Reliable robot localization: a constraint programming approach over dynamical systems

Simon Rohou

ENSTA Bretagne, Lab-STICC, UMR CNRS 6285, Brest, France
The University of Sheffield, Sheffield, UK
simon.rohou@gmail.com

26th January 2018

Outline

1. Motivations
2. SLAM formalization
3. Constraint programming
4. Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$
5. Bathymetric SLAM
6. Conclusions

Section 1

Motivations

Motivations

Motivations, robot localization: $\mathbf{p}(t) = ?$

Underwater exploration **without surfacing**:

- reasons of discretion and security (military missions)
- case of very deep environments (wrecks search)

Titanic wreck: 3821m deep

Lost MH370 aircraft: up to 6000m deep

Motivations

Motivations, wreck localization

Simultaneous destruction of *La Cordelière* and the *Regent*, $10^{\rm th}$ August 1512

Motivations

Motivations, wreck localization

Motivations

Motivations, wreck localization

Motivations

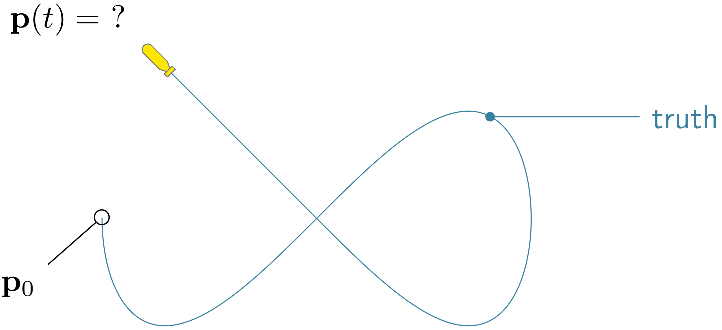
Motivations, wreck localization

Motivations

Motivations, robot localization: $\mathbf{p}(t) = ?$

Simple solution, **dead-reckoning**:

- navigation based on *proprioceptive* measurements
- fast drift on position estimation: strong errors

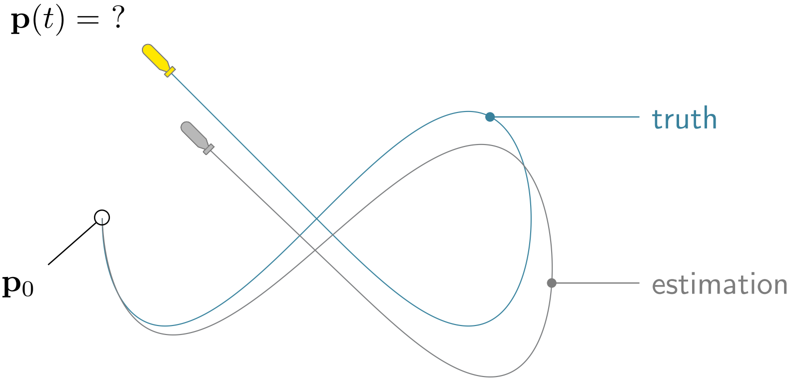


Motivations

Motivations, robot localization: $\mathbf{p}(t) = ?$

Simple solution, **dead-reckoning**:

- navigation based on *proprioceptive* measurements
- fast drift on position estimation: strong errors

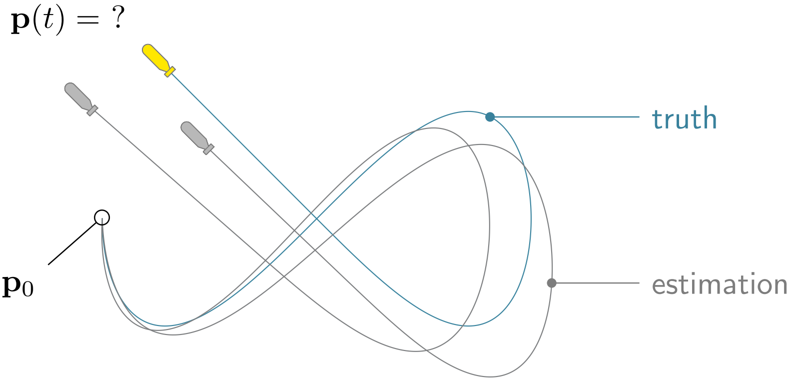


Motivations

Motivations, robot localization: $\mathbf{p}(t) = ?$

Simple solution, **dead-reckoning**:

- navigation based on *proprioceptive* measurements
- fast drift on position estimation: strong errors

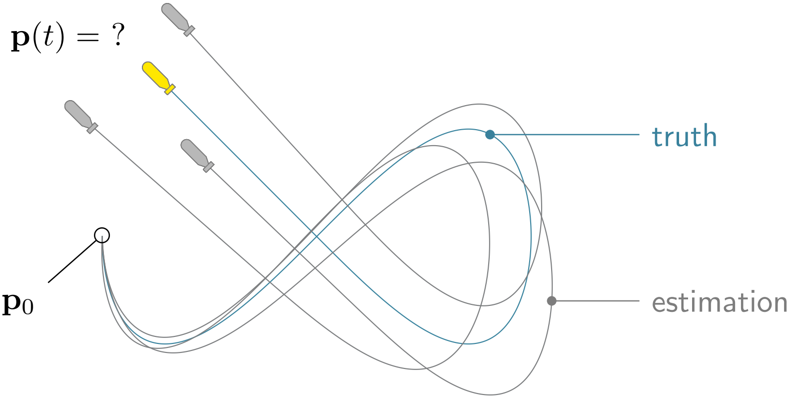


Motivations

Motivations, robot localization: $\mathbf{p}(t) = ?$

Simple solution, **dead-reckoning**:

- navigation based on *proprioceptive* measurements
- fast drift on position estimation: strong errors

