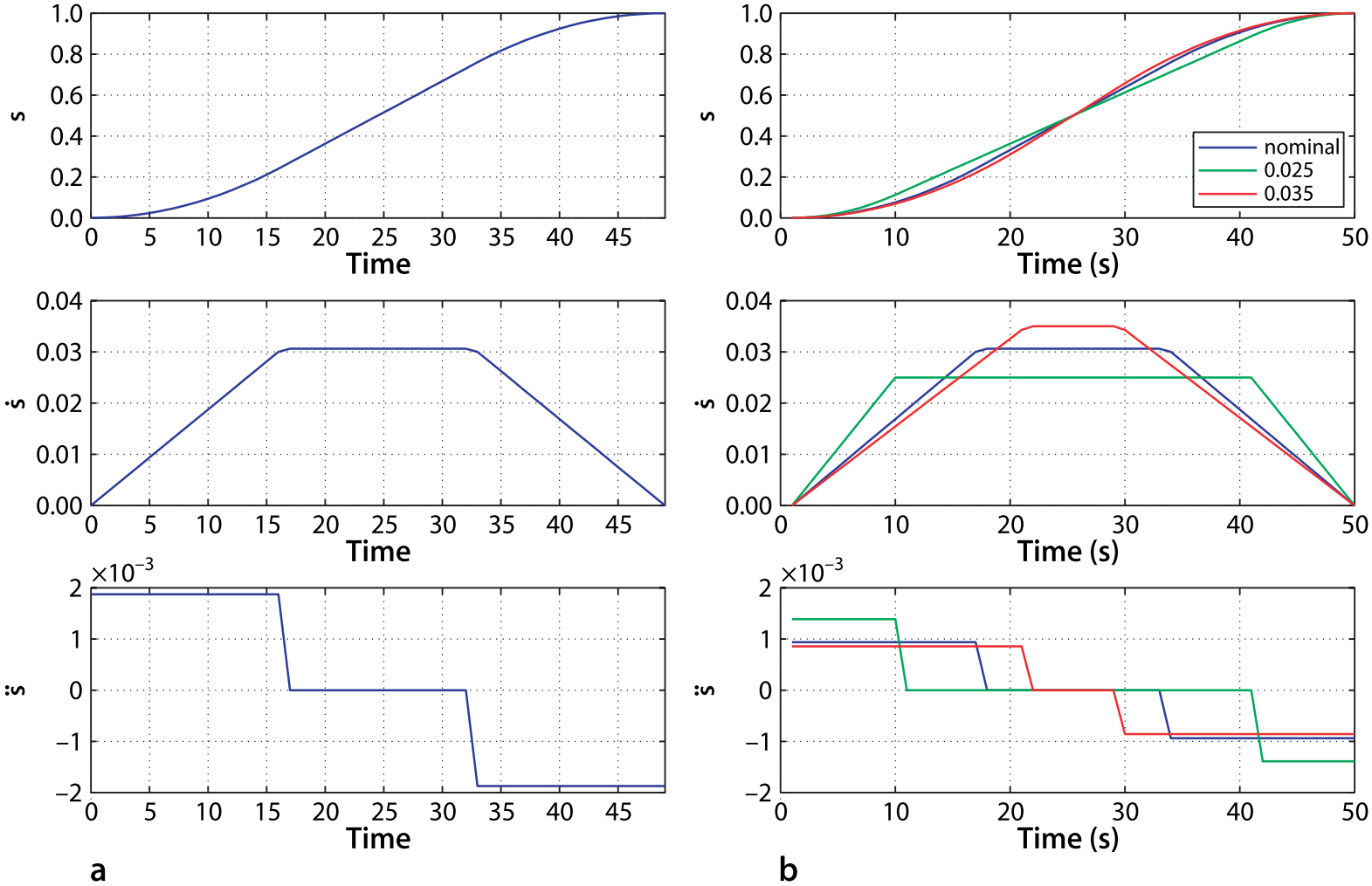
Solution 3: Line segments with parabolic blends

Principles:

- The trajectory is composed of
  - an initial segment at constant acceleration
  - a middle segment at constant velocity
  - a final segment at constant deceleration
- The velocity curve thus takes the form of a trapezoid.
- If the acceleration and deceleration constants are equal, then their value can be computed from the velocity at the middle segment.

