Reliable propagation of time uncertainties in dynamical systems

Simon Rohou, Luc Jaulin, Lyudmila Mihaylova,
Fabrice Le Bars, Sandor M. Veres

IMT Atlantique, LS2N, OGRE team, Nantes, France
simon.rohou@ls2n.fr

OGRE Seminar
12th July 2018

Outline

1. Motivations
2. Constraint programming for dynamical systems
3. Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$
4. Application: drifting clock
5. Conclusions

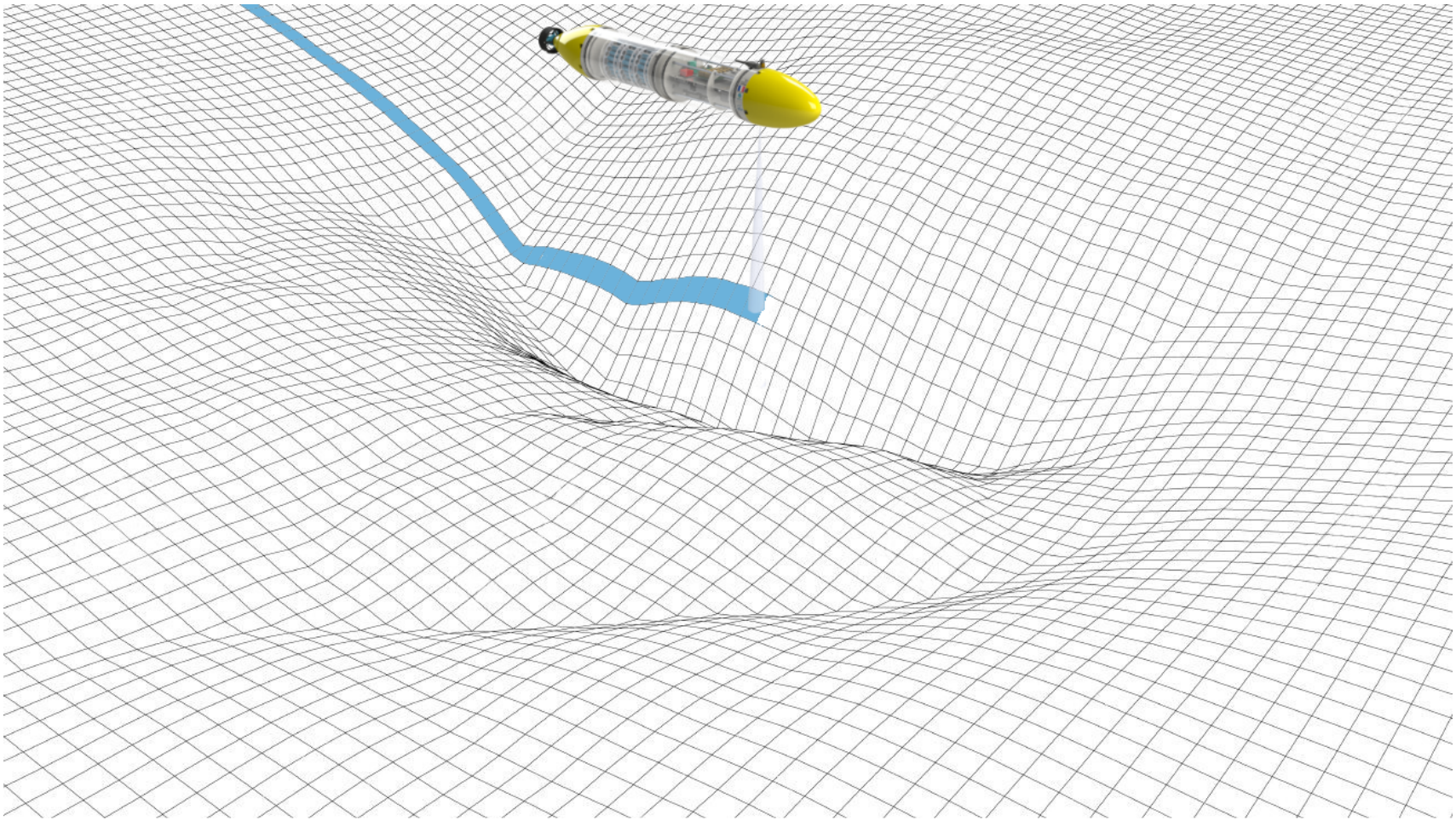
Section 1

Motivations

Motivations

Robot localization $\rightarrow$ temporal resolution

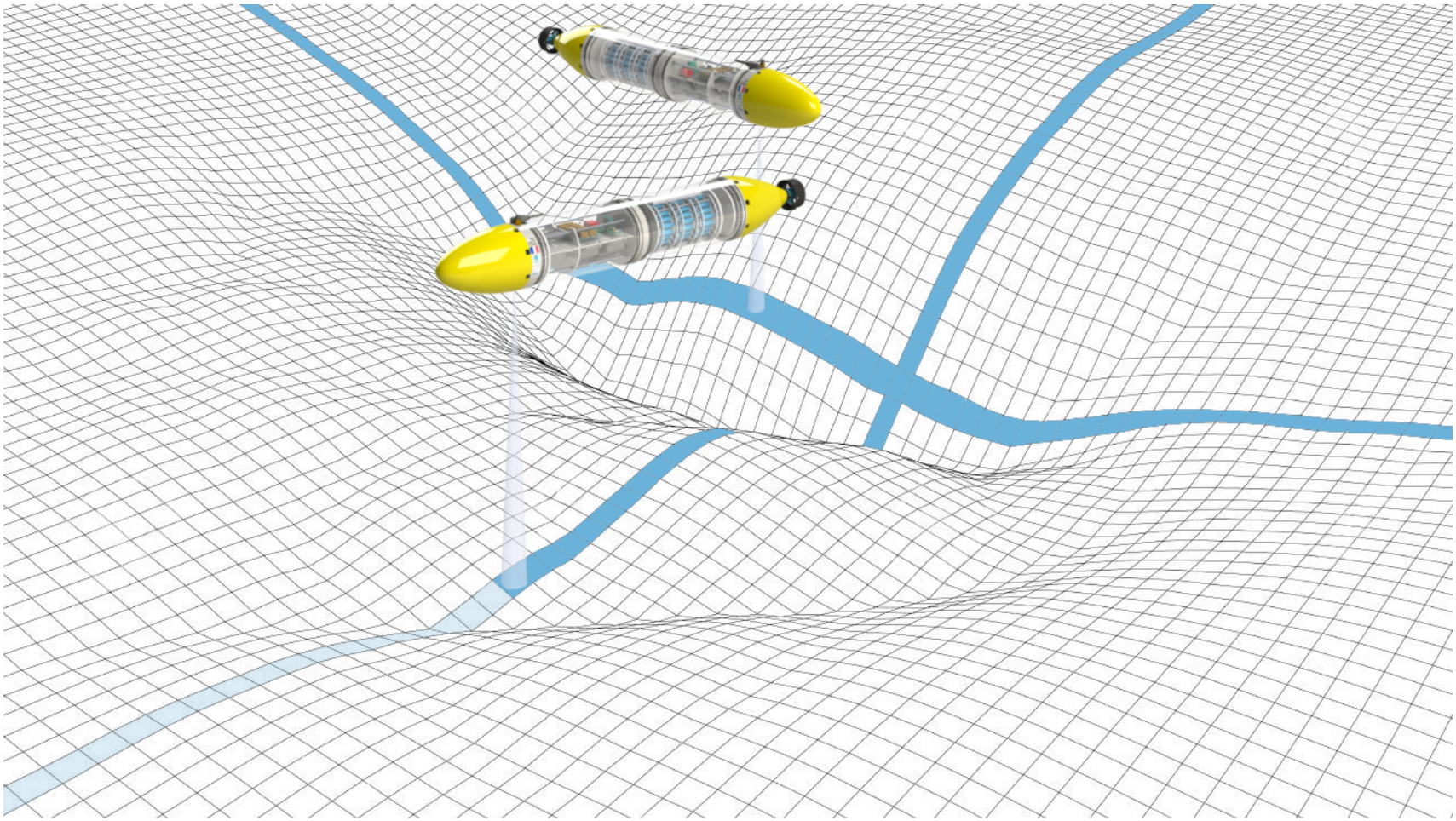
Trajectory $\mathbf{p}(\cdot) : \mathbb{R} \rightarrow \mathbb{R}^2$ crossed at times t_1, t_2: $\mathbf{p}(t_1) = \mathbf{p}(t_2)$.



Motivations

Robot localization $\rightarrow$ temporal resolution

Trajectory $\mathbf{p}(\cdot) : \mathbb{R} \rightarrow \mathbb{R}^2$ crossed at times t_1, t_2: $\mathbf{p}(t_1) = \mathbf{p}(t_2)$.



Motivations

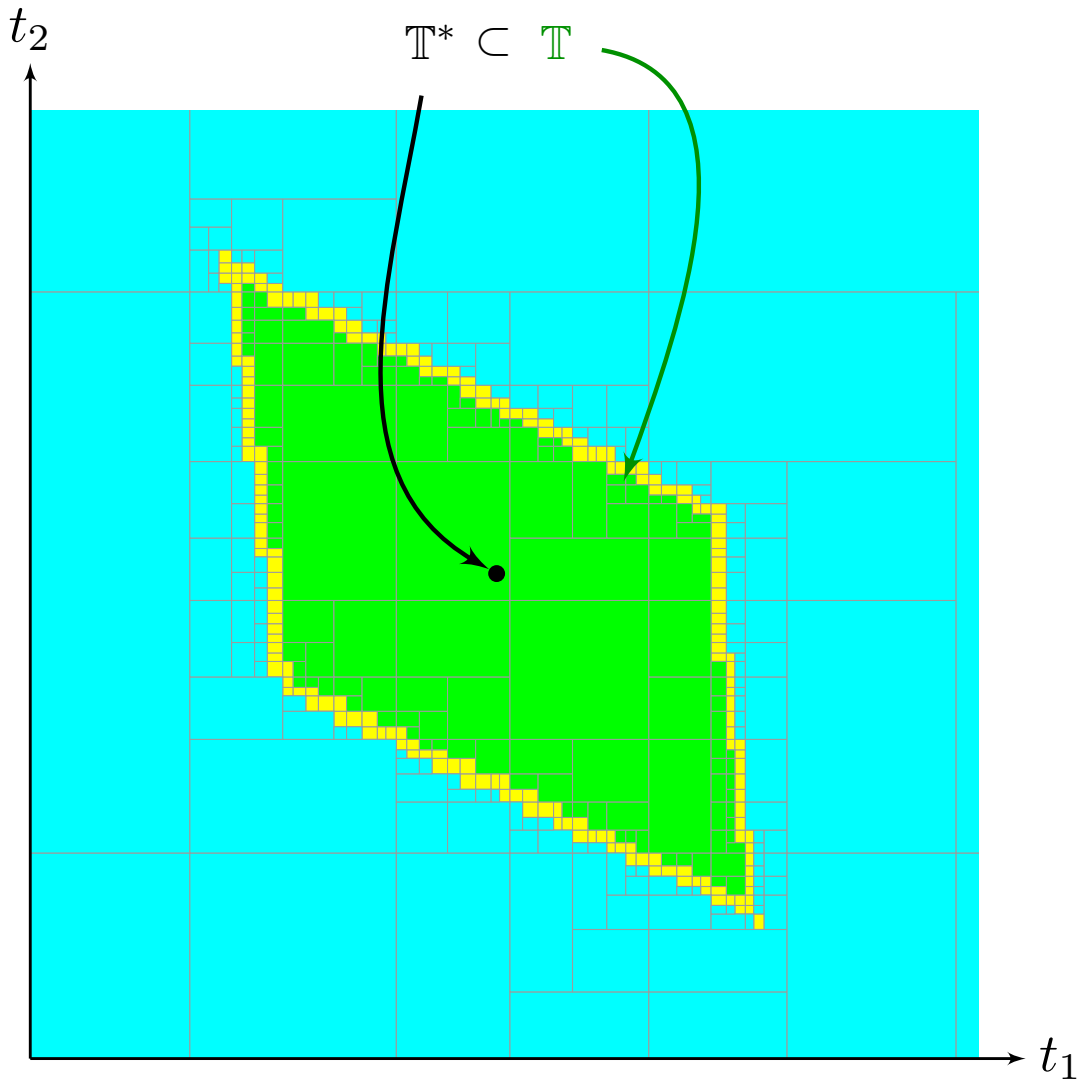
Robot localization $\rightarrow$ temporal resolution

Constraint:

- $\mathbf{p}(t_1) = \mathbf{p}(t_2)$
- $t_1 \in [t_1]$, $t_2 \in [t_2]$

1. approximation of a temporal set $\mathbb{T}$ with evolution constraints

2. contraction of $\mathbb{T}$ thanks to exteroceptive measurements (ex: bathymetry)



Motivations

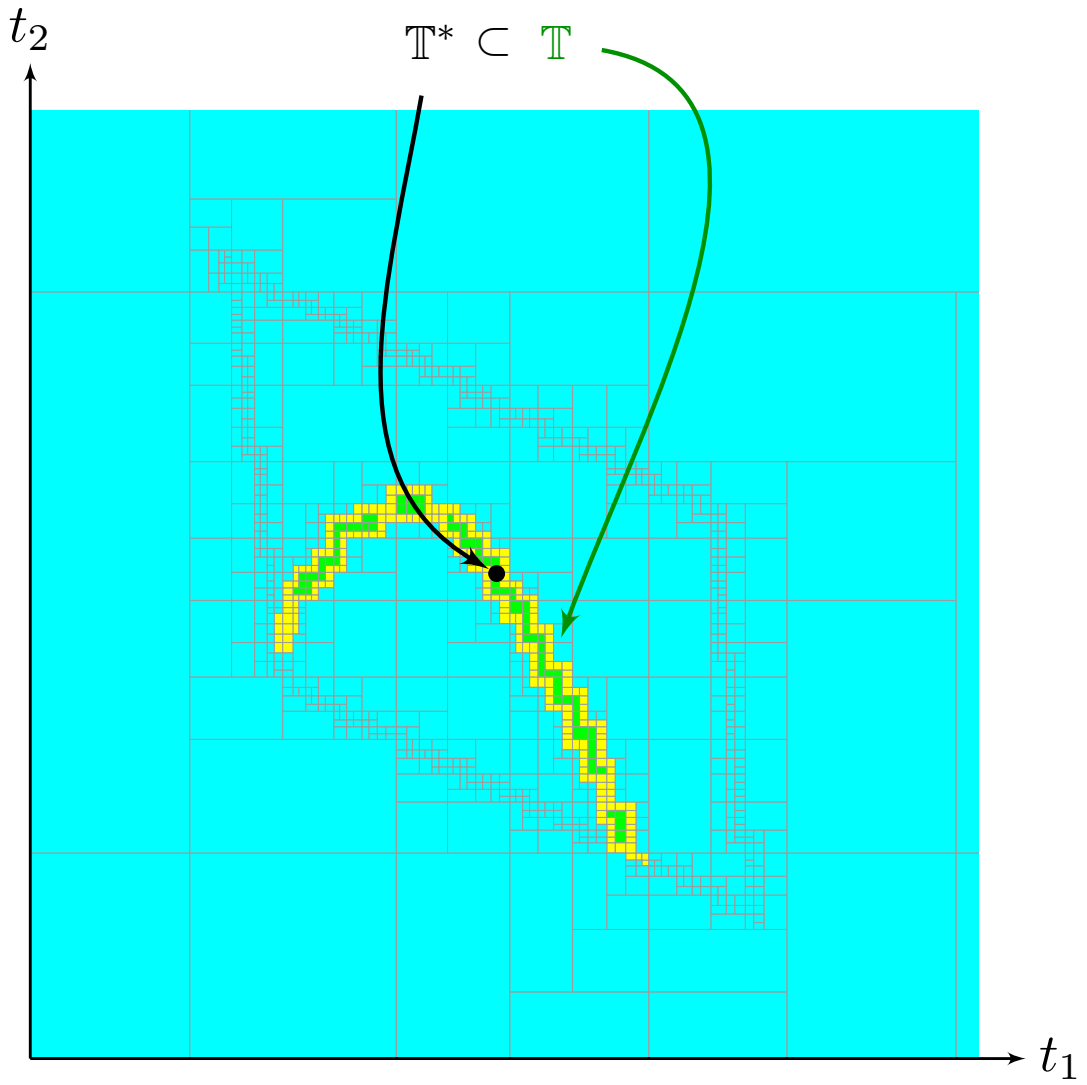
Robot localization $\rightarrow$ temporal resolution

Constraint:

- $\mathbf{p}(t_1) = \mathbf{p}(t_2)$
- $t_1 \in [t_1]$, $t_2 \in [t_2]$

1. approximation of a temporal set $\mathbb{T}$ with evolution constraints

2. contraction of $\mathbb{T}$ thanks to exteroceptive measurements (ex: bathymetry)



Motivations

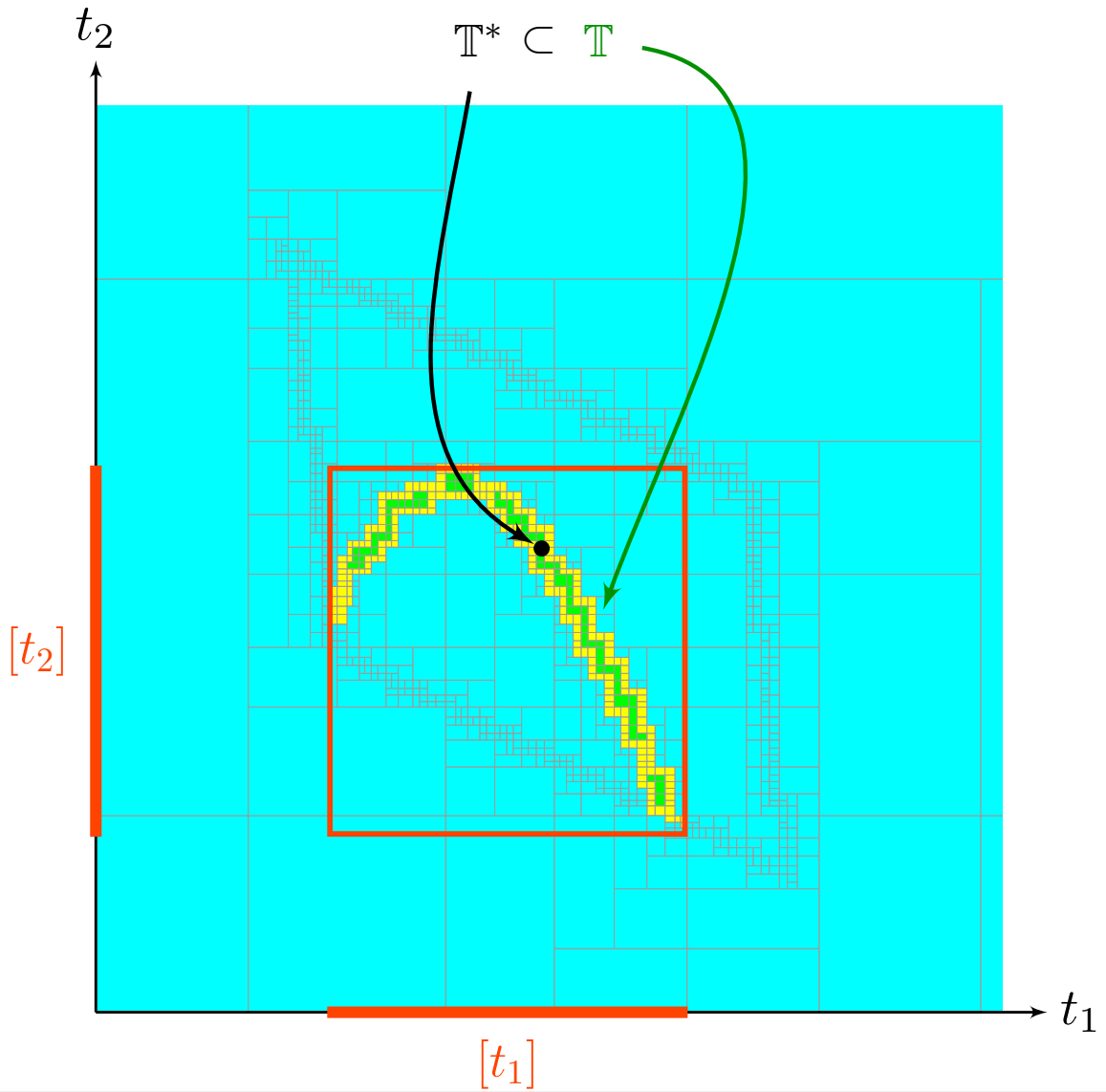
Robot localization → temporal resolution

Constraint:

- $\mathbf{p}(t_1) = \mathbf{p}(t_2)$
- $t_1 \in [t_1]$, $t_2 \in [t_2]$

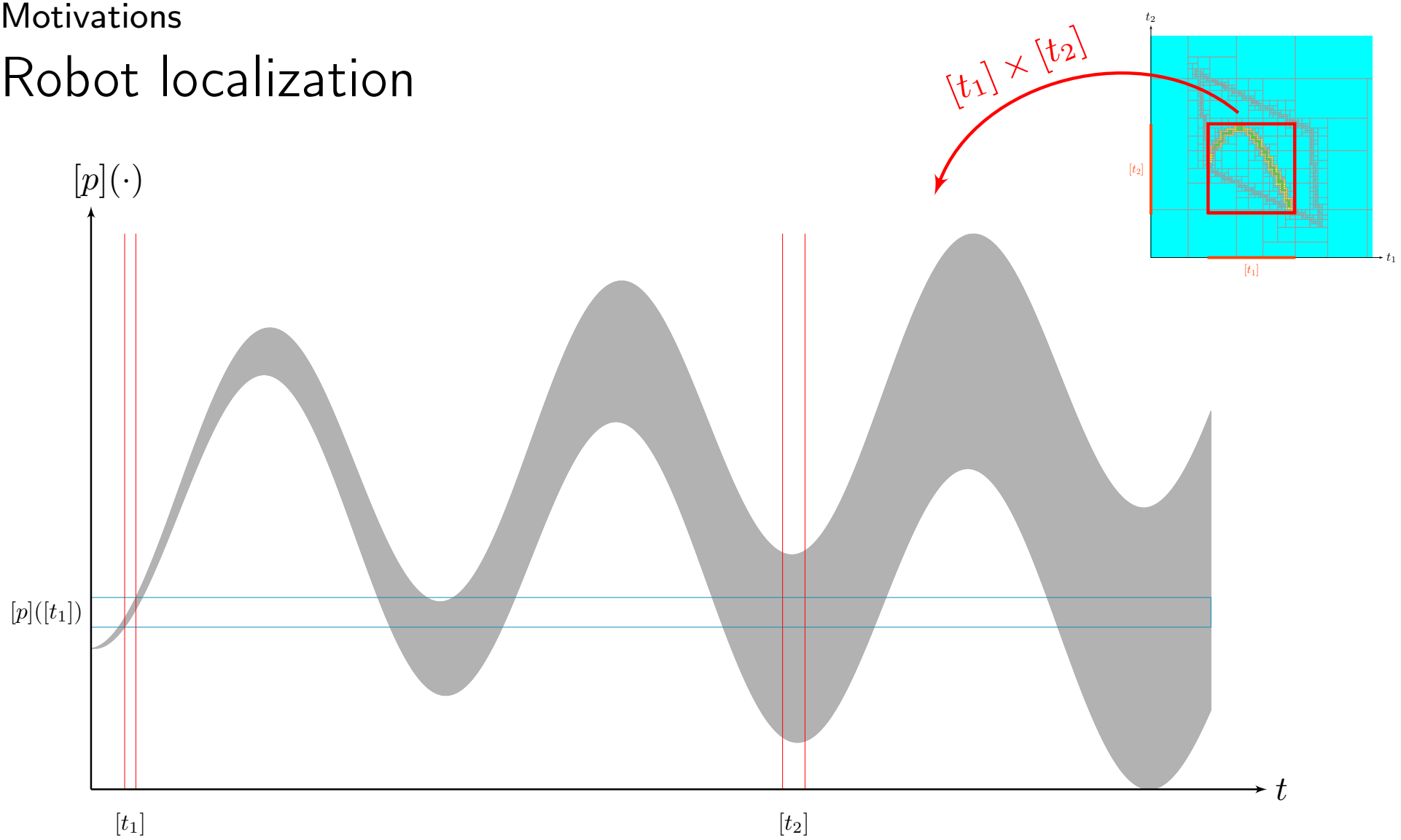
1. approximation of a temporal set $\mathbb{T}$ with evolution constraints

2. contraction of $\mathbb{T}$ thanks to exteroceptive measurements (ex: bathymetry)



Motivations

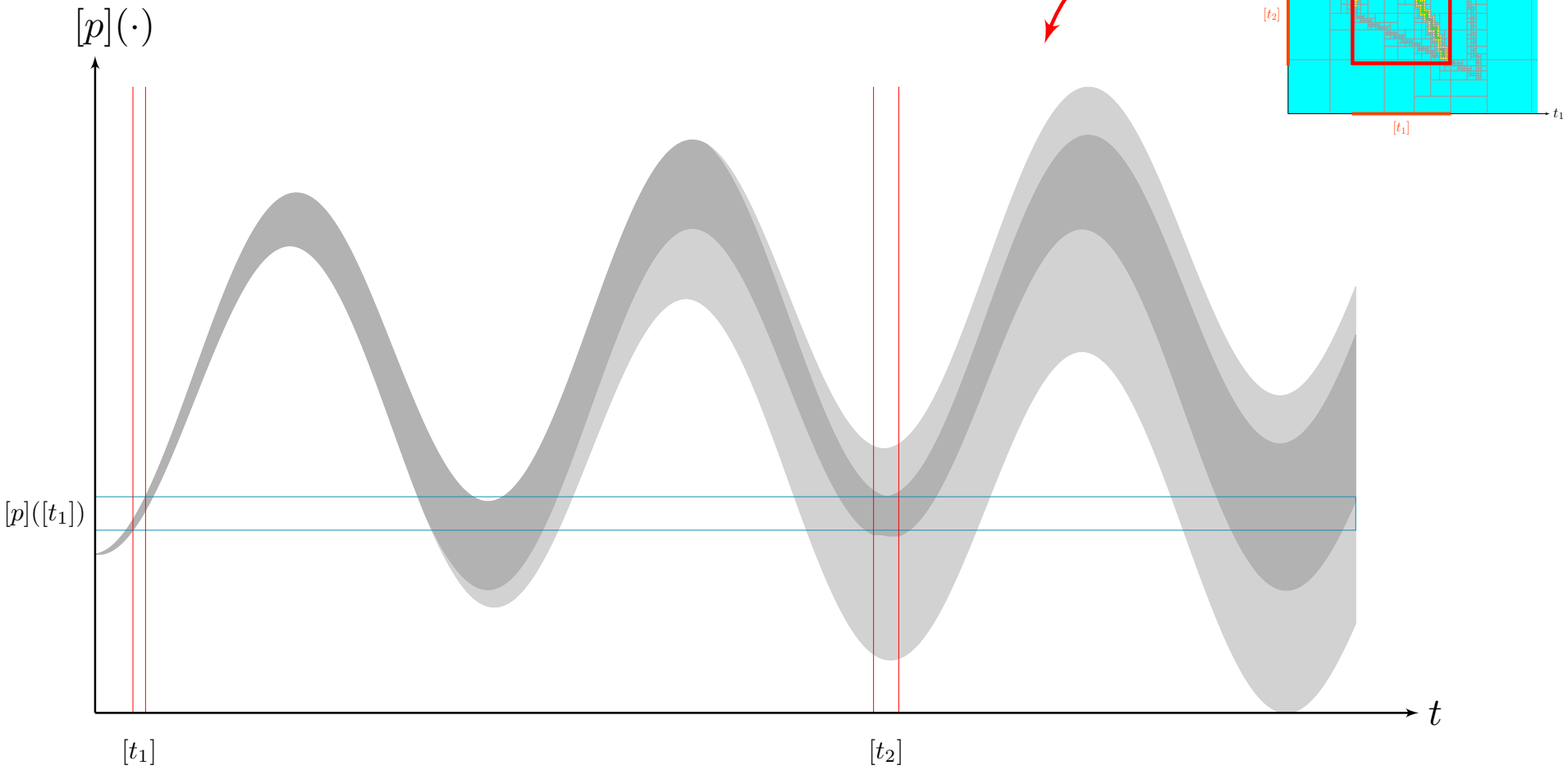
Robot localization



Constraint $\mathcal{L}_{t_1,t_2}\left(t_1, t_2, \mathbf{p}(\cdot), \mathbf{w}(\cdot)\right) : \left\{ \begin{array}{l} \mathbf{p}(t_1) = \mathbf{p}(t_2) \\ \dot{\mathbf{p}}(\cdot) = \mathbf{w}(\cdot) \end{array} \right.$

Motivations

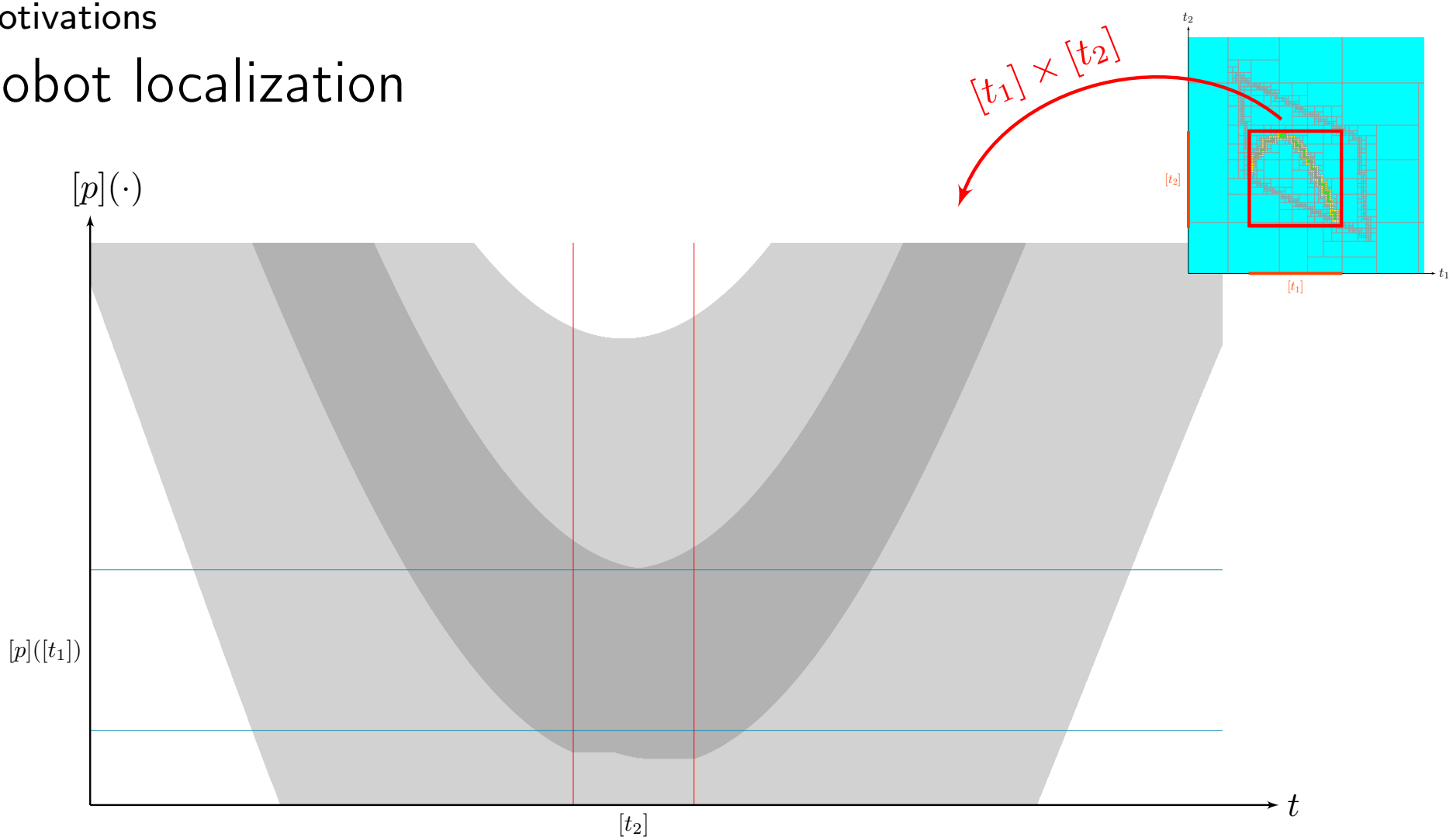
Robot localization



Constraint $\mathcal{L}_{t_1,t_2}\left(t_1, t_2, \mathbf{p}(\cdot), \mathbf{w}(\cdot)\right) : \left\{ \begin{array}{l} \mathbf{p}(t_1) = \mathbf{p}(t_2) \\ \dot{\mathbf{p}}(\cdot) = \mathbf{w}(\cdot) \end{array} \right.$

Motivations

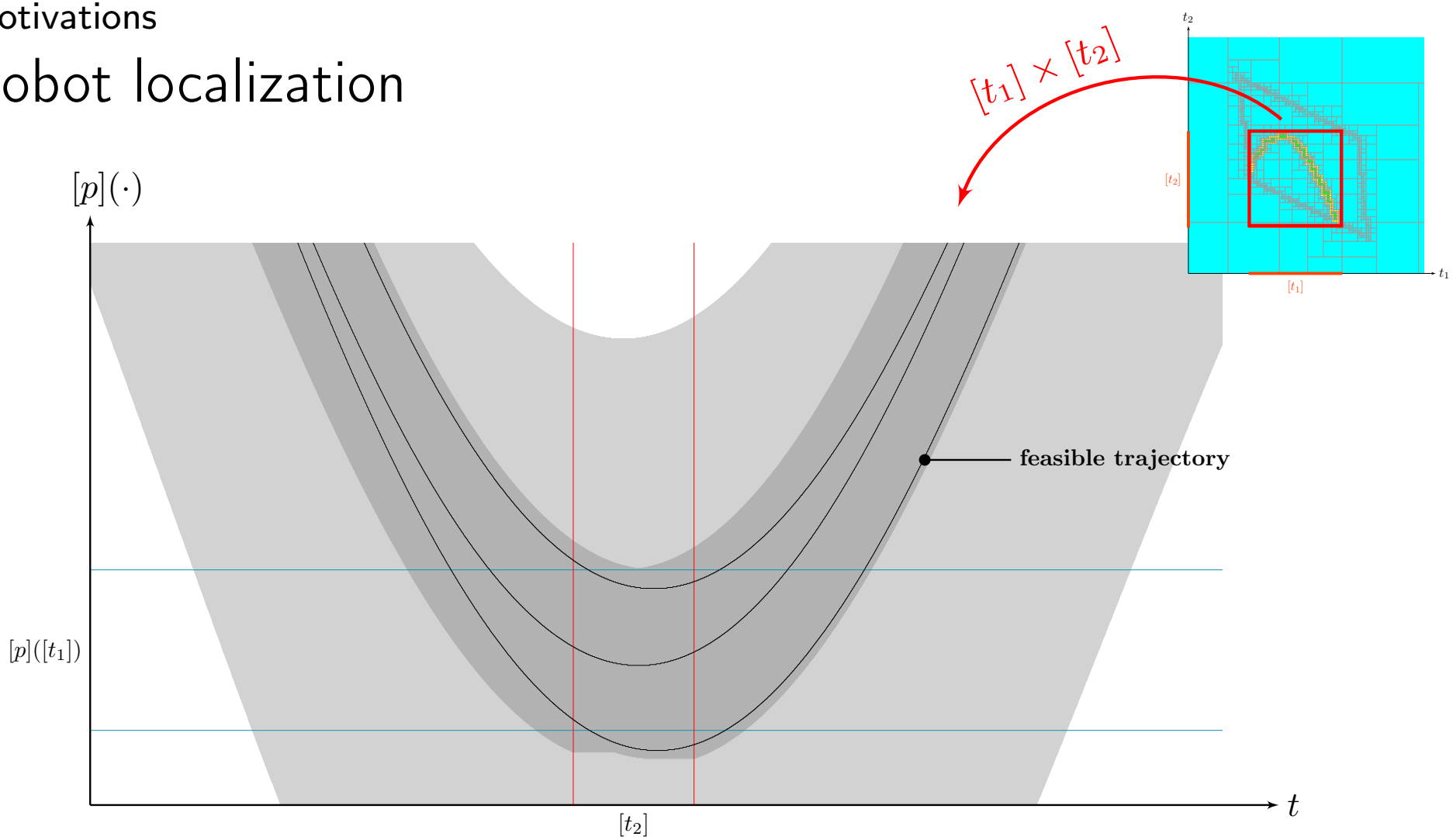
Robot localization



Constraint $\mathcal{L}_{t_1,t_2}\left(t_1, t_2, \mathbf{p}(\cdot), \mathbf{w}(\cdot)\right) : \left\{ \begin{array}{l} \mathbf{p}(t_1) = \mathbf{p}(t_2) \\ \dot{\mathbf{p}}(\cdot) = \mathbf{w}(\cdot) \end{array} \right.$

Motivations

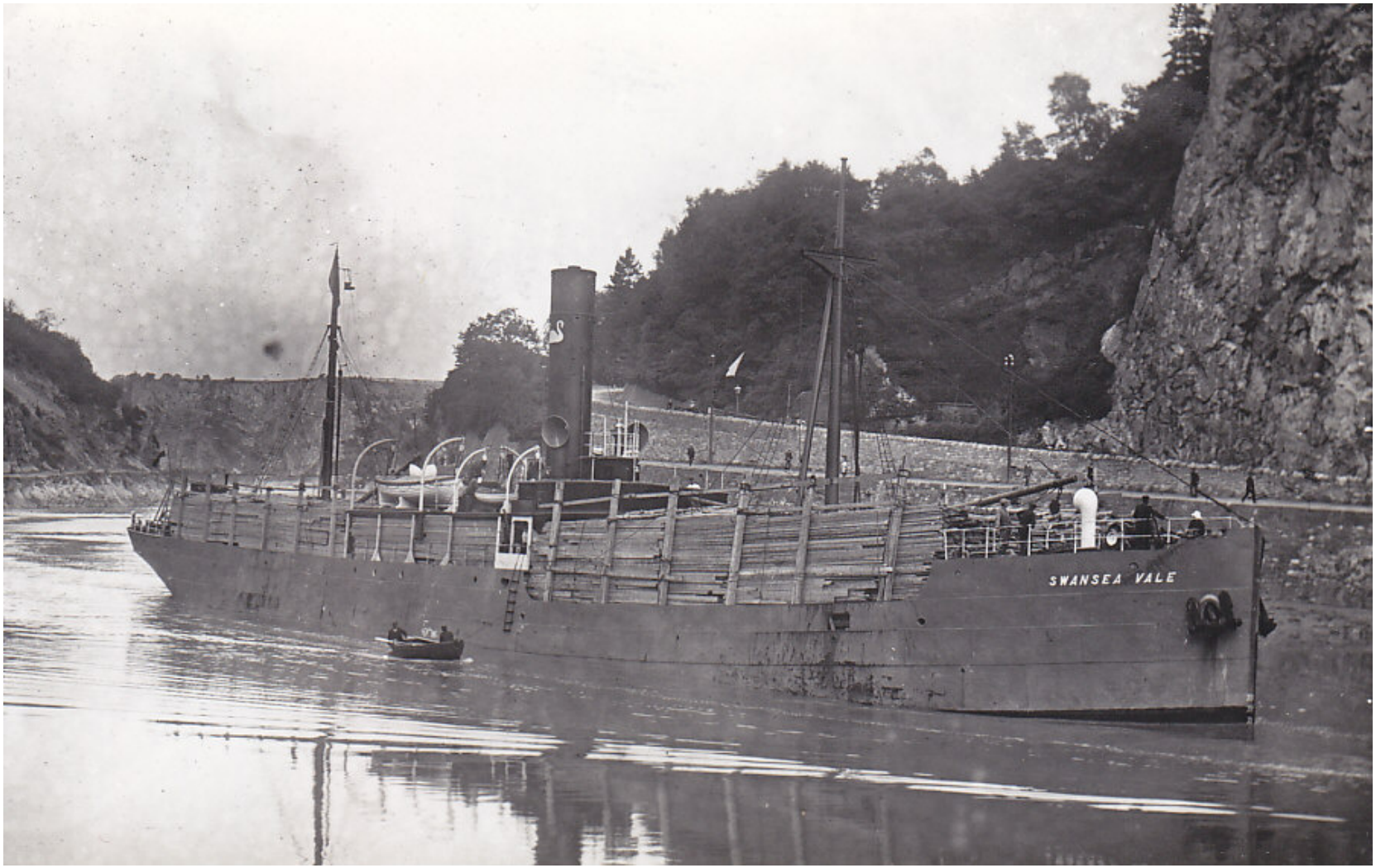
Robot localization



Constraint $\mathcal{L}_{t_1,t_2}\left(t_1, t_2, \mathbf{p}(\cdot), \mathbf{w}(\cdot)\right) : \left\{ \begin{array}{l} \mathbf{p}(t_1) = \mathbf{p}(t_2) \\ \dot{\mathbf{p}}(\cdot) = \mathbf{w}(\cdot) \end{array} \right.$