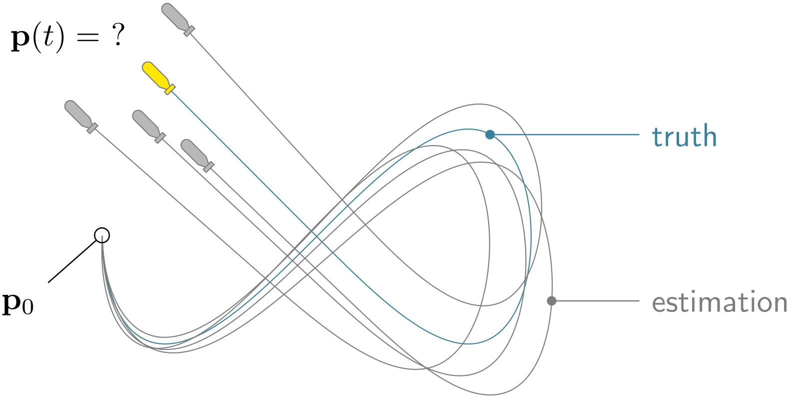


Motivations

Motivations, robot localization: $\mathbf{p}(t) = ?$

Simple solution, **dead-reckoning**:

- navigation based on *proprioceptive* measurements
- fast drift on position estimation: strong errors

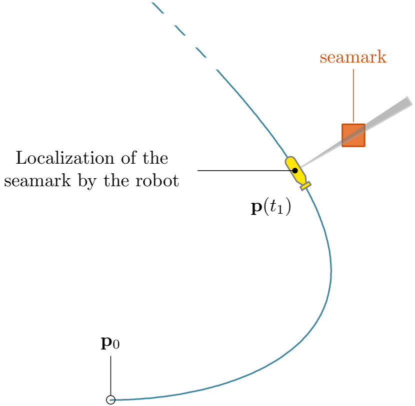


Motivations

Motivations, robot localization: $\mathbf{p}(t) = ?$

Exploration solution, **SLAM**:

- *Simultaneous Localization and Mapping*
- **come back** to a previous pose and **recognize** the environment

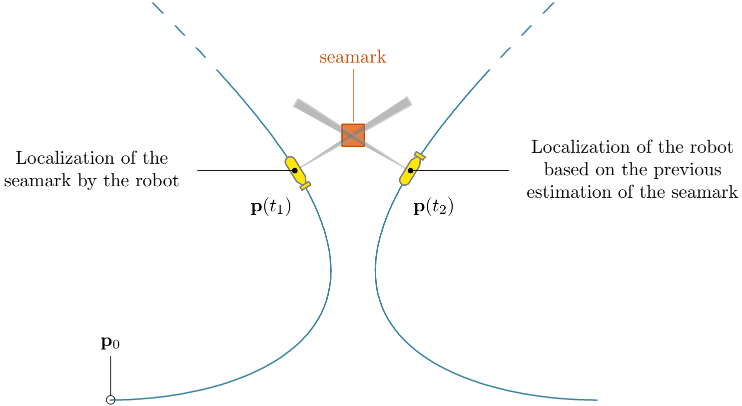


Motivations

Motivations, robot localization: $\mathbf{p}(t) = ?$

Exploration solution, **SLAM**:

- *Simultaneous Localization and Mapping*
- **come back** to a previous pose and **recognize** the environment

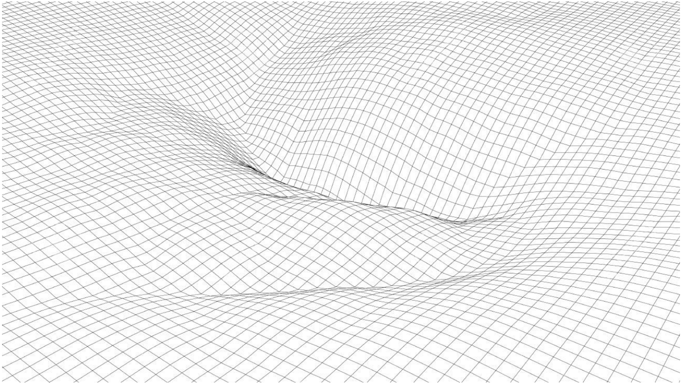


Motivations

Problem: homogeneous environments

Under the surface:

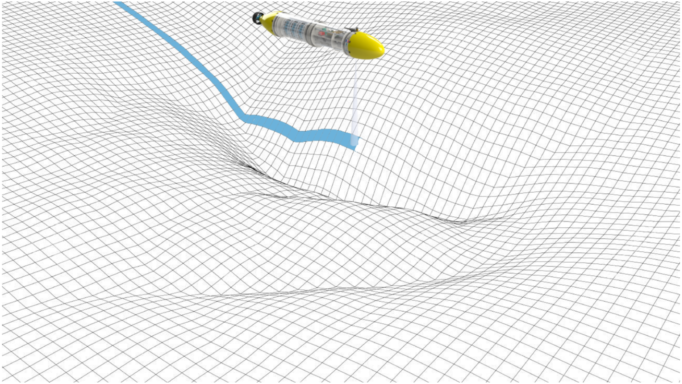
- **no seamarks** or points of interest
- usual SLAM methods do not apply



Motivations

Problem: homogeneous environments

- a robot coming back to a previous position should sense the same observations
- for instance, **bathymetric measurements**



Section 2

SLAM formalization

SLAM formalization

Mobile robotics

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

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

Where:

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

SLAM formalization

Mobile robotics

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

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

Where:

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

SLAM formalization

Mobile robotics

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

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

Where:

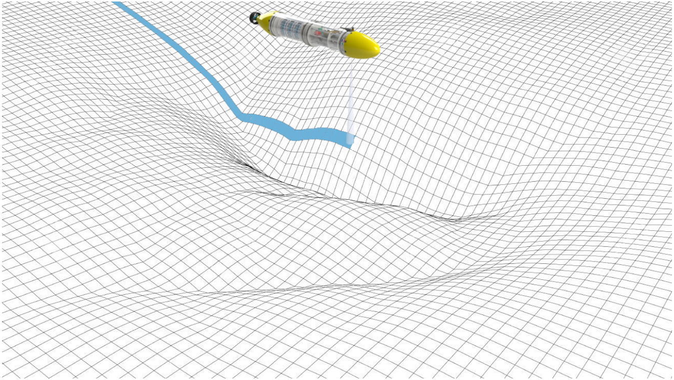
- $\mathbf{x} \in \mathbb{R}^n$ is the state vector (position, bearing, ...)
- $\mathbf{u} \in \mathbb{R}^m$ is the input vector (command)
- $\mathbf{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is the *evolution* function
- $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^p$ is the *observation*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