(Toolbox function: `lsbp`.)

Examples:



Advantages:

- The solution is efficient.
- The generated trajectory is physically feasible.

Drawbacks:

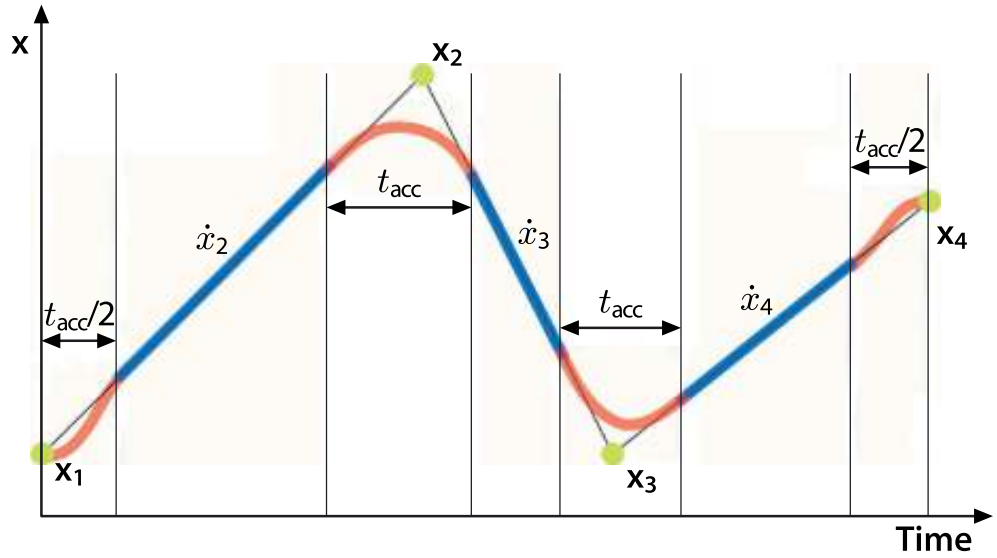
- Since the acceleration $\ddot{s}(t)$ is not continuous, a high level of *jerk* $\dddot{s}(t)$ is experienced at the boundaries between segments.
- One has to check that physical acceleration bounds are not exceeded.

Multi-segment trajectories

Illustration:

https://www.youtube.com/watch?v=JIGilyQ5I_o

Solution:



The trajectory is composed of:

- Constant-velocity segments (in blue).
- Transition segments of fixed duration t_{acc} around waypoints (in red).

Since each transition segment must satisfy six constraints (position, velocity and acceleration at each of its two endpoints), a good strategy is to interpolate them using a 5-th order polynomial.

(Toolbox function: `mstraj`, also for the n-dimensional case.)

Drawback: The value of t_{acc} has to be carefully chosen, so as to

- keep the path close to the waypoints, and
- respect acceleration bounds.

