Contractor Programming over Trajectories

Simon Rohou[1], Luc Jaulin[1], Lyudmila Mihaylova[2], Fabrice Le Bars[1], Sandor M. Veres[2]

[1] ENSTA Bretagne, Lab-STICC, UMR CNRS 6285, France
[2] University of Sheffield, Western Bank Sheffield S10 2TN, UK
simon.rohou@ensta-bretagne.org

Kickoff Contredo
March 2017

The University Of Sheffield.

Lab-STICC

LAGRANGIAN APPROACH

Contractor Programming over Trajectories

Simon Rohou[1], Luc Jaulin[1], Lyudmila Mihaylova[2], Fabrice Le Bars[1], Sandor M. Veres[2]

[1] ENSTA Bretagne, Lab-STICC, UMR CNRS 6285, France
[2] University of Sheffield, Western Bank Sheffield S10 2TN, UK
simon.rohou@ensta-bretagne.org

Kickoff Contredo
March 2017

The University Of Sheffield.

Lab-STICC

Section 1

Introduction

Introduction

Mobile Robotics: formalization

Robot localization = state estimation problem.
Classically, we have:

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) & (\text{navigation}) \\ y(t) = g(\mathbf{x}(t)) & (\text{measurements}) \end{cases}$$

Where:

- $\mathbf{x} \in \mathbb{R}^n$ is the state vector (position, bearing, ...)
- $\mathbf{u} \in \mathbb{R}^m$ is the input vector (command)
- $y \in \mathbb{R}$ is some exteroceptive measurement (*e.g.* bathymetry)
- $\mathbf{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is the *evolution* function
- $g : \mathbb{R}^n \to \mathbb{R}$ is the *observation* function

Introduction

Constraint Network based on trajectories

A problem involving constraints is classically presented with a **Constraint Network**:

- **Variables:**
- **Constraints:**
- **Domains:**

Introduction

Constraint Network based on trajectories

A problem involving constraints is classically presented with a **Constraint Network**:

- **Variables:**
- **Constraints:**
- **Domains:**

TCSP $\Leftrightarrow$ approach using:

Introduction

Constraint Network based on trajectories

A problem involving constraints is classically presented with a **Constraint Network**:

$$\begin{cases} \textbf{Variables: } \mathbf{x}(\cdot),\ \mathbf{u}(\cdot),\ y \\ \textbf{Constraints:} \\ \\ \\ \textbf{Domains:} \end{cases}$$

TCSP $\Leftrightarrow$ approach using:

- **trajectories** as variables: $\mathbf{x}(\cdot)$, $\mathbf{u}(\cdot)$

Introduction

Constraint Network based on trajectories

A problem involving constraints is classically presented with a **Constraint Network**:

$$\begin{cases} \textbf{Variables: } \mathbf{x}(\cdot),\ \mathbf{u}(\cdot),\ y \\ \textbf{Constraints:} \\ \\ \\ \textbf{Domains: } [\mathbf{x}](\cdot),\ [\mathbf{u}](\cdot),\ [y] \end{cases}$$

TCSP $\Leftrightarrow$ approach using:

- **trajectories** as variables: $\mathbf{x}(\cdot),\ \mathbf{u}(\cdot)$
- **tubes** as domains: $\mathbf{x}(\cdot) \in [\mathbf{x}](\cdot),\ \mathbf{u}(\cdot) \in [\mathbf{u}](\cdot)$

Introduction

Constraint Network based on trajectories

A problem involving constraints is classically presented with a **Constraint Network**:

$$\begin{cases} \textbf{Variables: } \mathbf{x}(\cdot),\ \mathbf{u}(\cdot),\ y \\ \textbf{Constraints:} \\ \quad - \quad \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \qquad \text{(evolution equation)} \\ \quad - \quad y(t) = g(\mathbf{x}(t)) \qquad \text{(observation equation)} \\ \textbf{Domains: } [\mathbf{x}](\cdot),\ [\mathbf{u}](\cdot),\ [y] \end{cases}$$

TCSP $\Leftrightarrow$ approach using:

- **trajectories** as variables: $\mathbf{x}(\cdot)$, $\mathbf{u}(\cdot)$
- **tubes** as domains: $\mathbf{x}(\cdot) \in [\mathbf{x}](\cdot)$, $\mathbf{u}(\cdot) \in [\mathbf{u}](\cdot)$

Need: consider constraints defined by **differential equations** or any other **non-linear operation** on trajectories.

Section 2

Tubes: Envelopes of Trajectories

Tubes: Envelopes of Trajectories

Definition

Tube $[x](\cdot)$**:** interval of functions $[x^-, x^+]$ such that $\forall t \in \mathbb{R},\ x^-(t) \leqslant x^+(t)$

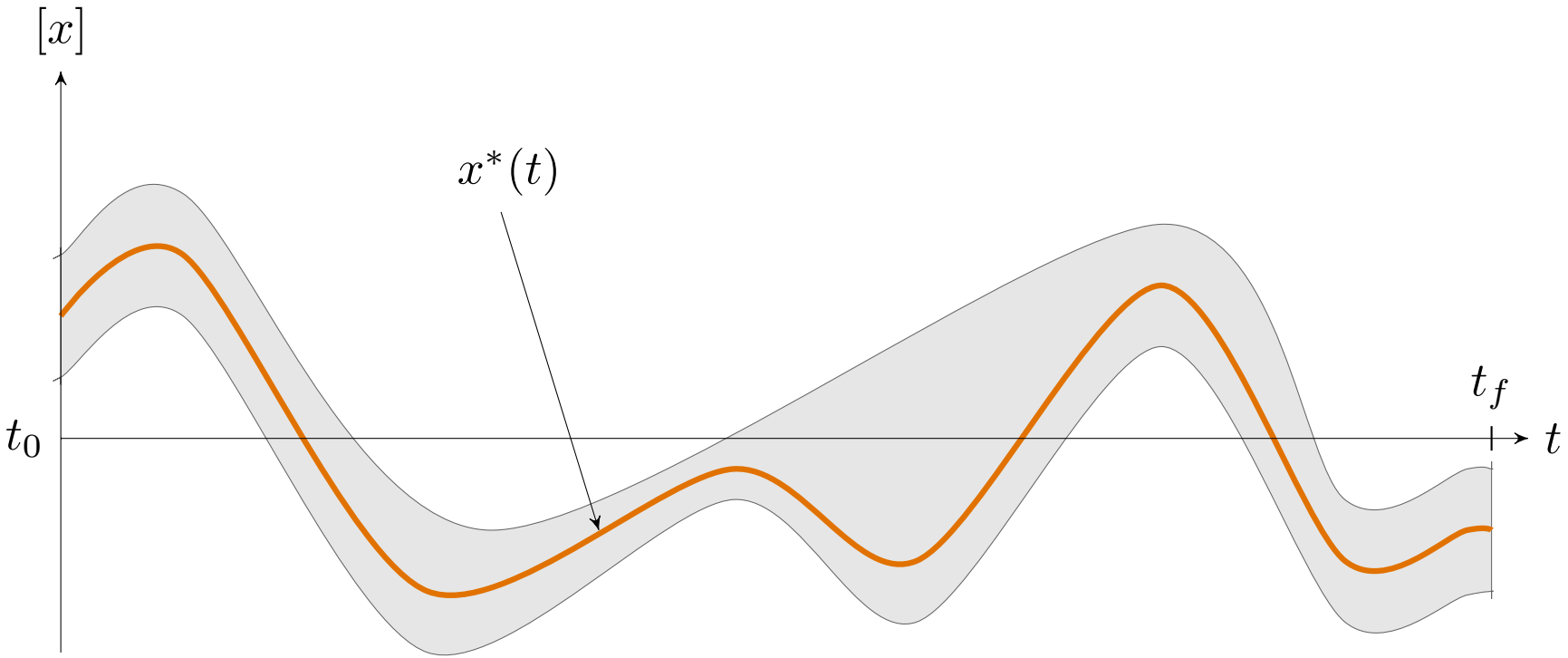


Figure: tube $[x](\cdot)$ enclosing an uncertain trajectory $x^*(\cdot)$

Tubes: Envelopes of Trajectories

Tubes arithmetic

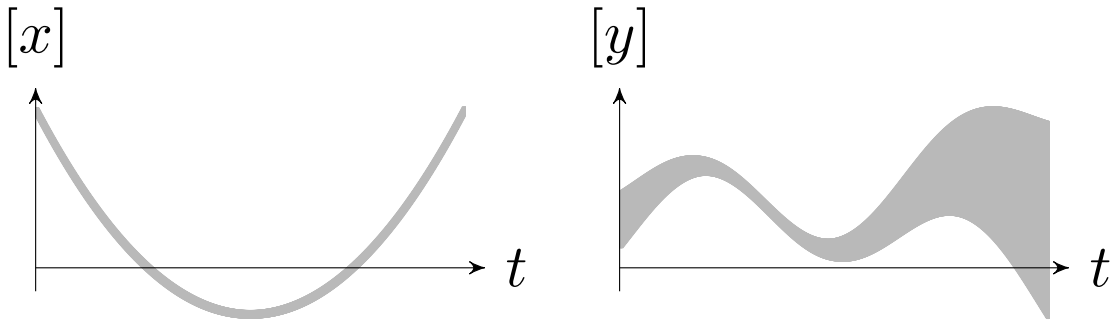
Example:
Tube arithmetic makes it possible to compute the following tubes:

$$\begin{aligned}[a](\cdot) &= [x](\cdot) + [y](\cdot)\\ [b](\cdot) &= \sin\big([x](\cdot)\big)\\ [c](\cdot) &= \int_0 [x](\tau)d\tau\end{aligned}$$

Definition:
If f is an elementary function such as $\sin$, $\cos$, ...,
$f\big([x](\cdot)\big)$ is the smallest tube containing all feasible values for
$f\big(x(\cdot)\big)$, $x(\cdot) \in [x](\cdot)$.

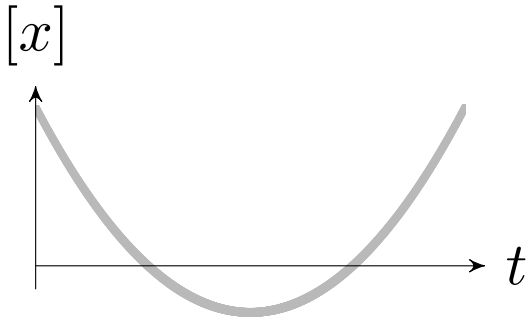
Tubes: Envelopes of Trajectories

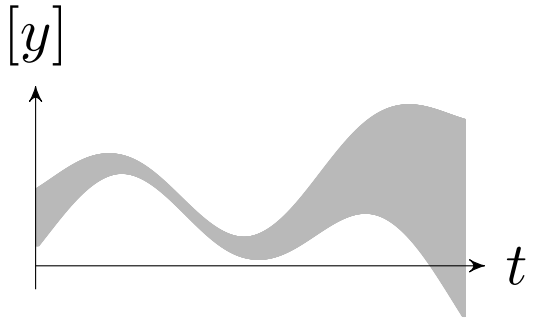
Tubes arithmetic: example

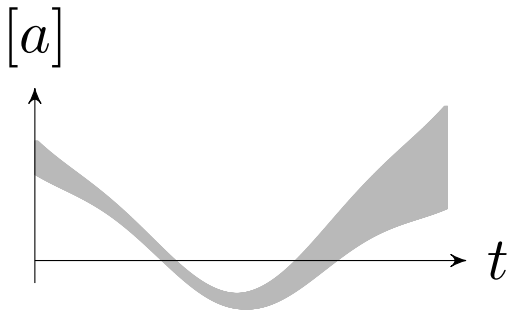


Tubes: Envelopes of Trajectories

Tubes arithmetic: example



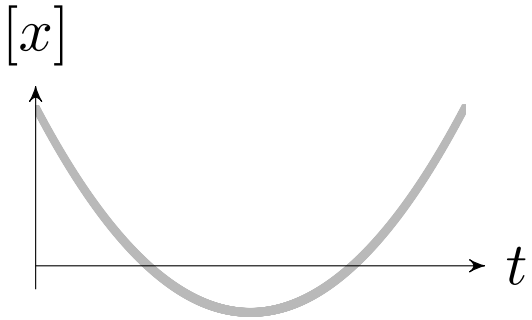


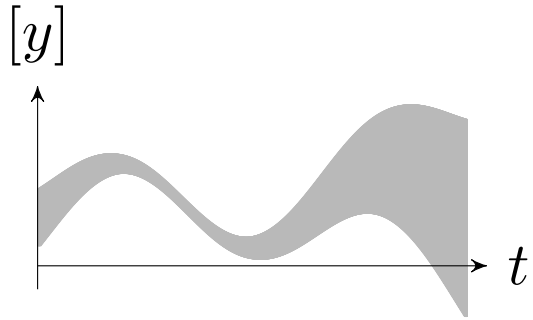


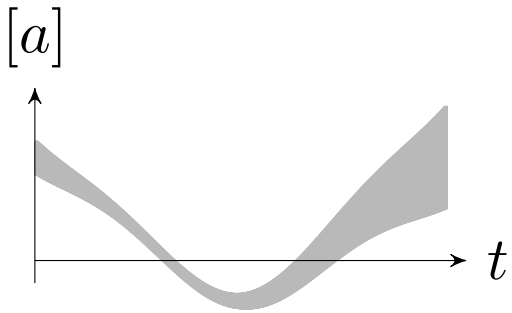
$$a(\cdot) = x(\cdot) + y(\cdot)$$

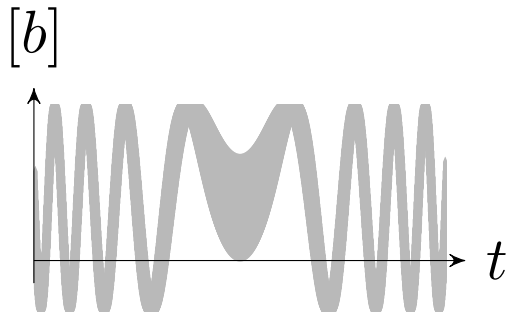
Tubes: Envelopes of Trajectories

Tubes arithmetic: example





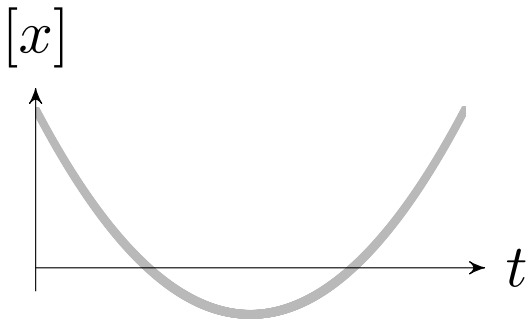


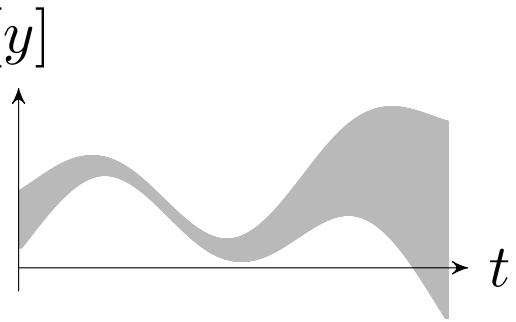


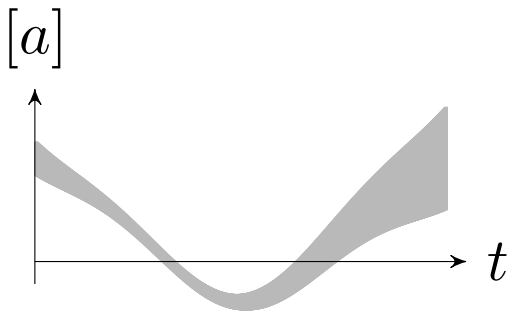
$$a(\cdot) = x(\cdot) + y(\cdot) \qquad b(\cdot) = \sin\big(x(\cdot)\big)$$

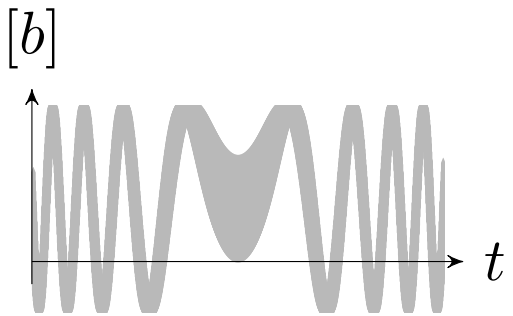
Tubes: Envelopes of Trajectories

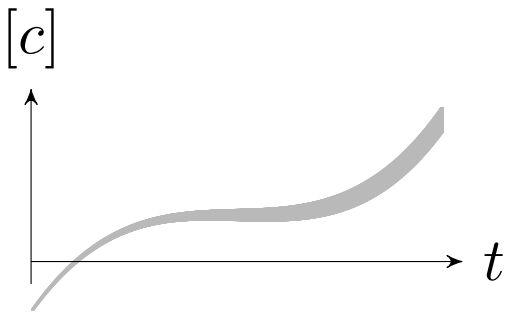
Tubes arithmetic: example











$$a(\cdot) = x(\cdot) + y(\cdot) \qquad b(\cdot) = \sin\big(x(\cdot)\big) \qquad c(\cdot) = \int_0 x(\tau)d\tau$$

Tubes: Envelopes of Trajectories

Integral of tubes

Definition: the integral of a tube $[x](\cdot) = [x^-, x^+]$ is an interval:

$$\int_a^b [x](\tau)d\tau = \left\{ \int_a^b x(\tau)d\tau \mid x \in [x] \right\} = \left[\int_a^b x^-(\tau)d\tau, \int_a^b x^+(\tau)d\tau \right]$$

[Aubry2013]

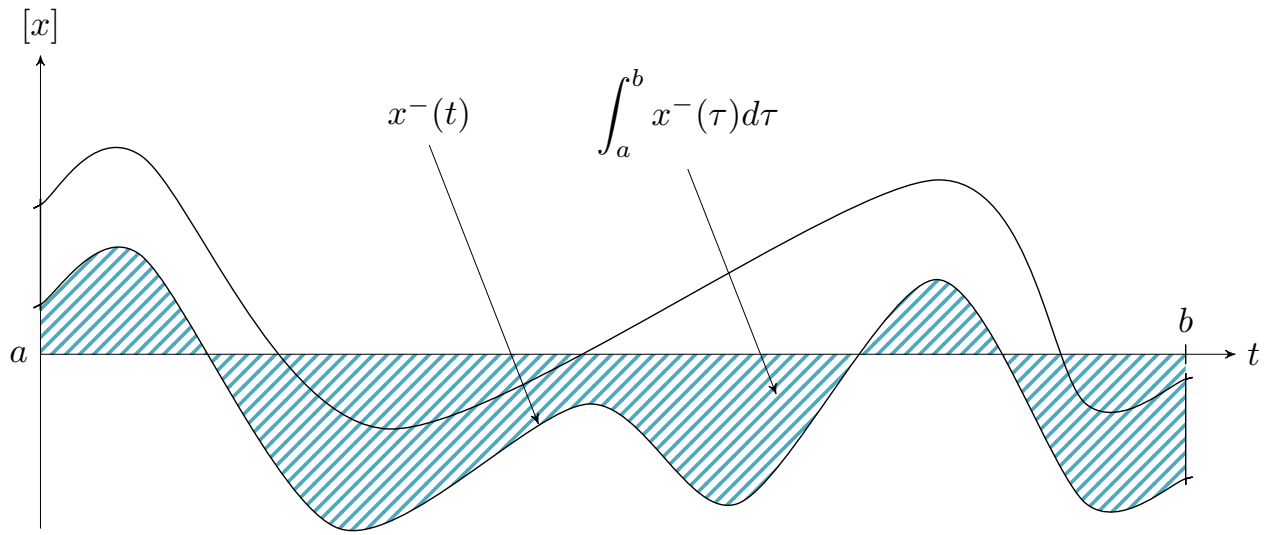


Figure: blue area: lower bound of the tube's integral

Tubes: Envelopes of Trajectories

Integral of tubes

Definition: the integral of a tube $[x](\cdot) = [x^-, x^+]$ is an interval:

$$\int_a^b [x](\tau)d\tau = \left\{ \int_a^b x(\tau)d\tau \mid x \in [x] \right\} = \left[\int_a^b x^-(\tau)d\tau, \int_a^b x^+(\tau)d\tau \right]$$

[Aubry2013]

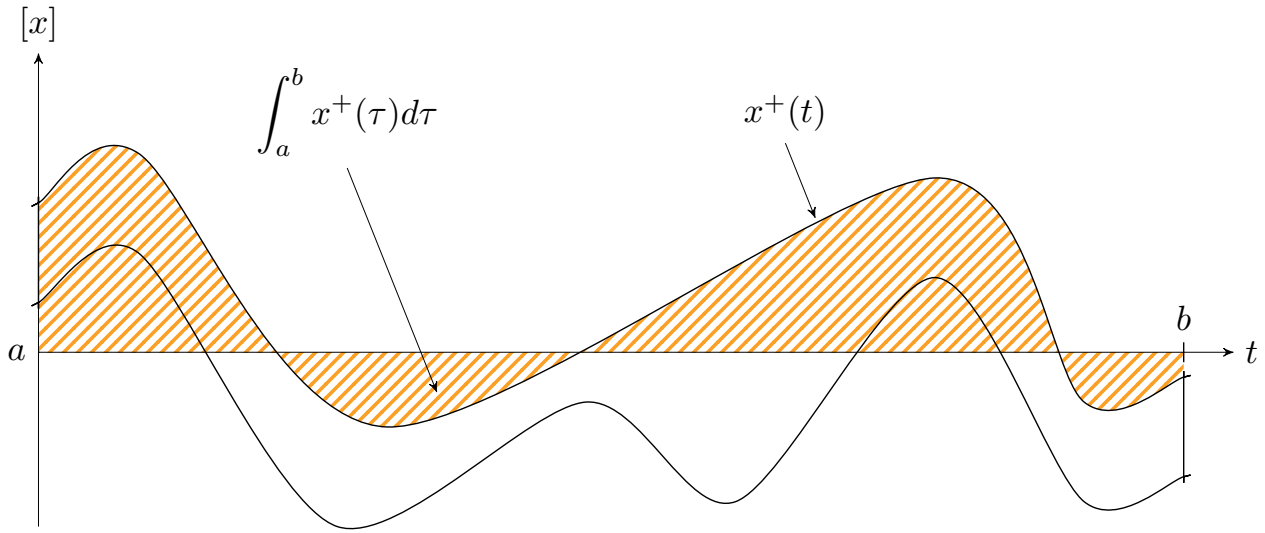


Figure: orange area: upper bound of the tube's integral

Section 3

Tube contractors

Tube contractors

Tubes: arithmetic and contractors

Example of constraints on trajectories:

$$\begin{aligned} a(\cdot) &= x(\cdot) + y(\cdot) \\ b(\cdot) &= \sin\big(x(\cdot)\big) \\ c(\cdot) &= \int_0 x(\tau)d\tau \end{aligned}$$

Tubes $[a](\cdot)$, $[b](\cdot)$, $[c](\cdot)$ respectively contain solutions $a(\cdot)$, $b(\cdot)$, $c(\cdot)$.