SLAM formalization

Bathymetric SLAM: observation function $\mathbf{g}$ not at hand

Observation equation:

- $\mathbf{z}(t) = \mathbf{g}(\mathbf{x}(t))$
- expression of $\mathbf{g}$ unknown $\implies$ no relation between $\mathbf{z}$ and $\mathbf{x}$

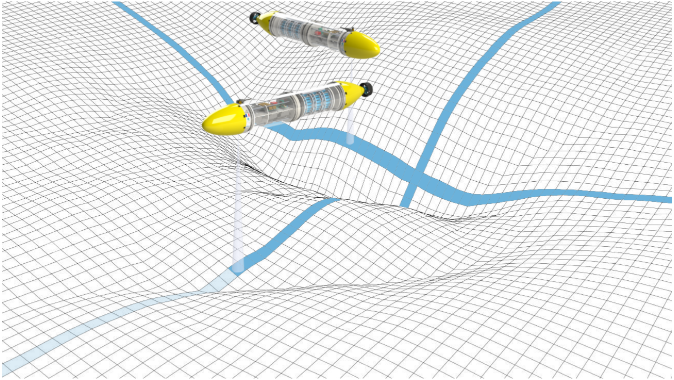


SLAM formalization

Bathymetric SLAM: observation function $\mathbf{g}$ not at hand

Observation equation:

- $\mathbf{z}(t) = \mathbf{g}(\mathbf{x}(t))$
- expression of $\mathbf{g}$ unknown $\implies$ no relation between $\mathbf{z}$ and $\mathbf{x}$
- main approach: **inter-temporal measurements**



SLAM formalization

New SLAM formalism: inter-temporal measurements

Raw-data SLAM equations:

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) & \text{(navigation)} \\ \mathbf{z}(t) \cancel{= \mathbf{g}(\mathbf{x}(t))} & \text{(observation)} \end{cases}$$

SLAM formalization

New SLAM formalism: inter-temporal measurements

Raw-data SLAM equations:

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) & \text{(navigation)} \\ \cancel{\mathbf{z}(t) = \mathbf{g}(\mathbf{x}(t))} & \text{(observation)} \\ \underbrace{\mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big)}_{\text{same } \textbf{state configurations}} \implies \underbrace{\mathbf{z}(t_1) = \mathbf{z}(t_2)}_{\text{same } \textbf{observations}} & \text{(inter-temporality)} \end{cases}$$

With:

- $\mathbf{h} : \mathbb{R}^n \to \mathbb{R}^{n'}$, the *configuration* function

SLAM formalization

New SLAM formalism: inter-temporal measurements

Raw-data SLAM equations:

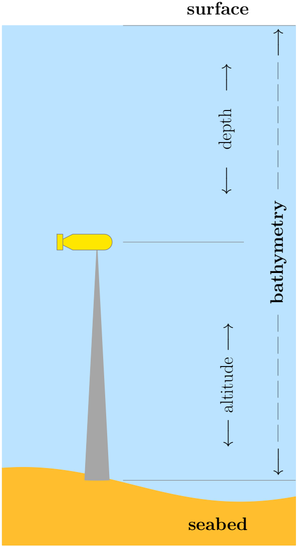
$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) & \text{(navigation)} \\ \cancel{\mathbf{z}(t) = \mathbf{g}(\mathbf{x}(t))} & \text{(observation)} \\ \underbrace{\mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big)}_{\text{same } \textbf{state configurations}} \implies \underbrace{\mathbf{z}(t_1) = \mathbf{z}(t_2)}_{\text{same } \textbf{observations}} & \text{(inter-temporality)} \end{cases}$$

With:

- $\mathbf{h} : \mathbb{R}^n \to \mathbb{R}^{n'}$, the *configuration* function
- $\mathbf{h}$ defined according to properties assumed on the unknown observation function $\mathbf{g}$
  - translational symmetries, spherical symmetries, ...

SLAM formalization

New SLAM formalism: inter-temporal measurements



Inter-temporal configuration:

$$\mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$$

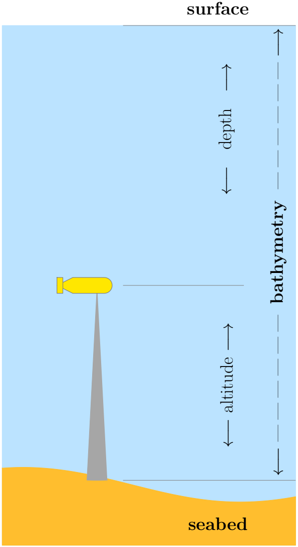
In the case of **bathymetric SLAM**:

- altitude measurements related to horizontal positions
- function $\mathbf{h}$ expressed as:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} \xmapsto{\mathbf{h}} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

SLAM formalization

New SLAM formalism: inter-temporal measurements



Inter-temporal configuration:

$$\mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$$

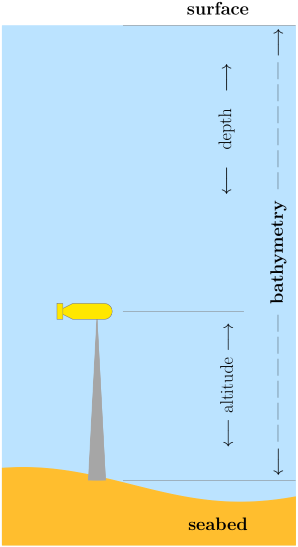
In the case of **bathymetric SLAM**:

- altitude measurements related to horizontal positions
- function $\mathbf{h}$ expressed as:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} \xmapsto{\mathbf{h}} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

SLAM formalization

New SLAM formalism: inter-temporal measurements



Inter-temporal configuration:

$$\mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$$

In the case of **bathymetric SLAM**:

- altitude measurements related to horizontal positions
- function $\mathbf{h}$ expressed as:

$$\begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} \xmapsto{\mathbf{h}} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