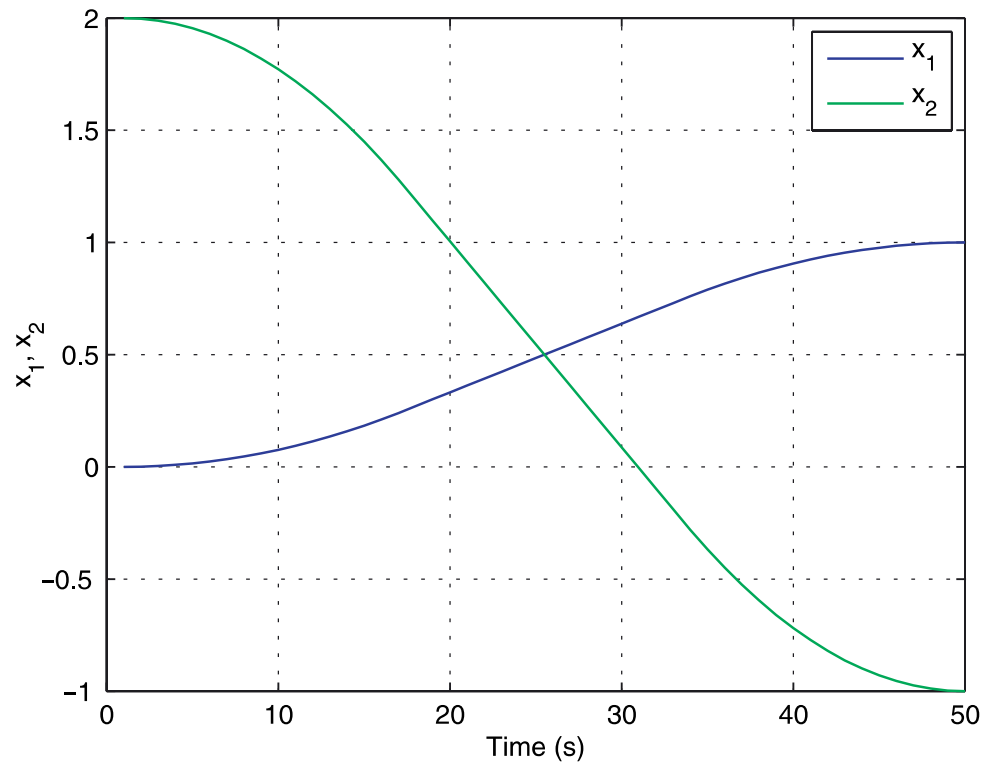
Generating trajectories: n-dimensional case

Illustration:

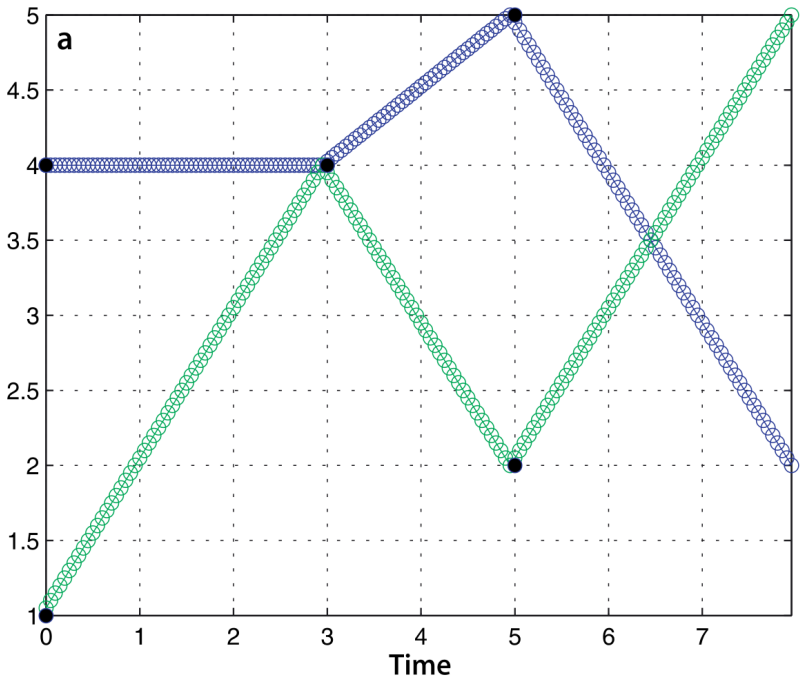
`https://www.youtube.com/watch?v=bxbjZiKAZP4`