Motivations

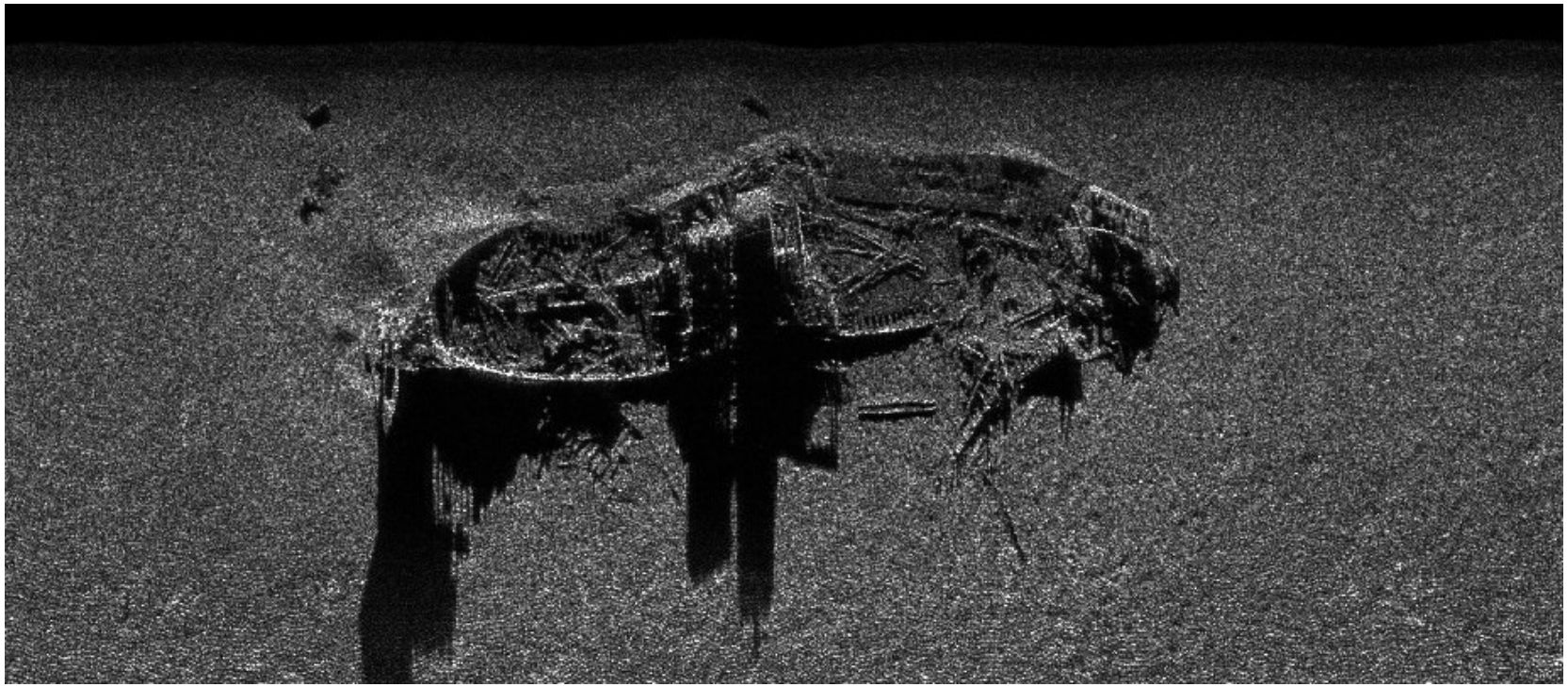
Another motivation: wreck based localization



The *Swansea* boat, during the First World War. *Unknown copyright.*

Motivations

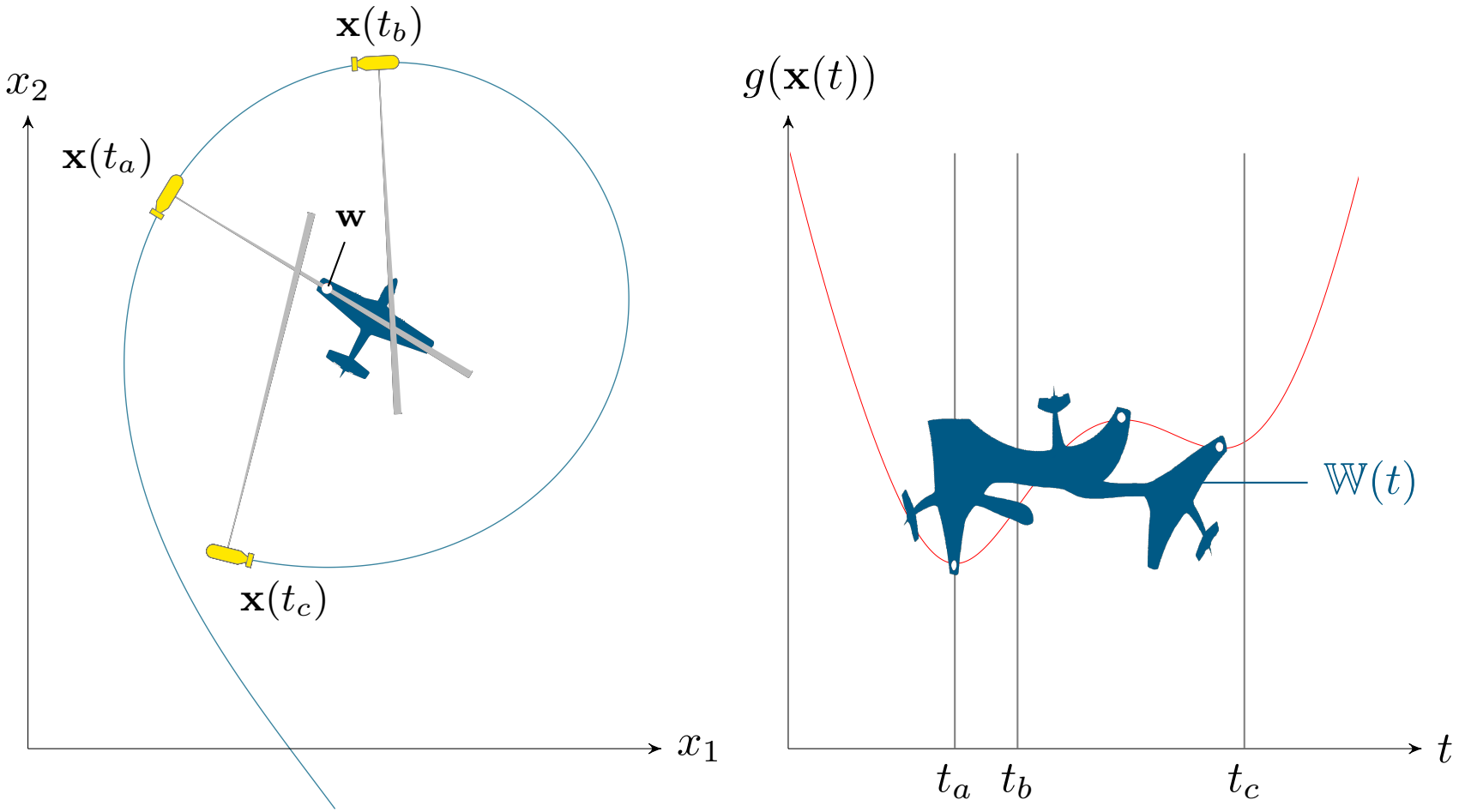
Another motivation: wreck based localization



The *Swansea* wreck perceived with a side scan sonar (Rade de Brest).
The ship's funnel and superstructures cause wide shadowed areas: the darkest parts of the sonar image.
Copyrights: SHOM, DGA-TN Brest, Michel Legris.

Motivations

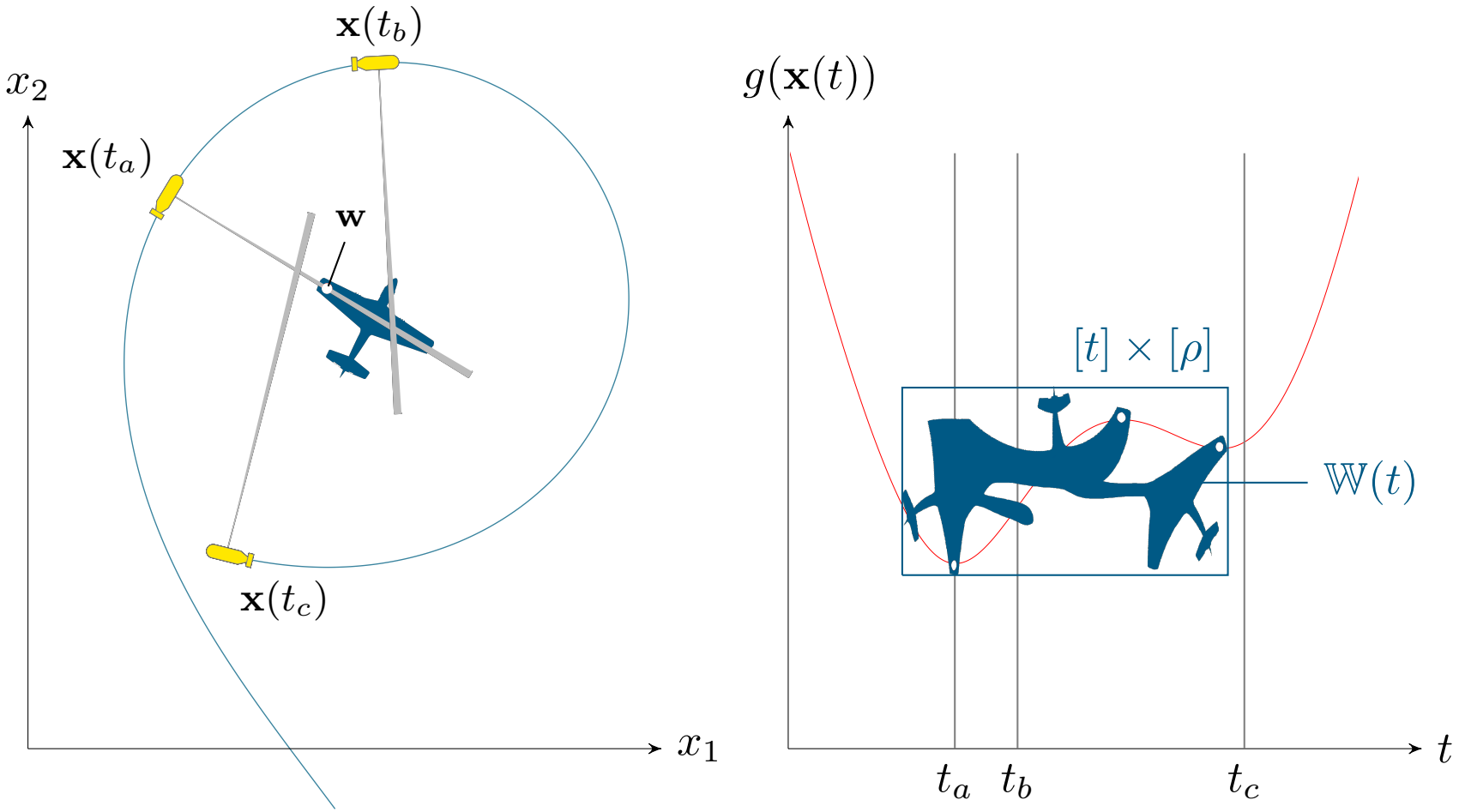
Another motivation: wreck based localization



A robot $\mathcal{R}$ perceiving a plane wreck with a side scan sonar.

Motivations

Another motivation: wreck based localization



A robot $\mathcal{R}$ perceiving a plane wreck with a side scan sonar.

Section 2

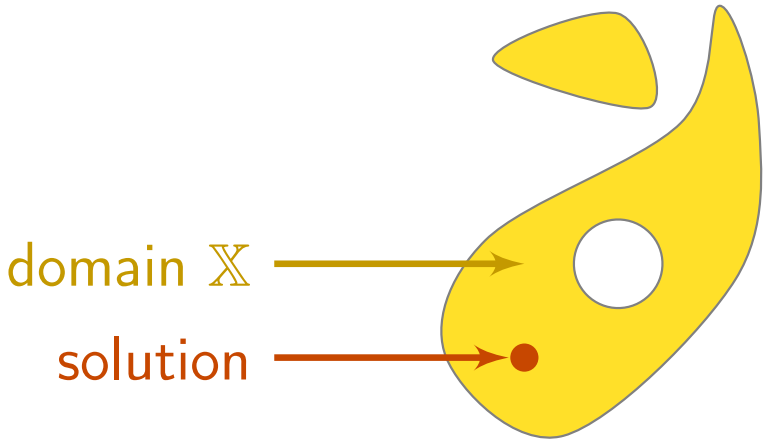
Constraint programming for dynamical systems

Constraint programming for dynamical systems

Main approach

Example in $\mathbb{R}^2$:

- system solving described by a *constraint network*
- **variables** (vectors $\mathbf{x} \in \mathbb{R}^n$) belonging to **domains** $\mathbb{X}$



Constraint network:

Variables: $\mathbf{x}$

Constraints:

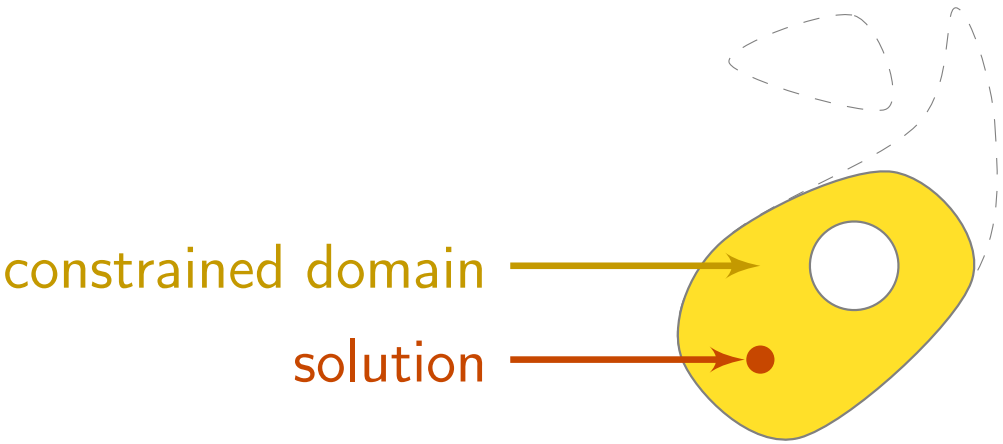
Domains: $\mathbb{X}$

Constraint programming for dynamical systems

Main approach

Example in $\mathbb{R}^2$:

- system solving described by a *constraint network*
- **variables** (vectors $\mathbf{x} \in \mathbb{R}^n$) belonging to **domains** $\mathbb{X}$
- continuous **constraints** $\mathcal{L}$: non-linear equations, inequalities, ...



Constraint network:

Variables: $\mathbf{x}$

Constraints:

1. $\mathcal{L}_1(\mathbf{x})$

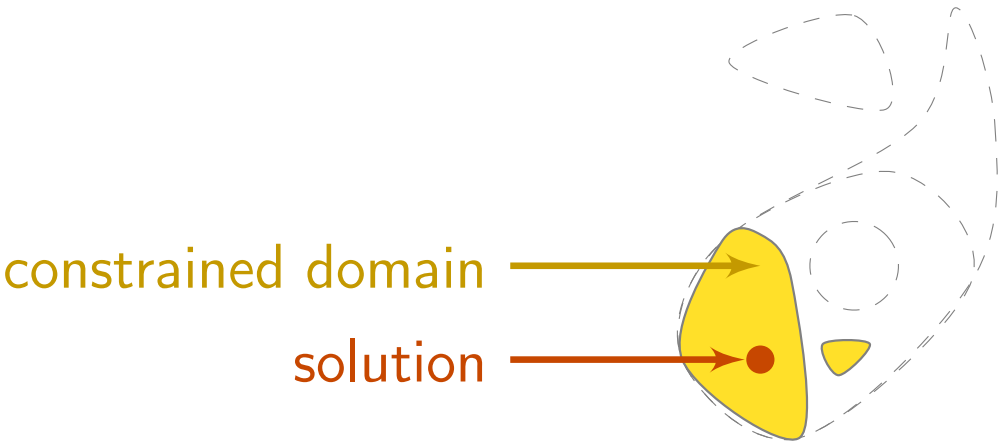
Domains: $\mathbb{X}$

Constraint programming for dynamical systems

Main approach

Example in $\mathbb{R}^2$:

- system solving described by a *constraint network*
- **variables** (vectors $\mathbf{x} \in \mathbb{R}^n$) belonging to **domains** $\mathbb{X}$
- continuous **constraints** $\mathcal{L}$: non-linear equations, inequalities, ...



Constraint network:

Variables: $\mathbf{x}$

Constraints:

1. $\mathcal{L}_1(\mathbf{x})$
2. $\mathcal{L}_2(\mathbf{x})$

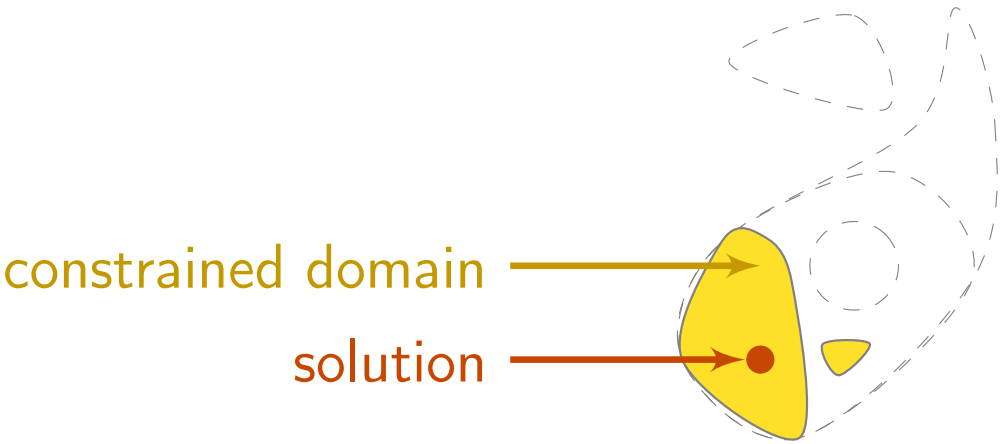
Domains: $\mathbb{X}$

Constraint programming for dynamical systems

Main approach

Example in $\mathbb{R}^2$:

- system solving described by a *constraint network*
- **variables** (vectors $\mathbf{x} \in \mathbb{R}^n$) belonging to **domains** $\mathbb{X}$
- continuous **constraints** $\mathcal{L}$: non-linear equations, inequalities, ...



Constraint network:

Variables: $\mathbf{x}$

Constraints:

1. $\mathcal{L}_1(\mathbf{x})$
2. $\mathcal{L}_2(\mathbf{x})$
3. ...

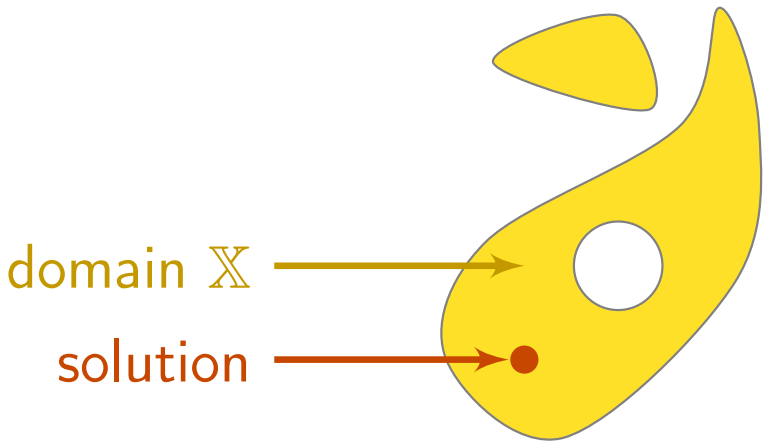
Domains: $\mathbb{X}$

Constraint programming for dynamical systems

Main approach

Example in $\mathbb{R}^2$:

- system solving described by a *constraint network*
- **variables** (vectors $\mathbf{x} \in \mathbb{R}^n$) belonging to **domains** $\mathbb{X}$
- continuous **constraints** $\mathcal{L}$: non-linear equations, inequalities, ...



Constraint network:

Variables: $\mathbf{x}$

Constraints:

1. $\mathcal{L}_1(\mathbf{x})$
2. $\mathcal{L}_2(\mathbf{x})$
3. ...

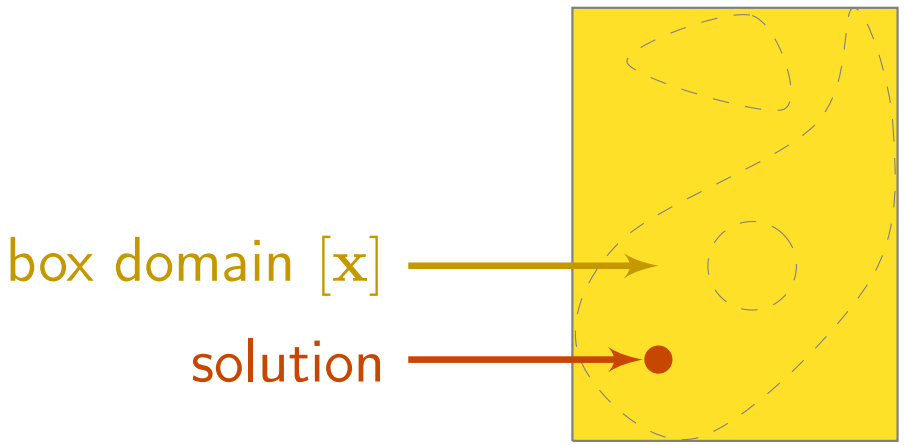
Domains: $\mathbb{X}$

Constraint programming for dynamical systems

Main approach

Example in $\mathbb{R}^2$:

- system solving described by a *constraint network*
- **variables** (vectors $\mathbf{x} \in \mathbb{R}^n$) belonging to **domains** $\mathbb{X}$
- continuous **constraints** $\mathcal{L}$: non-linear equations, inequalities, ...
- representable domains: interval-vectors $[\mathbf{x}] \in \mathbb{IR}^n$



Constraint network:

Variables: $\mathbf{x}$

Constraints:

1. $\mathcal{L}_1(\mathbf{x})$
2. $\mathcal{L}_2(\mathbf{x})$
3. ...