SLAM formalization

New SLAM formalism: inter-temporal measurements

Assumptions:

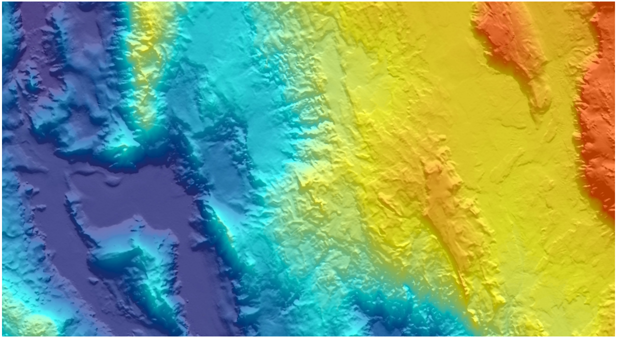
- bounded error context
- no unpredictable change in the environment

SLAM formalization

New SLAM formalism: inter-temporal measurements

Assumptions:

- bounded error context
- no unpredictable change in the environment
- sufficient spatial variations



Looking for MH370 – © 2014, Commonwealth of Australia

Section 3

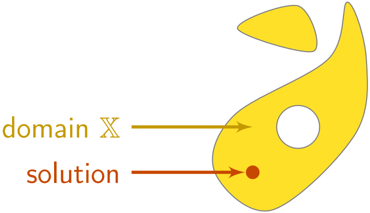
Constraint programming

Constraint programming

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:

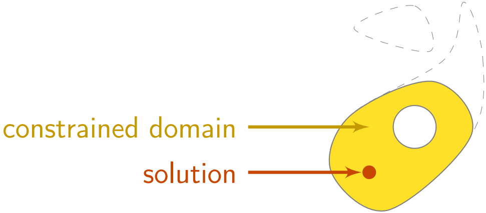
Domains: $\mathbb{X}$

Constraint programming

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

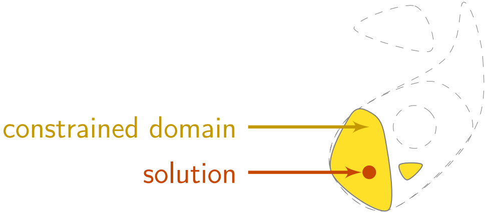
Domains: $\mathbb{X}$

Constraint programming

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:

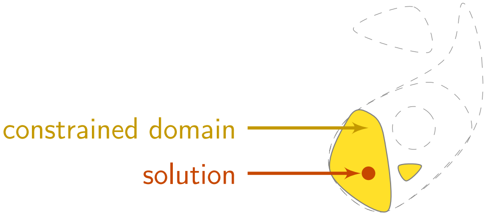
$$\begin{cases} \textbf{Variables: } \mathbf{x} \\ \textbf{Constraints:} \\ \quad 1.\ \mathcal{L}_1(\mathbf{x}) \\ \quad 2.\ \mathcal{L}_2(\mathbf{x}) \\ \\ \textbf{Domains: } \mathbb{X} \end{cases}$$

Constraint programming

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. ...

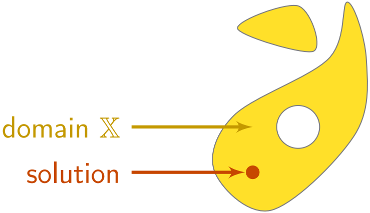
Domains: $\mathbb{X}$

Constraint programming

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. ...

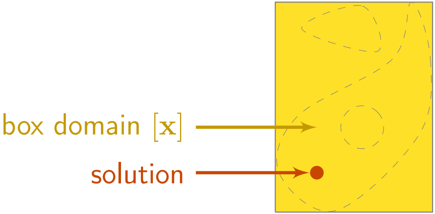
Domains: $\mathbb{X}$

Constraint programming

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. ...

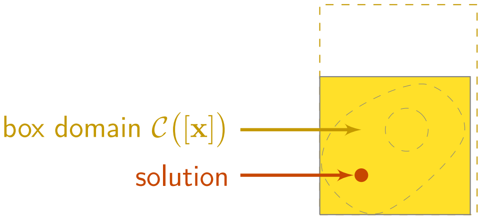
Domains: $[\mathbf{x}]$

Constraint programming

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:

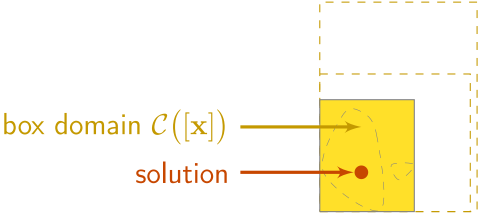
$$\left\{\begin{array}{l} \textbf{Variables: } \mathbf{x} \\ \textbf{Constraints:} \\ \quad 1.\ \mathcal{L}_1(\mathbf{x}) \\ \quad 2.\ \mathcal{L}_2(\mathbf{x}) \\ \quad 3.\ \ldots \\ \textbf{Domains: } [\mathbf{x}] \end{array}\right.$$

Constraint programming

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:

$$\begin{cases} \textbf{Variables: } \mathbf{x} \\ \textbf{Constraints:} \\ \quad 1.\ \mathcal{L}_1(\mathbf{x}) \\ \quad 2.\ \mathcal{L}_2(\mathbf{x}) \\ \quad 3.\ \dots \\ \textbf{Domains: } [\mathbf{x}] \end{cases}$$

Constraint programming

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

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

Constraint programming

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**, $[\mathbf{x}](\cdot) : \mathbb{R}^+ \to \mathbb{IR}^n$