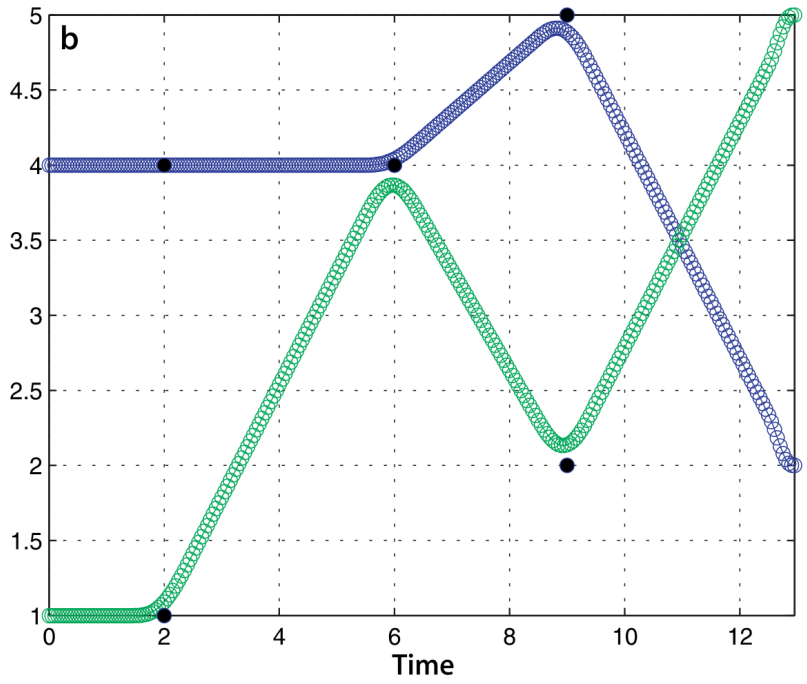
Solution: Interpolate separately each axis.

Example 1:



Example 2 (multi-segment):





Notes:

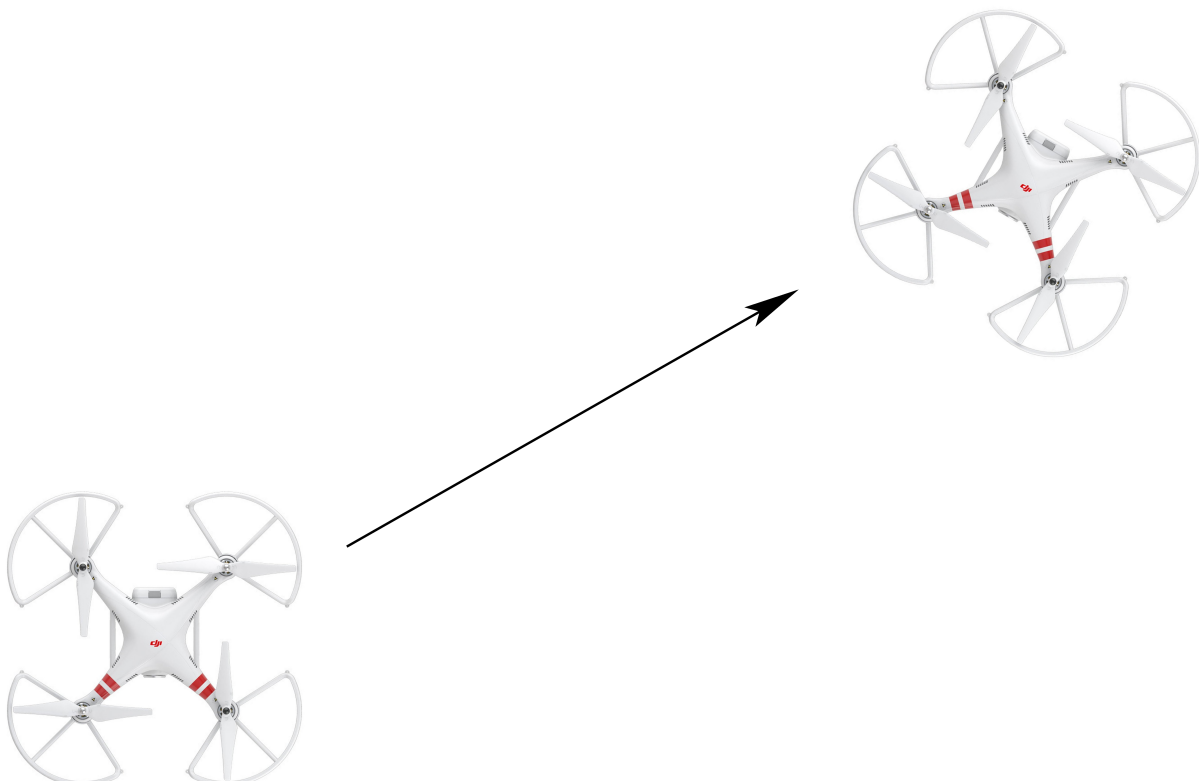
- Each axis usually has its own velocity and acceleration bounds.
- This means that the timing of a segment will usually be determined by the slowest axis, or the one that has to travel the longest distance.
- This interpolation strategy is the one mainly used in industrial robots.