Tube contractors

Tubes: arithmetic and contractors

Example of constraints on trajectories:

$$\begin{aligned} a(\cdot) &= x(\cdot) + y(\cdot) \\ b(\cdot) &= \sin\big(x(\cdot)\big) \\ c(\cdot) &= \int_0 x(\tau)d\tau \end{aligned}$$

Tubes $[a](\cdot)$, $[b](\cdot)$, $[c](\cdot)$ respectively contain solutions $a(\cdot)$, $b(\cdot)$, $c(\cdot)$.

Contractors for tubes:
For each primitive constraint, tubes from both sides of the equation are contracted without removing any feasible solution.

Tube contractors

Tubes: minimal and non-minimal contractors

Example:

Contractor $\mathcal{C}_+$ associated with the constraint $a(\cdot) = x(\cdot) + y(\cdot)$:

$$\begin{pmatrix} [a](\cdot) \\ [x](\cdot) \\ [y](\cdot) \end{pmatrix} \mapsto \begin{pmatrix} [a](\cdot) \cap ([x](\cdot) + [y](\cdot)) \\ [x](\cdot) \cap ([a](\cdot) - [y](\cdot)) \\ [y](\cdot) \cap ([a](\cdot) - [x](\cdot)) \end{pmatrix}$$

Tube contractors

Tubes: minimal and non-minimal contractors

Example:

Contractor $\mathcal{C}_+$ associated with the constraint $a(\cdot) = x(\cdot) + y(\cdot)$:

$$\begin{pmatrix} [a](\cdot) \\ [x](\cdot) \\ [y](\cdot) \end{pmatrix} \mapsto \begin{pmatrix} [a](\cdot) \cap ([x](\cdot) + [y](\cdot)) \\ [x](\cdot) \cap ([a](\cdot) - [y](\cdot)) \\ [y](\cdot) \cap ([a](\cdot) - [x](\cdot)) \end{pmatrix}$$

Example:

Contractor $\mathcal{C}_{\int_0}$ associated with the constraint $c(\cdot) = \int_0 x(\tau)d\tau$:

$$\begin{pmatrix} [x](\cdot) \\ [c](\cdot) \end{pmatrix} \mapsto \begin{pmatrix} [x](\cdot) \\ [c](\cdot) \cap \int_0 [x](\tau)\, d\tau \end{pmatrix}$$

Section 4

New tube contractors

New tube contractors

Contractors for state estimation

State equations:

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) & \text{(evolution)} \\ y(t) = g(\mathbf{x}(t)) & \text{(observation)} \end{cases}$$

Need for a **differential** contractor $\mathcal{C}_{\frac{d}{dt}}$ and a **probe** contractor $\mathcal{C}_{\text{obs}}$.

New tube contractors

Contractors for state estimation

State equations:

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) & (\text{evolution}) \\ y(t) = g(\mathbf{x}(t)) & (\text{observation}) \end{cases}$$

Need for a **differential** contractor $\mathcal{C}_{\frac{d}{dt}}$ and a **probe** contractor $\mathcal{C}_{\text{obs}}$.

- $\mathcal{C}_{\frac{d}{dt}}$: related to the constraint $\dot{\mathbf{x}} = \mathbf{v}$
  $\mathbf{x}(\cdot) \in [\mathbf{x}](\cdot)$, $\mathbf{v}(\cdot) \in [\mathbf{v}](\cdot)$

New tube contractors

Contractors for state estimation

State equations:

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) & \text{(evolution)} \\ y(t) = g(\mathbf{x}(t)) & \text{(observation)} \end{cases}$$

Need for a **differential** contractor $\mathcal{C}_{\frac{d}{dt}}$ and a **probe** contractor $\mathcal{C}_{\text{obs}}$.

- $\mathcal{C}_{\frac{d}{dt}}$: related to the constraint $\dot{\mathbf{x}} = \mathbf{v}$
  $\mathbf{x}(\cdot) \in [\mathbf{x}](\cdot)$, $\mathbf{v}(\cdot) \in [\mathbf{v}](\cdot)$
- $\mathcal{C}_{\text{obs}}$: related to the constraint $\mathbf{x}(t) = \mathbf{z}$
  $t \in [t]$, $\mathbf{z} \in [\mathbf{z}]$, $\mathbf{x}(\cdot) \in [\mathbf{x}](\cdot)$

Lab-STICC

New tube contractors

Contractors for state estimation

State equations:

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) & \text{(evolution)} \\ y(t) = g(\mathbf{x}(t)) & \text{(observation)} \end{cases}$$

Need for a **differential** contractor $\mathcal{C}_{\frac{d}{dt}}$ and a **probe** contractor $\mathcal{C}_{\text{obs}}$.

- $\mathcal{C}_{\frac{d}{dt}}$: related to the constraint $\dot{\mathbf{x}} = \mathbf{v}$
  $\mathbf{x}(\cdot) \in [\mathbf{x}](\cdot)$, $\mathbf{v}(\cdot) \in [\mathbf{v}](\cdot)$
- $\mathcal{C}_{\text{obs}}$: related to the constraint $\mathbf{x}(t) = \mathbf{z}$
  $t \in [t]$, $\mathbf{z} \in [\mathbf{z}]$, $\mathbf{x}(\cdot) \in [\mathbf{x}](\cdot)$
  $t \in [t] \implies$ constraint $\dot{\mathbf{x}} = \mathbf{v}$

New tube contractors

Contractors for state estimation

State equations:

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) & \text{(evolution)} \\ y(t) = g(\mathbf{x}(t)) & \text{(observation)} \end{cases}$$

Need for a **differential** contractor $\mathcal{C}_{\frac{d}{dt}}$ and a **probe** contractor $\mathcal{C}_{\text{obs}}$.

- $\mathcal{C}_{\frac{d}{dt}}$: related to the constraint $\dot{\mathbf{x}} = \mathbf{v}$
  $\mathbf{x}(\cdot) \in [\mathbf{x}](\cdot)$, $\mathbf{v}(\cdot) \in [\mathbf{v}](\cdot)$
- $\mathcal{C}_{\text{obs}}$: related to the constraint $\mathbf{x}(t) = \mathbf{z}$
  $t \in [t]$, $\mathbf{z} \in [\mathbf{z}]$, $\mathbf{x}(\cdot) \in [\mathbf{x}](\cdot)$, $\mathbf{v}(\cdot) \in [\mathbf{v}](\cdot)$
  $t \in [t] \implies$ constraint $\dot{\mathbf{x}} = \mathbf{v}$

New tube contractors

Contractors for state estimation

$\mathcal{C}_{\text{obs}}$: contraction based on the observation $[z_1]$ made at time $[t_1]$.

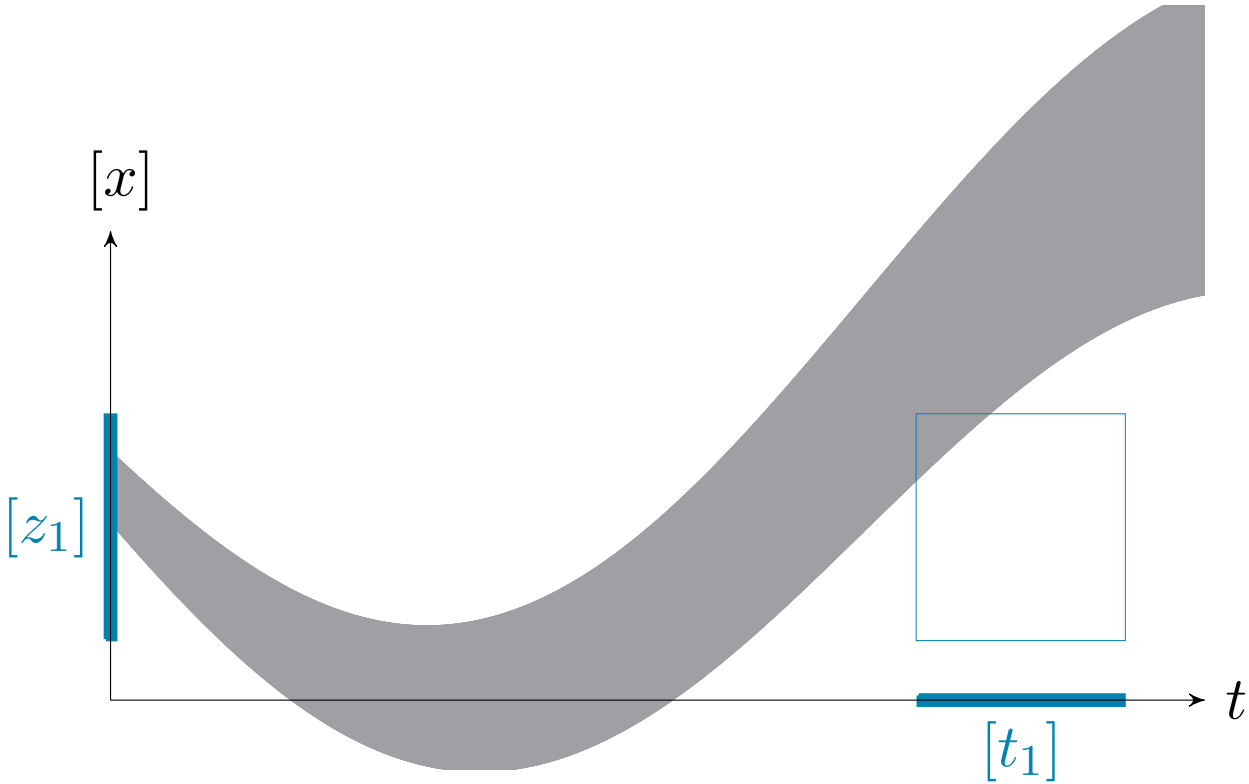


Figure: tube $[x](\cdot)$ before contraction

New tube contractors

Contractors for state estimation

$\mathcal{C}_{\text{obs}}$: contraction based on the observation $[z_1]$ made at time $[t_1]$.

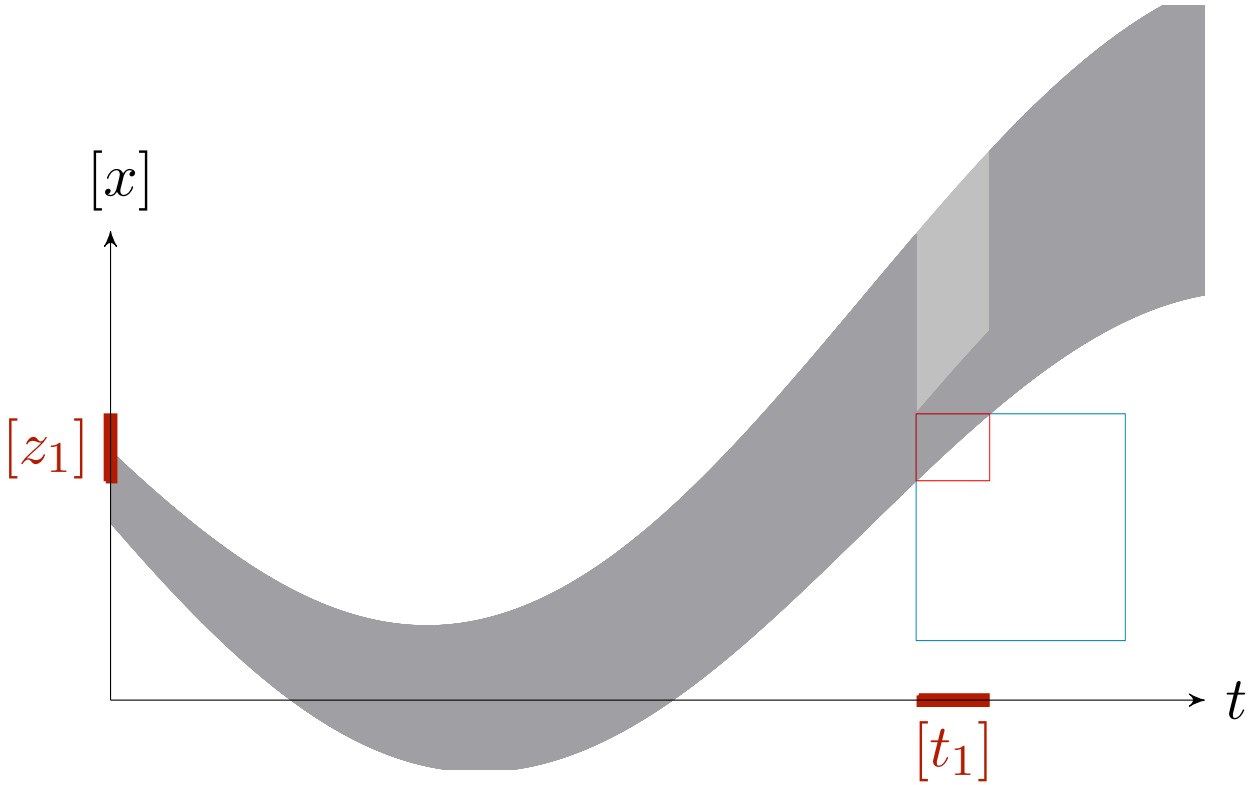


Figure: contraction of tube $[x](\cdot)$ and both $[z_1]$ and $[t_1]$

New tube contractors

Contractors for state estimation

$\mathcal{C}_{\text{obs}}$: contraction based on the observation $[z_1]$ made at time $[t_1]$.

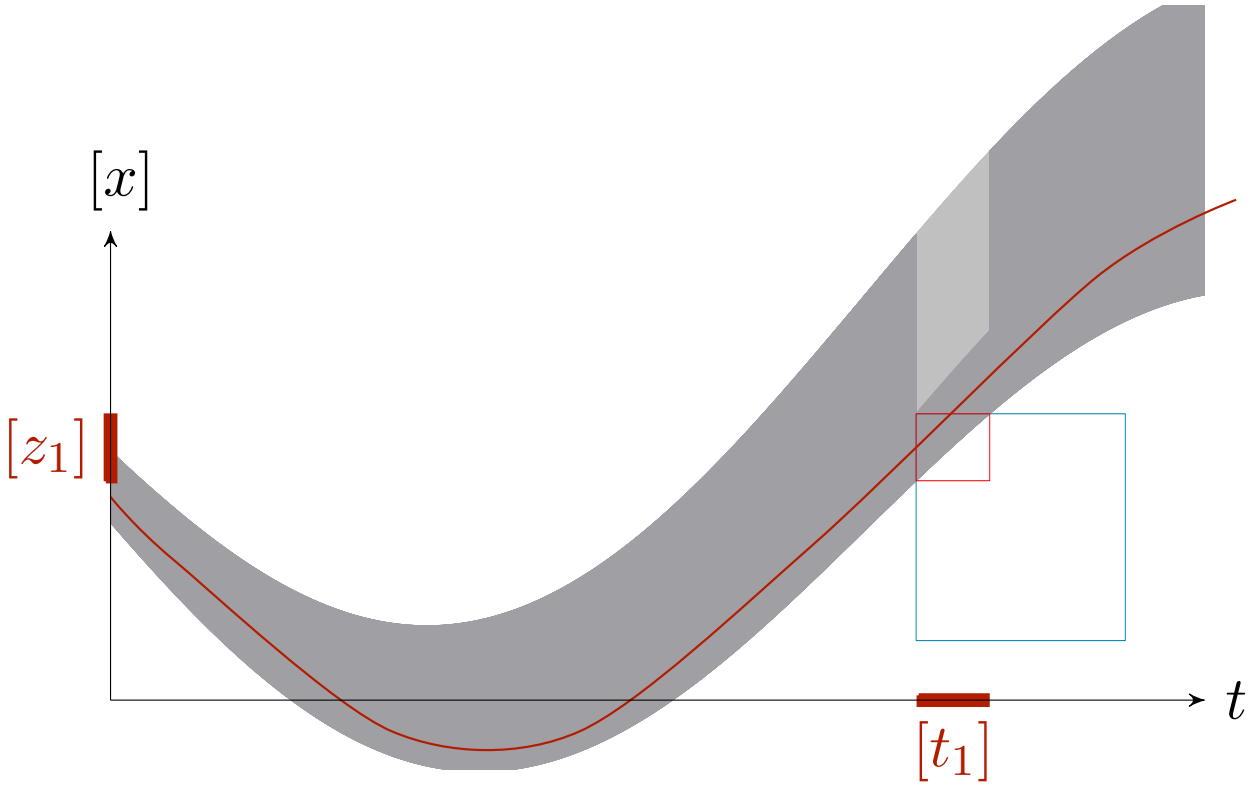


Figure: contraction of tube $[x](\cdot)$ and both $[z_1]$ and $[t_1]$

New tube contractors

Contractors for state estimation

$\mathcal{C}_{\frac{d}{dt}}$: contraction based on the differential constraint:

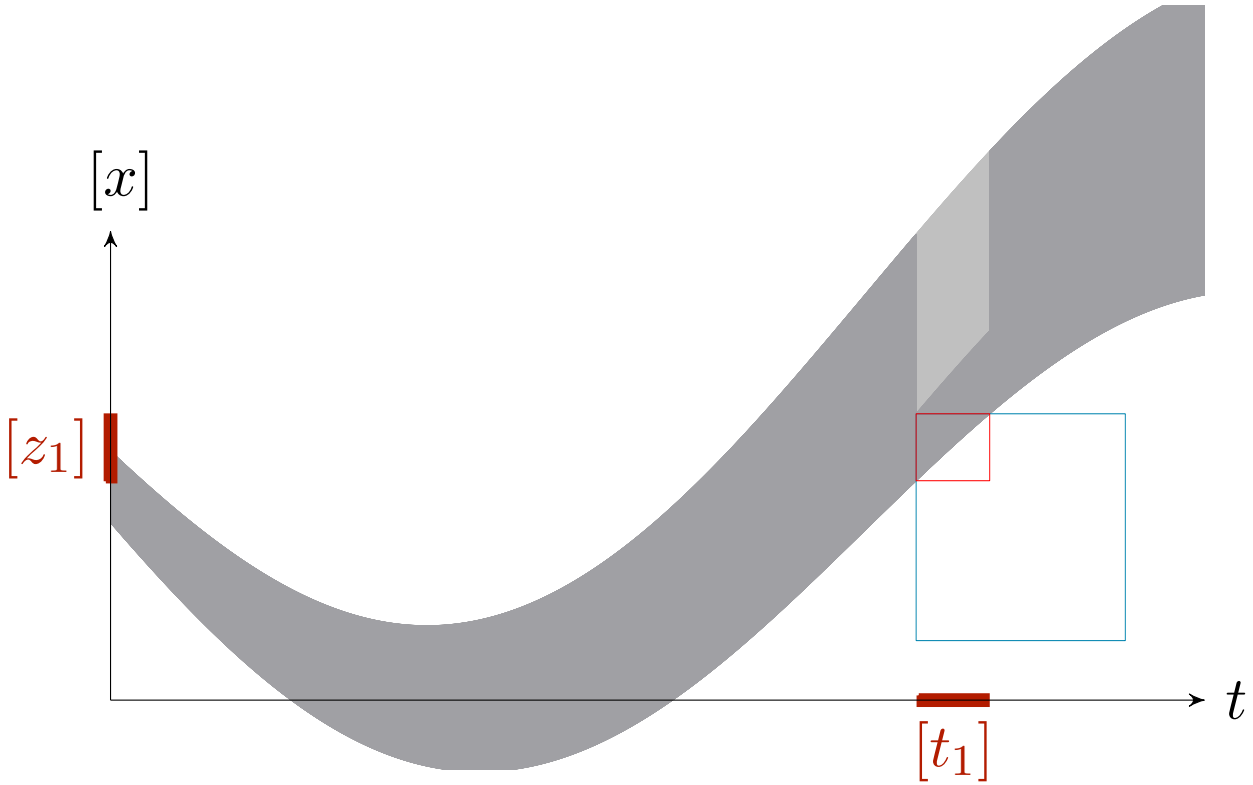


Figure: contraction of tube $[x](\cdot)$ and both $[z_1]$ and $[t_1]$

New tube contractors

Contractors for state estimation

$\mathcal{C}_{\frac{d}{dt}}$: contraction based on the differential constraint:

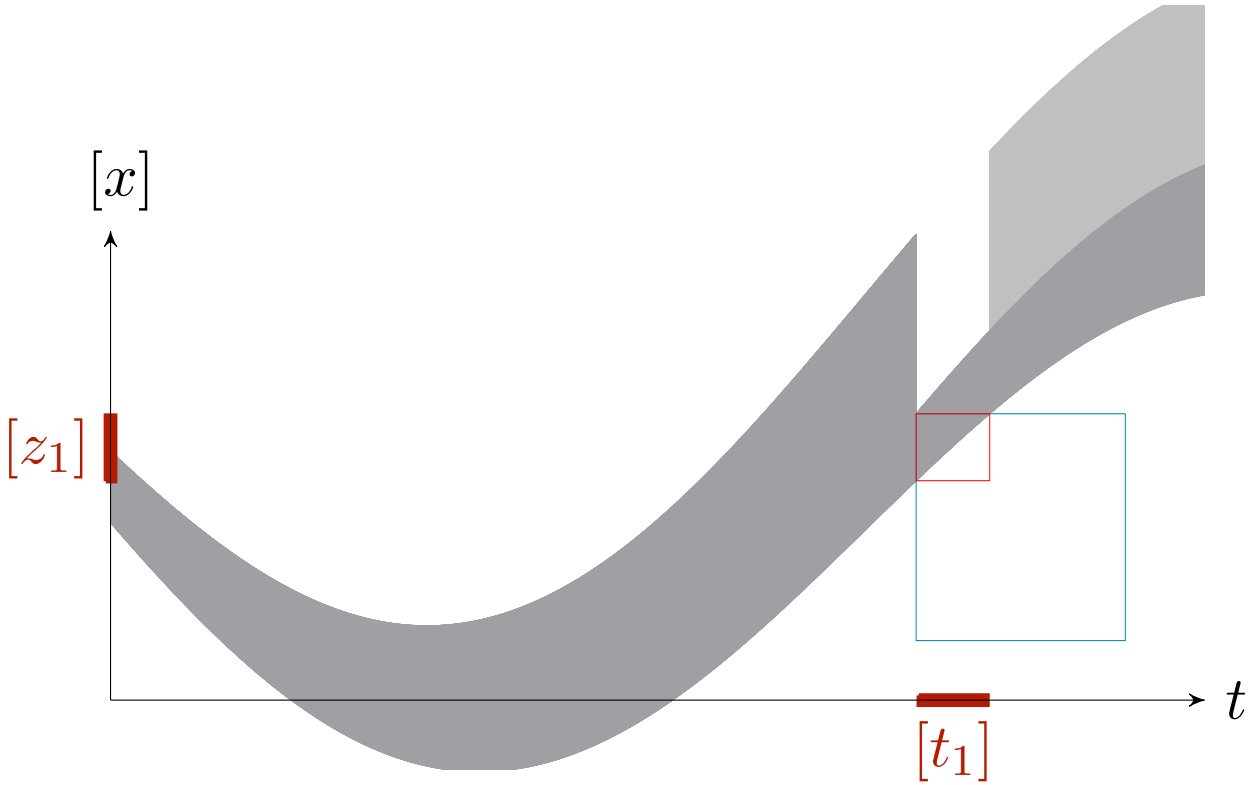


Figure: tube contraction in forward

New tube contractors

Contractors for state estimation

$\mathcal{C}_{\frac{d}{dt}}$: contraction based on the differential constraint:

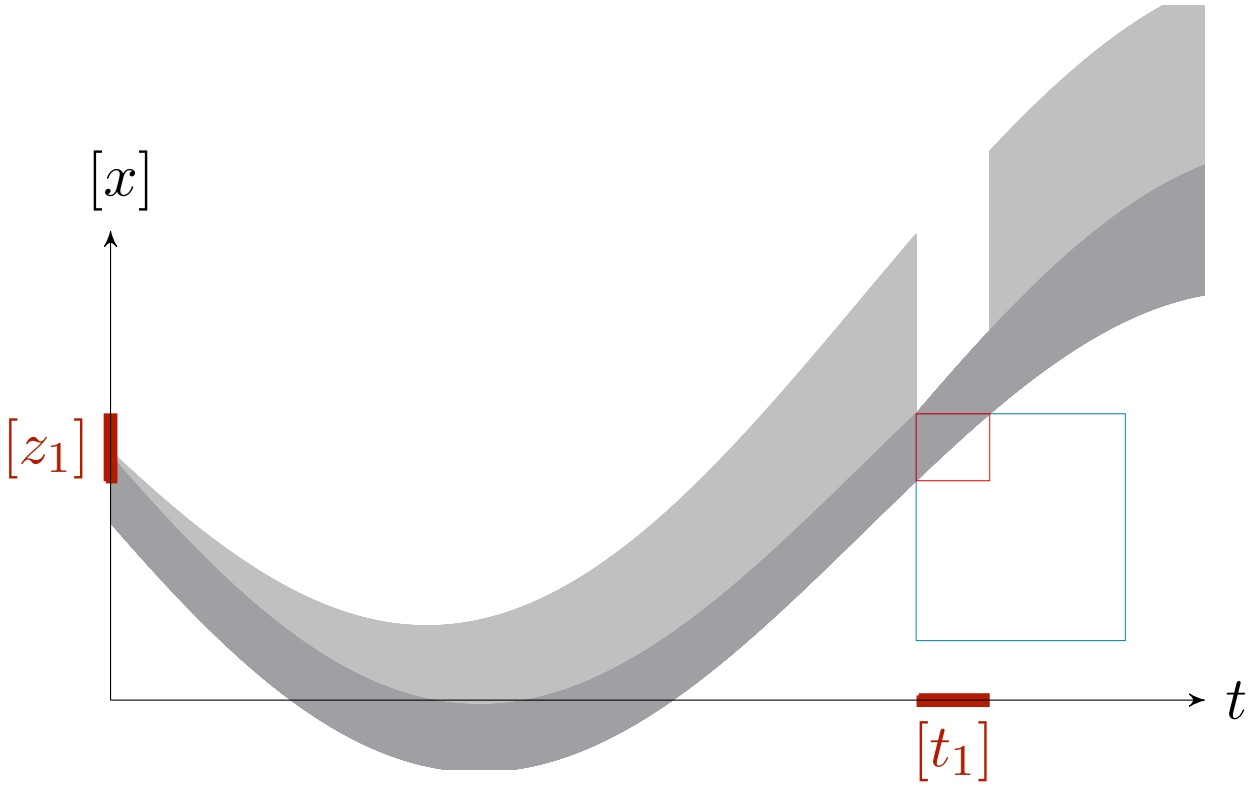


Figure: tube contraction in forward/backward

New tube contractors

Contractors for state estimation

Contractor $\mathcal{C}_{\frac{d}{dt}}$ related to the constraint $\dot{x} = v$:

$$\begin{pmatrix} [x](t) \\ [v](t) \end{pmatrix} \mapsto \begin{pmatrix} \bigcap_{t_1=t_0}^{t_f} \left([x](t_1) + \int_{t_1}^{t} [v](\tau) d\tau \right) \\ [v](t) \end{pmatrix}$$

New tube contractors

Contractors for state estimation

Contractor $\mathcal{C}_{\frac{d}{dt}}$ related to the constraint $\dot{x} = v$:

$$\begin{pmatrix} [x](t) \\ [v](t) \end{pmatrix} \mapsto \begin{pmatrix} \bigcap_{t_1=t_0}^{t_f} \left([x](t_1) + \int_{t_1}^{t} [v](\tau) d\tau \right) \\ [v](t) \end{pmatrix}$$

Contractor $\mathcal{C}_{\text{obs}}$ related to the constraint $y = x(t)$:

$$\begin{pmatrix} [t] \\ [y] \\ [x]\,(t) \\ [v]\,(\cdot) \end{pmatrix} \mapsto \begin{pmatrix} [t] \cap [x]^{-1}([y]) \\ [y] \cap [x]([t]) \\ [x](t) \cap \bigsqcup_{t_1 \in [t]} \left(([x](t_1) \cap [y]) + \int_{t_1}^{t} [v](\tau) d\tau \right) \\ [v]\,(\cdot) \end{pmatrix}$$

New tube contractors

Storyboard of a forward/backward tube contraction

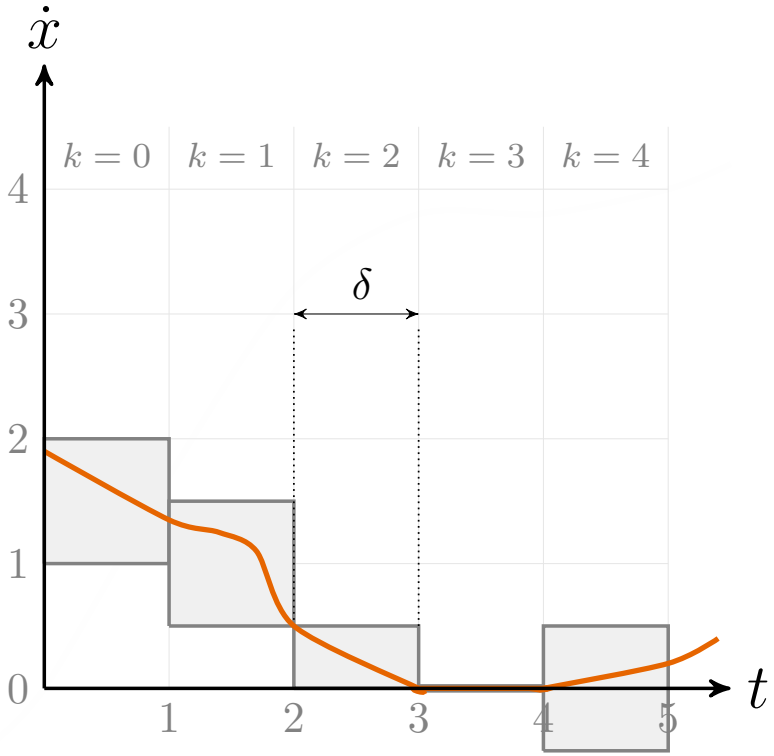


Figure: Derivative tube $[\dot{x}](\cdot)$ used to contract $[x](\cdot)$ initialized to $[-\infty, \infty] \forall t$

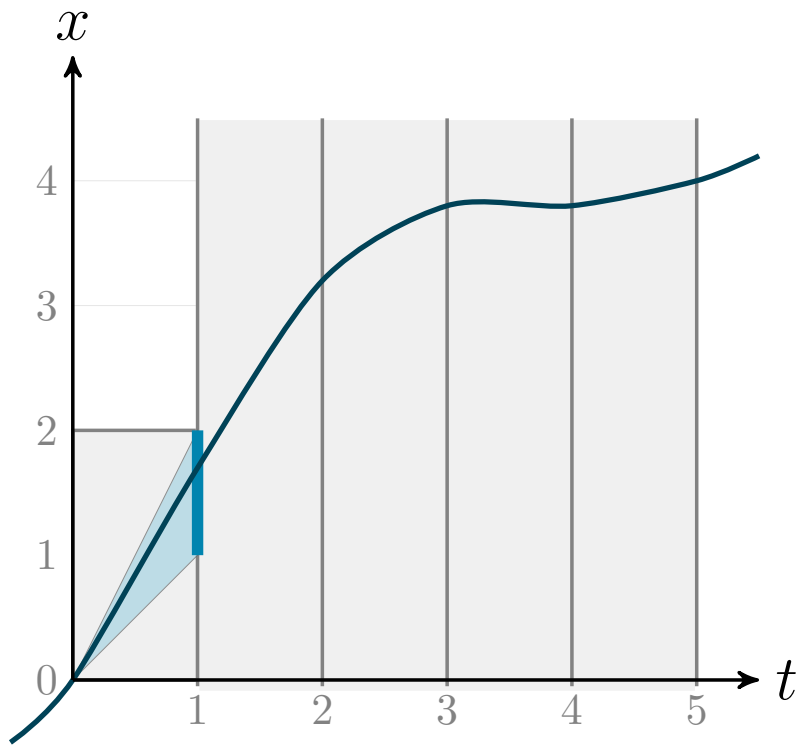


Figure: First step of forward integration contractor: first slice k_0 is contracted from $[-\infty, \infty]$ to $[0, 2]$

New tube contractors

Storyboard of a forward/backward tube contraction

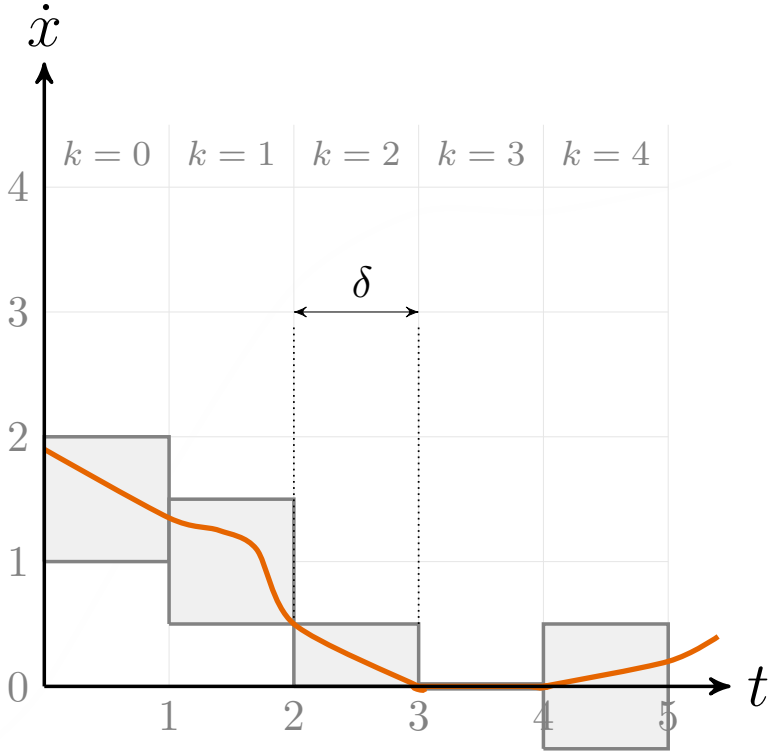


Figure: Derivative tube $[\dot{x}](\cdot)$ used to contract $[x](\cdot)$ initialized to $[-\infty, \infty]\forall t$

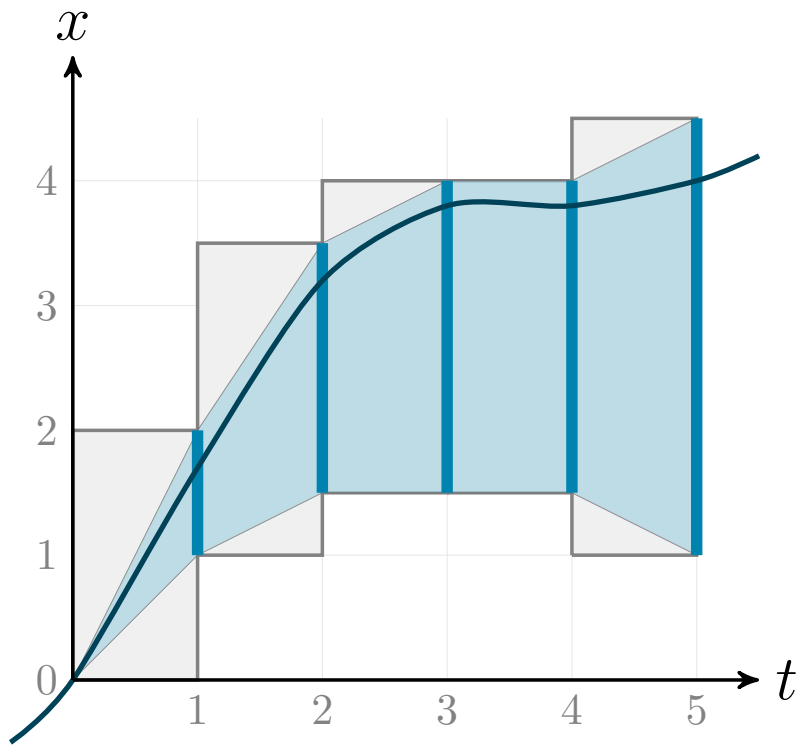


Figure: End of forward contractions. The minimal envelope of solutions compliant with $[\dot{x}](\cdot)$ is pictured in blue. Slices representation – outer approximation – is depicted in gray

New tube contractors

Storyboard of a forward/backward tube contraction

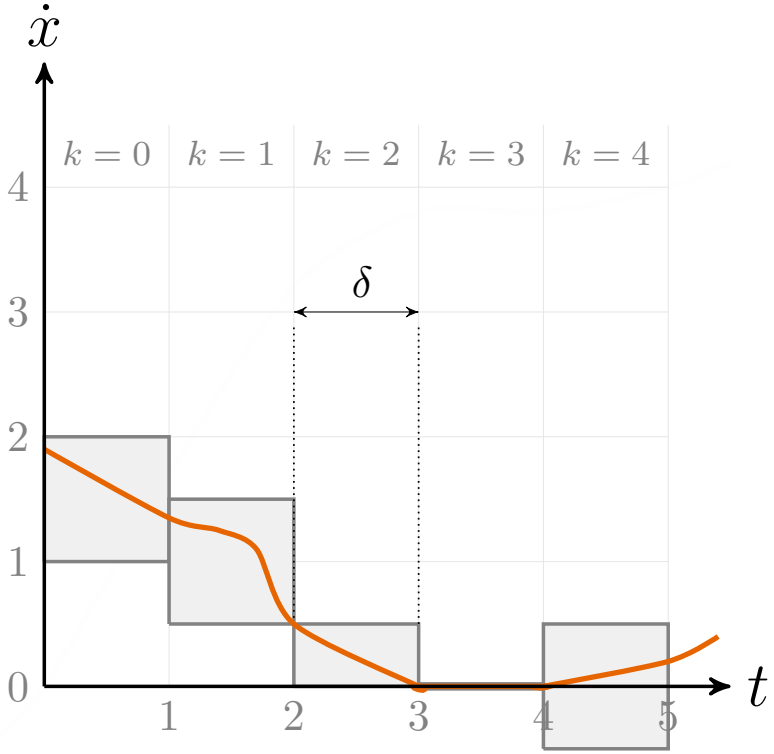


Figure: Derivative tube $[\dot{x}](\cdot)$ used to contract $[x](\cdot)$ initialized to $[-\infty, \infty]\forall t$

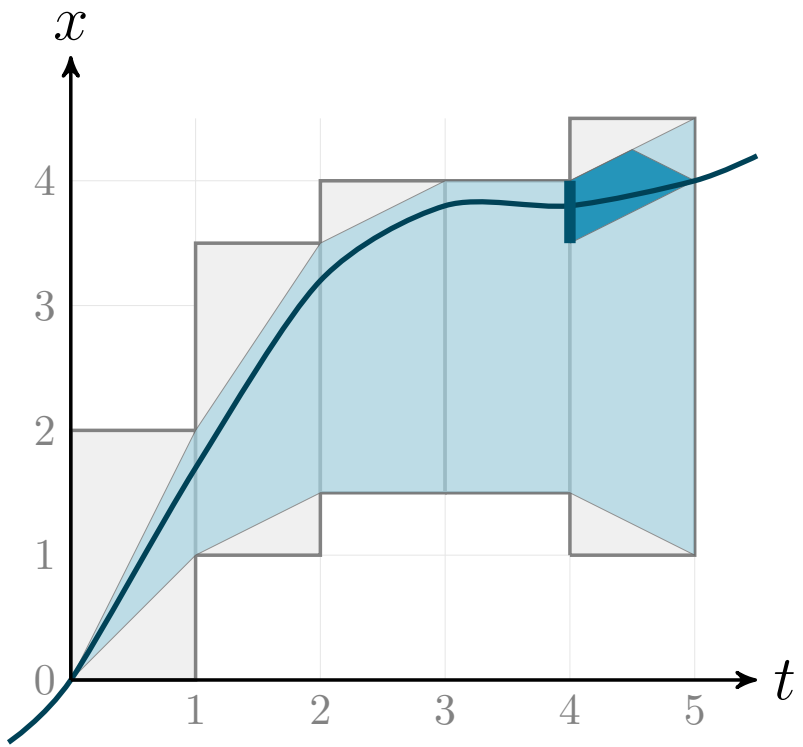


Figure: First step of backward integration contractor, with the last condition $x(5) = 4$. A reduced envelope, pictured in dark blue, is computed from $t = 5$ to $t = 0$

New tube contractors

Storyboard of a forward/backward tube contraction

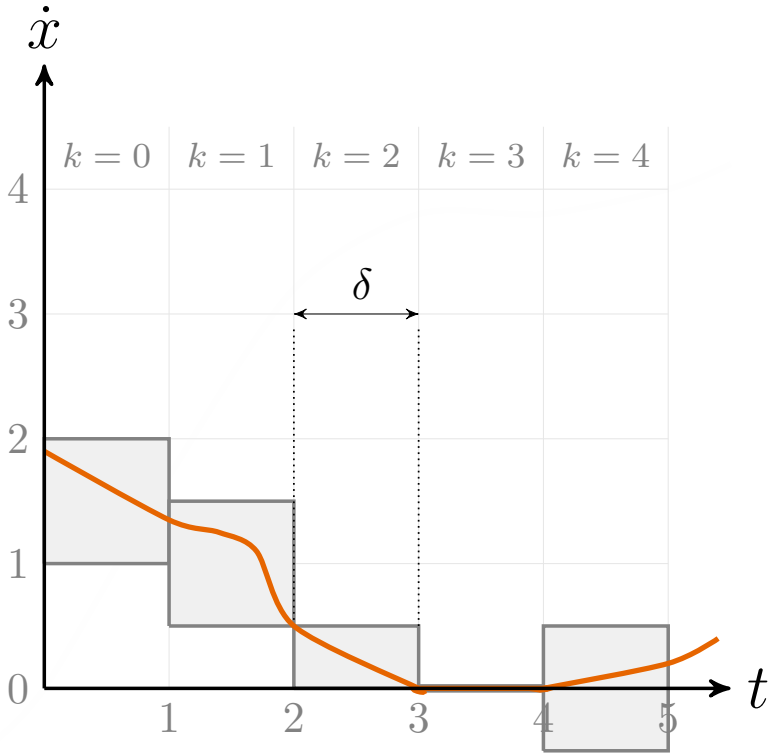


Figure: Derivative tube $[\dot{x}](\cdot)$ used to contract $[x](\cdot)$ initialized to $[-\infty, \infty]\forall t$