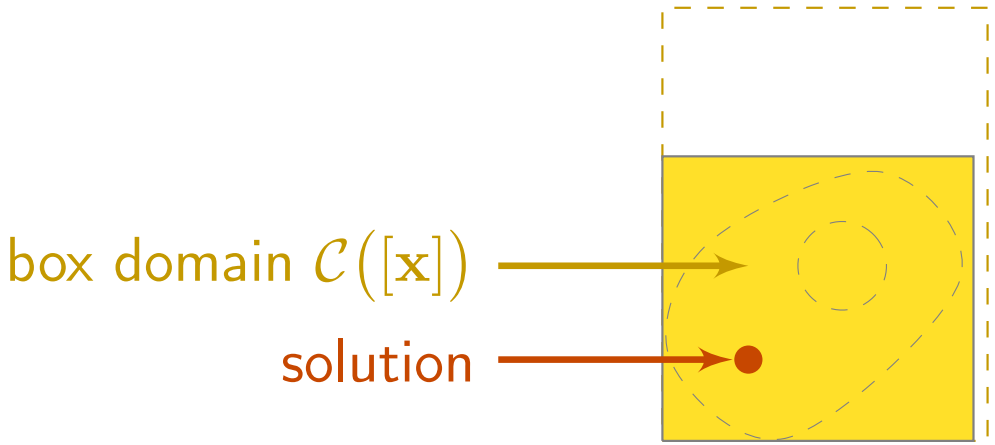
Domains: $[\mathbf{x}]$

Constraint programming for dynamical systems

Main approach

Example in $\mathbb{R}^2$:

- system solving described by a *constraint network*
- **variables** (vectors $\mathbf{x} \in \mathbb{R}^n$) belonging to **domains** $\mathbb{X}$
- continuous **constraints** $\mathcal{L}$: non-linear equations, inequalities, ...
- representable domains: interval-vectors $[\mathbf{x}] \in \mathbb{IR}^n$
- resolution by **contractors**, $\mathcal{C}_{\mathcal{L}}([\mathbf{x}])$



Constraint network:

Variables: $\mathbf{x}$

Constraints:

1. $\mathcal{L}_1(\mathbf{x})$
2. $\mathcal{L}_2(\mathbf{x})$
3. ...

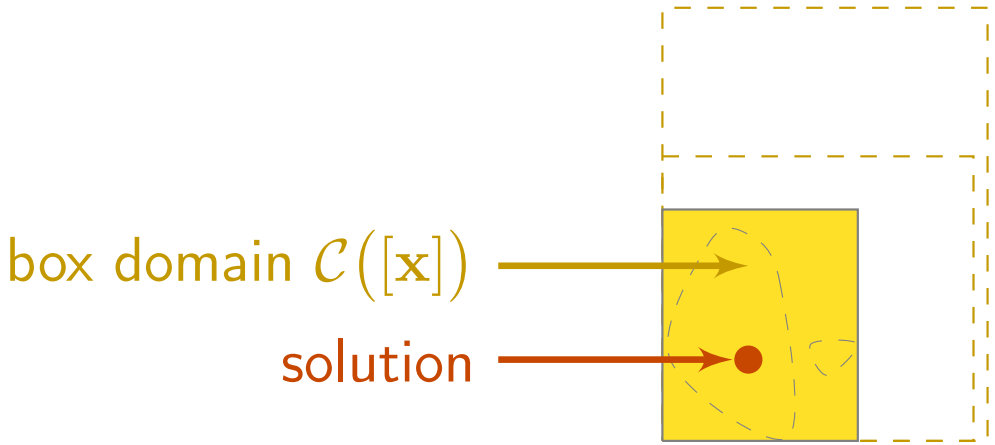
Domains: $[\mathbf{x}]$

Constraint programming for dynamical systems

Main approach

Example in $\mathbb{R}^2$:

- system solving described by a *constraint network*
- **variables** (vectors $\mathbf{x} \in \mathbb{R}^n$) belonging to **domains** $\mathbb{X}$
- continuous **constraints** $\mathcal{L}$: non-linear equations, inequalities, ...
- representable domains: interval-vectors $[\mathbf{x}] \in \mathbb{IR}^n$
- resolution by **contractors**, $\mathcal{C}_{\mathcal{L}}([\mathbf{x}])$



Constraint network:

Variables: $\mathbf{x}$

Constraints:

1. $\mathcal{L}_1(\mathbf{x})$
2. $\mathcal{L}_2(\mathbf{x})$
3. ...

Domains: $[\mathbf{x}]$

Constraint programming for dynamical systems

Extension to dynamical systems

Only few work on **constraints for dynamical systems**:

- Janssen, Van Hentenryck, and Deville 2002
- Hickey 2000
- Cruz and Barahona 2003

Constraint programming for dynamical systems

Extension to dynamical systems

Only few work on **constraints for dynamical systems**:

- Janssen, Van Hentenryck, and Deville 2002
- Hickey 2000
- Cruz and Barahona 2003

New approach: Le Bars et al. 2012; Bethencourt and Jaulin 2014

- variables: **trajectories**, $\mathbf{x}(\cdot) : \mathbb{R} \to \mathbb{R}^n$
- domains: **tubes**, $[\mathbf{x}](\cdot) : \mathbb{R} \to \mathbb{IR}^n$

Constraint programming for dynamical systems

Extension to dynamical systems

Only few work on **constraints for dynamical systems**:

- Janssen, Van Hentenryck, and Deville 2002
- Hickey 2000
- Cruz and Barahona 2003

New approach: Le Bars et al. 2012; Bethencourt and Jaulin 2014

- variables: **trajectories**, $\mathbf{x}(\cdot) : \mathbb{R} \to \mathbb{R}^n$
- domains: **tubes**, $[\mathbf{x}](\cdot) : \mathbb{R} \to \mathbb{IR}^n$

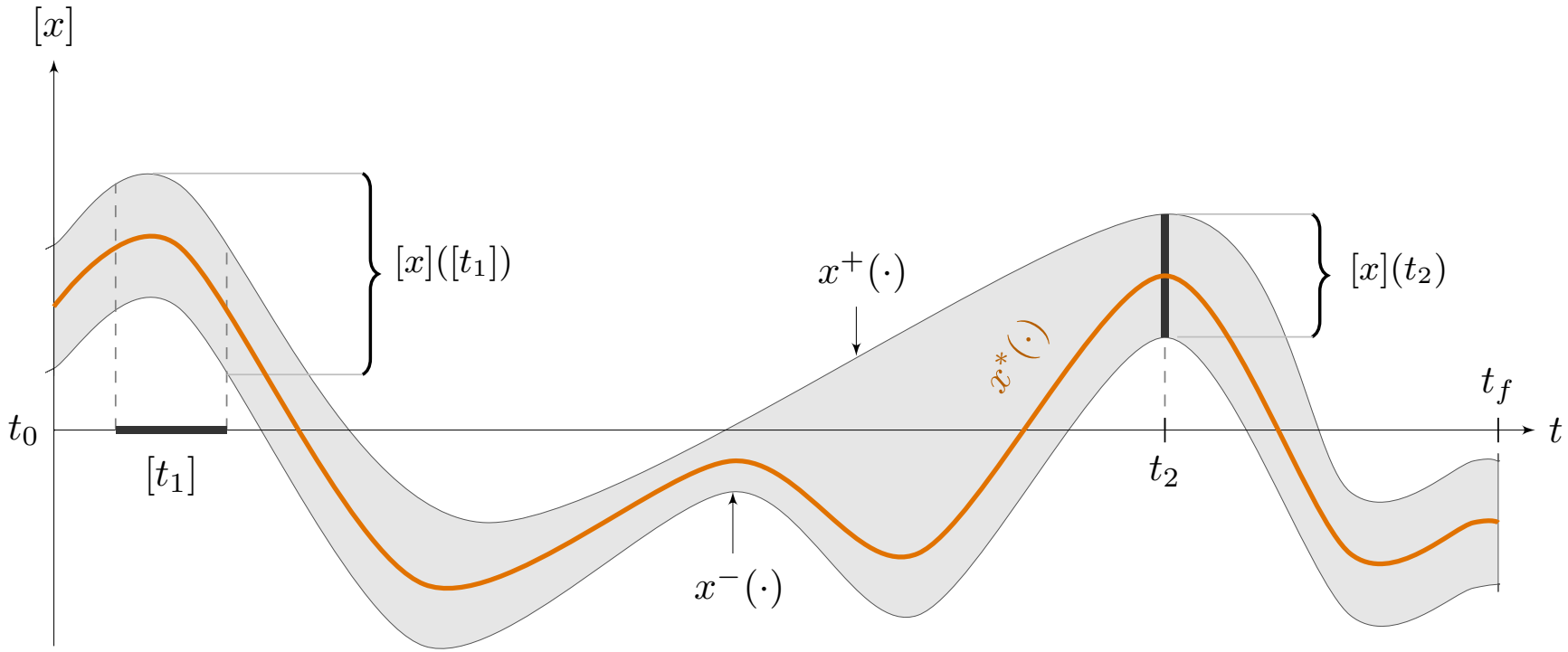
Objectives:

- develop **primitive dynamical contractors**
- application to **robot localization**

Constraint programming for dynamical systems

Tubes

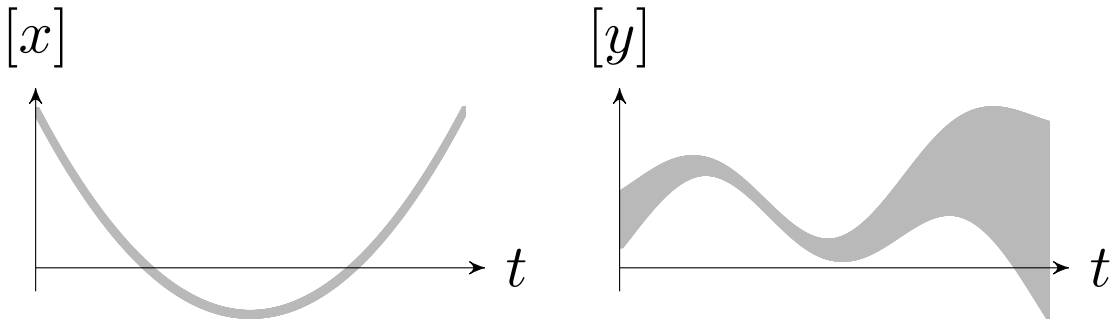
Tube $[x](\cdot)$: interval of trajectories $[x^-(\cdot), x^+(\cdot)]$
such that $\forall t \in \mathbb{R},\ x^-(t) \leqslant x^+(t)$



Tube $[x](\cdot)$ enclosing an uncertain trajectory $x^*(\cdot)$

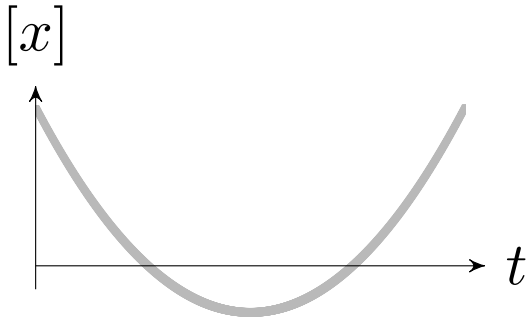
Constraint programming for dynamical systems

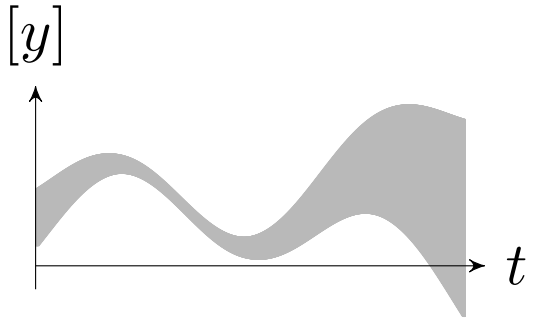
Tubes arithmetic

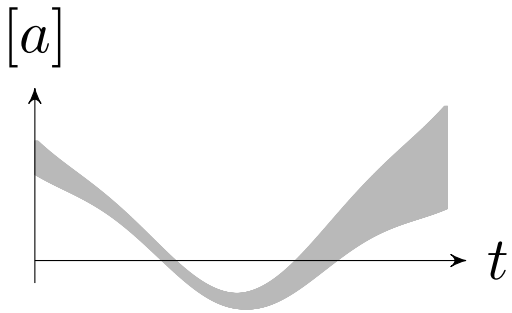


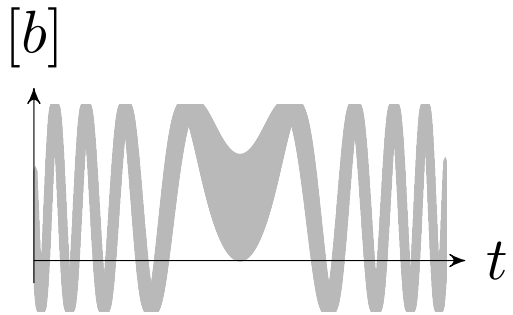
Constraint programming for dynamical systems

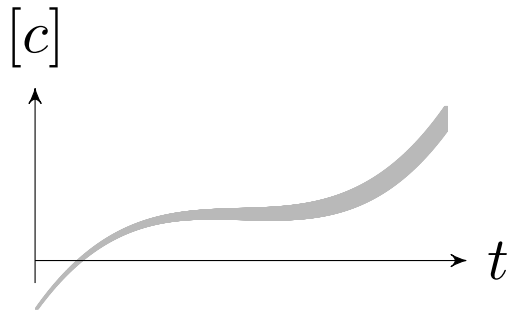
Tubes arithmetic











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

Constraint programming for dynamical systems

Tube contractor

Contractor on boxes can be extended to sets of trajectories (tubes).

Definition

A contractor $\mathcal{C}_{\mathcal{L}}$ applied on a tube $[x](\cdot)$ aims at removing infeasible trajectories according to a given constraint $\mathcal{L}$ so that:

$$(i) \quad \forall t \in [t_0, t_f],\ \mathcal{C}_{\mathcal{L}}\big([x](t)\big) \subseteq [x](t) \qquad \text{(contraction)}$$

$$(ii) \quad \begin{pmatrix} \mathcal{L}\big(x(\cdot)\big) \\ x(\cdot) \in [x](\cdot) \end{pmatrix} \implies x(\cdot) \in \mathcal{C}_{\mathcal{L}}\big([x](\cdot)\big) \qquad \text{(consistency)}$$

Constraint programming for dynamical systems

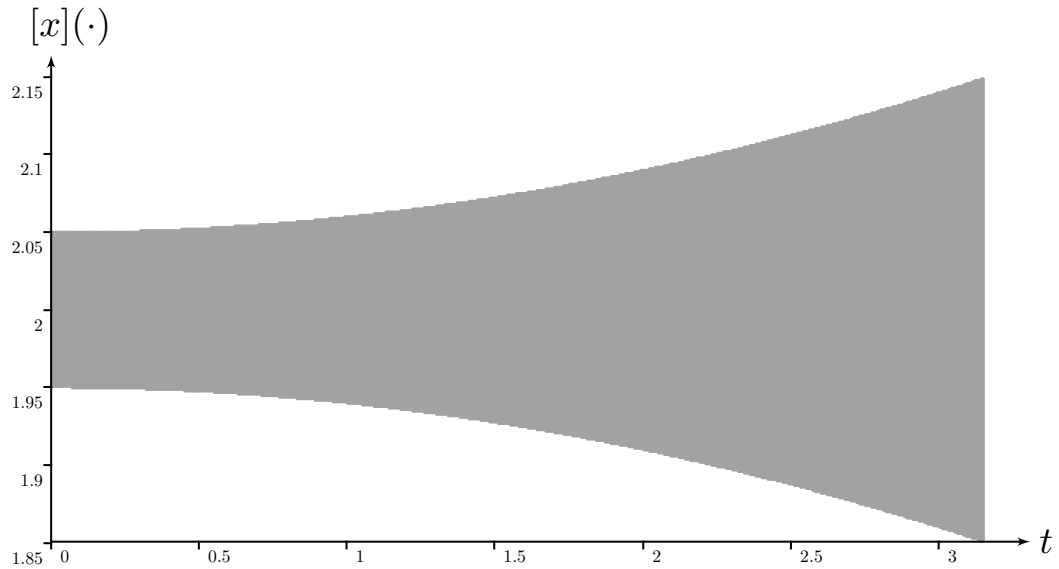
Constraint $\dot{x}(\cdot) = v(\cdot)$

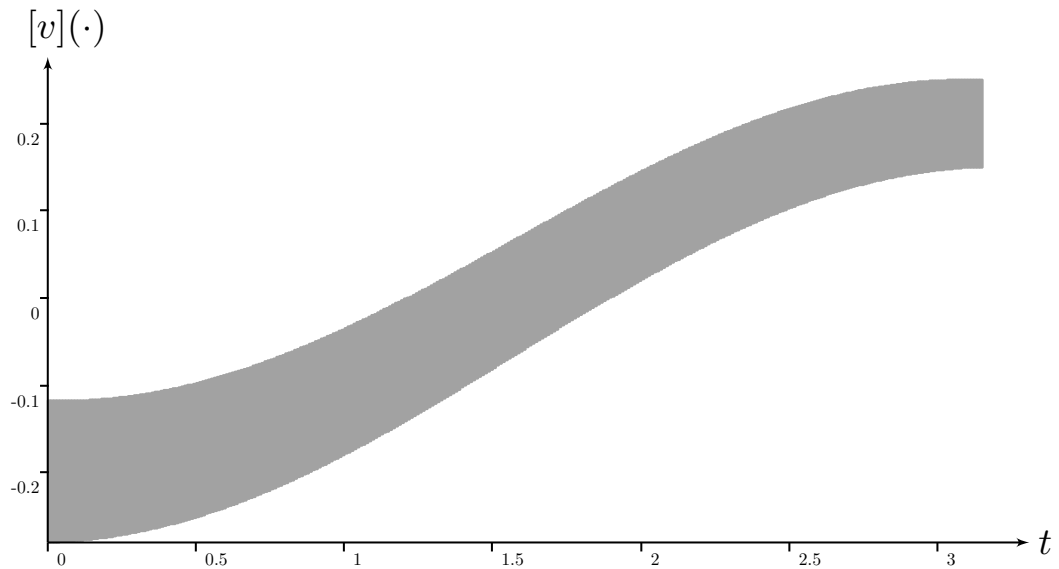
Differential constraint:

- $\mathcal{L}_{\frac{d}{dt}}\big(x(\cdot), v(\cdot)\big) : \dot{x}(\cdot) = v(\cdot)$
- elementary constraint

Related contractor $\mathcal{C}_{\frac{d}{dt}}$:

- one tube $[x](\cdot)$
- one tube $[v](\cdot)$





Constraint programming for dynamical systems

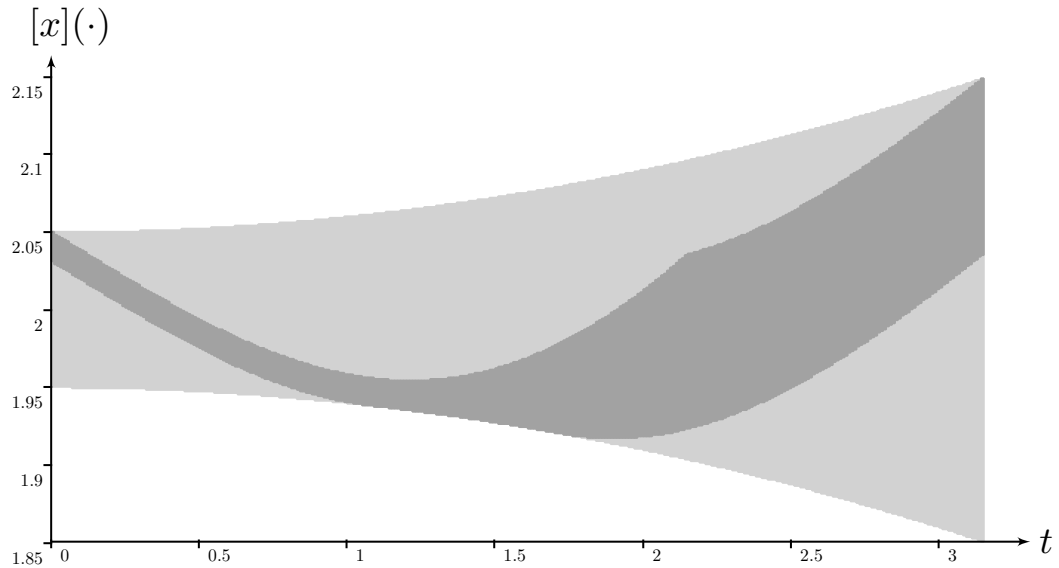
Constraint $\dot{x}(\cdot) = v(\cdot)$

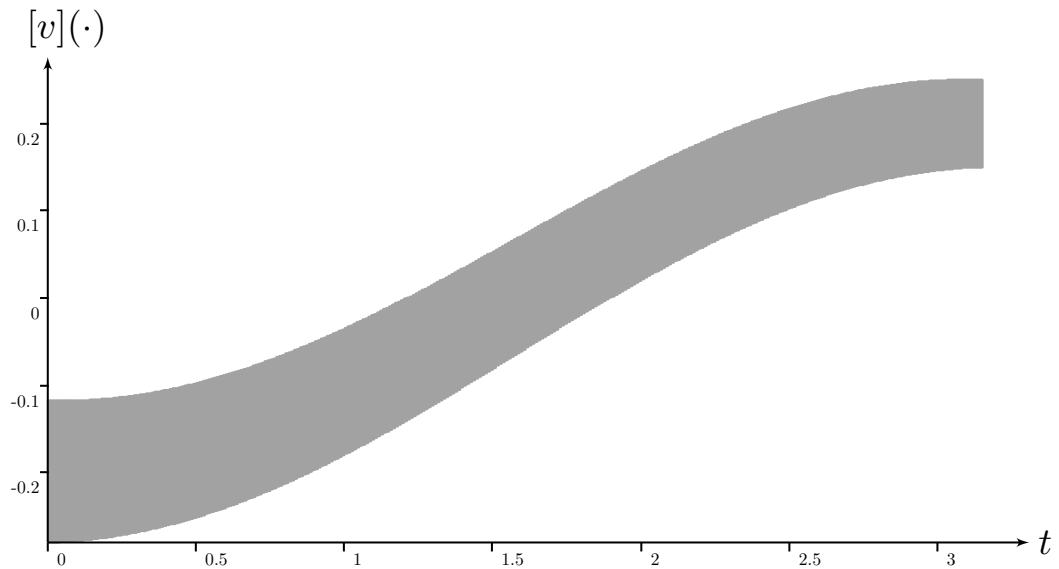
Differential constraint:

- $\mathcal{L}_{\frac{d}{dt}}\left(x(\cdot), v(\cdot)\right) : \dot{x}(\cdot) = v(\cdot)$
- elementary constraint

Related contractor $\mathcal{C}_{\frac{d}{dt}}$:

- one tube $[x](\cdot)$
- one tube $[v](\cdot)$
- $\mathcal{C}_{\frac{d}{dt}}\left([x](\cdot), [v](\cdot)\right)$





Constraint programming for dynamical systems

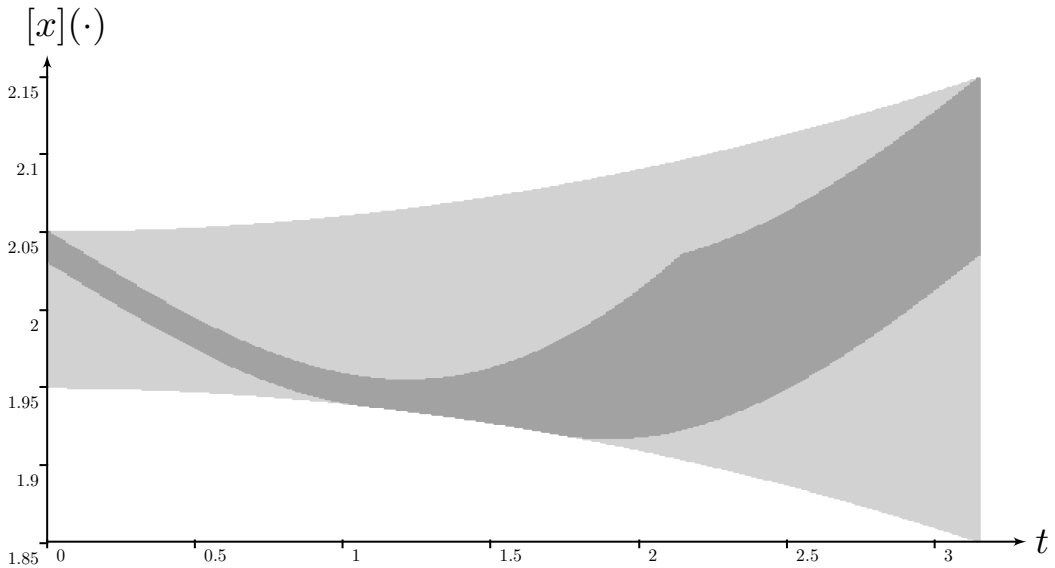
Constraint $\dot{x}(\cdot) = v(\cdot)$

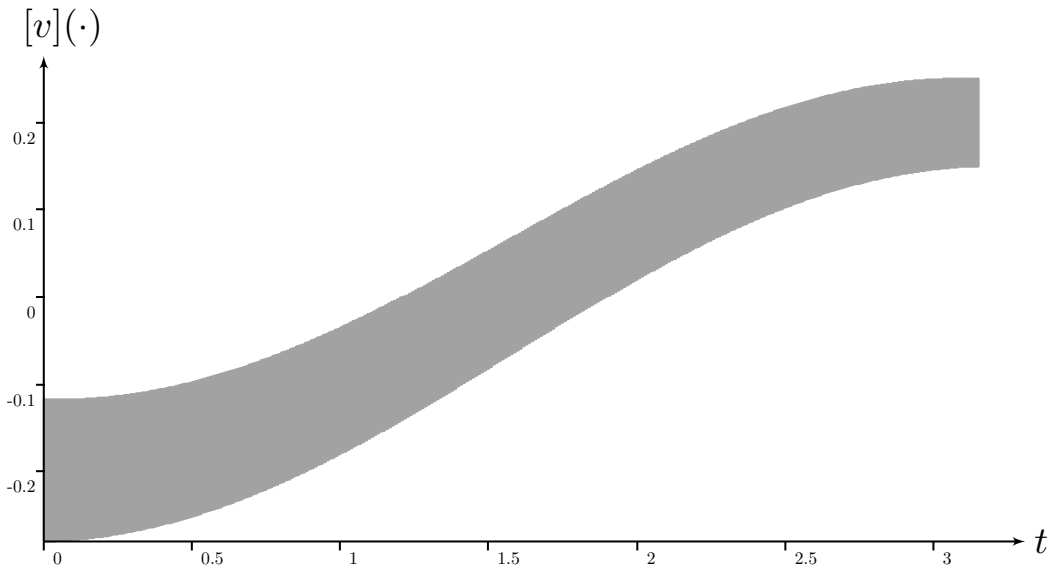
Differential constraint:

- $\mathcal{L}_{\frac{d}{dt}}\left(x(\cdot), v(\cdot)\right) : \dot{x}(\cdot) = v(\cdot)$
- elementary constraint

Related contractor $\mathcal{C}_{\frac{d}{dt}}$:

- one tube $[x](\cdot)$
- one tube $[v](\cdot)$
- $\mathcal{C}_{\frac{d}{dt}}\left([x](\cdot), [v](\cdot)\right)$





Guaranteed computation of robot trajectories

Rohou, Jaulin, Mihaylova, Le Bars, Veres

Robotics and Autonomous Systems, 2017

Section 3

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

Definition

$$\mathcal{L}_{\text{eval}} : \begin{cases} \textbf{Variables: } t,\ z,\ y(\cdot) \\ \textbf{Constraints:} \\ \quad 1.\ \ z = y(t) \\ \\ \textbf{Domains: } [t],\ [z],\ [y](\cdot) \end{cases}$$

$\mathcal{L}_{\text{eval}}$ equivalent to:
$\exists t \in [t],\ \exists z \in [z],\ \exists y(\cdot) \in [y](\cdot)\ \mid\ z = y(t)$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

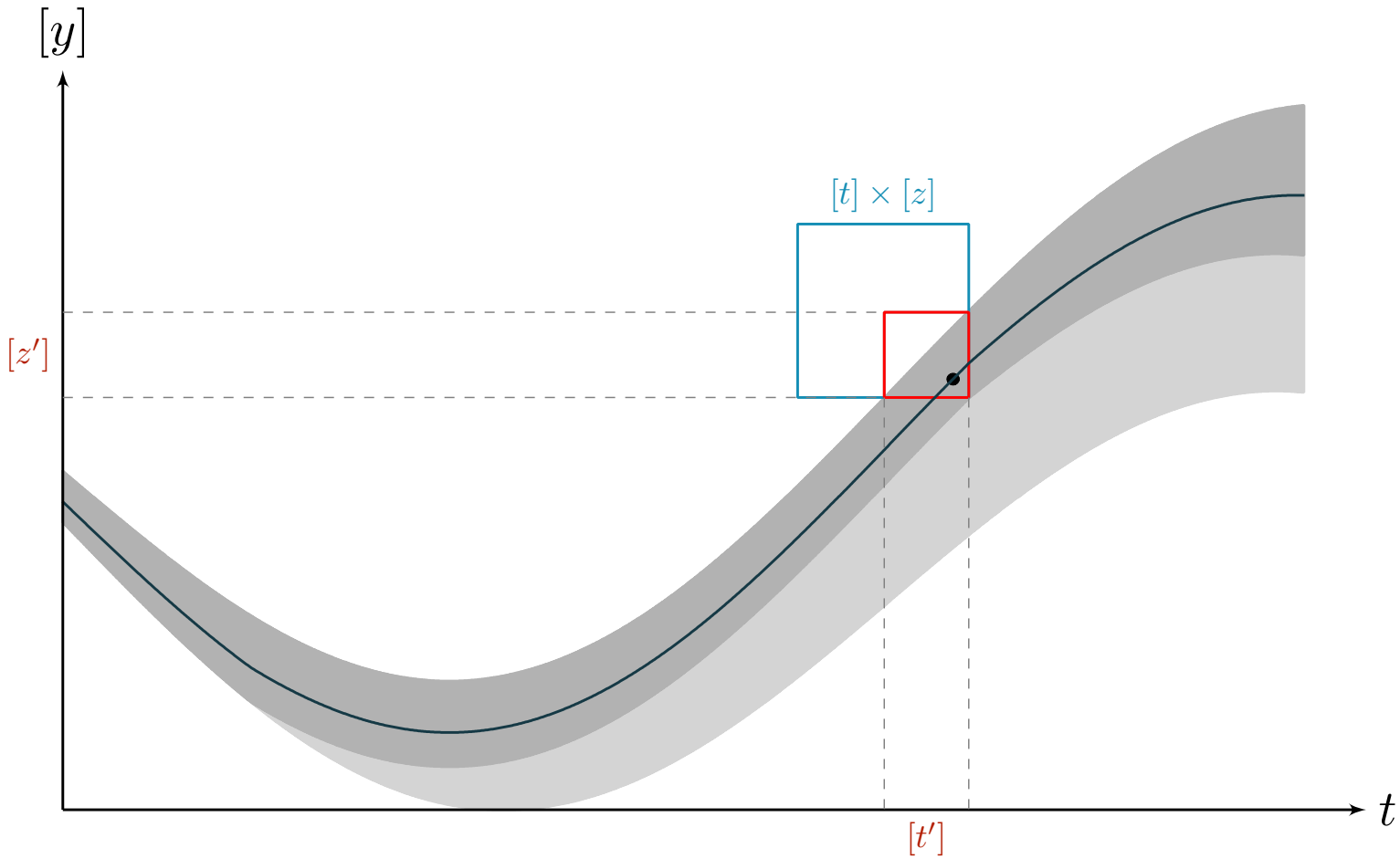
Definition

$$\mathcal{L}_{\text{eval}} : \begin{cases} \textbf{Variables: } t,\ z,\ y(\cdot),\ w(\cdot) \\ \textbf{Constraints:} \\ \quad 1.\ z = y(t) \\ \quad 2.\ \dot{y}(\cdot) = w(\cdot) \\ \textbf{Domains: } [t],\ [z],\ [y](\cdot),\ [w](\cdot) \end{cases}$$

$\mathcal{L}_{\text{eval}}$ equivalent to:
$\exists t \in [t],\ \exists z \in [z],\ \exists y(\cdot) \in [y](\cdot)\ \mid\ z = y(t)$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

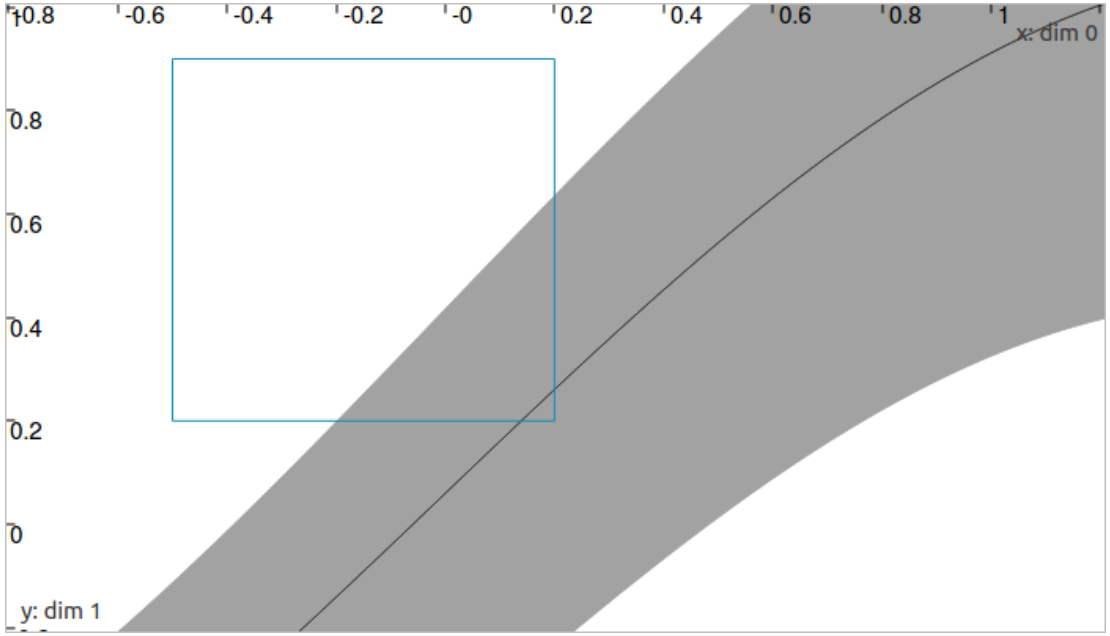
Definition



Bounded evaluation with contractions of $[y](\cdot)$ and both $[t]$ and $[z]$ by means of $\mathcal{C}_{\text{eval}}$. The tube's contracted part is depicted in light gray.

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

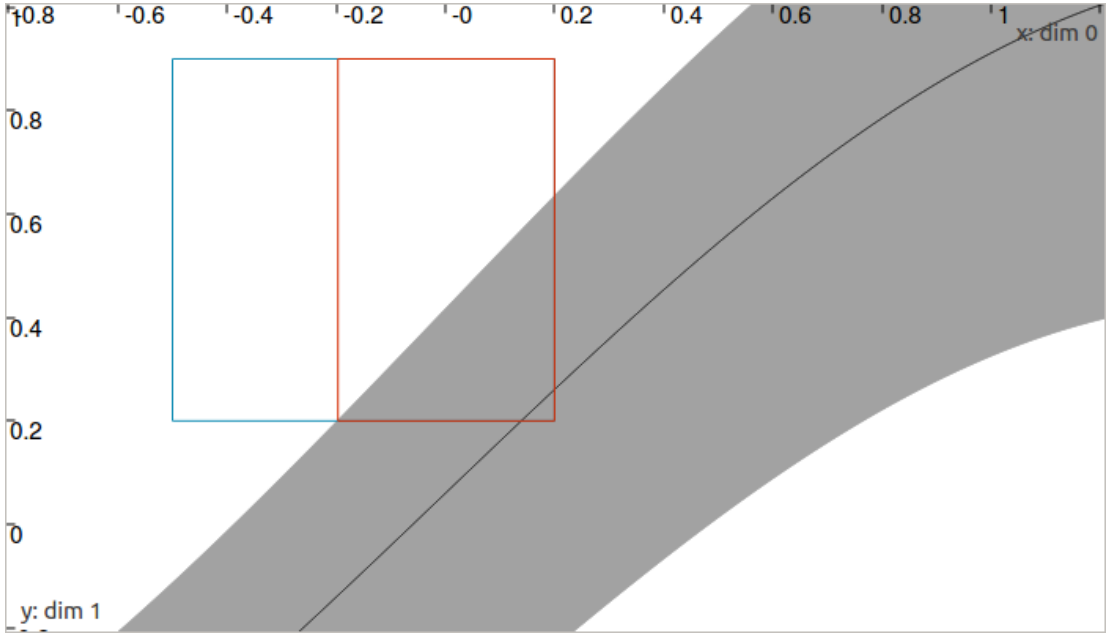
$$\mathcal{C}_{\text{eval}}\big([t],[z],[y](\cdot),[w](\cdot)\big)$$



$$\begin{pmatrix} [t] \\ [z] \\ [y](\cdot) \\ \\ [w](\cdot) \end{pmatrix} \xmapsto{\mathcal{C}_{\text{eval}}} \begin{pmatrix} \\ \\ \\ \\ \\ \end{pmatrix}$$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

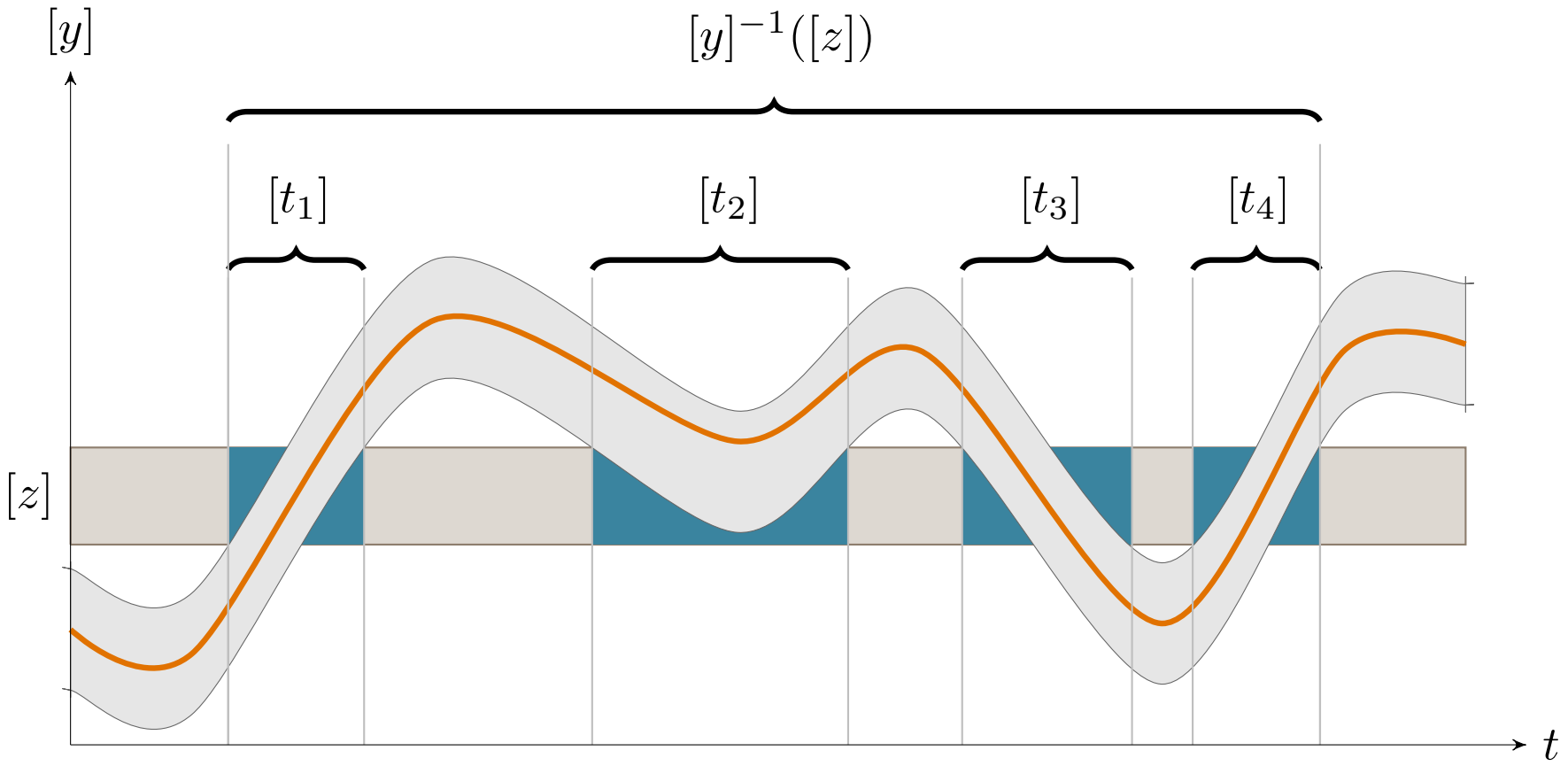
$$\mathcal{C}_{\text{eval}}\big([t],[z],[y](\cdot),[w](\cdot)\big)$$



$$\begin{pmatrix} [t] \\ [z] \\ [y](\cdot) \\ [w](\cdot) \end{pmatrix} \xmapsto{\mathcal{C}_{\text{eval}}} \begin{pmatrix} [t] \cap [y]^{-1}([z]) \end{pmatrix}$$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

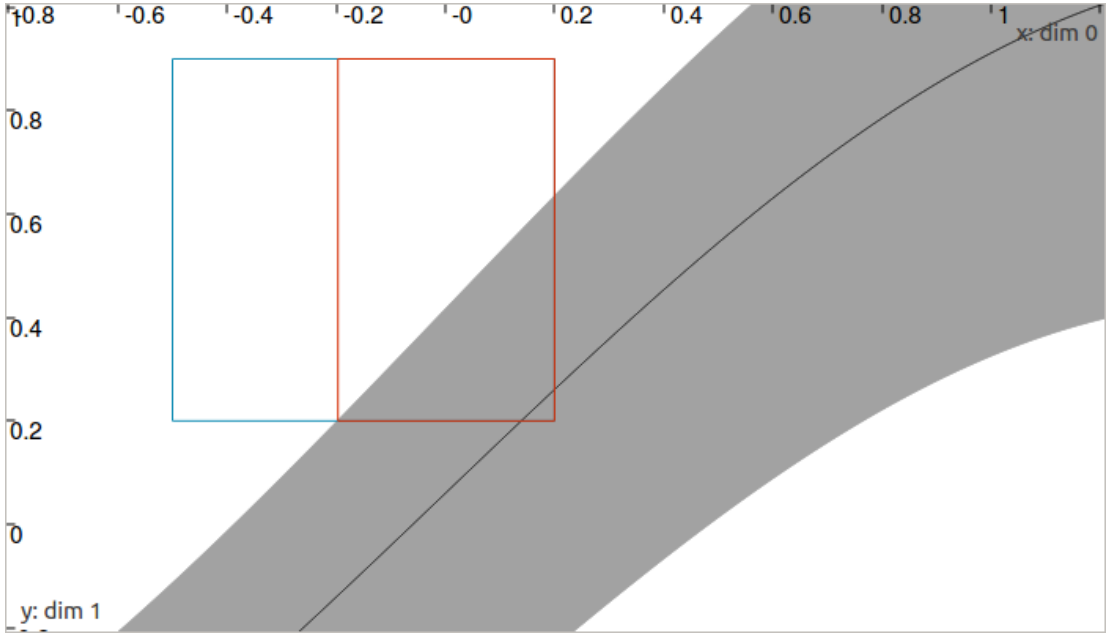
$$\mathcal{C}_{\text{eval}}\big([t],[z],[y](\cdot),[w](\cdot)\big)$$



Tube set-inversion $[y]^{-1}([z]) = \bigsqcup_{z\in[z]} \{t \mid y \in [y](t)\}$.