Example: Albert (Eurobot)

`http://www.montefiore.ulg.ac.be/~boigelot/tunnel/albert.mov`

Cartesian motion

Problem statement: Interpolate smoothly between two given poses.



2D case: Interpolate separately

- the position (x, y) (e.g., straight path with lspb trajectory).
- the orientation θ (1D problem).

Note: The orientation difference should be kept within $[-\pi, \pi]$.

2D case: Interpolate separately

- the position (x, y) (e.g., straight path with lspb trajectory).
- the orientation θ (1D problem).

Note: The orientation difference should be kept within $[-\pi, \pi]$.

3D case: We can follow the same approach, but we need a method for interpolating 3D orientations.

Interpolation in $SO(3)$

First solution: Interpolate separately Euler or Tait-Bryan angles.

- OK for small rotations.
- Problematic near singularities.

Illustration: $(x,\, y,\, z) : (0°,\, 0°,\, 0°) \longrightarrow (180°,\, 180°,\, 90°)$.

`http://www.montefiore.ulg.ac.be/~boigelot/tunnel/ulg-1.avi`

Better solution: Interpolate quaternions.

`http://www.montefiore.ulg.ac.be/~boigelot/tunnel/ulg-2.avi`

Better solution: Interpolate quaternions.

`http://www.montefiore.ulg.ac.be/~boigelot/tunnel/ulg-2.avi`

Principles:

- Recall that unit quaternions represent 3D orientations (with $\mathring{q}$ and $-\mathring{q}$ being equivalent).
- Unit quaternions form a 3-sphere in 4D space.
- One can move from $\mathring{q}_0$ to $\mathring{q}_1$ by following the shortest path between them on this 3-sphere: *Spherical linear interpolation (Slerp)*.

Toolbox functions:

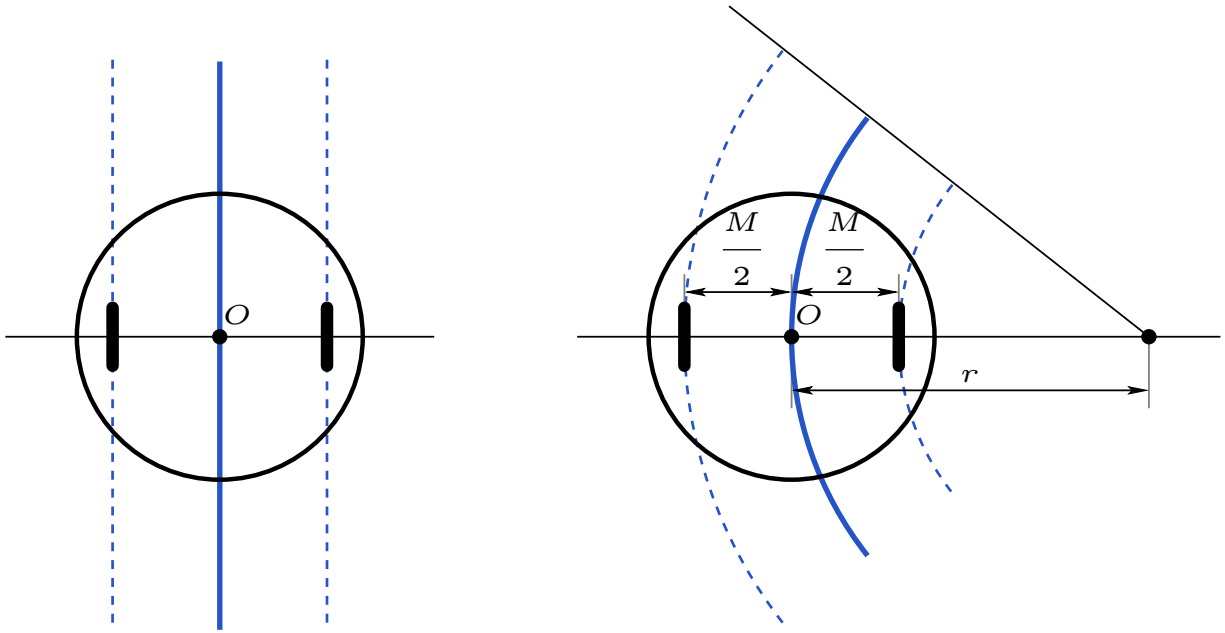
- `interp` for interpolating quaternions.
- `trinterp` for interpolating Cartesian motion.
- `ctraj` for combining `trinterp` with `lspb`.

Non-holonomic mobile robots

Some robotic drives are not able to achieve Cartesian motion.
Example: A car must keep at all times its orientation tangent to its travel path.

- A holonomic drive is sufficiently actuated to be able to follow any path in its configuration space.
- A non-holonomic drive has restricted mobility.

Differential drive



- The path followed by the robot can be described by the motion of a single reference point O in a 2D plane (velocity v, rotational velocity $\dot{\theta}$).
- Motion is entirely determined by the velocities $v_L(t)$ and $v_R(t)$ of the two wheels.

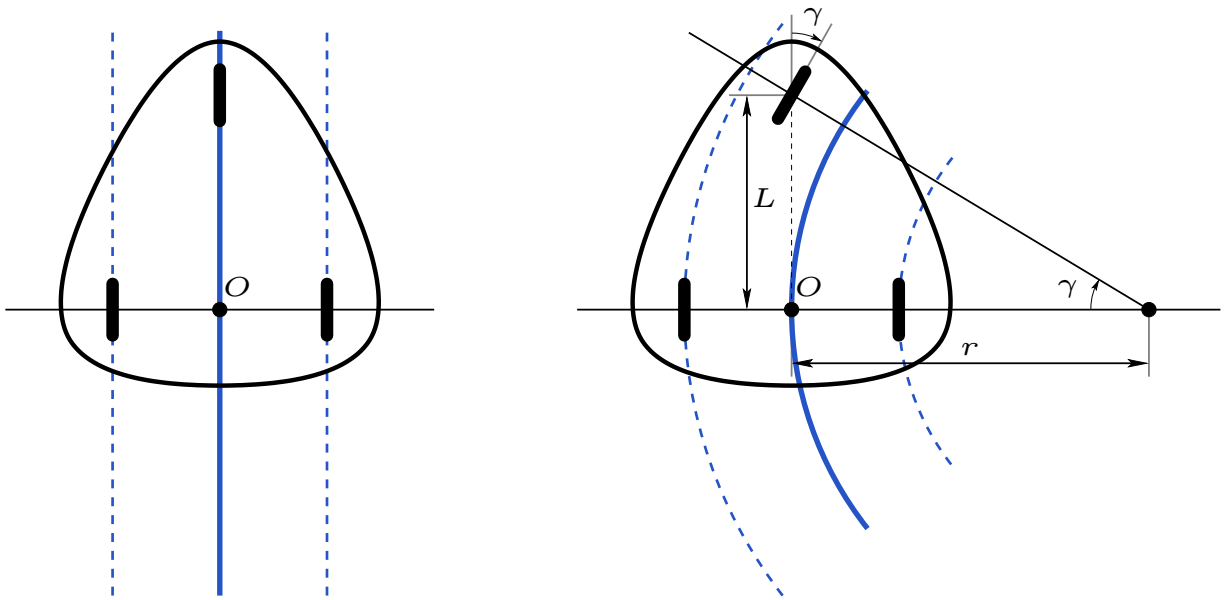
- If $v_L = v_R$, then $v = v_L = v_R$ and $\dot{\theta} = 0$.
- If $v_L \neq v_R$:
  - The radius r of the circle followed by O satisfies