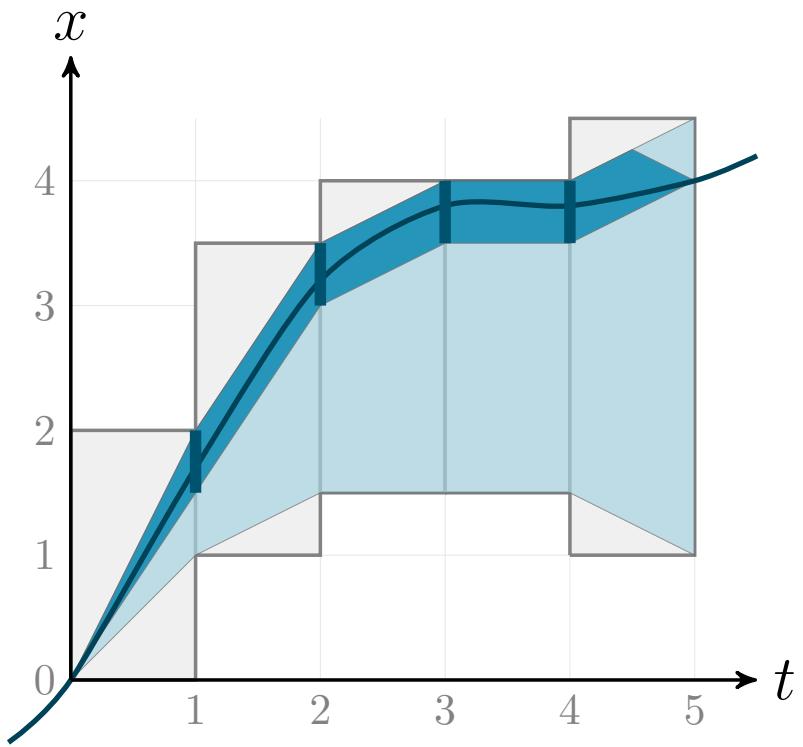


Figure: End of backward processing

New tube contractors

Storyboard of a forward/backward tube contraction

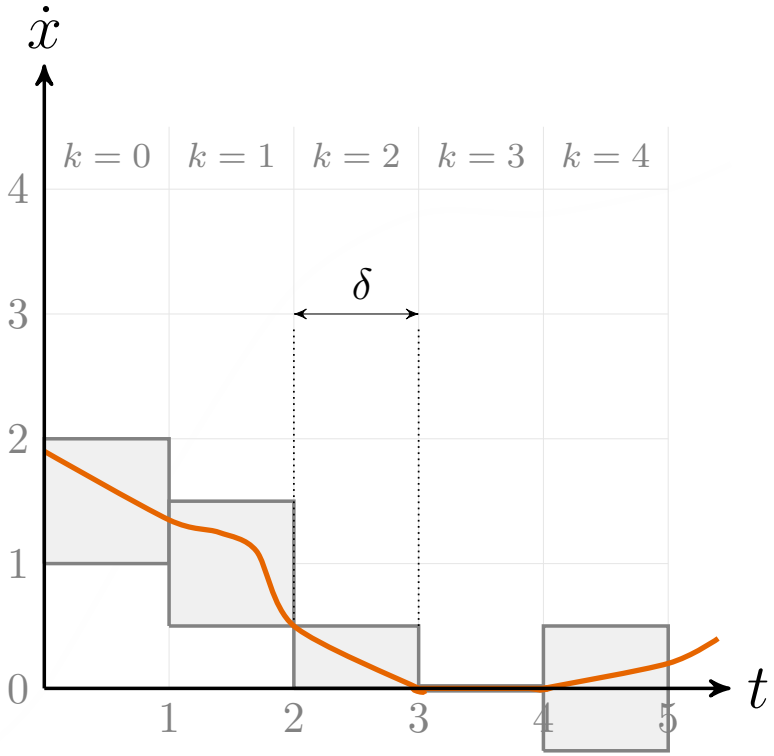


Figure: Derivative tube $[\dot{x}](\cdot)$ used to contract $[x](\cdot)$ initialized to $[-\infty, \infty] \forall t$

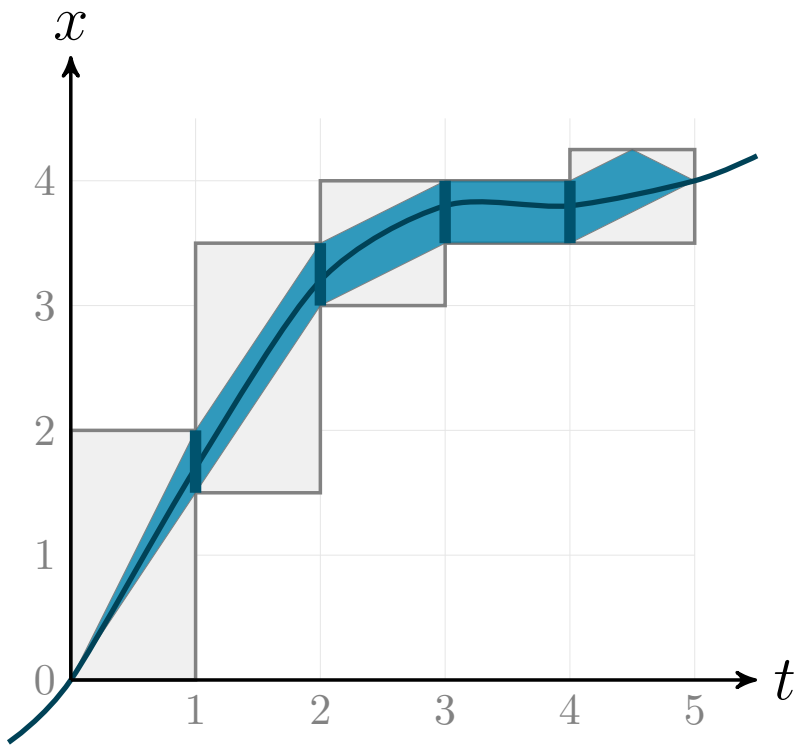


Figure: Tube $[x](\cdot)$ is contracted to optimally enclose the thin envelope pictured in dark blue

Section 5

Simulation of Mobile Robots

Simulation of Mobile Robots

Dead-reckoning

Considering a car $\mathcal{R}$ which state is $\mathbf{x} = (x, y, \theta)^\intercal$.

$$\mathcal{R} \begin{cases} \dot{x}(t) &= v \cdot \cos(\theta) \\ \dot{y}(t) &= v \cdot \sin(\theta) \\ \dot{\theta}(t) &= u(t) \end{cases}$$

Simulation of Mobile Robots

Dead-reckoning

Considering a car $\mathcal{R}$ which state is $\mathbf{x} = (x, y, \theta)^\intercal$.

$$\mathcal{R} \begin{cases} \dot{x}(t) &= v \cdot \cos(\theta) \\ \dot{y}(t) &= v \cdot \sin(\theta) \\ \dot{\theta}(t) &= u(t) \end{cases}$$

- car's speed: $v = 10m.s^{-1}$
- $\mathbf{x}_0 \in [-1, 1] \times [-1, 1] \times [\frac{-6\pi}{5} - 0.002, \frac{-6\pi}{5} + 0.002]$
- $u(t) \in -\cos\left(\frac{t + 33}{5}\right) + [-0.02, 0.02]$

Simulation of Mobile Robots

Dead-reckoning

Considering a car $\mathcal{R}$ which state is $\mathbf{x} = (x, y, \theta)^\intercal$.

$$\mathcal{R} \begin{cases} \dot{x}(t) &= v \cdot \cos(\theta) \\ \dot{y}(t) &= v \cdot \sin(\theta) \\ \dot{\theta}(t) &= u(t) \end{cases}$$

- car's speed: $v = 10m.s^{-1}$
- $\mathbf{x}_0 \in [-1, 1] \times [-1, 1] \times [\frac{-6\pi}{5} - 0.002, \frac{-6\pi}{5} + 0.002]$
- $u(t) \in -\cos\left(\frac{t + 33}{5}\right) + [-0.02, 0.02]$

Initial tube $[u](\cdot)$ is obtained with the analytical expression.
Other tubes $[\theta](\cdot)$, $[\dot{x}](\cdot)$, $[x](\cdot)$, ..., are initialized to $[-\infty, +\infty]$.

Simulation of Mobile Robots

Dead-reckoning

Dead-reckoning: state estimation without observation.

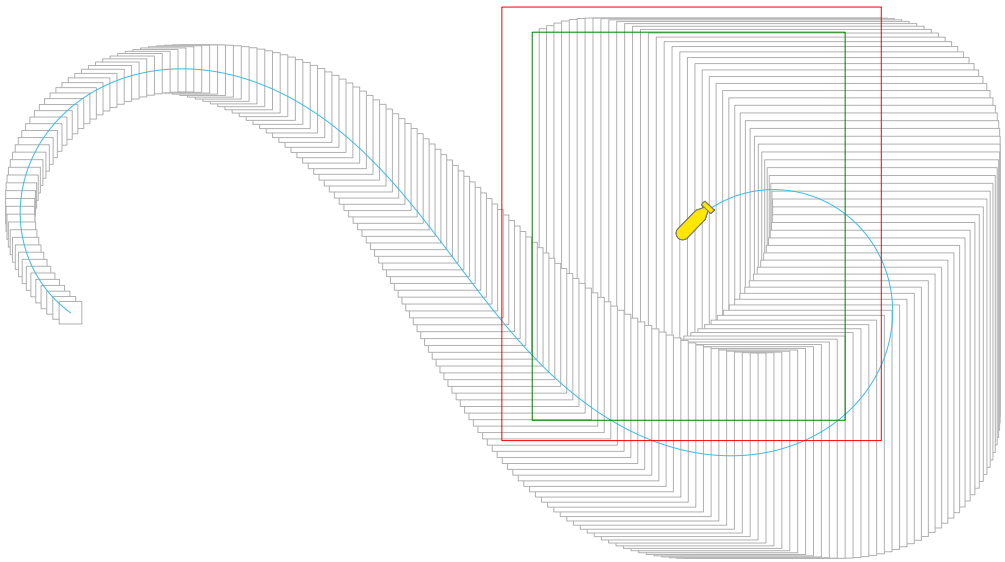


Figure: Interval simulation of the robot following car equations. A strong drift is observed, depicting cumulative uncertainties.

Simulation of Mobile Robots

Observations

Same simulation considering constraints on both initial and final states:

$$\begin{aligned} \mathbf{x}(0) &\in [-1,1] \times [-1,1] \times [\tfrac{-6\pi}{5} - 0.002, \tfrac{-6\pi}{5} + 0.002] \\ \mathbf{x}(14) &\in [53.9, 55.9] \times [6.9, 8.9] \times [-2.36, -2.32] \end{aligned}$$

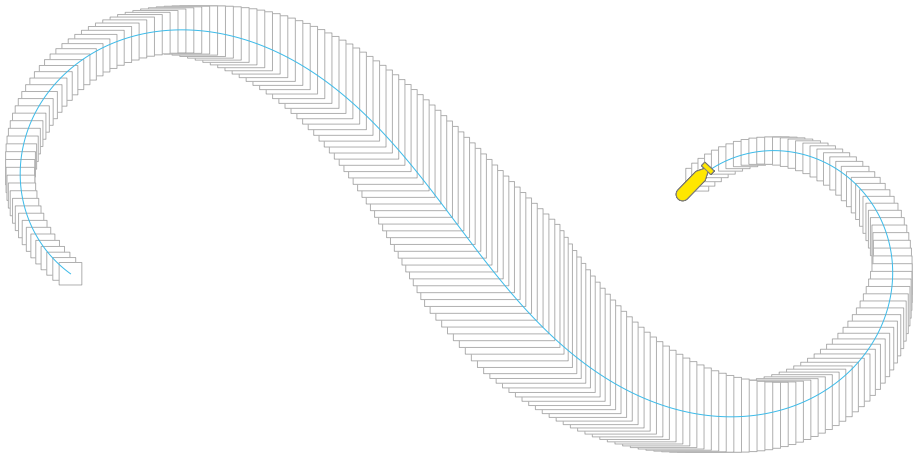


Figure: With a forward/backward contraction, uncertainties are maximal in the middle of the mission.

Simulation of Mobile Robots

Observations

Same simulation considering constraints on both initial and final states:

$$\begin{aligned} \mathbf{x}(0) &\in [-1,1] \times [-1,1] \times [\tfrac{-6\pi}{5} - 0.002, \tfrac{-6\pi}{5} + 0.002] \\ \mathbf{x}(14) &\in [53.9, 55.9] \times [6.9, 8.9] \times [-2.36, -2.32] \end{aligned}$$

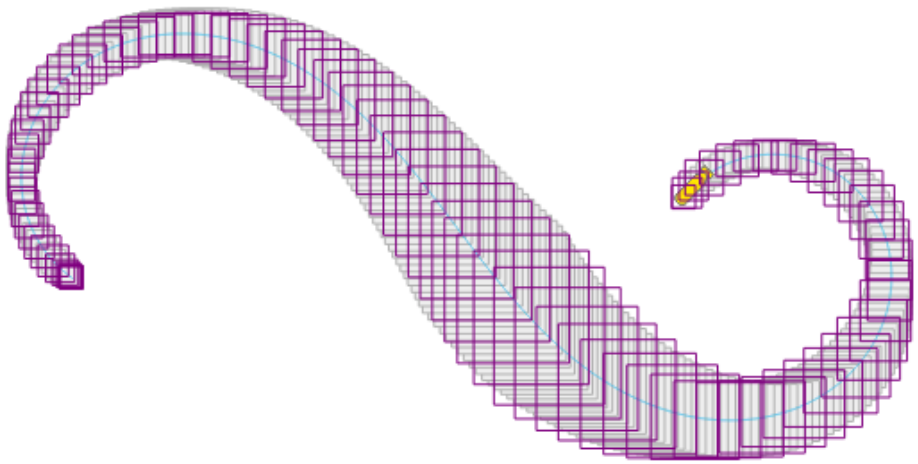


Figure: With a forward/backward contraction.
Comparison with DynIbex (purple boxes).

Simulation of Mobile Robots

Observations

Same simulation adding measurements from beacons.

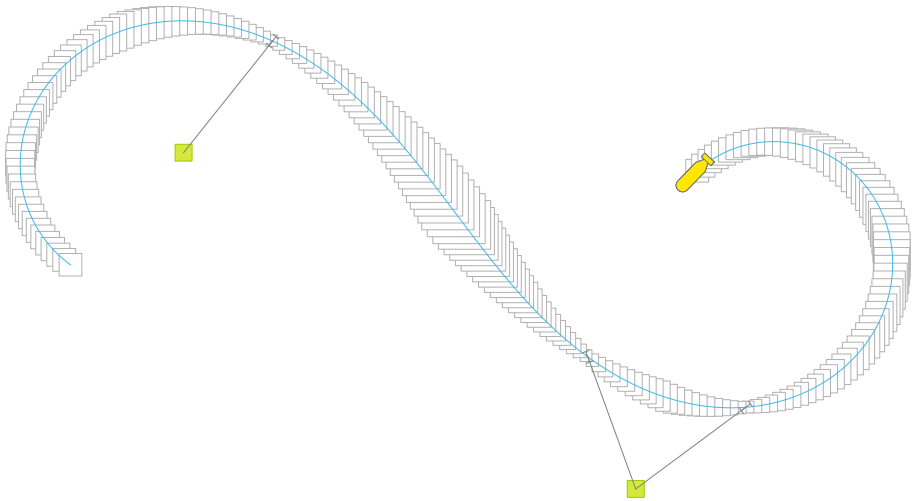


Figure: With a forward/backward contraction, uncertainties are maximal in the middle of the mission.

Simulation of Mobile Robots

Observations

Same simulation without any knowledge on initial or final states:

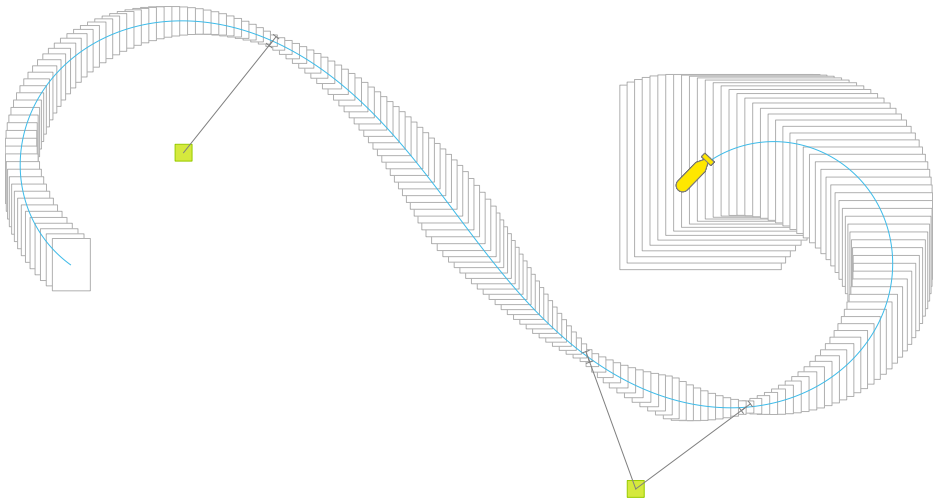


Figure: With a forward/backward contraction, uncertainties are maximal between precise constraints.

Simulation of Mobile Robots

Lissajous example: complex differential equations

Example: a robot described by its state $\mathbf{x}^\intercal = \{x, y\}$ and following a Lissajous trajectory:

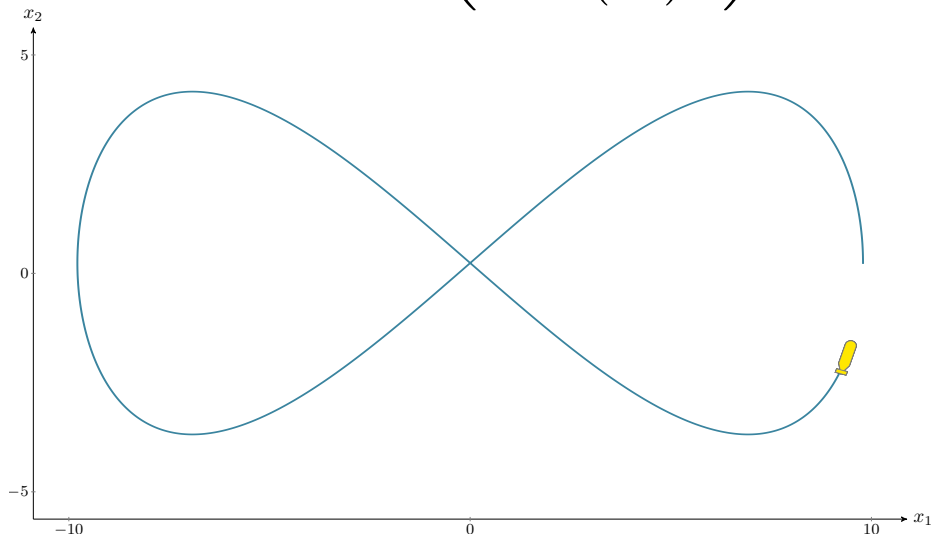
$$\mathbf{x}(t) = 5 \cdot \begin{pmatrix} 2\cos(t) \\ \sin(2t) \end{pmatrix}$$

Simulation of Mobile Robots

Lissajous example: complex differential equations

Example: a robot described by its state $\mathbf{x}^\intercal = \{x, y\}$ and following a Lissajous trajectory:

$$\mathbf{x}(t) = 5 \cdot \begin{pmatrix} 2\cos(t) \\ \sin(2t) \end{pmatrix}$$



Initial conditions: $\mathbf{x}(0) \in \begin{bmatrix} 9.8, 10.2 \\ -0.2, 0.2 \end{bmatrix}$, $\dot{\mathbf{x}}(0) = \begin{pmatrix} 0 \\ 10 \end{pmatrix}$

Simulation of Mobile Robots

Lissajous example: defining constraints

Defining a set of constraints based on the trajectory equation:

$$\mathbf{x}(t) = \begin{pmatrix} 10\cos(t) \\ 5\sin(2t) \end{pmatrix}$$

Simulation of Mobile Robots

Lissajous example: defining constraints

Defining a set of constraints based on the trajectory equation:

$$\mathbf{x}(t) = \begin{pmatrix} 10\cos(t) \\ 5\sin(2t) \end{pmatrix}$$

1. Calculating derivatives:
   - $\dot{\mathbf{x}}(t) = \begin{pmatrix} -10\sin(t) \\ 10\cos(2t) \end{pmatrix}$
   - $\ddot{\mathbf{x}}(t) = \begin{pmatrix} -10\cos(t) \\ -20\sin(2t) \end{pmatrix}$

Simulation of Mobile Robots

Lissajous example: defining constraints

Defining a set of constraints based on the trajectory equation:

$$\mathbf{x}(t) = \begin{pmatrix} 10\cos(t) \\ 5\sin(2t) \end{pmatrix}$$

1. Calculating derivatives:
   - $\dot{\mathbf{x}}(t) = \begin{pmatrix} -10\sin(t) \\ 10\cos(2t) \end{pmatrix}$
   - $\ddot{\mathbf{x}}(t) = \begin{pmatrix} -10\cos(t) \\ -20\sin(2t) \end{pmatrix}$
2. Inter-variable relations:
   - $\ddot{x}_2(t) = -20\sin(t)$