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

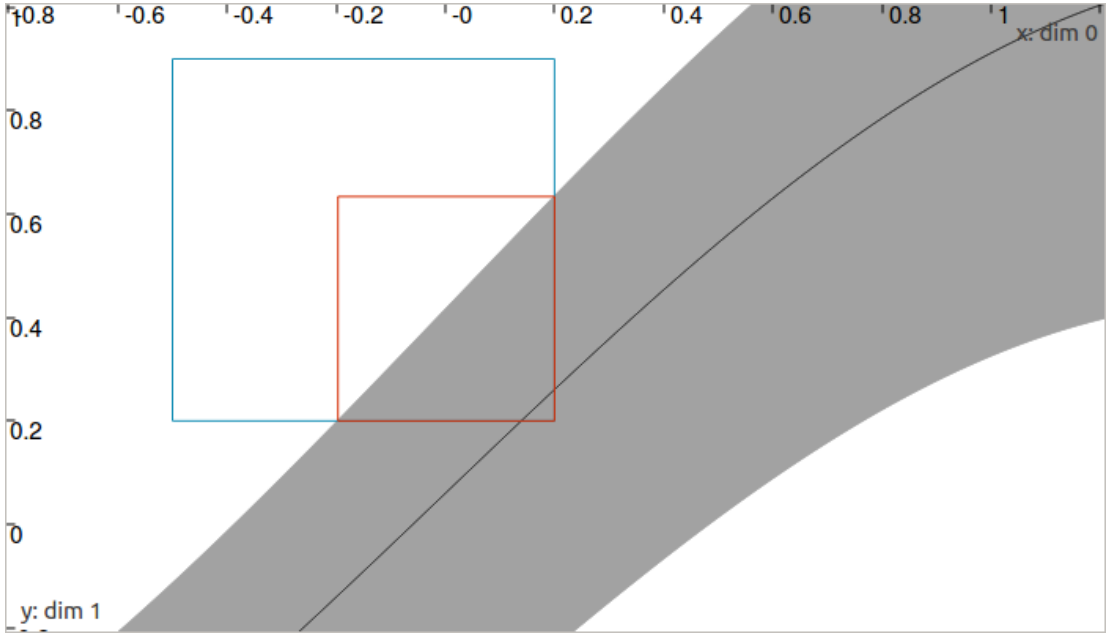
$$\mathcal{C}_{\text{eval}}\big([t],[z],[y](\cdot),[w](\cdot)\big)$$



$$\begin{pmatrix} [t] \\ [z] \\ [y](\cdot) \\ [w](\cdot) \end{pmatrix} \xmapsto{\mathcal{C}_{\text{eval}}} \begin{pmatrix} [t] \cap [y]^{-1}([z]) \end{pmatrix}$$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

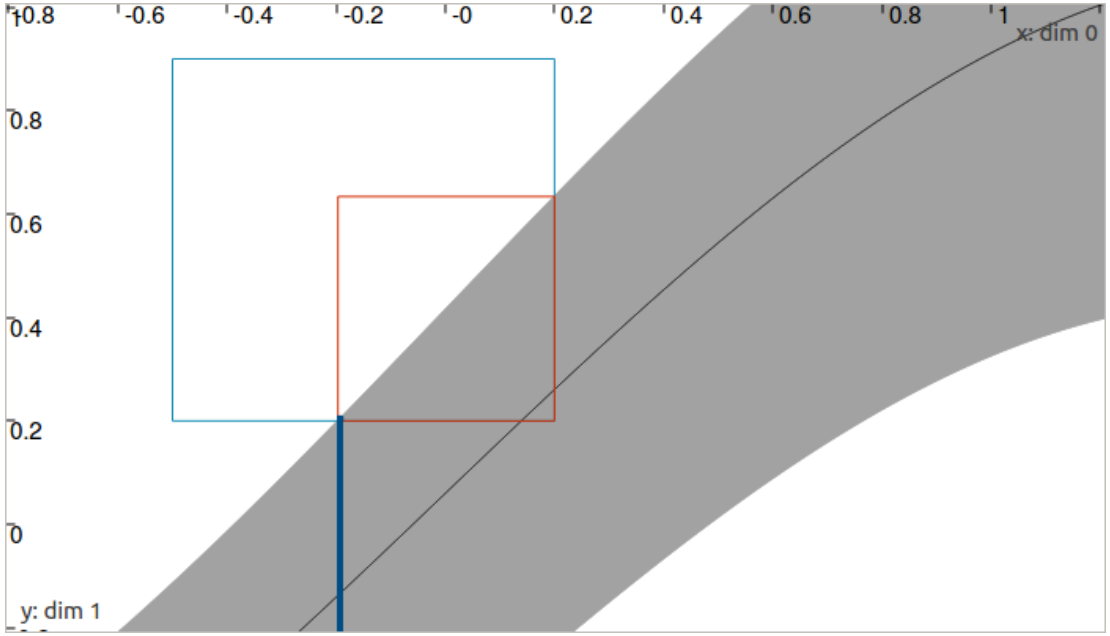
$$\mathcal{C}_{\text{eval}}\big([t],[z],[y](\cdot),[w](\cdot)\big)$$



$$\begin{pmatrix} [t] \\ [z] \\ [y](\cdot) \\ [w](\cdot) \end{pmatrix} \xmapsto{\mathcal{C}_{\text{eval}}} \begin{pmatrix} [t] \cap [y]^{-1}([z]) \\ [z] \cap [y]([t]) \end{pmatrix}$$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

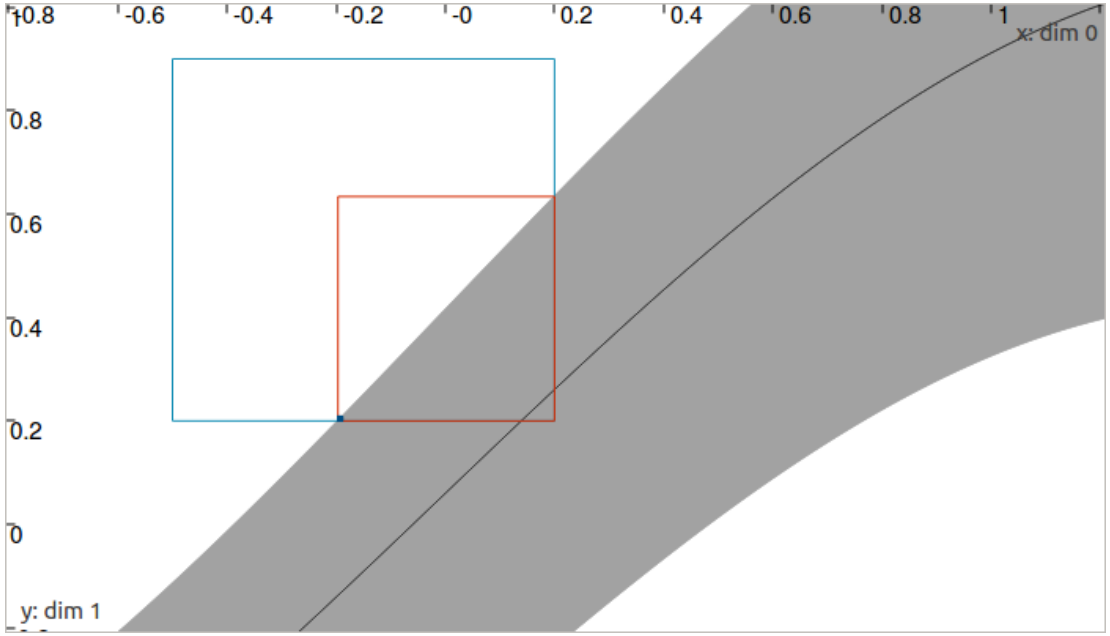
$$\mathcal{C}_{\text{eval}}\big([t],[z],[y](\cdot),[w](\cdot)\big)$$



$$\begin{pmatrix} [t] \\ [z] \\ [y](\cdot) \\ [w](\cdot) \end{pmatrix} \xmapsto{\mathcal{C}_{\text{eval}}} \begin{pmatrix} [t] \cap [y]^{-1}([z]) \\ [z] \cap [y]([t]) \\ \begin{pmatrix} [y](t_1) \end{pmatrix} \end{pmatrix}$$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

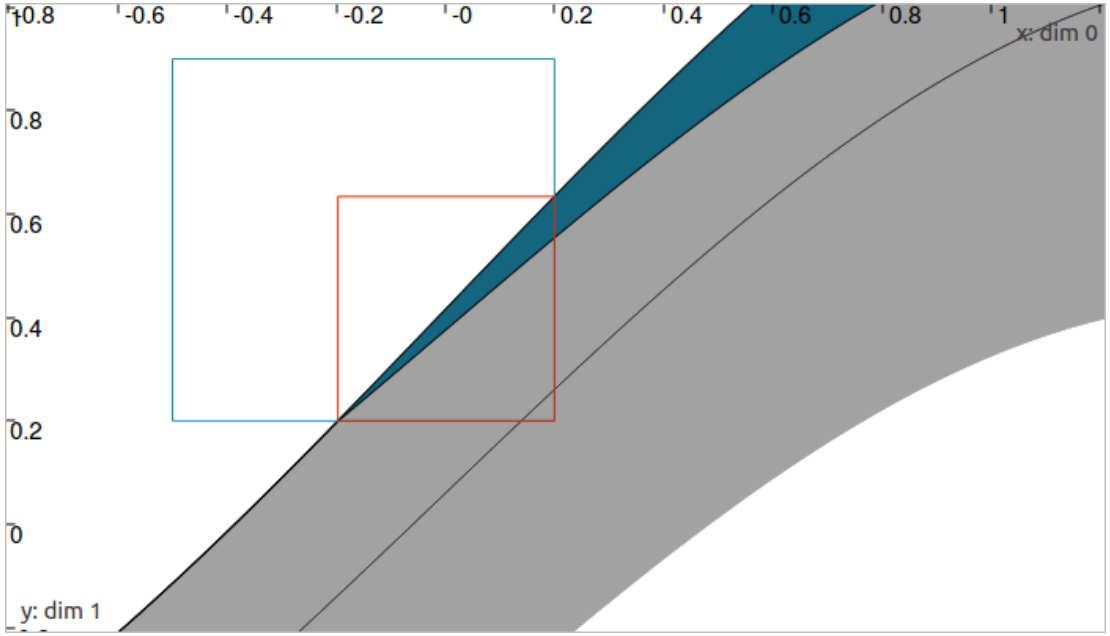
$$\mathcal{C}_{\text{eval}}\big([t],[z],[y](\cdot),[w](\cdot)\big)$$



$$\begin{pmatrix} [t] \\ [z] \\ [y](\cdot) \\ [w](\cdot) \end{pmatrix} \xmapsto{\mathcal{C}_{\text{eval}}} \begin{pmatrix} [t] \cap [y]^{-1}([z]) \\ [z] \cap [y]([t]) \\ \begin{pmatrix} ([y](t_1) \cap [z]) \end{pmatrix} \end{pmatrix}$$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

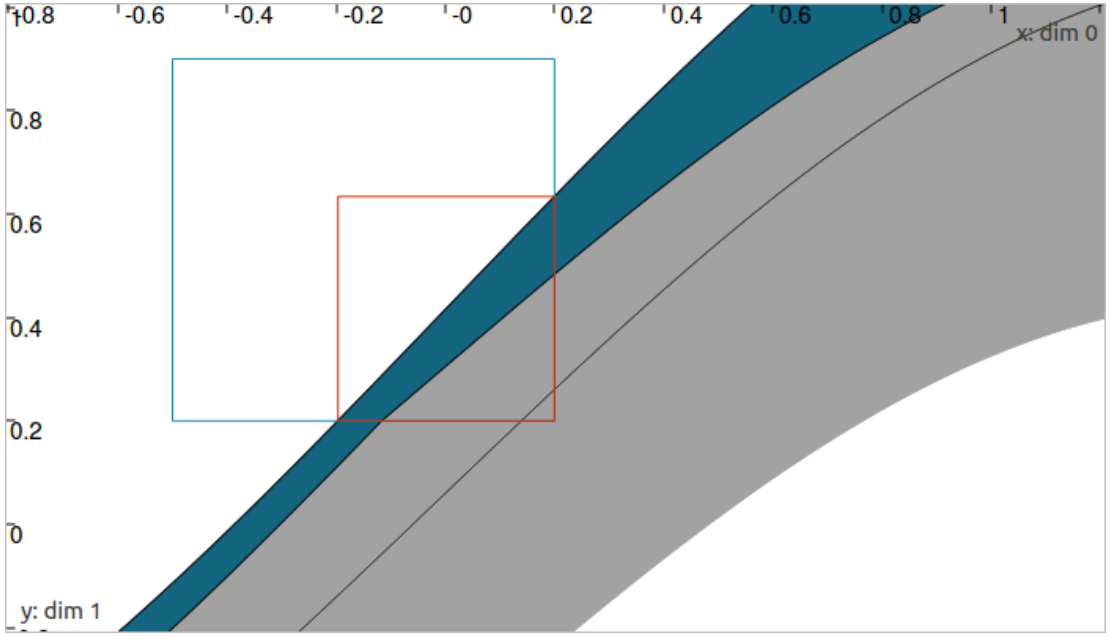
$$\mathcal{C}_{\text{eval}}\big([t],[z],[y](\cdot),[w](\cdot)\big)$$



$$\begin{pmatrix} [t] \\ [z] \\ [y](\cdot) \\ \\ [w](\cdot) \end{pmatrix} \xmapsto{\mathcal{C}_{\text{eval}}} \begin{pmatrix} [t] \cap [y]^{-1}([z]) \\ [z] \cap [y]([t]) \\ \left(([y](t_1) \cap [z]) + \int_{t_1}^{\cdot} [w](\tau) d\tau \right) \end{pmatrix}$$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

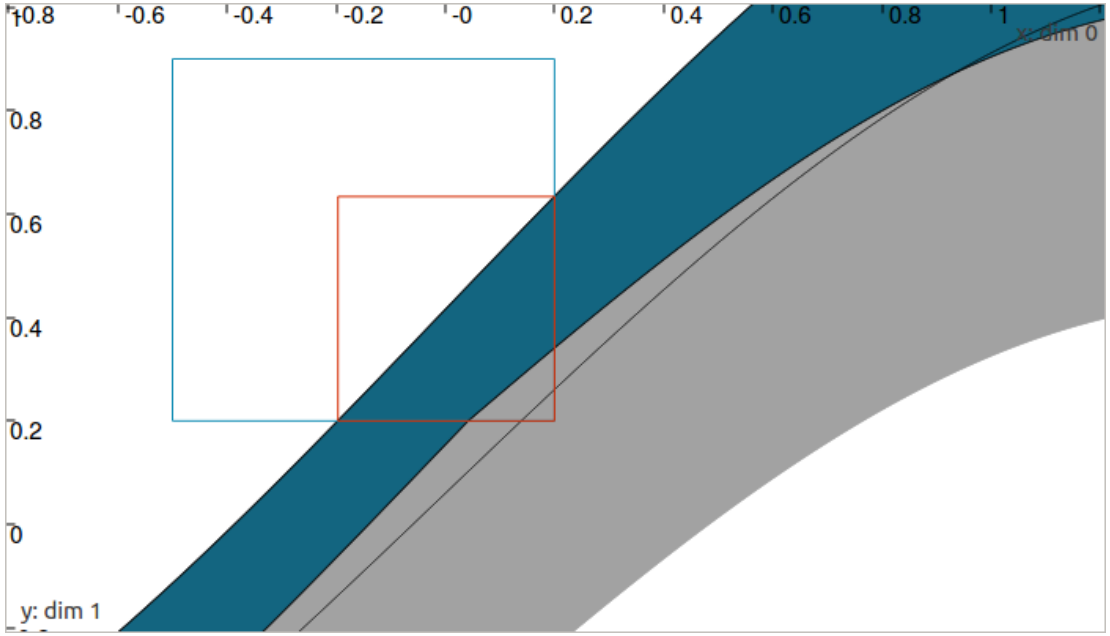
$$\mathcal{C}_{\text{eval}}\big([t],[z],[y](\cdot),[w](\cdot)\big)$$



$$\begin{pmatrix} [t] \\ [z] \\ [y](\cdot) \\ \\ [w](\cdot) \end{pmatrix} \xrightarrow{\mathcal{C}_{\text{eval}}} \begin{pmatrix} [t] \cap [y]^{-1}([z]) \\ [z] \cap [y]([t]) \\ \bigsqcup_{t_1 \in [t]} \left(([y](t_1) \cap [z]) + \int_{t_1}^{\cdot} [w](\tau) d\tau \right) \end{pmatrix}$$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

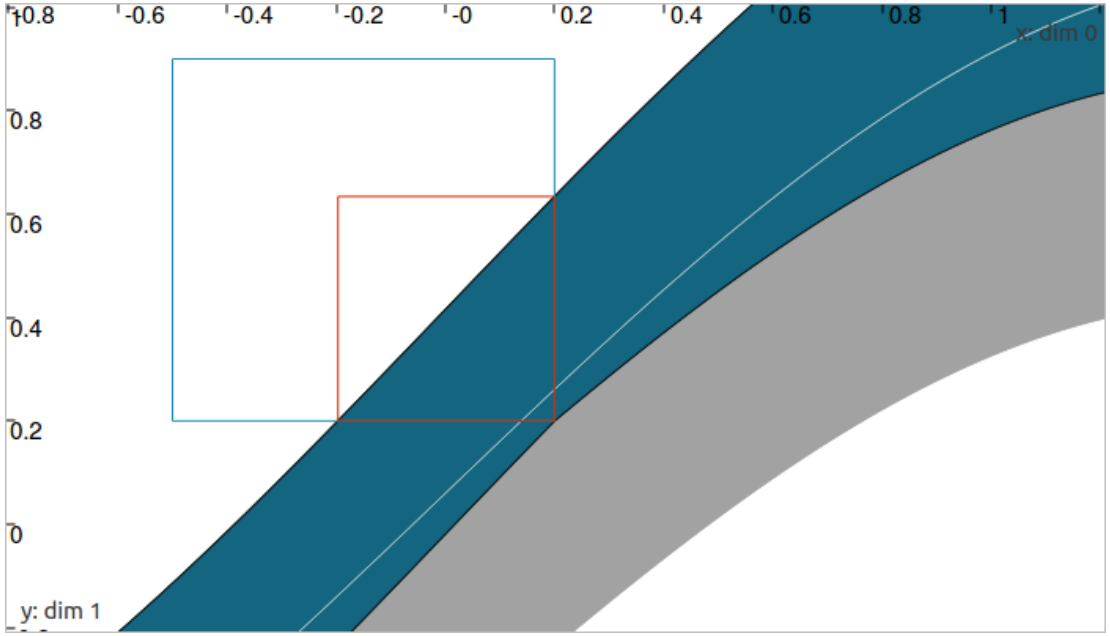
$$\mathcal{C}_{\text{eval}}\big([t],[z],[y](\cdot),[w](\cdot)\big)$$



$$\begin{pmatrix} [t] \\ [z] \\ [y](\cdot) \\ \\ [w](\cdot) \end{pmatrix} \longmapsto_{\mathcal{C}_{\text{eval}}} \begin{pmatrix} [t] \cap [y]^{-1}([z]) \\ [z] \cap [y]([t]) \\ \bigsqcup_{t_1 \in [t]} \left(([y](t_1) \cap [z]) + \int_{t_1}^{\cdot} [w](\tau) d\tau \right) \end{pmatrix}$$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