$$v_L\left(r + \frac{M}{2}\right) = v_R\left(r - \frac{M}{2}\right)$$

  - We thus have

$$\begin{aligned} r &= \frac{M}{2} \cdot \frac{v_R + v_L}{v_R - v_L} \\ v &= \frac{v_R + v_L}{2} \\ \dot{\theta} &= \frac{v_R - v_L}{M} \end{aligned}$$

Tricycle platform



- The reference point O is located at the middle of the rear axle.
- Motion is entirely characterized by the velocity v_S of the steering wheel and the steering angle $\gamma \in [-\frac{\pi}{2}, \frac{\pi}{2}]$.

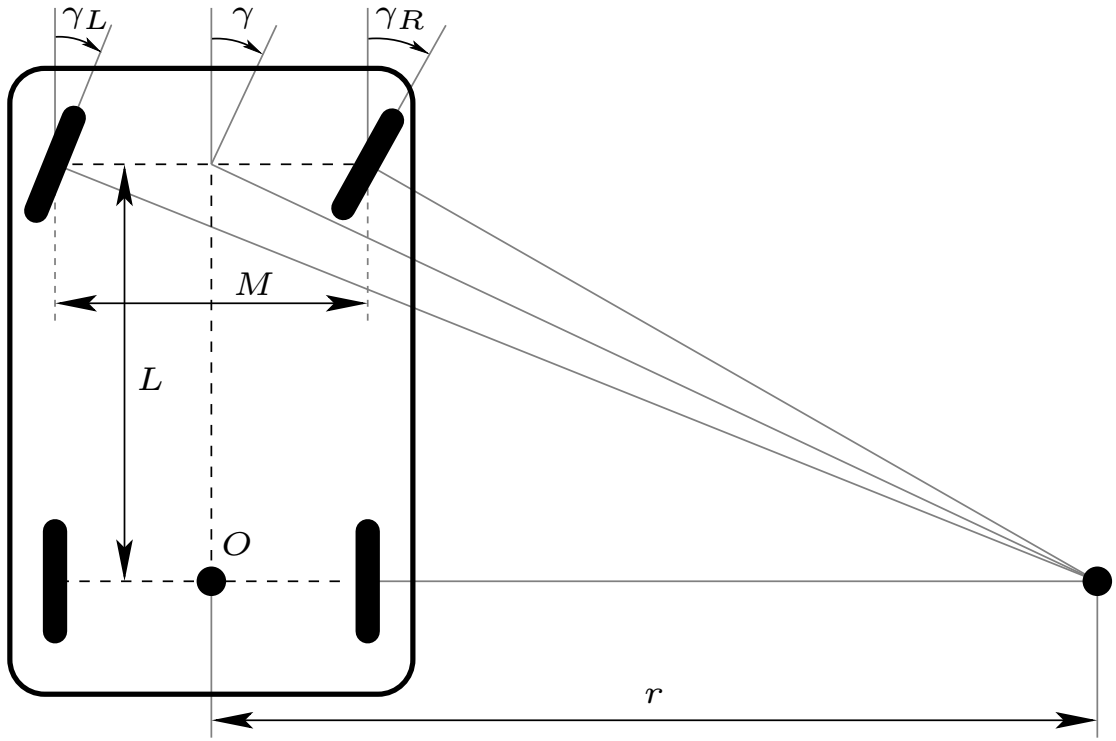
- If $\gamma = 0$, then $v = v_S$ and $\dot{\theta} = 0$.
- If $\gamma \neq 0$:
  - The radius r of the circle followed by O satisfies