Simulation of Mobile Robots

Lissajous example: defining constraints

Defining a set of constraints based on the trajectory equation:

$$\mathbf{x}(t) = \begin{pmatrix} 10\cos(t) \\ 5\sin(2t) \end{pmatrix}$$

1. Calculating derivatives:
   - $\dot{\mathbf{x}}(t) = \begin{pmatrix} -10\sin(t) \\ 10\cos(2t) \end{pmatrix}$
   - $\ddot{\mathbf{x}}(t) = \begin{pmatrix} -10\cos(t) \\ -20\sin(2t) \end{pmatrix}$

2. Inter-variable relations:
   - $\ddot{x}_2(t) = -20\sin(t)$
   - $\ddot{x}_2(t) = -20 \cdot 2 \cdot \sin(t) \cdot \cos(t)$
   - $\ddot{x}_2(t) = -40 \cdot \frac{\dot{x}_1(t)}{-10} \cdot \frac{\ddot{x}_1(t)}{-10}$
   - $\ddot{x}_2(t) = -0.4 \cdot \dot{x}_1(t) \cdot \ddot{x}_1(t)$

Simulation of Mobile Robots

Lissajous example: defining constraints

Defining a set of constraints based on the trajectory equation:

$$\mathbf{x}(t) = \begin{pmatrix} 10\cos(t) \\ 5\sin(2t) \end{pmatrix}$$

1. Calculating derivatives:
   - $\dot{\mathbf{x}}(t) = \begin{pmatrix} -10\sin(t) \\ 10\cos(2t) \end{pmatrix}$
   - $\ddot{\mathbf{x}}(t) = \begin{pmatrix} -10\cos(t) \\ -20\sin(2t) \end{pmatrix}$
2. Inter-variable relations:
   - $\ddot{x}_2(t) = -20\sin(t)$
   - $\ddot{x}_2(t) = -20 \cdot 2 \cdot \sin(t) \cdot \cos(t)$
   - $\ddot{x}_2(t) = -40 \cdot \frac{\dot{x}_1(t)}{-10} \cdot \frac{\ddot{x}_1(t)}{-10}$
   - $\ddot{x}_2(t) = -0.4 \cdot \dot{x}_1(t) \cdot \ddot{x}_1(t)$

Selected constraint:

$$\ddot{\mathbf{x}}(t) = \begin{pmatrix} -10\cos(t) \\ -0.4 \cdot \dot{x}_1 \cdot \ddot{x}_1 \end{pmatrix}$$

Simulation of Mobile Robots

Lissajous example: CSP

The following CSP will be used for state estimation:

$$\begin{cases} \textbf{Variables: } \mathbf{x}(\cdot),\ \dot{\mathbf{x}}(\cdot),\ \ddot{\mathbf{x}}(\cdot) \\ \textbf{Constraints:} \\ \quad - \quad \ddot{x}_1(t) = -10\cos(t) \\ \quad - \quad \ddot{x}_2(t) = -0.4 \cdot \dot{x}_1(t) \cdot \ddot{x}_1(t) \\ \quad - \quad \mathbf{x}(0) \in \begin{bmatrix} 9.8, 10.2 \\ -0.2, 0.2 \end{bmatrix},\ \dot{\mathbf{x}}(0) = \begin{pmatrix} 0 \\ 10 \end{pmatrix} \\ \textbf{Domains: } [\mathbf{x}](\cdot),\ [\dot{\mathbf{x}}](\cdot),\ [\ddot{\mathbf{x}}](\cdot) \end{cases}$$

Domains (tubes) are initialized to $[-\infty, \infty]\ \forall t$.

Simulation of Mobile Robots

Lissajous example: CSP

The following CSP will be used for state estimation:

$$\begin{cases} \textbf{Variables: } \mathbf{x}(\cdot),\ \dot{\mathbf{x}}(\cdot),\ \ddot{\mathbf{x}}(\cdot) \\ \textbf{Constraints:} \\ \quad - \quad \ddot{x}_1(t) \in -10\cos(t) + [-0.001, 0.001] \\ \quad - \quad \ddot{x}_2(t) = -0.4 \cdot \dot{x}_1(t) \cdot \ddot{x}_1(t) \\ \quad - \quad \mathbf{x}(0) \in \begin{bmatrix} 9.8, 10.2 \\ -0.2, 0.2 \end{bmatrix},\ \dot{\mathbf{x}}(0) = \begin{pmatrix} 0 \\ 10 \end{pmatrix} \\ \textbf{Domains: } [\mathbf{x}](\cdot),\ [\dot{\mathbf{x}}](\cdot),\ [\ddot{\mathbf{x}}](\cdot) \end{cases}$$

Domains (tubes) are initialized to $[-\infty, \infty]\ \forall t$.

Simulation of Mobile Robots

Lissajous example: computations

Variables: $\mathbf{x}(\cdot)$, $\dot{\mathbf{x}}(\cdot)$, $\ddot{\mathbf{x}}(\cdot)$

Constraints:

- $\ddot{x}_1(t) \in -10\cos(t) + [-0.001, 0.001]$
- $\ddot{x}_2(t) = -0.4 \cdot \dot{x}_1(t) \cdot \ddot{x}_1(t)$
- $\mathbf{x}(0) \in \begin{bmatrix} 9.8, 10.2 \\ -0.2, 0.2 \end{bmatrix}$, $\dot{\mathbf{x}}(0) = \begin{pmatrix} 0 \\ 10 \end{pmatrix}$

Domains: $[\mathbf{x}](\cdot)$, $[\dot{\mathbf{x}}](\cdot)$, $[\ddot{\mathbf{x}}](\cdot)$

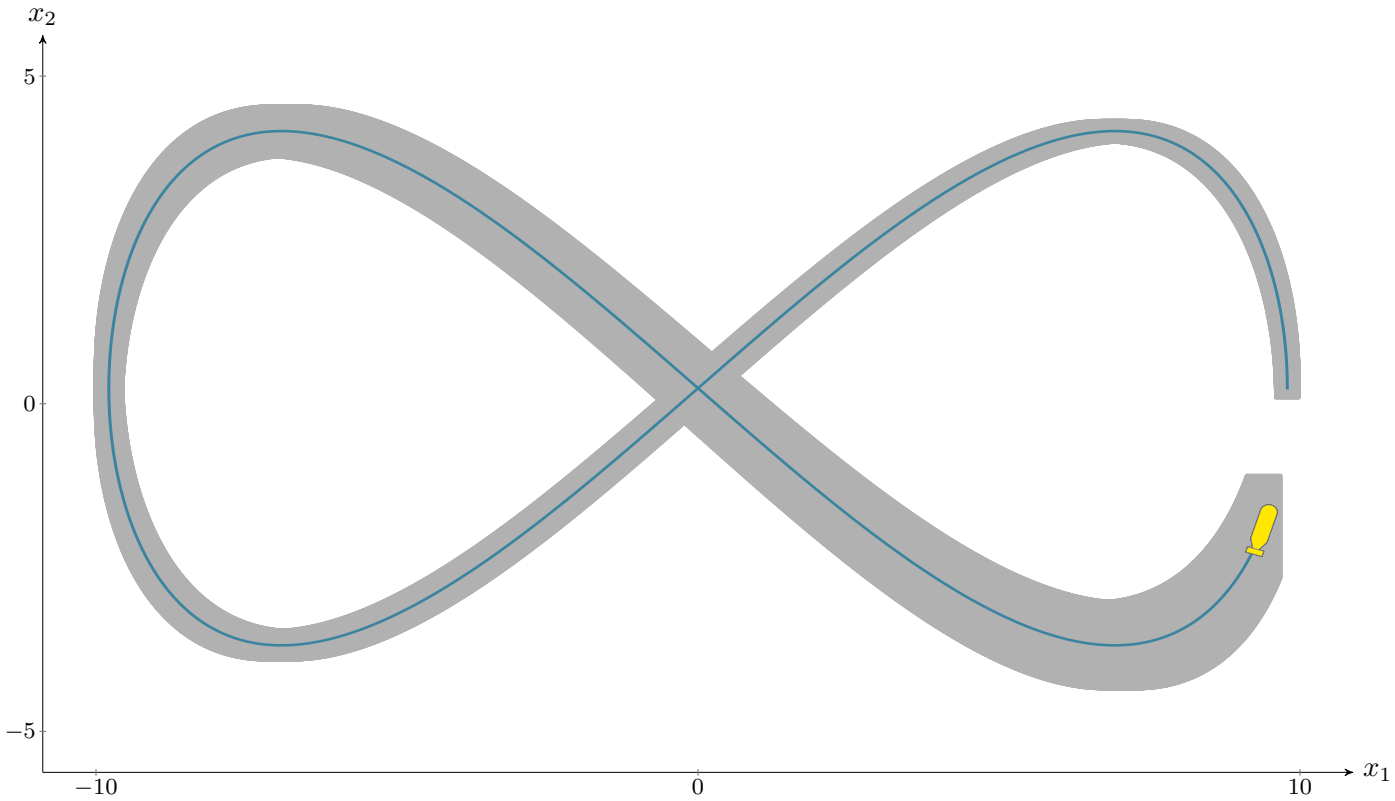


Figure: Set-membership state estimation of a Lissajous robot

Simulation of Mobile Robots

Lissajous example: computations

Other kind of constraints are easily added, such as periodic constraints: $x_2(t) = x_2(t + \pi)$

Variables: $\mathbf{x}(\cdot)$, $\dot{\mathbf{x}}(\cdot)$, $\ddot{\mathbf{x}}(\cdot)$

Constraints:

- $\ddot{x}_1(t) \in -10\cos(t) + [-0.001, 0.001]$
- $\ddot{x}_2(t) = -0.4 \cdot \dot{x}_1(t) \cdot \ddot{x}_1(t)$
- $x_2(t) = x_2(t + \pi)$
- $\mathbf{x}(0) \in \begin{bmatrix} 9.8, 10.2 \\ -0.2, 0.2 \end{bmatrix}$, $\dot{\mathbf{x}}(0) = \begin{pmatrix} 0 \\ 10 \end{pmatrix}$

Domains: $[\mathbf{x}](\cdot)$, $[\dot{\mathbf{x}}](\cdot)$, $[\ddot{\mathbf{x}}](\cdot)$

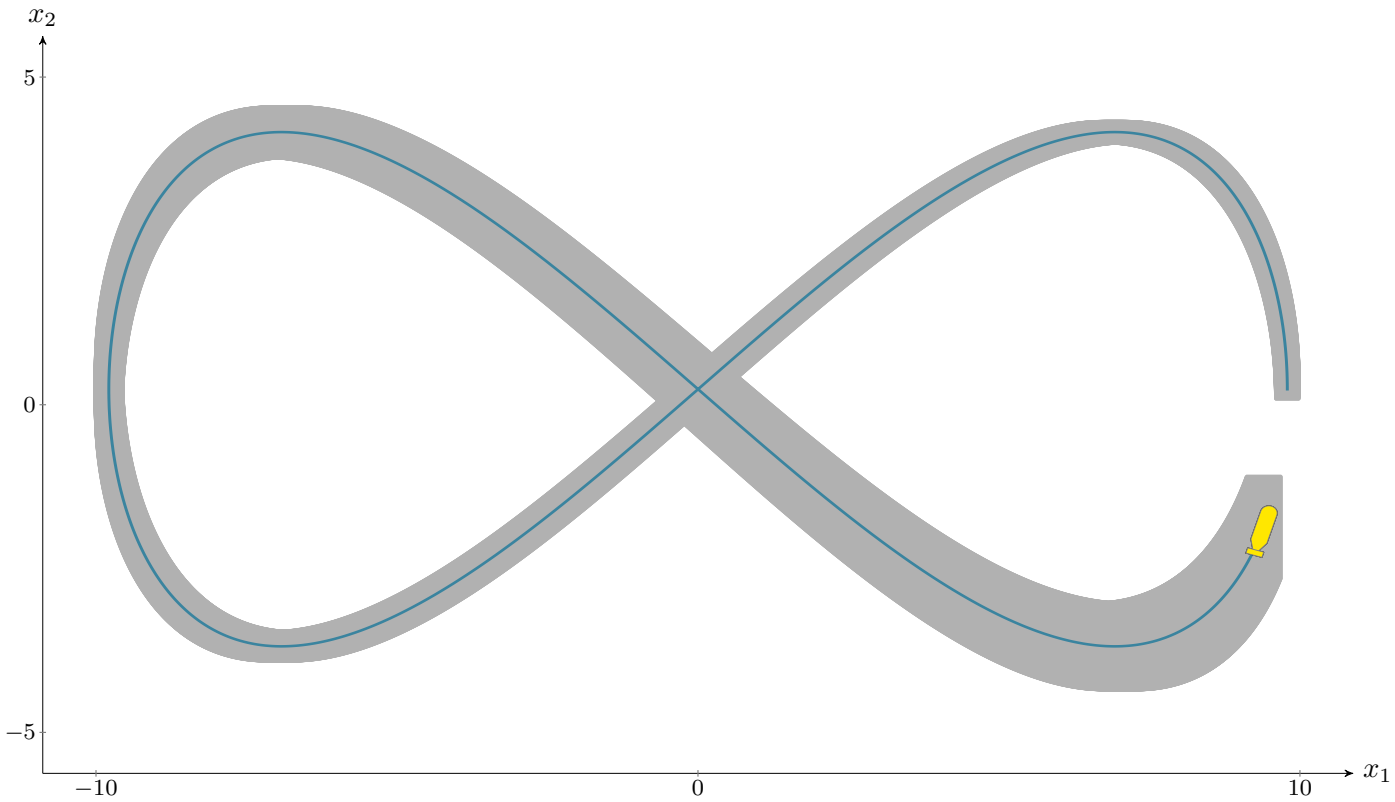


Figure: Set-membership state estimation of a Lissajous robot

Simulation of Mobile Robots

Lissajous example: computations

Other kind of constraints are easily added, such as periodic constraints: $x_2(t) = x_2(t + \pi)$

Variables: $\mathbf{x}(\cdot)$, $\dot{\mathbf{x}}(\cdot)$, $\ddot{\mathbf{x}}(\cdot)$

Constraints:

- $\ddot{x}_1(t) \in -10\cos(t) + [-0.001, 0.001]$
- $\ddot{x}_2(t) = -0.4 \cdot \dot{x}_1(t) \cdot \ddot{x}_1(t)$
- $x_2(t) = x_2(t + \pi)$
- $\mathbf{x}(0) \in \begin{bmatrix} 9.8, 10.2 \\ -0.2, 0.2 \end{bmatrix}$, $\dot{\mathbf{x}}(0) = \begin{pmatrix} 0 \\ 10 \end{pmatrix}$

Domains: $[\mathbf{x}](\cdot)$, $[\dot{\mathbf{x}}](\cdot)$, $[\ddot{\mathbf{x}}](\cdot)$

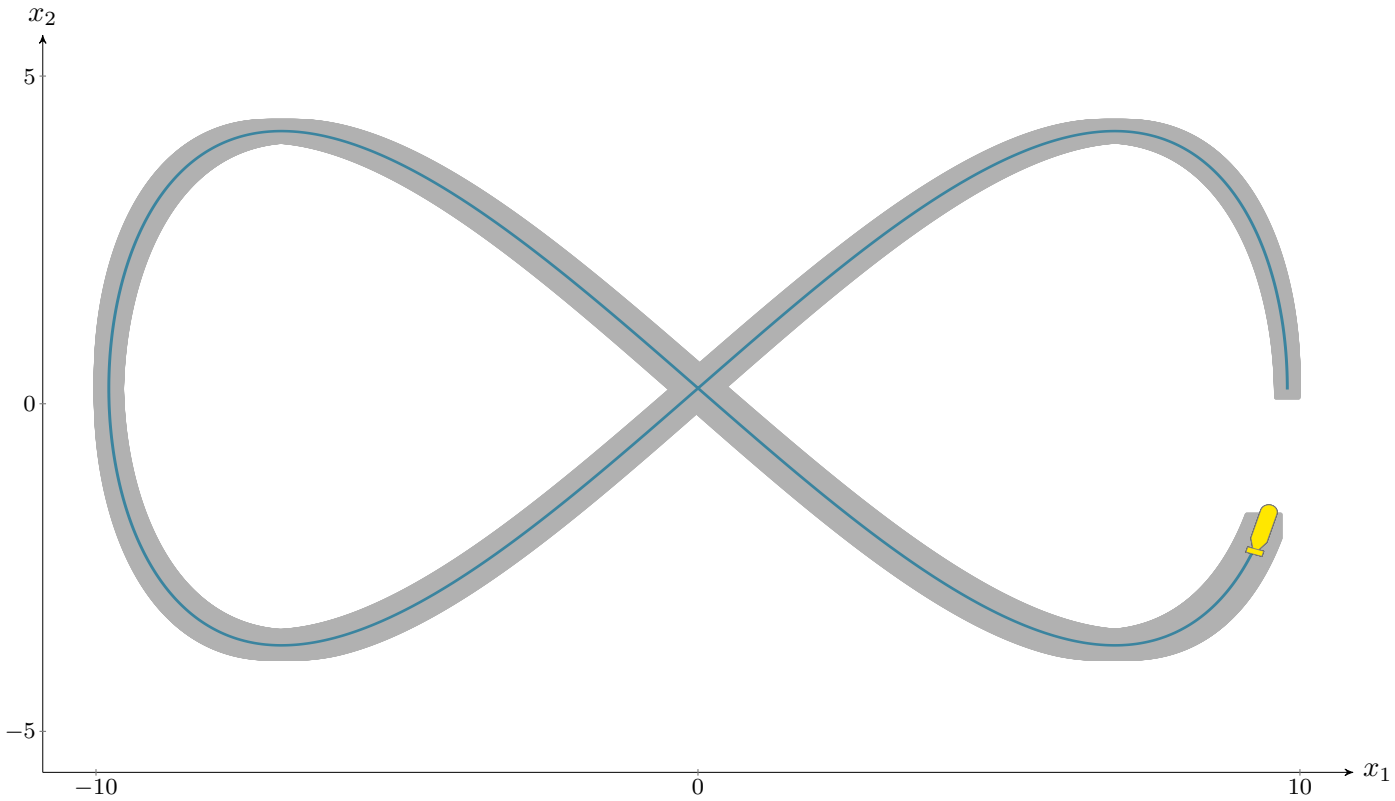


Figure: Set-membership state estimation of a Lissajous robot

Section 6

Application on Real Data-Sets

Application on Real Data-Sets

State estimation of a real underwater robot

Figure: DAURADE AUV managed by DGA Techniques Navales Brest and the Service Hydrographique et Océanographique de la Marine, during an experiment in the Rade de Brest, October 2015.

Application on Real Data-Sets

State estimation of a real underwater robot

Figure: DAURADE AUV managed by DGA Techniques Navales Brest and the Service Hydrographique et Océanographique de la Marine, during an experiment in the Rade de Brest, October 2015.