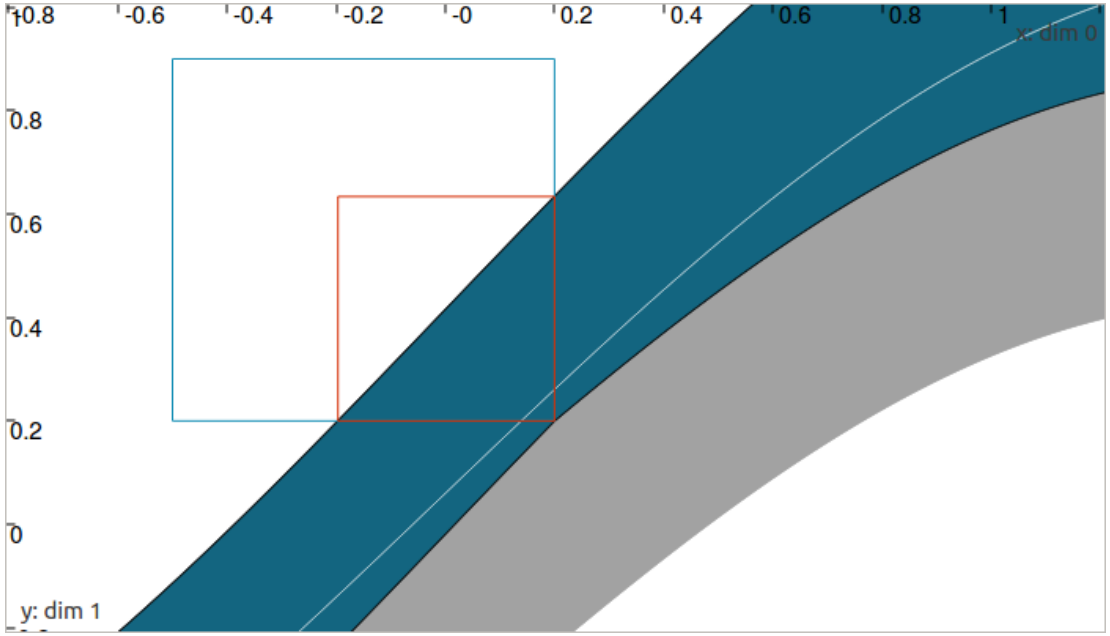
$$\mathcal{C}_{\text{eval}}\big([t],[z],[y](\cdot),[w](\cdot)\big)$$



$$\begin{pmatrix} [t] \\ [z] \\ [y](\cdot) \\ \\ [w](\cdot) \end{pmatrix} \longmapsto^{\mathcal{C}_{\text{eval}}} \begin{pmatrix} [t] \cap [y]^{-1}([z]) \\ [z] \cap [y]([t]) \\ \bigsqcup_{t_1 \in [t]} \left(([y](t_1) \cap [z]) + \int_{t_1}^{\cdot} [w](\tau) d\tau \right) \end{pmatrix}$$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

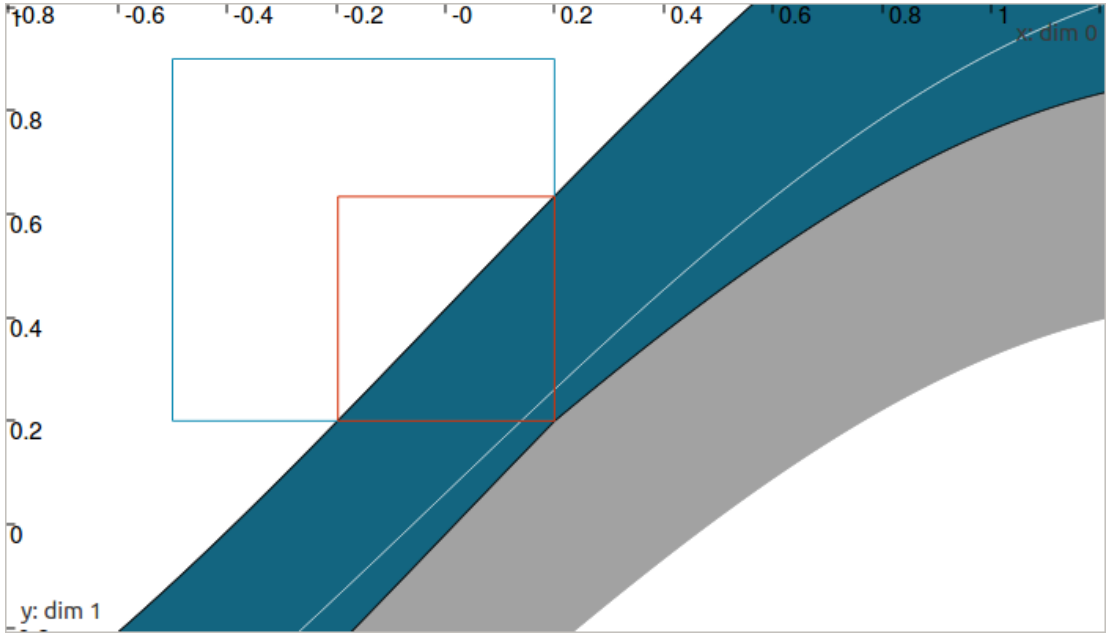
$$\mathcal{C}_{\text{eval}}\big([t],[z],[y](\cdot),[w](\cdot)\big)$$



$$\begin{pmatrix} [t] \\ [z] \\ [y](\cdot) \\ \\ [w](\cdot) \end{pmatrix} \xrightarrow{\mathcal{C}_{\text{eval}}} \begin{pmatrix} [t] \cap [y]^{-1}([z]) \\ [z] \cap [y]([t]) \\ [y](\cdot) \cap \bigsqcup_{t_1 \in [t]} \left(([y](t_1) \cap [z]) + \int_{t_1}^{\cdot} [w](\tau) d\tau \right) \end{pmatrix}$$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

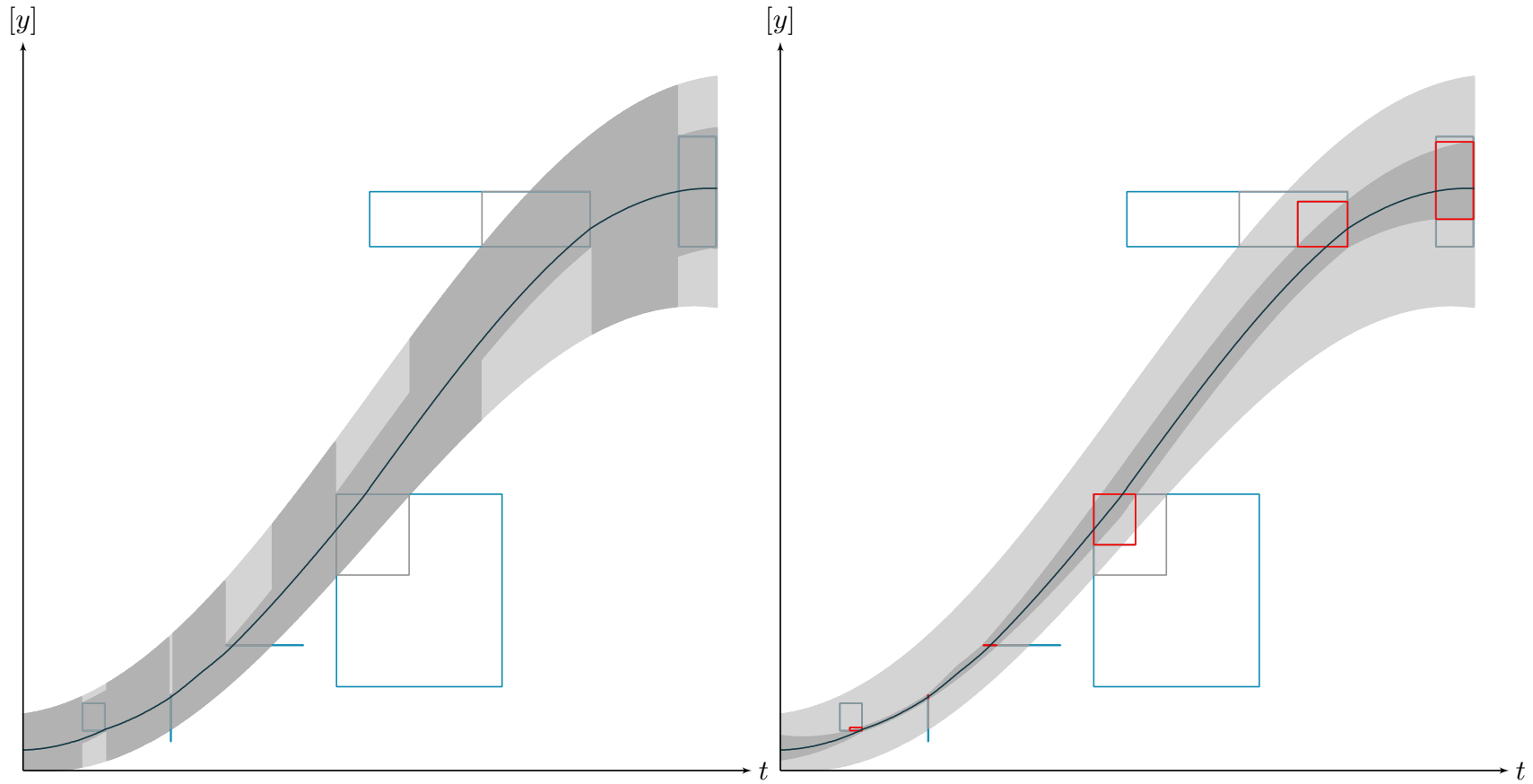
$$\mathcal{C}_{\text{eval}}\big([t],[z],[y](\cdot),[w](\cdot)\big)$$



$$\begin{pmatrix} [t] \\ [z] \\ [y](\cdot) \\ \\ [w](\cdot) \end{pmatrix} \xrightarrow{\mathcal{C}_{\text{eval}}} \begin{pmatrix} [t] \cap [y]^{-1}([z]) \\ [z] \cap [y]([t]) \\ [y](\cdot) \cap \bigsqcup_{t_1 \in [t]} \left(([y](t_1) \cap [z]) + \int_{t_1}^{\cdot} [w](\tau) d\tau \right) \\ [w](\cdot) \end{pmatrix}$$

Constraint $\mathcal{L}_{\text{eval}}$: $z = y(t)$

Several evaluations: fixed point iteration



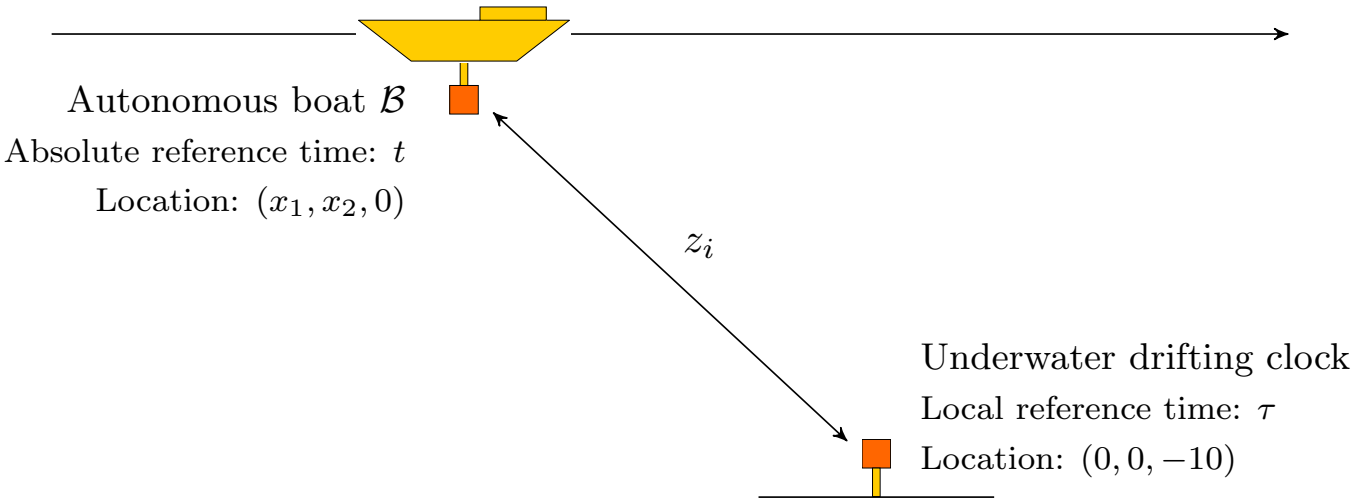
Left: one iteration. Right: fixed point result.

Section 4

Application: drifting clock

Application: drifting clock

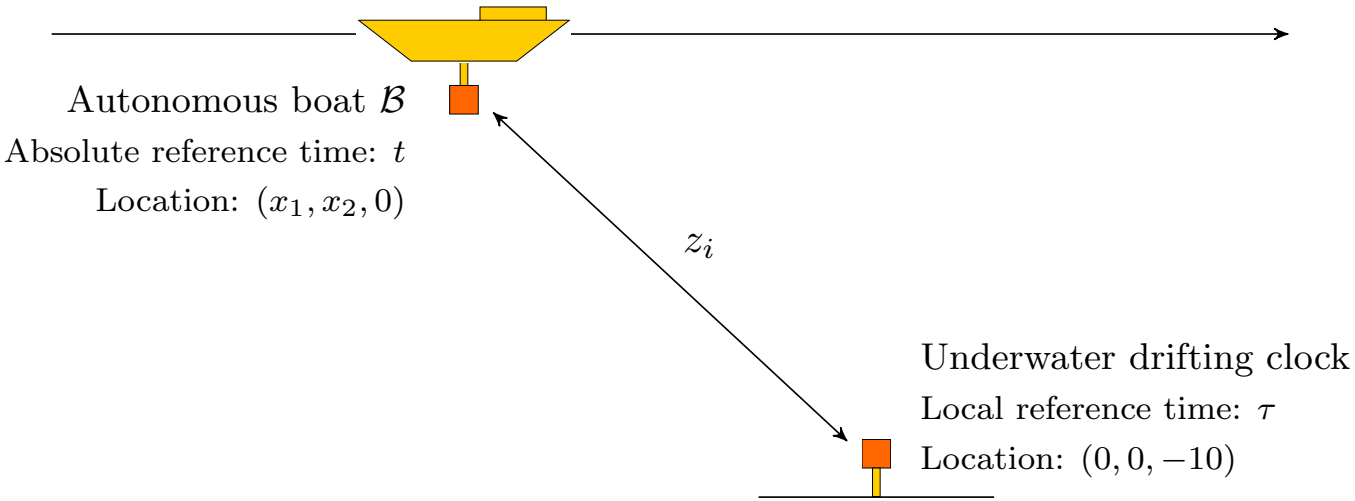
Underwater system equipped with a low-cost drifting clock



- absolute time reference represented by t
- underwater clock providing a drifting value τ:
  - $\tau = h(t)$
  - unknown: $h(t) = 0.045t^2 + 0.98t$

Application: drifting clock

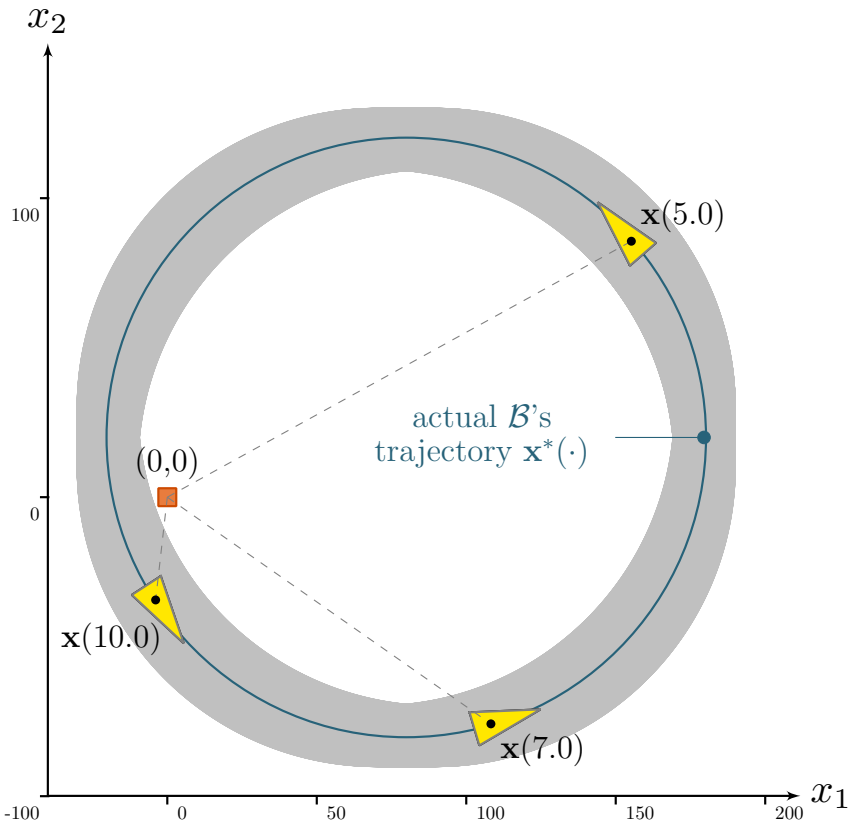
Underwater system equipped with a low-cost drifting clock



- absolute time reference represented by t
- underwater clock providing a drifting value τ:
  - $\tau = h(t)$
  - unknown: $h(t) = 0.045t^2 + 0.98t$
  - clock's datasheet: $\dot{h}(t) \in [0.08, 0.12] \cdot t + [0.97, 1.08]$

Application: drifting clock

Boat $\mathcal{B}$ following a preprogrammed trajectory



Top view of $\mathcal{B}$'s trajectory around the underwater beacon.

Preprogrammed trajectory $\mathbf{x}(\cdot)$ (ephemeris) assumed as:

$$\mathbf{x}(\cdot) \in \begin{pmatrix} [70,90] \\ [10,30] \end{pmatrix} + 100 \begin{pmatrix} \cos(\cdot) \\ \sin(\cdot) \end{pmatrix}$$

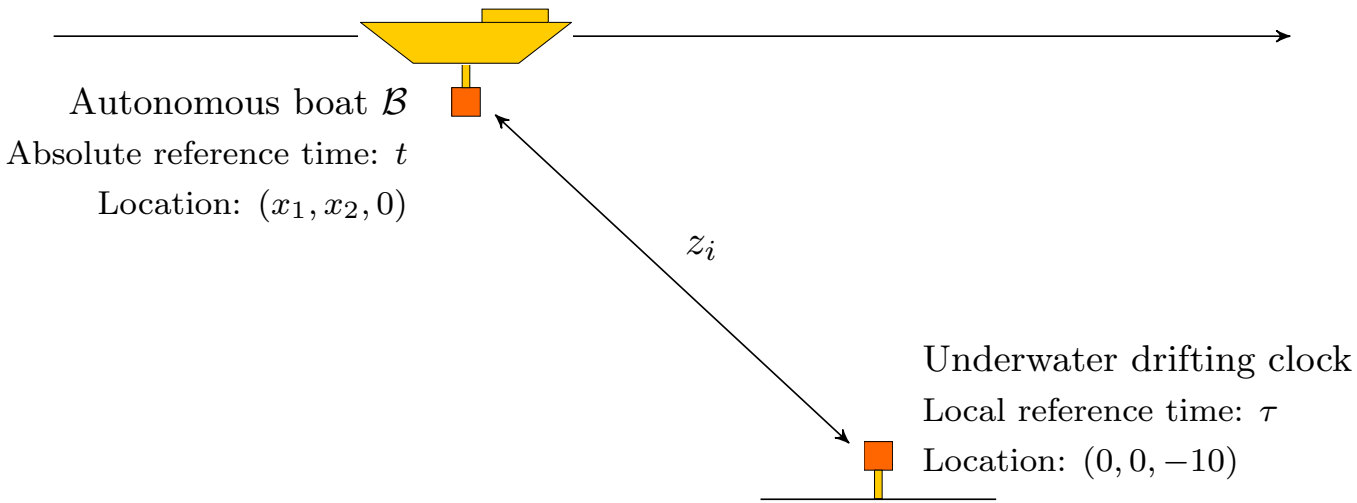
Bounded $\mathcal{B}$'s velocities:

$$\mathbf{v}(\cdot) \in \tfrac{1}{10} \begin{pmatrix} [-1,1] \\ [-1,1] \end{pmatrix} + 100 \begin{pmatrix} -\sin(\cdot) \\ \cos(\cdot) \end{pmatrix}$$

Application: drifting clock

List of measurements $(\tau_i, [z_i])$

i	τ_i	$[z_i]$	i	τ_i	$[z_i]$
1	1.57	$[152.47, 156.47]$	5	9.88	$[167.09, 171.09]$
2	3.34	$[34.67, 38.67]$	6	12.46	$[60.03, 64.03]$
3	5.32	$[102.38, 106.38]$	7	15.25	$[78.76, 82.76]$
4	7.50	$[184.45, 188.45]$	8	18.24	$[175.88, 179.88]$



Application: drifting clock

Resolution

Constraint Network:

Variables:

Domains:

Constraints:

Application: drifting clock

Resolution

Constraint Network:

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

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

Constraints:

1. Ephemerides (*i.e.* boat's locations):
   - $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$

Application: drifting clock

Resolution

Constraint Network:

Variables: $\mathbf{x}(\cdot)$, $\mathbf{v}(\cdot)$, $y(\cdot)$

Domains: $[\mathbf{x}](\cdot)$, $[\mathbf{v}](\cdot)$, $[y](\cdot)$

Constraints:

1. Ephemerides (*i.e.* boat's locations):
   - $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$
2. Beacon-boat distance function:
   - $y(\cdot) = \sqrt{x_1(\cdot)^2 + x_2(\cdot)^2 + (-10)^2}$