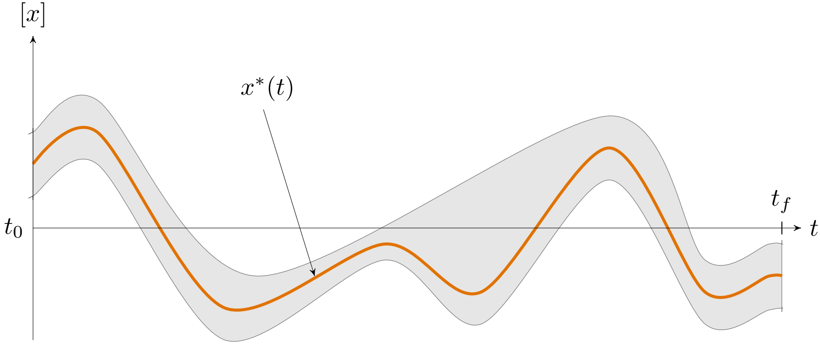
PhD thesis **objectives**:

- develop **primitive dynamical contractors**
- application to **robot localization**
- dynamical constraint $\not\Leftrightarrow$ set of continuous constraints

Constraint programming

Tubes

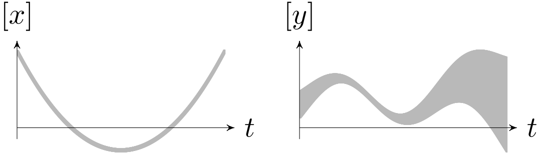
Tube $[x](\cdot)$: interval of trajectories $[x^-(\cdot), x^+(\cdot)]$
such that $\forall t \in \mathbb{R},\ x^-(t) \leqslant x^+(t)$



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

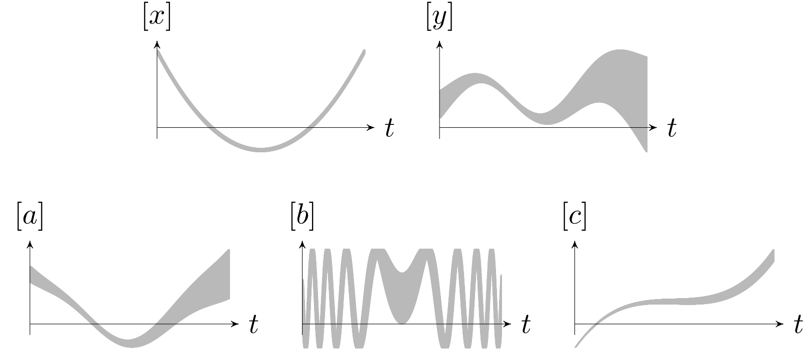
Constraint programming

Tubes arithmetic



Constraint programming

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

SLAM under constraints

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM: constraint problem over trajectories

SLAM :

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

Constraint programming

SLAM under constraints

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM: constraint problem over trajectories

SLAM :
- **Variables:** $\mathbf{x}(\cdot)$, $\mathbf{z}(\cdot)$
- **Constraints:**
- **Domains:** $[\mathbf{x}](\cdot)$, $[\mathbf{z}](\cdot)$

Constraint programming

SLAM under constraints

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM: constraint problem over trajectories

SLAM :

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

Constraints:

1. Evolution constraint:
   - $\mathbf{v}(\cdot) = \mathbf{f}\big(\mathbf{x}(\cdot)\big)$

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

Constraint programming

SLAM under constraints

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM: constraint problem over trajectories

$$\text{SLAM}: \begin{cases} \textbf{Variables: } \mathbf{x}(\cdot),\ \mathbf{z}(\cdot),\ \mathbf{v}(\cdot) \\ \textbf{Constraints:} \\ \quad 1.\ \text{Evolution constraint:} \\ \qquad \blacktriangleright\ \mathbf{v}(\cdot) = \mathbf{f}\big(\mathbf{x}(\cdot)\big) \\ \qquad \blacktriangleright\ \dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot) \\ \textbf{Domains: } [\mathbf{x}](\cdot),\ [\mathbf{z}](\cdot),\ [\mathbf{v}](\cdot) \end{cases}$$

Constraint programming

SLAM under constraints

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM: constraint problem over trajectories

$$\mathrm{SLAM} : \left\{ \begin{array}{l} \textbf{Variables: } \mathbf{x}(\cdot),\ \mathbf{z}(\cdot),\ \mathbf{v}(\cdot),\ \mathbf{p}(\cdot) \\ \textbf{Constraints:} \\ \quad 1.\ \text{Evolution constraint:} \\ \qquad \blacktriangleright\ \mathbf{v}(\cdot) = \mathbf{f}\big(\mathbf{x}(\cdot)\big) \\ \qquad \blacktriangleright\ \dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot) \\ \quad 2.\ \text{Inter-temporal constraint:} \\ \qquad \blacktriangleright\ \mathbf{p}(\cdot) = \mathbf{h}\big(\mathbf{x}(\cdot)\big) \\ \\ \textbf{Domains: } [\mathbf{x}](\cdot),\ [\mathbf{z}](\cdot),\ [\mathbf{v}](\cdot),\ [\mathbf{p}](\cdot) \end{array} \right.$$

Constraint programming

SLAM under constraints

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM: constraint problem over trajectories

$$\text{SLAM}: \left\{ \begin{array}{l} \textbf{Variables: } \mathbf{x}(\cdot),\ \mathbf{z}(\cdot),\ \mathbf{v}(\cdot),\ \mathbf{p}(\cdot) \\ \textbf{Constraints:} \\ \quad 1.\ \text{Evolution constraint:} \\ \qquad \blacktriangleright\ \mathbf{v}(\cdot) = \mathbf{f}\big(\mathbf{x}(\cdot)\big) \\ \qquad \blacktriangleright\ \dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot) \\ \quad 2.\ \text{Inter-temporal constraint:} \\ \qquad \blacktriangleright\ \mathbf{p}(\cdot) = \mathbf{h}\big(\mathbf{x}(\cdot)\big) \\ \qquad \blacktriangleright\ \mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \\ \textbf{Domains: } [\mathbf{x}](\cdot),\ [\mathbf{z}](\cdot),\ [\mathbf{v}](\cdot),\ [\mathbf{p}](\cdot) \end{array} \right.$$

Constraint programming

SLAM under constraints

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM: constraint problem over trajectories

SLAM :

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

Constraints:

1. Evolution constraint:
   - $\mathbf{v}(\cdot) = \mathbf{f}\big(\mathbf{x}(\cdot)\big)$ $\longleftarrow$ algebraic constraint
   - $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$
2. Inter-temporal constraint:
   - $\mathbf{p}(\cdot) = \mathbf{h}\big(\mathbf{x}(\cdot)\big)$ $\longleftarrow$ algebraic constraint
   - $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

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

Constraint programming

SLAM under constraints

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM: constraint problem over trajectories

SLAM :

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

Constraints:

1. Evolution constraint:
   - $\mathbf{v}(\cdot) = \mathbf{f}\big(\mathbf{x}(\cdot)\big)$ $\longleftarrow$ algebraic constraint
   - $\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot), \mathbf{v}(\cdot)\big)$: $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$
2. Inter-temporal constraint:
   - $\mathbf{p}(\cdot) = \mathbf{h}\big(\mathbf{x}(\cdot)\big)$ $\longleftarrow$ algebraic constraint
   - $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

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

Constraint programming

SLAM under constraints

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}(\mathbf{x}(t_1)) = \mathbf{h}(\mathbf{x}(t_2)) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM: constraint problem over trajectories

$\mathrm{SLAM}:$

- **Variables:** $\mathbf{x}(\cdot)$, $\mathbf{z}(\cdot)$, $\mathbf{v}(\cdot)$, $\mathbf{p}(\cdot)$
- **Constraints:**
  1. Evolution constraint:
     - $\mathbf{v}(\cdot) = \mathbf{f}\big(\mathbf{x}(\cdot)\big)$ $\longleftarrow$ algebraic constraint
     - $\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot), \mathbf{v}(\cdot)\big)$: $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$
  2. Inter-temporal constraint:
     - $\mathbf{p}(\cdot) = \mathbf{h}\big(\mathbf{x}(\cdot)\big)$ $\longleftarrow$ algebraic constraint
     - $\mathcal{L}_{\mathrm{inter}}\big(\mathbf{p}(\cdot), \mathbf{z}(\cdot)\big)$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$
- **Domains:** $[\mathbf{x}](\cdot)$, $[\mathbf{z}](\cdot)$, $[\mathbf{v}](\cdot)$, $[\mathbf{p}](\cdot)$

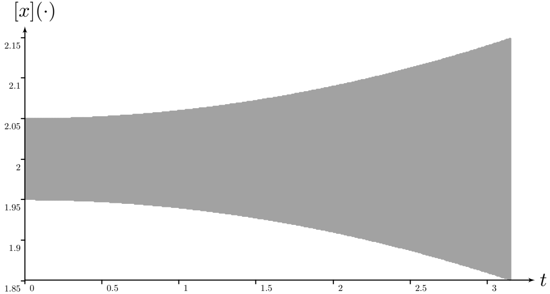
Constraint programming
$\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot),\mathbf{v}(\cdot)\big)$

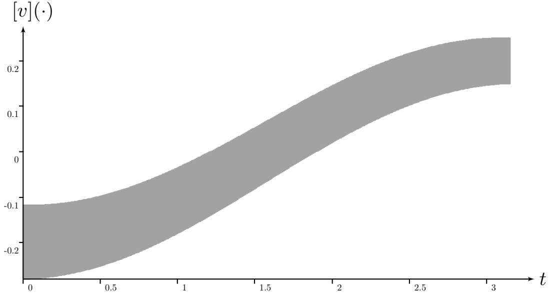
Differential constraint:

- $\dot{\mathbf{x}}(\cdot)=\mathbf{v}(\cdot)$
- elementary constraint

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

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





Constraint programming

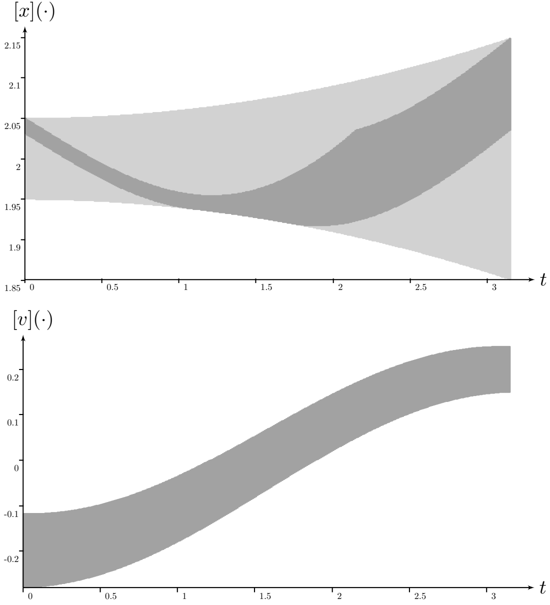
$$\mathcal{L}_{\frac{d}{dt}}\Big(\mathbf{x}(\cdot),\mathbf{v}(\cdot)\Big)$$

Differential constraint:

- $\dot{\mathbf{x}}(\cdot)=\mathbf{v}(\cdot)$
- elementary constraint

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

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



Constraint programming

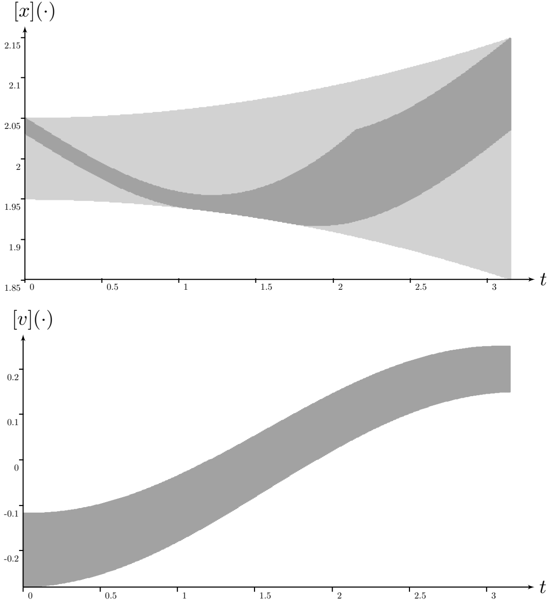
$$\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot),\mathbf{v}(\cdot)\big)$$

Differential constraint:

- $\dot{\mathbf{x}}(\cdot)=\mathbf{v}(\cdot)$
- elementary constraint

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

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



Guaranteed computation of robot trajectories

Rohou, Jaulin, Mihaylova, Le Bars, Veres

Robotics and Autonomous Systems, 2017

Section 4

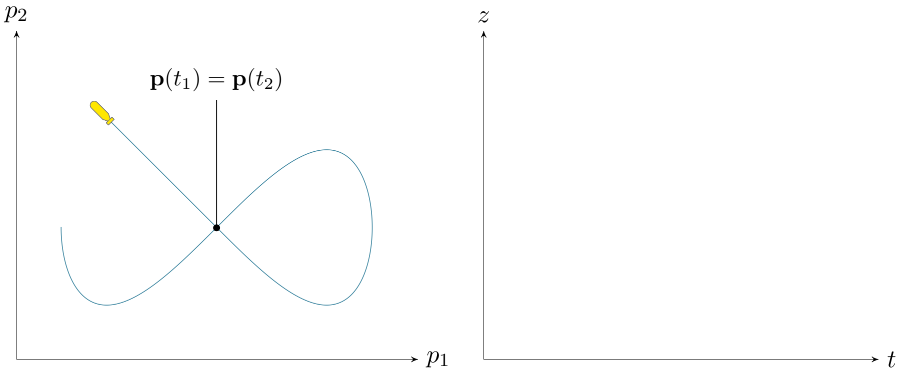
Constraint $\mathcal{L}_{\text{inter}}$:

$$\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: physical interpretation

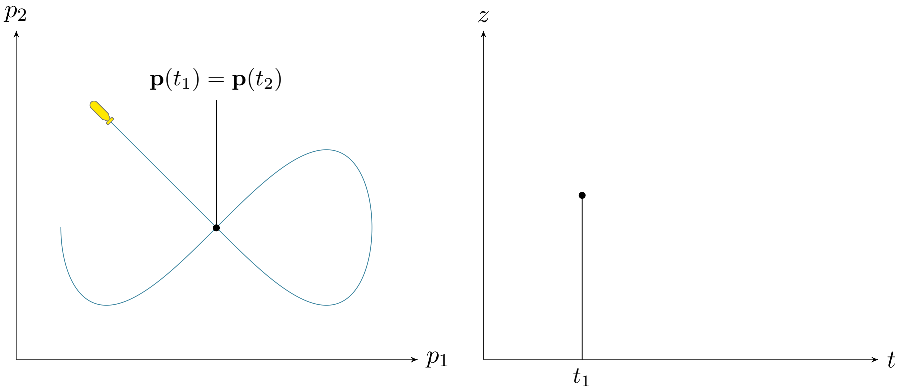
A robot coming back to a previous position $\mathbf{p}$ should sense the same observation $\mathbf{z}$.



Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1)=\mathbf{p}(t_2) \implies \mathbf{z}(t_1)=\mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: physical interpretation

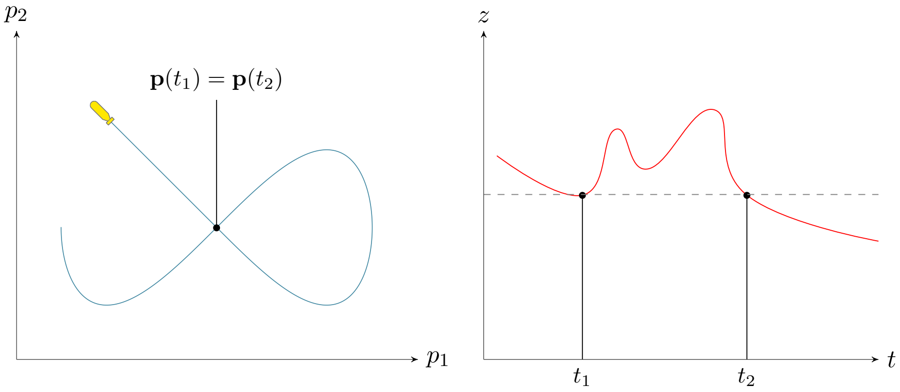
A robot coming back to a previous position $\mathbf{p}$ should sense the same observation $\mathbf{z}$.



Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: physical interpretation

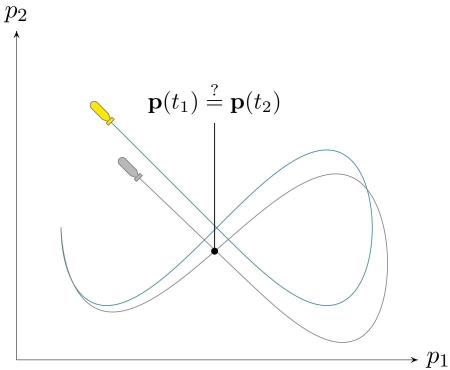
A robot coming back to a previous position $\mathbf{p}$ should sense the same observation $\mathbf{z}$.

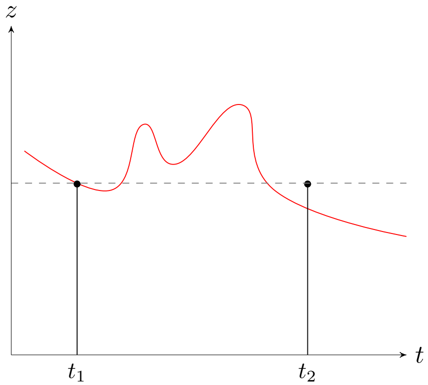


Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: physical interpretation

A robot coming back to a previous position $\mathbf{p}$ should sense the same observation $\mathbf{z}$.