Application on Real Data-Sets

State estimation of a real underwater robot

Classical kinematic model of an underwater robot:

$$\begin{cases} \dot{\mathbf{p}} &= \mathbf{R}(\psi,\theta,\varphi) \cdot \mathbf{v}_r \\ \dot{\mathbf{v}}_r &= \mathbf{a}_r - \boldsymbol{\omega}_r \wedge \mathbf{v}_r \end{cases}$$

Where:

- $\mathcal{R}_0$ is the absolute inertial coordinate system
- $\mathcal{R}_1$ is robot's own coordinate system
- (ψ,θ,φ) and $\mathbf{R}(\psi,\theta,\varphi)$ are Euler angles and matrix
- $\mathbf{p} = (p_x, p_y, p_z)$ gives the center of the robot in $\mathcal{R}_0$
- $\mathbf{v}_r$ is the speed vector in $\mathcal{R}_1$
- $\mathbf{a}_r$ is the acceleration vector in $\mathcal{R}_1$
- $\boldsymbol{\omega}_r = (\omega_x, \omega_y, \omega_z)$ is the rotation vector of the robot relative to $\mathcal{R}_0$ expressed in $\mathcal{R}_1$

Application on Real Data-Sets

State estimation of a real underwater robot

Classical kinematic model of an underwater robot:

$$\begin{cases} \dot{\mathbf{p}} &= \mathbf{R}(\psi,\theta,\varphi) \cdot \mathbf{v}_r \\ \dot{\mathbf{v}}_r &= \mathbf{a}_r - \boldsymbol{\omega}_r \wedge \mathbf{v}_r \end{cases}$$

Sensed values:
Speed vector $\mathbf{v}_r$, acceleration vector $\mathbf{a}_r$, Euler angles (ψ,θ,φ)

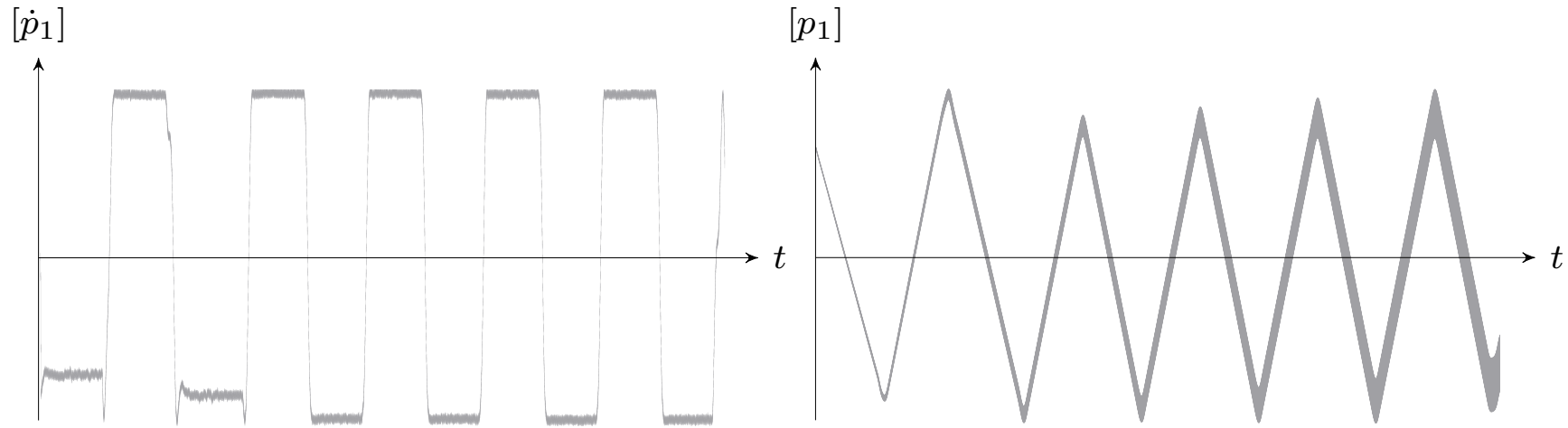
Corresponding tubes:

- feeded with sensor data:
  $[\mathbf{v}_r](\cdot)$, $[\mathbf{a}_r](\cdot)$, $[\psi](\cdot)$, $[\theta](\cdot)$, $[\varphi](\cdot)$
- to be computed:
  $[\mathbf{p}](\cdot)$

Application on Real Data-Sets

State estimation of a real underwater robot

Daurade's tubes:



- $[\dot{p}_1](\cdot)$ – velocity along x in $\mathcal{R}_0$
  tube's thickness coming from measurements remains constant
- $[p_1](\cdot)$ – position along x in $\mathcal{R}_0$
  tube becomes thicker depicting cumulative errors due to the dead-reckoning method and $[\dot{p}_1](\cdot)$ non-zero thickness

Application on Real Data-Sets

1. Dead-reckoning

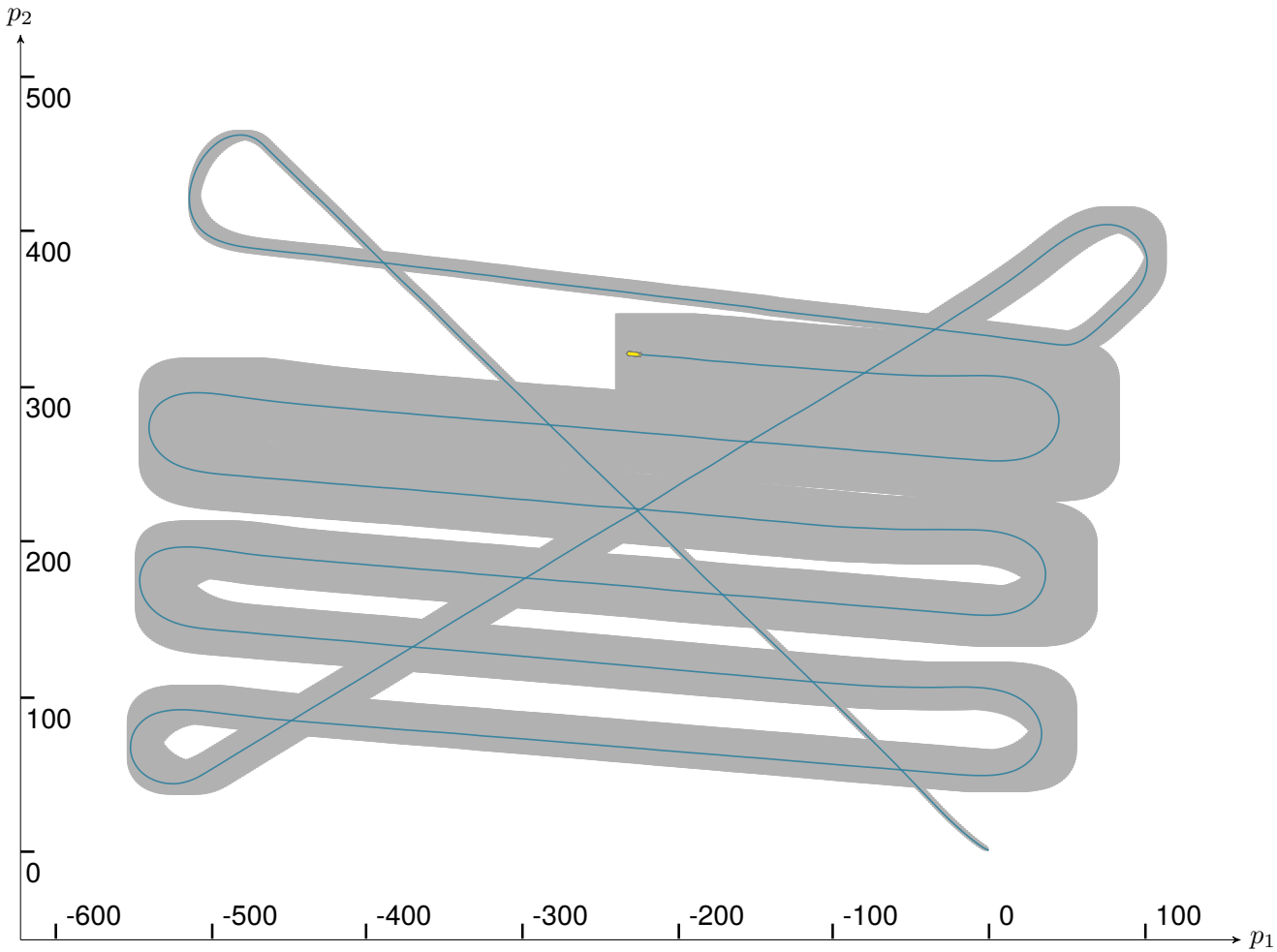


Figure: Mission map: DAURADE exploring a 25 hectares seabed area.

Application on Real Data-Sets

2. Observations (USBL measurements)

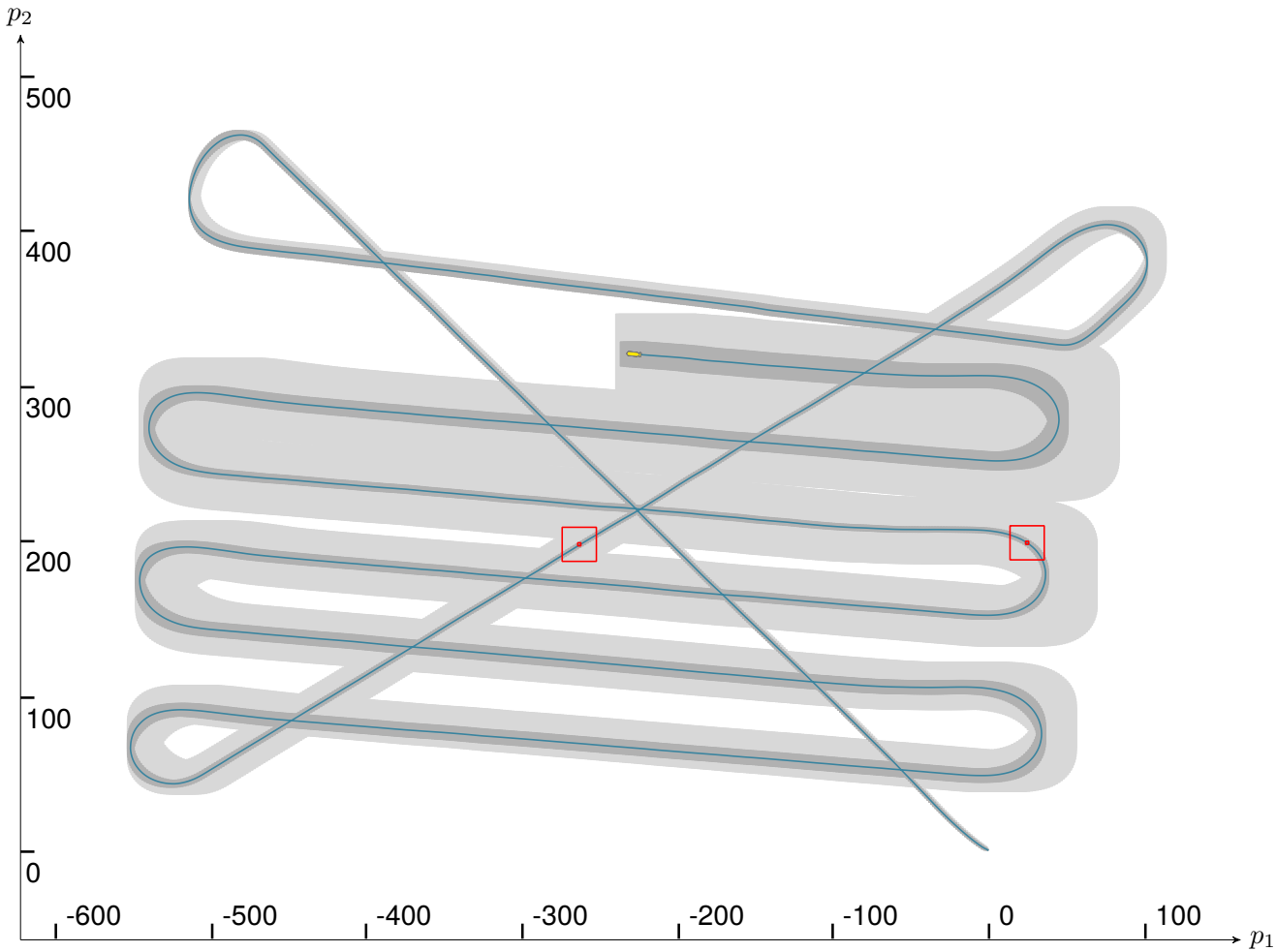


Figure: Mission map: DAURADE exploring a 25 hectares seabed area.

Application on Real Data-Sets

2. Observations (USBL measurements)

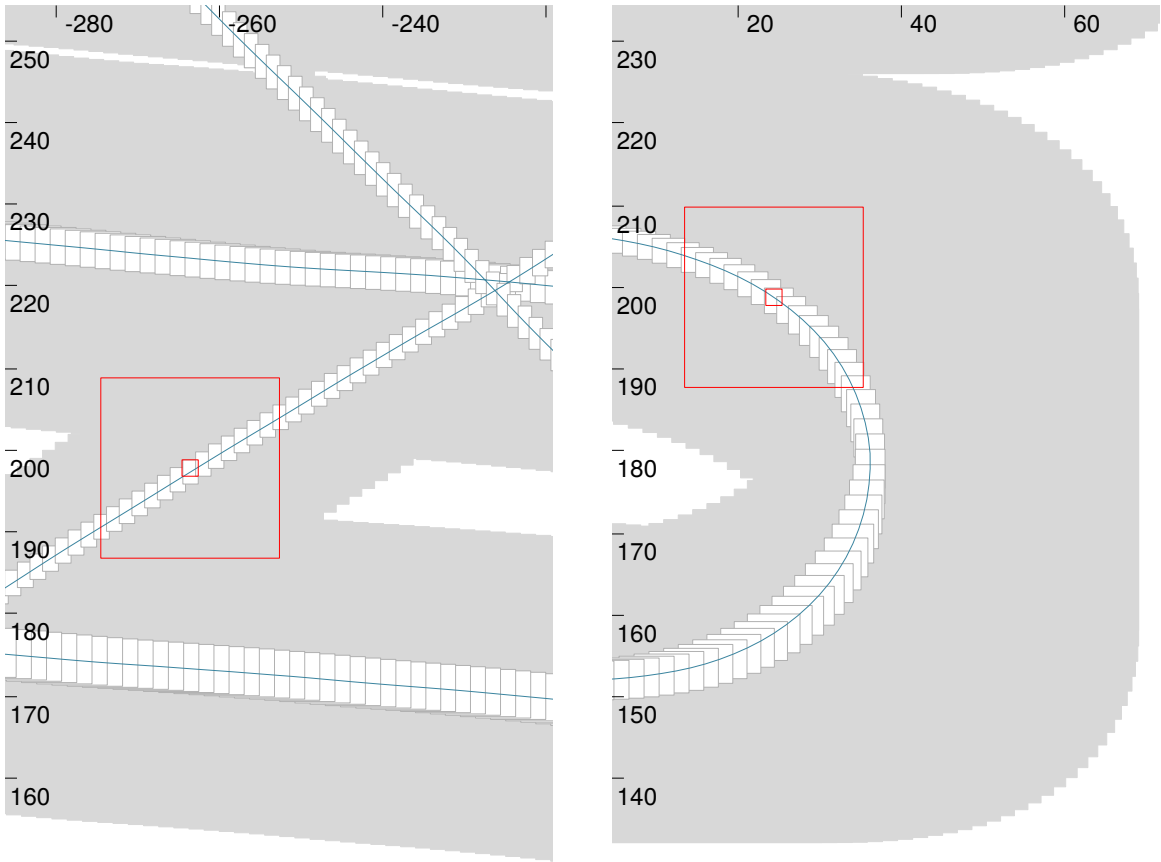


Figure: Zoom on the two state observations: these are bounded positioning measurements obtained at $t_1 = 14\,\mathrm{min}$ and $t_2 = 32\,\mathrm{min}$.

Section 7

Conclusion

Conclusion

Tubes, pros and cons

Tubes and related **contractors** provide a simple and efficient constraint-based method for the computation of **uncertain**, **non-linear** and **dynamical** systems.

Conclusion

Tubes, pros and cons

Tubes and related **contractors** provide a simple and efficient constraint-based method for the computation of **uncertain**, **non-linear** and **dynamical** systems.

Advantages:

- simple problem description by constraints definition
- competitive method for robotics applications
- easily applicable on data-sets
- reliable results

Conclusion

Tubes, pros and cons

Tubes and related **contractors** provide a simple and efficient constraint-based method for the computation of **uncertain**, **non-linear** and **dynamical** systems.

Advantages:

- simple problem description by constraints definition
- competitive method for robotics applications
- easily applicable on data-sets
- reliable results

Drawbacks:

- envelope of solutions only
- pessimism added by implementation
- hybrid systems?
- computation time?

Lab-STICC

Conclusion

PhD thesis

Contractor Programming over Trajectories, application to Underwater Localization

Conclusion

PhD thesis

Contractor Programming over Trajectories, application to Underwater Localization

Theory:

- constraint $\dot{x} = v$, contractor $\mathcal{C}_{\frac{d}{dt}}$

Conclusion

PhD thesis

Contractor Programming over Trajectories, application to Underwater Localization

Theory:

- constraint $\dot{x} = v$, contractor $\mathcal{C}_{\frac{d}{dt}}$
- constraint $x(t) = z$, contractor $\mathcal{C}_{\text{obs}}$

Lab-STICC

Conclusion

PhD thesis

Contractor Programming over Trajectories, application to Underwater Localization

Theory:

- constraint $\dot{x} = v$, contractor $\mathcal{C}_{\frac{d}{dt}}$
- constraint $x(t) = z$, contractor $\mathcal{C}_{\text{obs}}$
- topological degree test (0 verification, Peter Franek)

Conclusion

PhD thesis

Contractor Programming over Trajectories, application to Underwater Localization

Theory:

- constraint $\dot{x} = v$, contractor $\mathcal{C}_{\frac{d}{dt}}$
- constraint $x(t) = z$, contractor $\mathcal{C}_{\text{obs}}$
- topological degree test (0 verification, Peter Franek)

Applications:

- loop based localization for mobile robots

Conclusion

PhD thesis

Contractor Programming over Trajectories, application to Underwater Localization

Theory:

- constraint $\dot{x} = v$, contractor $\mathcal{C}_{\frac{d}{dt}}$
- constraint $x(t) = z$, contractor $\mathcal{C}_{\text{obs}}$
- topological degree test (0 verification, Peter Franek)

Applications:

- loop based localization for mobile robots
- wreck-based localization

Conclusion

PhD thesis

Contractor Programming over Trajectories, application to Underwater Localization

Theory:

- constraint $\dot{x} = v$, contractor $\mathcal{C}_{\frac{d}{dt}}$
- constraint $x(t) = z$, contractor $\mathcal{C}_{\text{obs}}$
- topological degree test (0 verification, Peter Franek)

Applications:

- loop based localization for mobile robots
- wreck-based localization
- robot localization in an unknown but symmetric environment

Lab-STICC

Conclusion

PhD thesis

Contractor Programming over Trajectories, application to Underwater Localization

Theory:

- constraint $\dot{x} = v$, contractor $\mathcal{C}_{\frac{d}{dt}}$
- constraint $x(t) = z$, contractor $\mathcal{C}_{\text{obs}}$
- topological degree test (0 verification, Peter Franek)

Applications:

- loop based localization for mobile robots
- wreck-based localization
- robot localization in an unknown but symmetric environment
- ...

Lab-STICC

Conclusion

PhD thesis

Contractor Programming over Trajectories, application to Underwater Localization

Theory:

- constraint $\dot{x} = v$, contractor $\mathcal{C}_{\frac{d}{dt}}$
- constraint $x(t) = z$, contractor $\mathcal{C}_{\text{obs}}$
- topological degree test (0 verification, Peter Franek)

Applications:

- loop based localization for mobile robots
- wreck-based localization
- robot localization in an unknown but symmetric environment
- ...

Main topics: dynamical syst., time uncertainties, inter-temporalities

Lab-STICC

Conclusion

PhD thesis

3 equations, 3 main contributions:

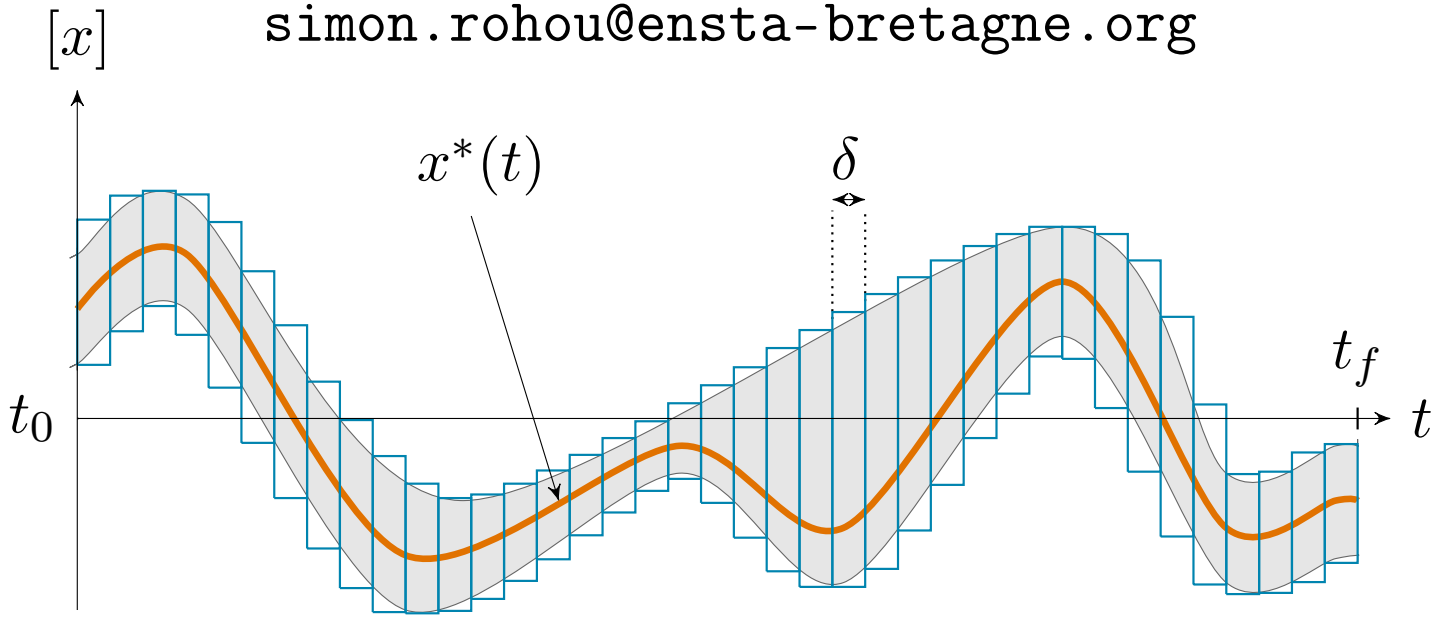
$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) & \text{(navigation)} \\ y(t) = g(\mathbf{x}(t)) & \text{(measurements)} \\ y(t_1) \neq y(t_2) \implies h\left(\mathbf{x}(t_1), \mathbf{x}(t_2)\right) \neq 0 & \text{(inter-temporality)} \end{cases}$$

Applied on real underwater robot localization problems.

Conclusion

Tubes: library

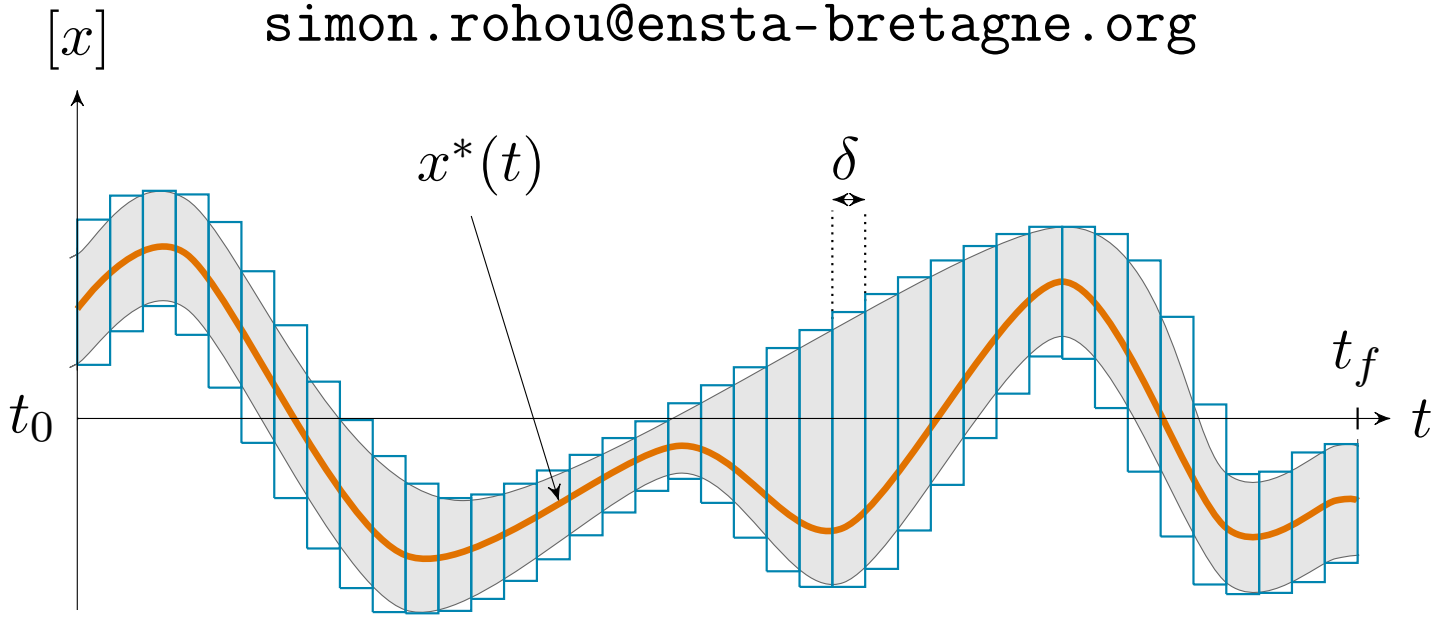
An open-source C++/python tube library soon available.