Application: drifting clock

Resolution

Constraint Network:

Variables: $\mathbf{x}(\cdot)$, $\mathbf{v}(\cdot)$, $y(\cdot)$, $h(\cdot)$, $\phi(\cdot)$

Domains: $[\mathbf{x}](\cdot)$, $[\mathbf{v}](\cdot)$, $[y](\cdot)$, $[h](\cdot)$, $[\phi](\cdot)$

Constraints:

1. Ephemerides (*i.e.* boat's locations):
   - $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$
2. Beacon-boat distance function:
   - $y(\cdot) = \sqrt{x_1(\cdot)^2 + x_2(\cdot)^2 + (-10)^2}$
3. Drifting time function:
   - $\dot{h}(\cdot) = \phi(\cdot)$
   - $h(0) = 0$

Application: drifting clock

Resolution

Constraint Network:

Variables: $\mathbf{x}(\cdot)$, $\mathbf{v}(\cdot)$, $y(\cdot)$, $h(\cdot)$, $\phi(\cdot)$, $\{(t_i, z_i)\}$

Domains: $[\mathbf{x}](\cdot)$, $[\mathbf{v}](\cdot)$, $[y](\cdot)$, $[h](\cdot)$, $[\phi](\cdot)$, $\{([t_i], [z_i])\}$

Constraints:

1. Ephemerides (*i.e.* boat's locations):
   - $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$
2. Beacon-boat distance function:
   - $y(\cdot) = \sqrt{x_1(\cdot)^2 + x_2(\cdot)^2 + (-10)^2}$
3. Drifting time function:
   - $\dot{h}(\cdot) = \phi(\cdot)$
   - $h(0) = 0$
4. Measurements:
   - $z_i = y(t_i)$
   - $\tau_i = h(t_i)$

Application: drifting clock

Resolution

Constraint Network:

Variables: $\mathbf{x}(\cdot)$, $\mathbf{v}(\cdot)$, $y(\cdot)$, $h(\cdot)$, $\phi(\cdot)$, $\{(t_i, z_i)\}$, $w(\cdot)$

Domains: $[\mathbf{x}](\cdot)$, $[\mathbf{v}](\cdot)$, $[y](\cdot)$, $[h](\cdot)$, $[\phi](\cdot)$, $\{([t_i], [z_i])\}$, $[w](\cdot)$

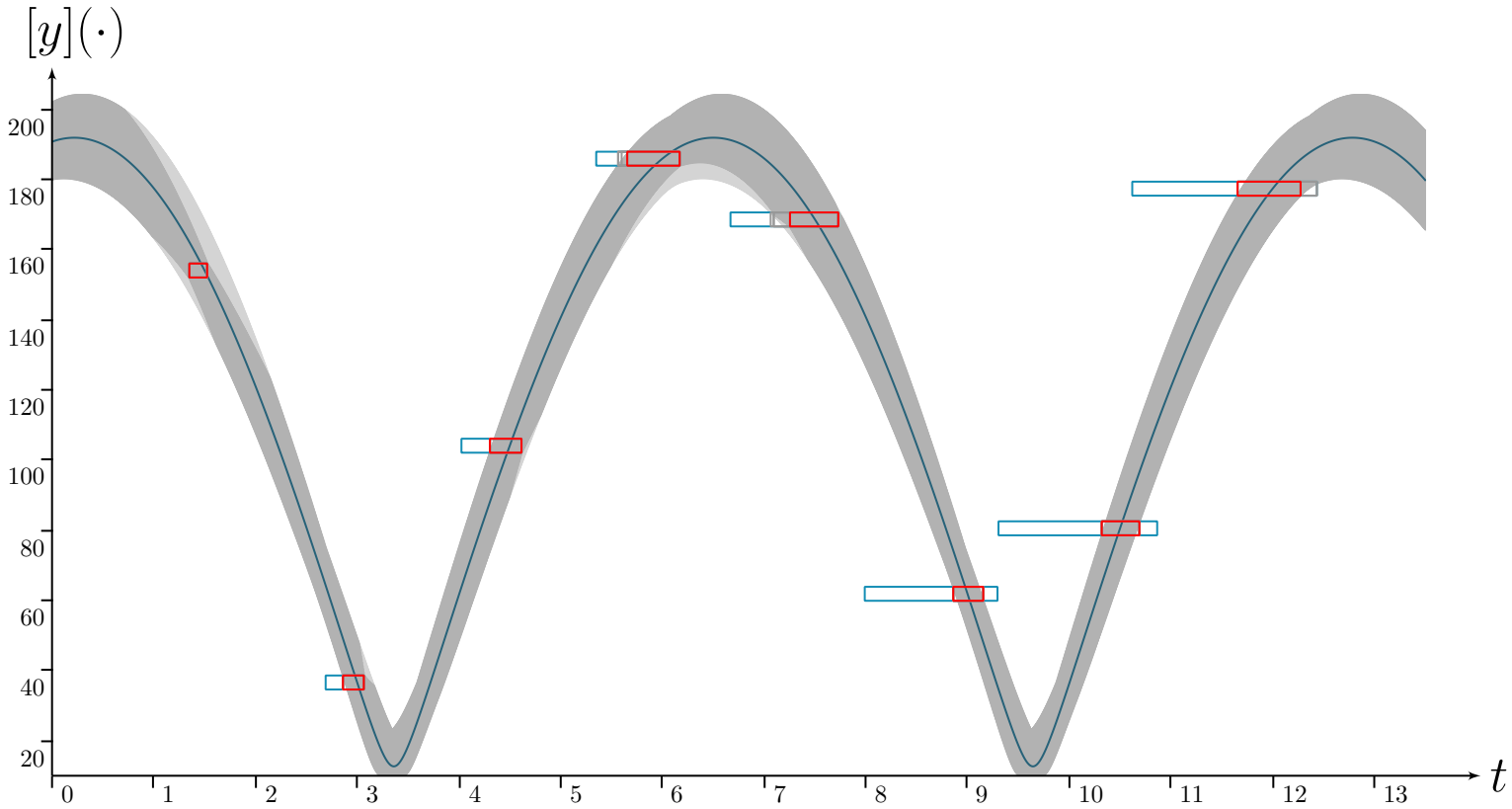
Constraints:

1. Ephemerides (*i.e.* boat's locations):
   - $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$
2. Beacon-boat distance function:
   - $y(\cdot) = \sqrt{x_1(\cdot)^2 + x_2(\cdot)^2 + (-10)^2}$
   - $w_k(\cdot) = \big(x_1(\cdot) \cdot v_1(\cdot) + x_2(\cdot) \cdot v_2(\cdot)\big)/y(\cdot)$
   - $\dot{y}_k(\cdot) = w_k(\cdot)$
3. Drifting time function:
   - $\dot{h}(\cdot) = \phi(\cdot)$
   - $h(0) = 0$
4. Measurements:
   - $z_i = y(t_i)$
   - $\tau_i = h(t_i)$

Application: drifting clock

Resolution: contracting the measurements from $[y](\cdot)$

1. $[t_i] = [h]^{-1}(\tau_i)$
2. $\big([t_i], [z_i], [y](\cdot), [w](\cdot)\big) \xmapsto{\mathcal{C}_{\text{eval}}} \big([t_i], [z_i], [y](\cdot), [w](\cdot)\big)$

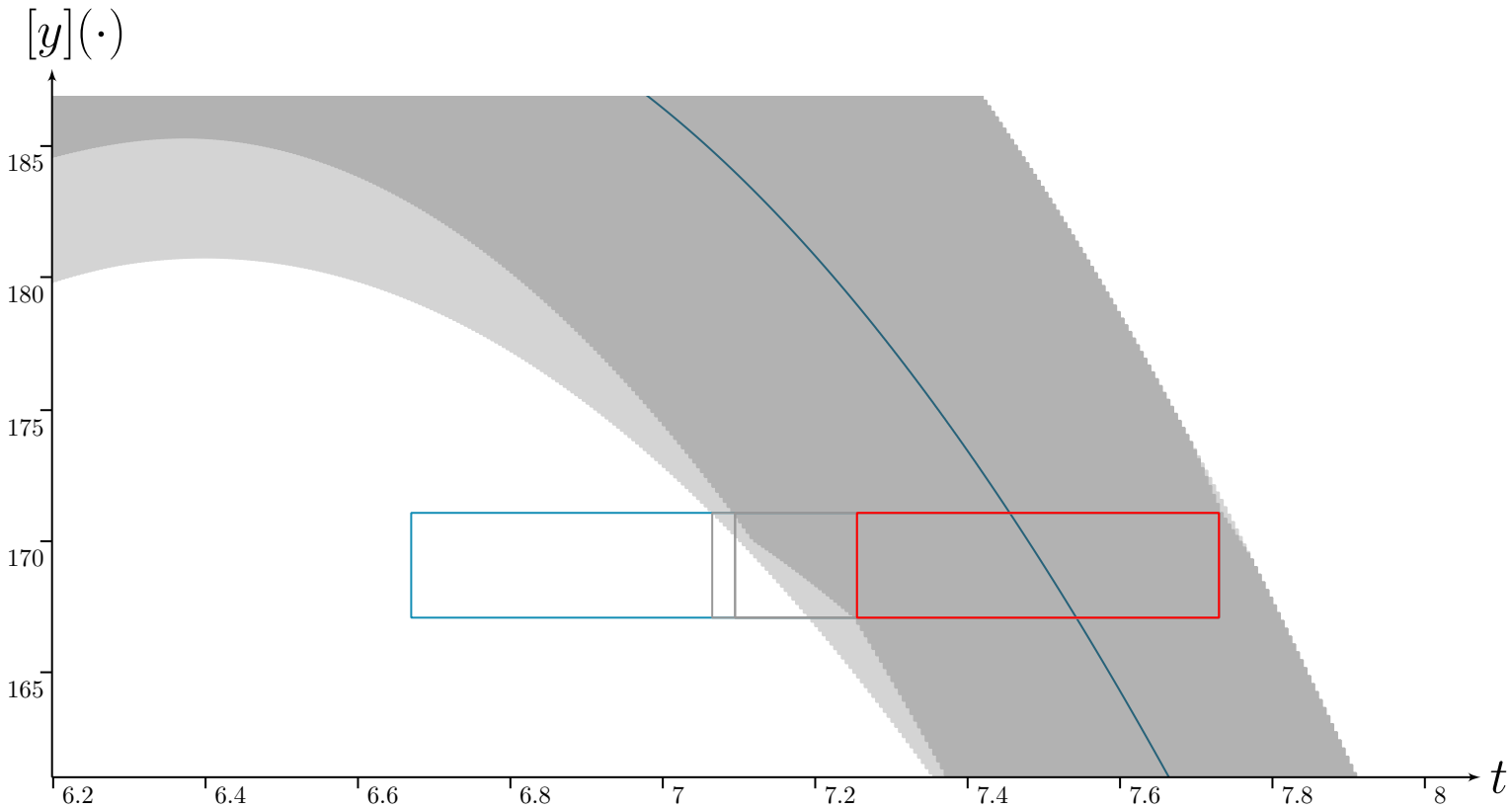


Tube $[y](\cdot)$: reliable prevision of the distances between the boat and the beacon.

Application: drifting clock

Resolution: contracting the measurements from $[y](\cdot)$

1. $[t_i] = [h]^{-1}(\tau_i)$
2. $\left([t_i], [z_i], [y](\cdot), [w](\cdot)\right) \xmapsto{\mathcal{C}_{\text{eval}}} \left([t_i], [z_i], [y](\cdot), [w](\cdot)\right)$

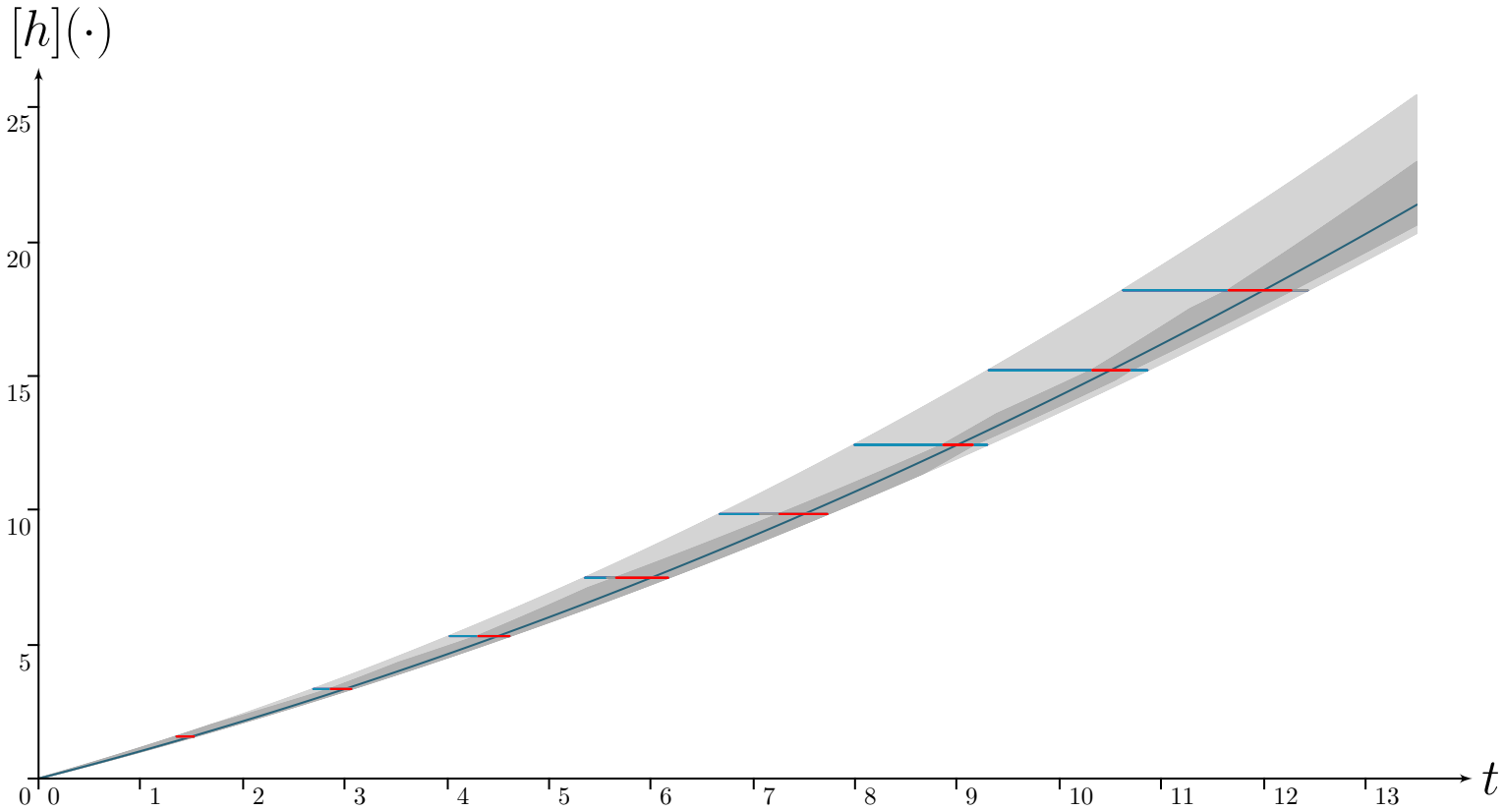


Tube $[y](\cdot)$: reliable prevision of the distances between the boat and the beacon.

Application: drifting clock

Resolution: contracting $[h](\cdot)$ from the measurements

$$\big([t_i], \tau_i, [h](\cdot), [\phi](\cdot)\big) \xmapsto{\mathcal{C}_{\text{eval}}} \big([t_i], \tau_i, [h](\cdot), [\phi](\cdot)\big)$$

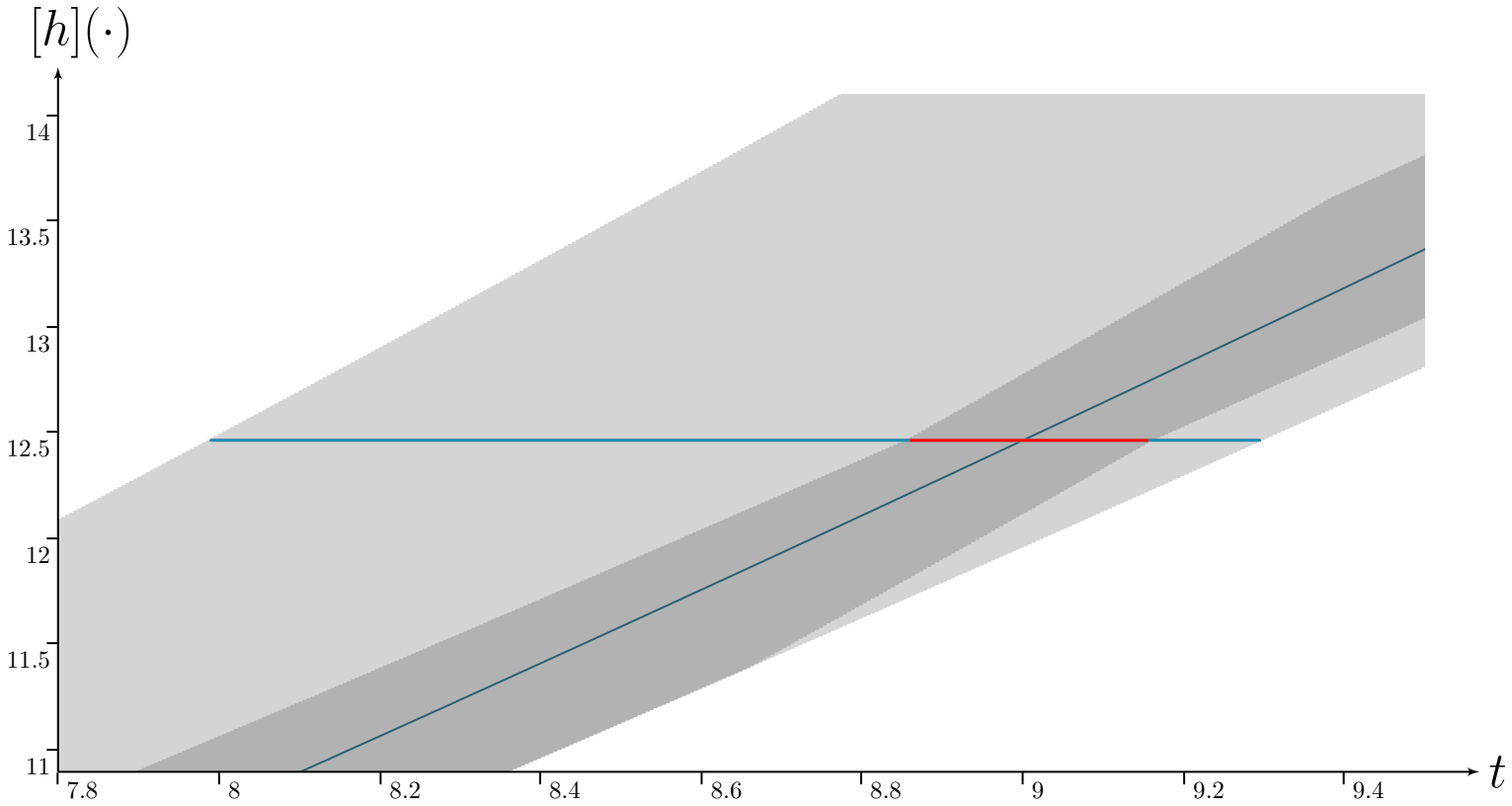


Tube $[h](\cdot)$: clock's drift.

Application: drifting clock

Resolution: contracting $[h](\cdot)$ from the measurements

$$\big([t_i], \tau_i, [h](\cdot), [\phi](\cdot)\big) \xmapsto{\mathcal{C}_{\text{eval}}} \big([t_i], \tau_i, [h](\cdot), [\phi](\cdot)\big)$$



Tube $[h](\cdot)$: clock's drift.

Section 5

Conclusions

Conclusion

To conclude:

- original method to deal with time uncertainties
- systems can be differential and non-linear
- elementary tool in the contractor programming framework
- $\mathcal{C}_{\text{eval}}$ now allows one to consider state estimation problems from a temporal point of view where the time t becomes an unknown variable to be estimated

Prospects:

- wreck-based localization problem

References

Contractor Programming
G. Chabert, L. Jaulin. *Artificial Intelligence*, 2009

A Constraint Satisfaction Approach for Enclosing Solutions to Parametric ODEs
M. Janssen, P. Van Hentenryck, Y. Deville. *SIAM Journal on Numerical Analysis*, 2002

Analytic constraint solving and interval arithmetic
T. J. Hickey. *ACM Press*, 2000

Constraint Satisfaction Differential Problems
J. Cruz, P. Barahona. *Springer Berlin Heidelberg*, 2003

Set-membership state estimation with fleeting data
F. Le Bars, J. Sliwka, L. Jaulin, O. Reynet *Automatica*, 2012

Solving Non-Linear Constraint Satisfaction Problems Involving Time-Dependant Functions
A. Bethencourt, L. Jaulin. *Mathematics in Computer Science*, 2014

Guaranteed computation of robot trajectories
S. Rohou, L. Jaulin, L. Mihaylova, F. Le Bars, S. M. Veres. *Robotics and Autonomous Systems*, 2017

Reliable non-linear state estimation involving time uncertainties
S. Rohou, L. Jaulin, L. Mihaylova, F. Le Bars, S. M. Veres. *Automatica*, 2018