$$r = L \cot \gamma$$

  - We thus have

$$\begin{aligned} v &= v_S \cos \gamma \\ \dot{\theta} &= \frac{v_S}{L} \sin \gamma \end{aligned}$$

Car-like robot



- The principles are similar to the tricycle platform.

- In order for the center of rotation to exist, the steering angles must satisfy

$$\tan \gamma_L = \frac{L}{r - \frac{M}{2}}$$

$$\tan \gamma_R = \frac{L}{r + \frac{M}{2}}$$

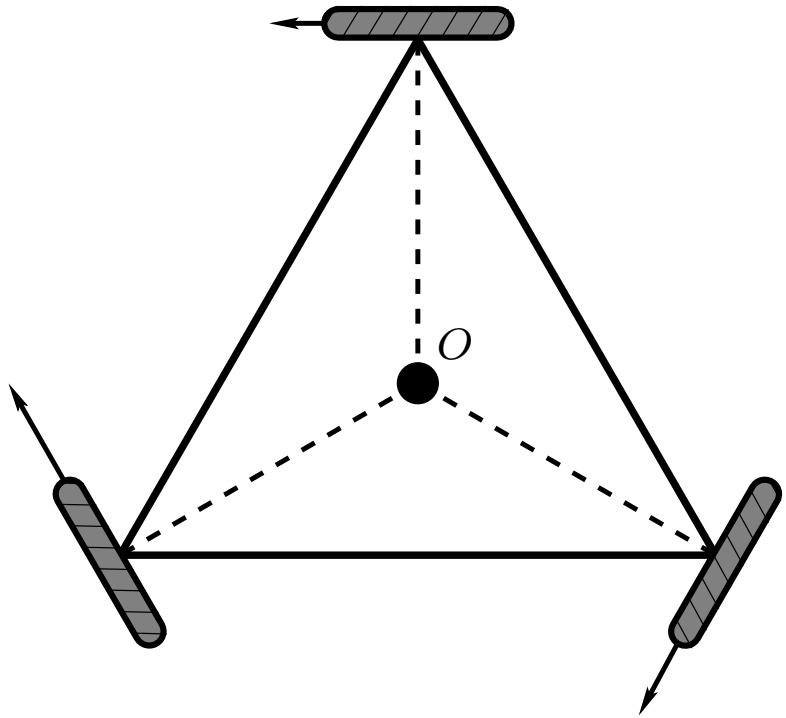
  (This is usually achieved by mechanical means.)

- An equivalent tricycle steering angle can be computed:

$$\tan \gamma = \frac{L}{r}.$$

Note: Unlike the tricyle drive, in-place turns ($v = 0$ and $\dot{\theta} \neq 0$) are usually impossible with this platform.

Holonomic mobile robots



By using *swedish wheels*, a robot drive can be made holonomic:

- Each wheel exerts a longitudinal reaction force with the ground, and remains free to slide sideways.
- Three wheels are thus potentially able to generate any (2D) global force and global torque at O.