Conclusion

Tubes: library

An open-source C++/python tube library soon available.

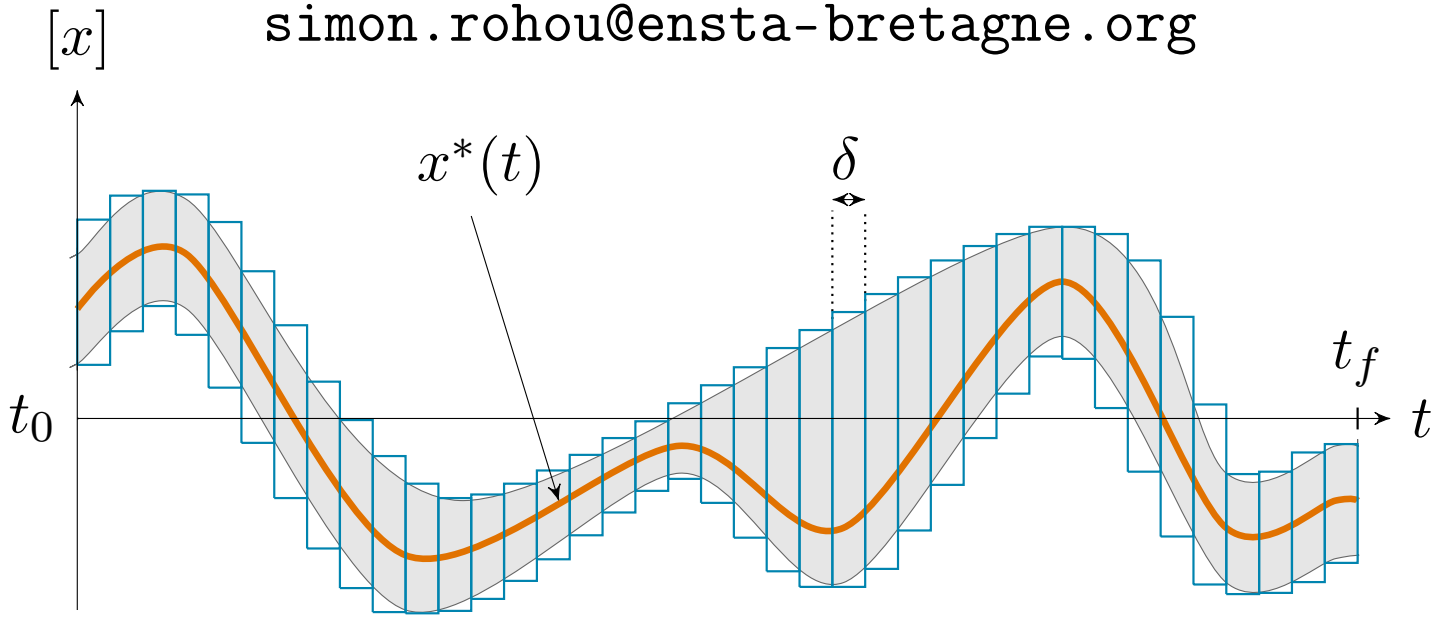


- tube arithmetic, algebraic contractors, integral computations

Conclusion

Tubes: library

An open-source C++/python tube library soon available.

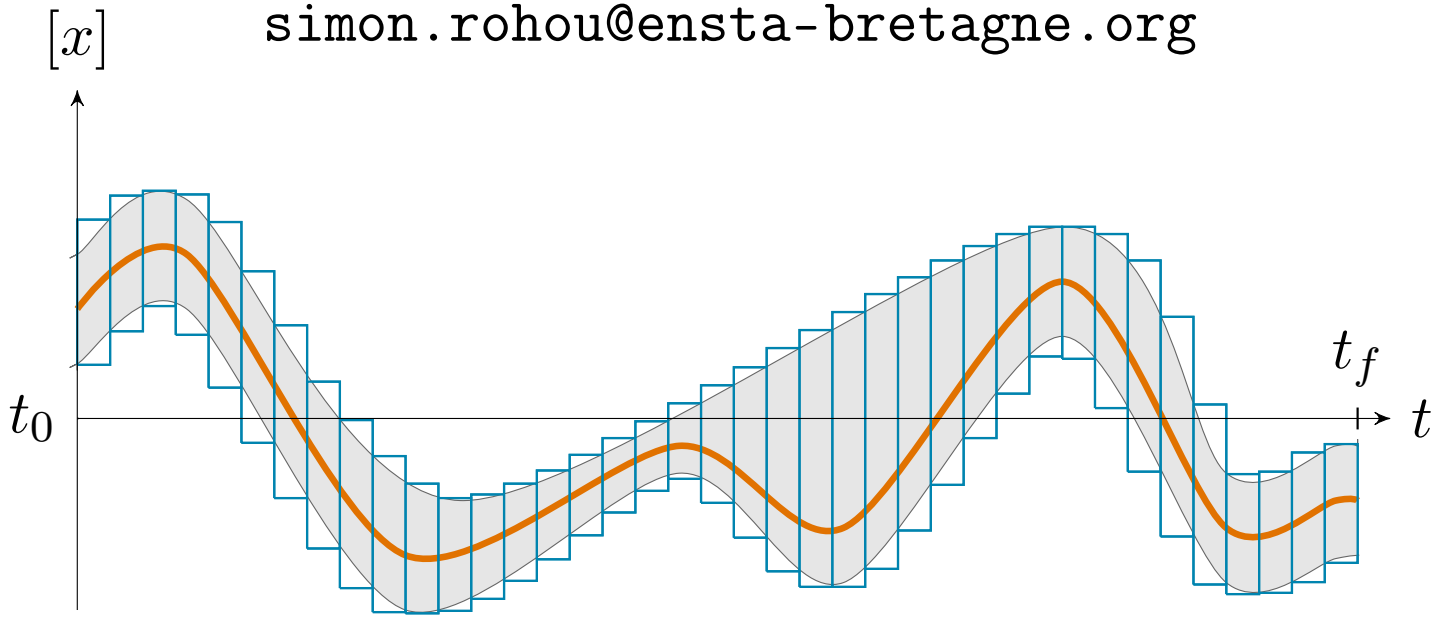


- tube arithmetic, algebraic contractors, integral computations
- contractors $\mathcal{C}_{\frac{d}{dt}}$, $\mathcal{C}_{\text{obs}}$, $\mathcal{C}_{t_1,t_2}$, $\mathcal{C}_{\text{period}}$, ...

Conclusion

Tubes: library

An open-source C++/python tube library soon available.

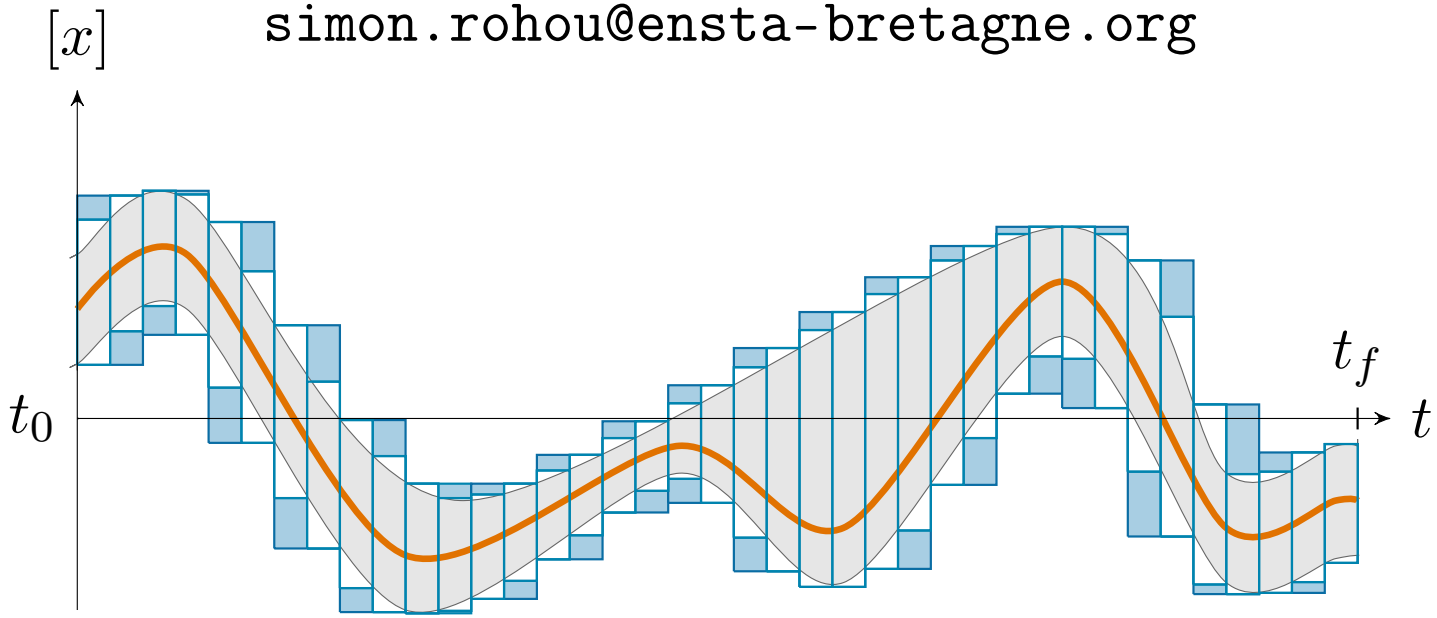


- tube arithmetic, algebraic contractors, integral computations
- contractors $\mathcal{C}_{\frac{d}{dt}}$, $\mathcal{C}_{\text{obs}}$, $\mathcal{C}_{t_1,t_2}$, $\mathcal{C}_{\text{period}}$, ...
- binary tree structure

Conclusion

Tubes: library

An open-source C++/python tube library soon available.

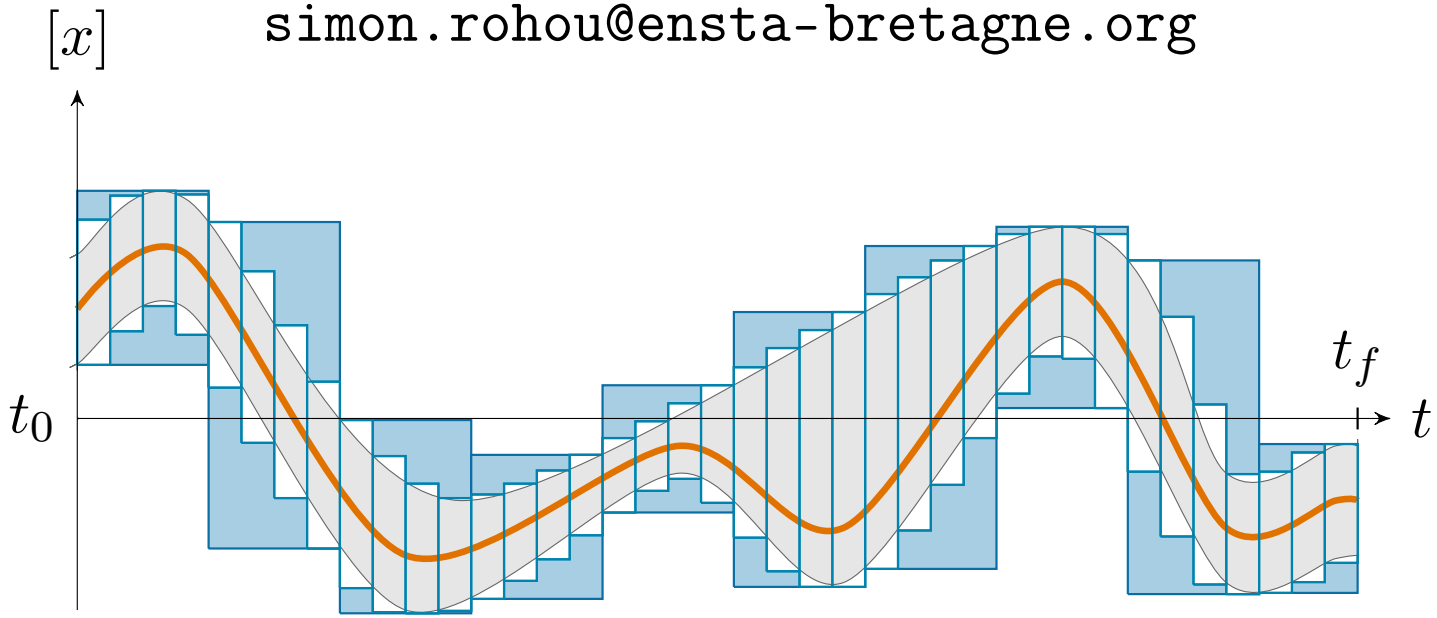


- tube arithmetic, algebraic contractors, integral computations
- contractors $\mathcal{C}_{\frac{d}{dt}}$, $\mathcal{C}_{\text{obs}}$, $\mathcal{C}_{t_1,t_2}$, $\mathcal{C}_{\text{period}}$, ...
- binary tree structure

Conclusion

Tubes: library

An open-source C++/python tube library soon available.

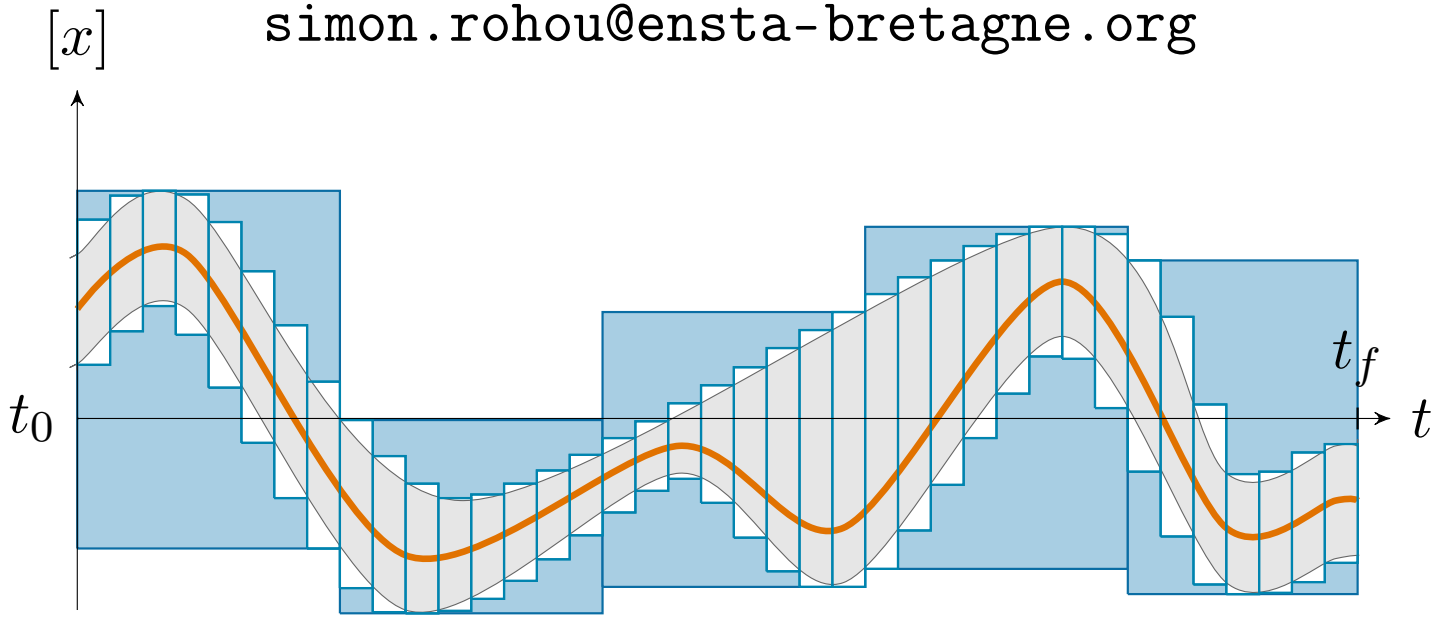


- tube arithmetic, algebraic contractors, integral computations
- contractors $\mathcal{C}_{\frac{d}{dt}}$, $\mathcal{C}_{\text{obs}}$, $\mathcal{C}_{t_1,t_2}$, $\mathcal{C}_{\text{period}}$, ...
- binary tree structure

Conclusion

Tubes: library

An open-source C++/python tube library soon available.

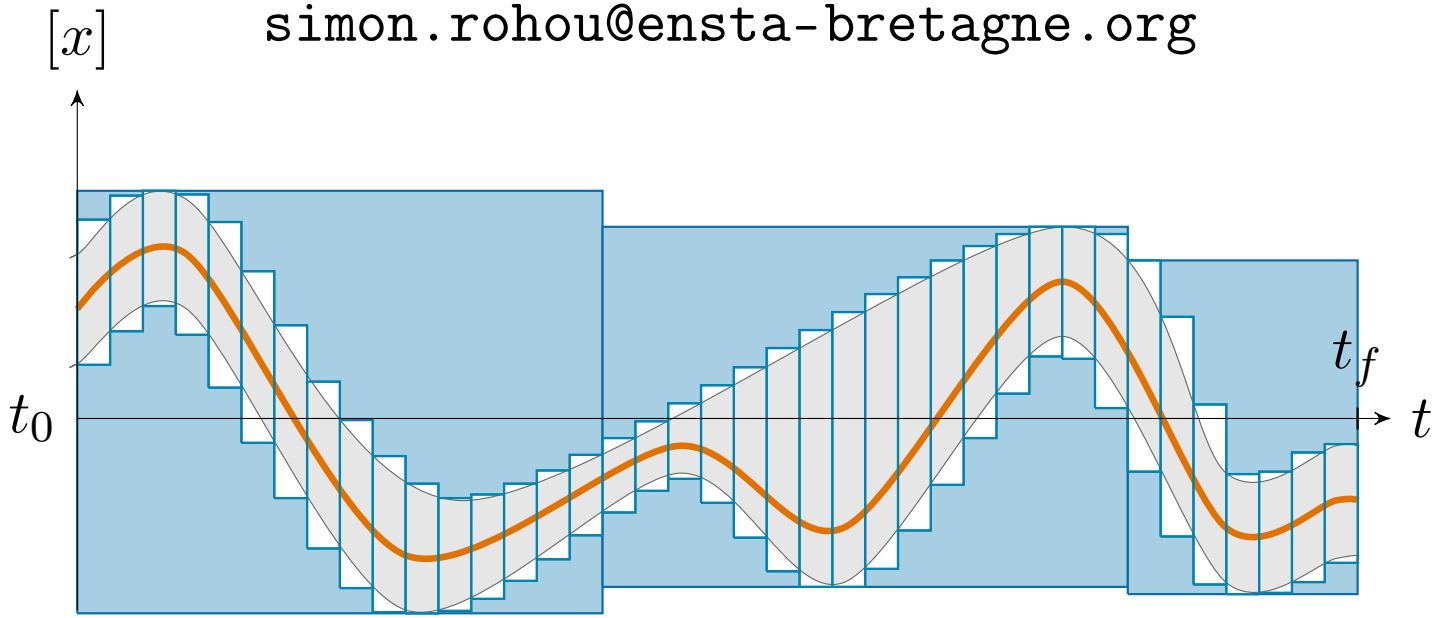


- tube arithmetic, algebraic contractors, integral computations
- contractors $\mathcal{C}_{\frac{d}{dt}}$, $\mathcal{C}_{\text{obs}}$, $\mathcal{C}_{t_1,t_2}$, $\mathcal{C}_{\text{period}}$, ...
- binary tree structure

Conclusion

Tubes: library

An open-source C++/python tube library soon available.

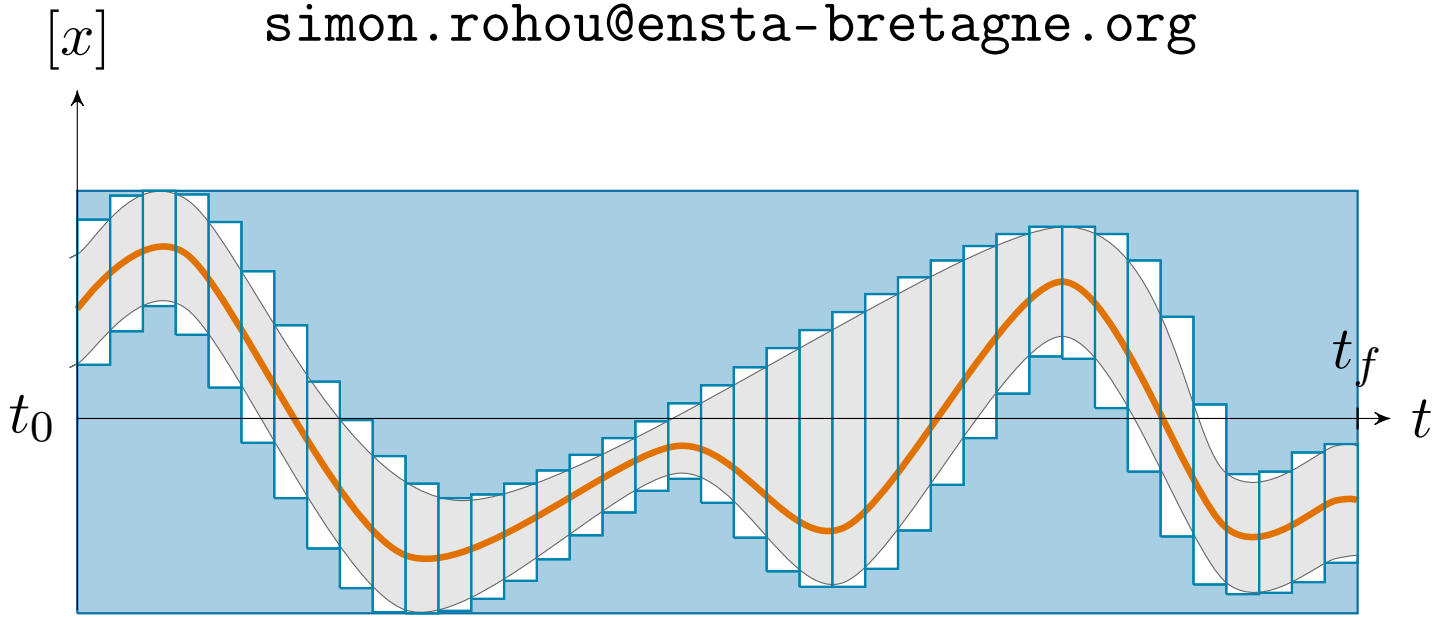


- tube arithmetic, algebraic contractors, integral computations
- contractors $\mathcal{C}_{\frac{d}{dt}}$, $\mathcal{C}_{\text{obs}}$, $\mathcal{C}_{t_1,t_2}$, $\mathcal{C}_{\text{period}}$, ...
- binary tree structure

Conclusion

Tubes: library

An open-source C++/python tube library soon available.

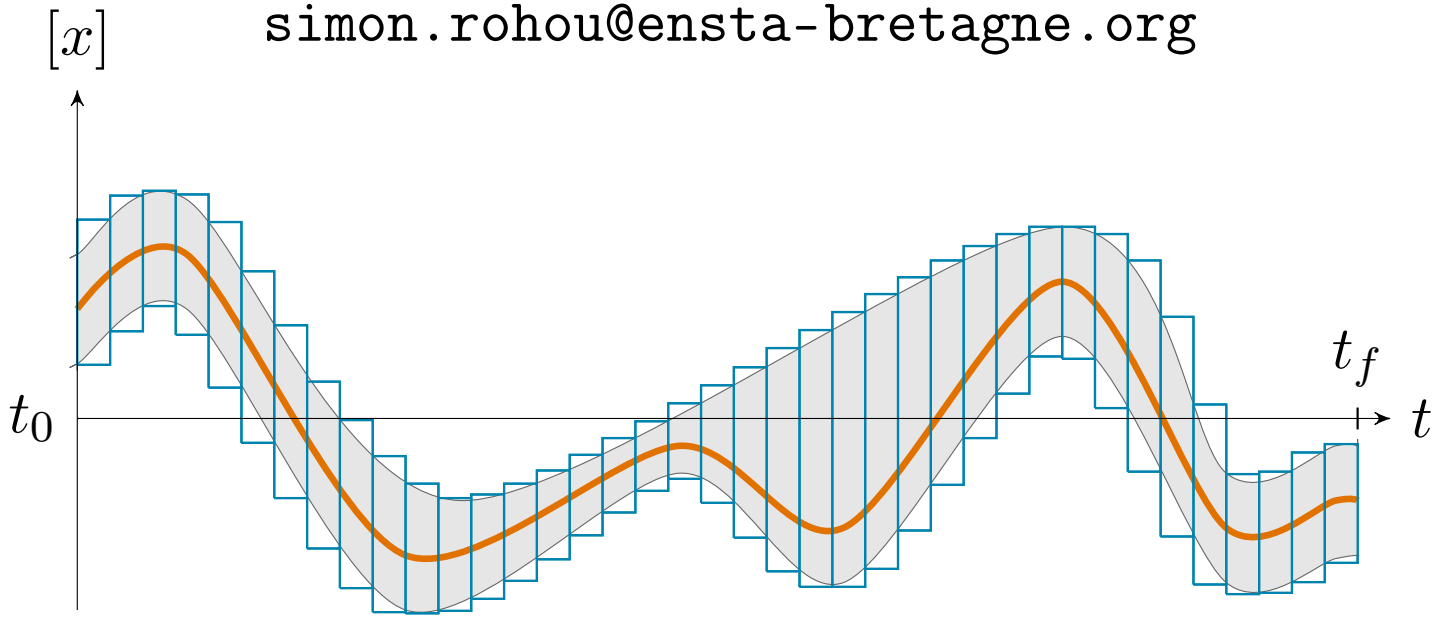


- tube arithmetic, algebraic contractors, integral computations
- contractors $\mathcal{C}_{\frac{d}{dt}}$, $\mathcal{C}_{\text{obs}}$, $\mathcal{C}_{t_1,t_2}$, $\mathcal{C}_{\text{period}}$, ...
- binary tree structure

Conclusion

Tubes: library

An open-source C++/python tube library soon available.

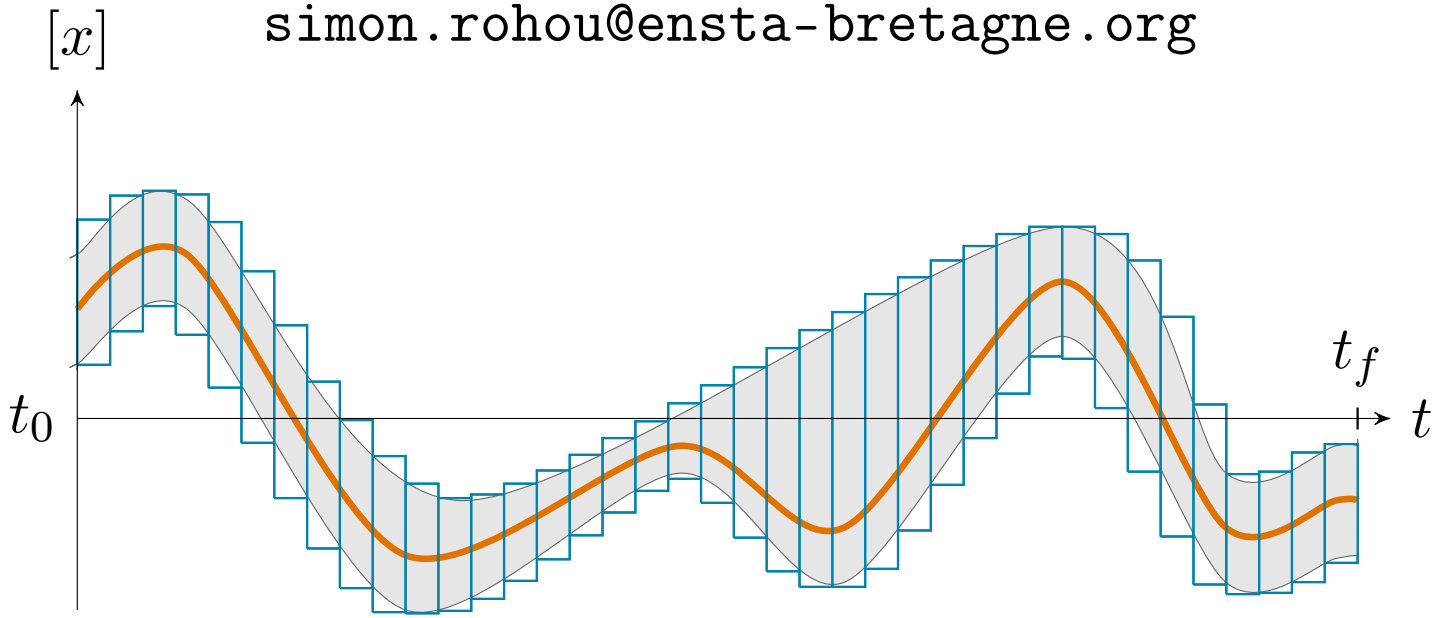


- tube arithmetic, algebraic contractors, integral computations
- contractors $\mathcal{C}_{\frac{d}{dt}}$, $\mathcal{C}_{\text{obs}}$, $\mathcal{C}_{t_1,t_2}$, $\mathcal{C}_{\text{period}}$, ...
- binary tree structure
- utilities: graphical library, serialization, ...

Conclusion

Tubes: library

An open-source C++/python tube library soon available.



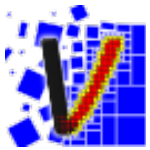
- tube arithmetic, algebraic contractors, integral computations
- contractors $\mathcal{C}_{\frac{d}{dt}}$, $\mathcal{C}_{\text{obs}}$, $\mathcal{C}_{t_1,t_2}$, $\mathcal{C}_{\text{period}}$, ...
- binary tree structure
- utilities: graphical library, serialization, ...
- examples, documentation, unit tests

Support:

DGA
Direction Générale de l'Armement

Tools:

IBEX library
used for interval arithmetic, contractor programming



VIBES
used for rendering

References:

[Mackworth1977] A. K. Mackworth,
Consistency in networks of relations,
Artificial Intelligence, 8(1):99–118, 1977.

[Chabert2009] G. Chabert, L. Jaulin,
Contractor Programming,
Artificial Intelligence, 173:1079–1100, 2009.

[LeBars2012] F. Le Bars, J. Sliwka, O. Reynet, L. Jaulin,
State estimation with fleeting data,
Automatica, 48(2):381–387, 2012.

[Bethencourt2014] A. Bethencourt, L. Jaulin,
Solving non-linear constraint satisfaction problems involving time-dependant functions,
Mathematics in Computer Science, 8(3), 2014.

[Aubry2013] C. Aubry, R. Desmare, L. Jaulin,
Loop detection of mobile robots using interval analysis,
Automatica, 2013.

[tubint] S. Rohou, L. Jaulin, L. Mihaylova, F. Le Bars, S. M. Veres,
Guaranteed Computation of Robot Trajectories,
Robotics and Autonomous Systems, to be published.

[tubobs] S. Rohou, L. Jaulin, L. Mihaylova, F. Le Bars, S. M. Veres,
Reliable Non-Linear State Estimation involving Time Uncertainties,
to be submitted.

Section 9

Appendix

Appendix

Tubes: implementation

The computer representation of a tube encloses $[x^-(\cdot), x^+(\cdot)]$ inside an interval of step functions $[\underline{x^-}(\cdot), \overline{x^+}(\cdot)]$ such that:

$$\forall t \in \mathbb{R},\ \underline{x^-}(t) \leqslant x^-(t) \leqslant x^+(t) \leqslant \overline{x^+}(t)$$

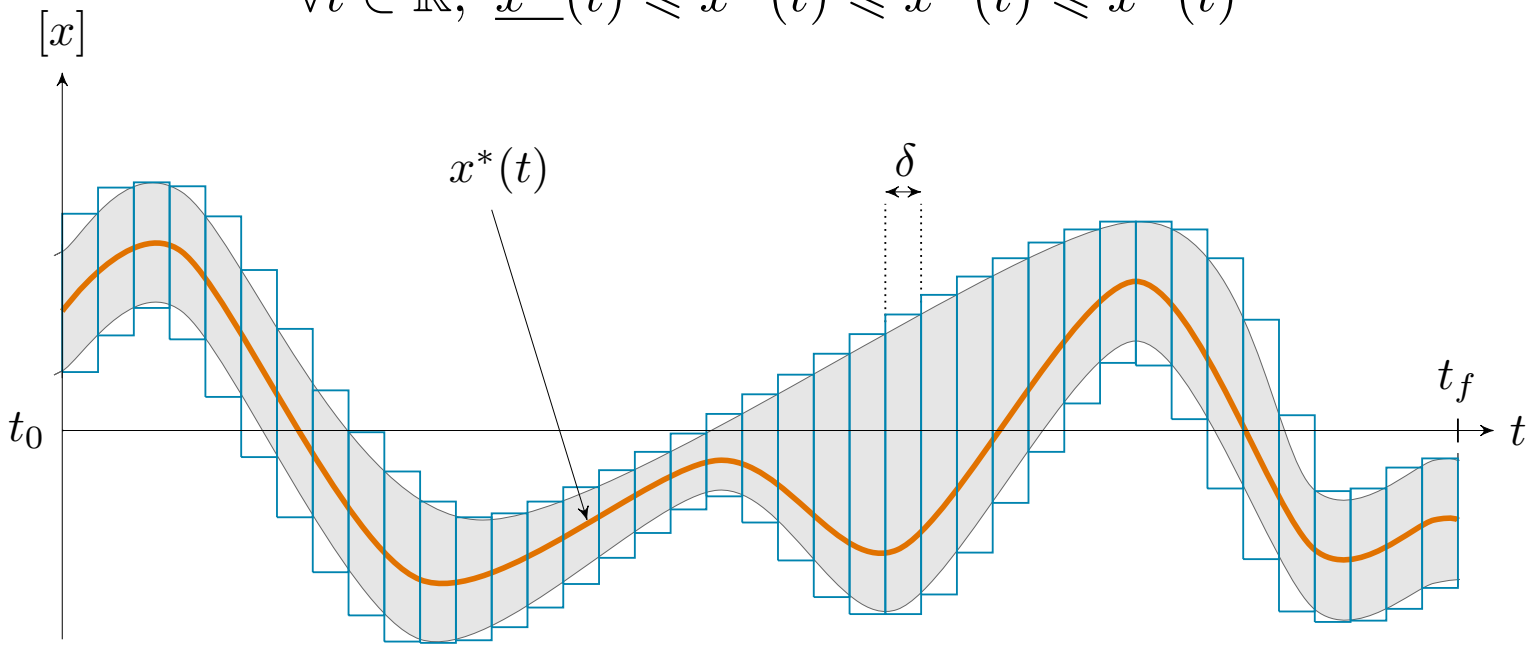


Figure: tube implementation with a set of boxes – this outer representation adds pessimism but enables guaranteed and simple computations

Appendix

Definition

A contractor applied on a tube $[a](\cdot)$ aims at removing unfeasible trajectories according to a given constraint $\mathcal{L}$:

[Bethencourt2014]

$$[a](\cdot) \xrightarrow{\mathcal{C}_{\mathcal{L}}} [b](\cdot)$$

Appendix

Definition

A contractor applied on a tube $[a](\cdot)$ aims at removing unfeasible trajectories according to a given constraint $\mathcal{L}$:

[Bethencourt2014]

$$[a](\cdot) \xrightarrow{\mathcal{C}_{\mathcal{L}}} [b](\cdot)$$

The output of the contractor $\mathcal{C}_{\mathcal{L}}$ is the tube $[b](\cdot)$ such that:

$$\forall t,\ [b](t) \subseteq [a](t) \qquad \text{(contraction)}$$

$$\begin{pmatrix} \mathcal{L}\big(a(\cdot)\big) \\ a(\cdot) \in [a](\cdot) \end{pmatrix} \Longrightarrow a(\cdot) \in [b](\cdot) \qquad \text{(completeness)}$$

Appendix

Tubes: integral and its implementation

Implementation: an outer approximation of the integral is computed

$$\int_a^b [x](\tau)d\tau \subset \left[\int_a^b \underline{x^-}(\tau)d\tau, \int_a^b \overline{x^+}(\tau)d\tau\right]$$

[Aubry2013]

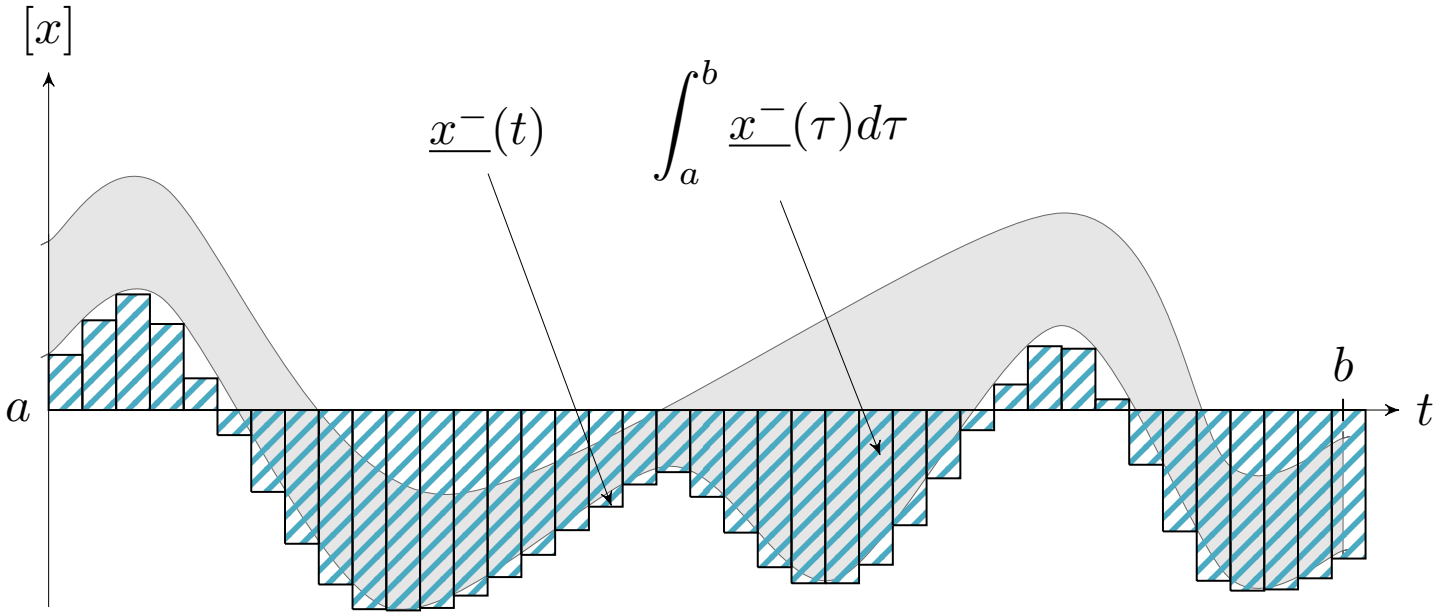


Figure: blue area: outer approximation of the lower bound of the tube's integral

Appendix

Tubes: integral and its implementation

Implementation: an outer approximation of the integral is computed

$$\int_a^b [x](\tau)d\tau \subset \left[\int_a^b \underline{x^-}(\tau)d\tau, \int_a^b \overline{x^+}(\tau)d\tau\right]$$

[Aubry2013]

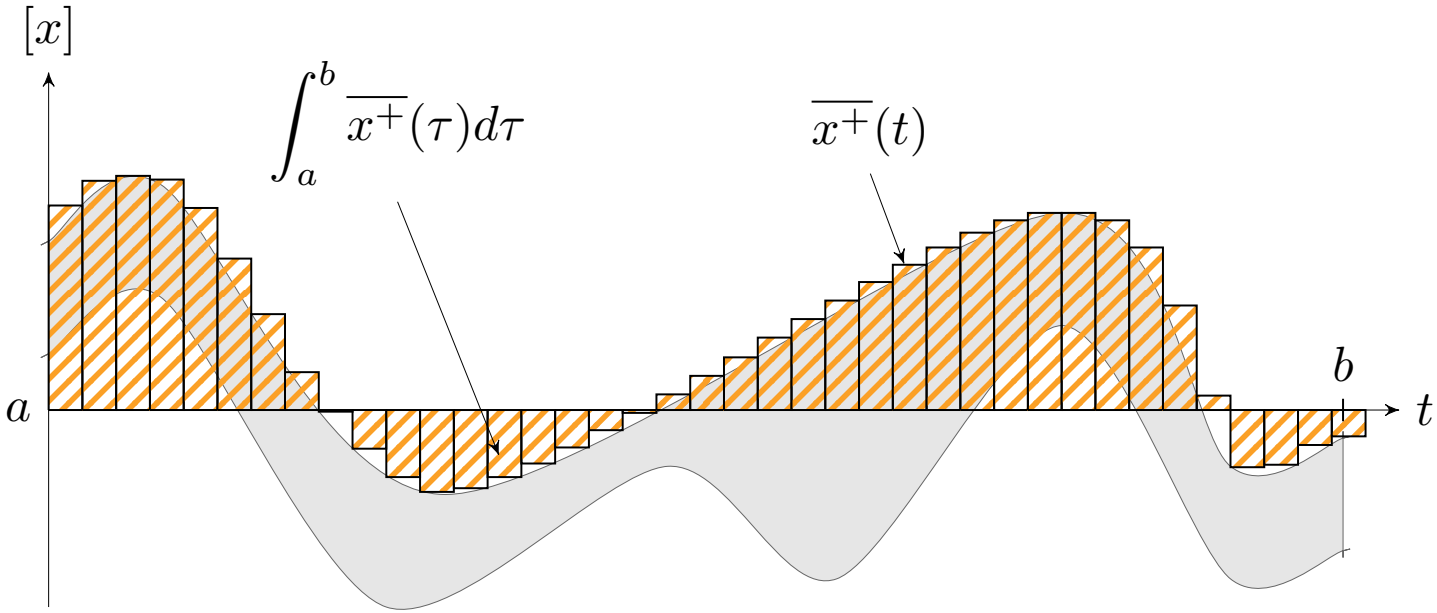


Figure: red area: outer approximation of the upper bound of the tube's integral