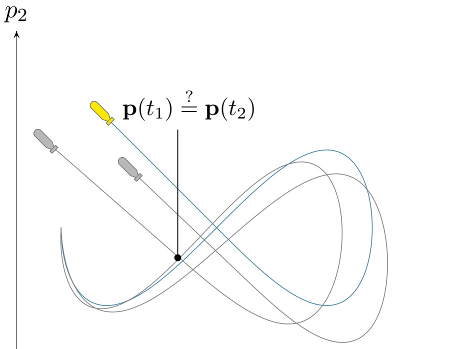


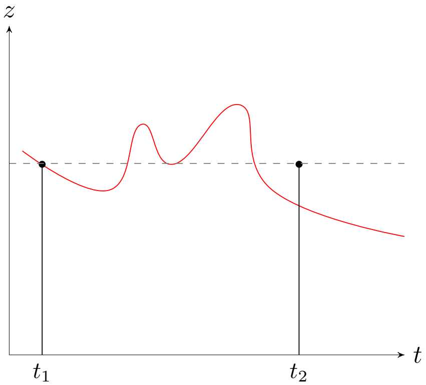


Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: physical interpretation

A robot coming back to a previous position $\mathbf{p}$ should sense the same observation $\mathbf{z}$.

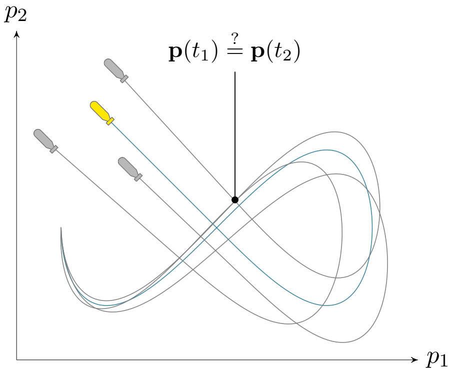


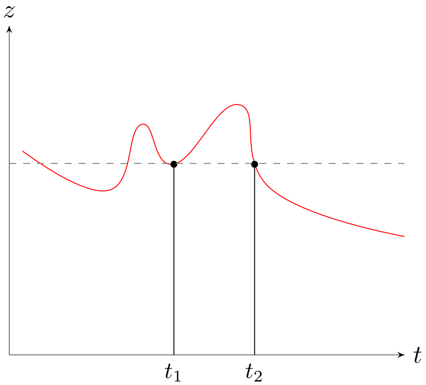


Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: physical interpretation

A robot coming back to a previous position $\mathbf{p}$ should sense the same observation $\mathbf{z}$.

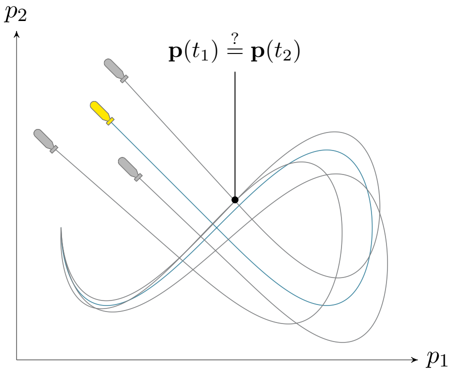


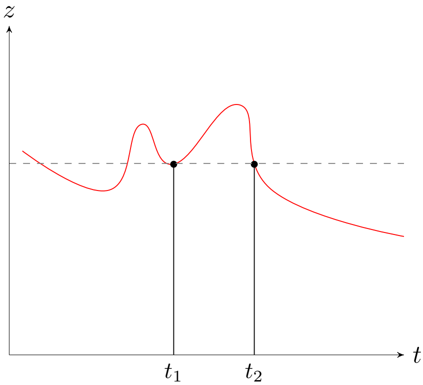


Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: physical interpretation

A robot coming back to a previous position $\mathbf{p}$ should sense the same observation $\mathbf{z}$.





Method: temporal resolution, estimation of feasible pairs (t_1, t_2)

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

Temporal decomposition

$$\underbrace{\mathbf{p}(t_1) = \mathbf{p}(t_2)}_{①} \implies \underbrace{\mathbf{z}(t_1) = \mathbf{z}(t_2)}_{②}$$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

Temporal decomposition

$$\underbrace{\mathbf{p}(t_1) = \mathbf{p}(t_2)}_{①} \implies \underbrace{\mathbf{z}(t_1) = \mathbf{z}(t_2)}_{②}$$

Temporal space. Sets of t-pairs defined by:

- the cause ①:
  $\mathbb{T}^*_{\mathbf{p}} = \left\{(t_1, t_2) \in [t_0, t_f]^2 \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\ ,\ t_1 < t_2\right\}$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

Temporal decomposition

$$\underbrace{\mathbf{p}(t_1) = \mathbf{p}(t_2)}_{①} \implies \underbrace{\mathbf{z}(t_1) = \mathbf{z}(t_2)}_{②}$$

Temporal space. Sets of t-pairs defined by:

- the cause ①:
  $\mathbb{T}^*_{\mathbf{p}} = \left\{(t_1, t_2) \in [t_0, t_f]^2 \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\ ,\ t_1 < t_2\right\}$
- the effect ②:
  $\mathbb{T}^*_{\mathbf{z}} = \left\{(t_1, t_2) \in [t_0, t_f]^2 \mid \mathbf{z}(t_1) = \mathbf{z}(t_2)\ ,\ t_1 < t_2\right\}$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

Temporal decomposition

$$\underbrace{\mathbf{p}(t_1) = \mathbf{p}(t_2)}_{①} \implies \underbrace{\mathbf{z}(t_1) = \mathbf{z}(t_2)}_{②}$$

Temporal space. Sets of t-pairs defined by:

- the cause ①:
  $\mathbb{T}^*_{\mathbf{p}} = \left\{(t_1, t_2) \in [t_0, t_f]^2 \ \mid\ \mathbf{p}(t_1) = \mathbf{p}(t_2)\ ,\ t_1 < t_2\right\}$
- the effect ②:
  $\mathbb{T}^*_{\mathbf{z}} = \left\{(t_1, t_2) \in [t_0, t_f]^2 \ \mid\ \mathbf{z}(t_1) = \mathbf{z}(t_2)\ ,\ t_1 < t_2\right\}$

From the implication ① $\implies$ ②:

$$\mathbb{T}^*_{\mathbf{p}} \subset \mathbb{T}^*_{\mathbf{z}}$$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1)=\mathbf{p}(t_2) \implies \mathbf{z}(t_1)=\mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: constraint network

$\mathcal{L}_{\text{inter}}$:

Variables: $\mathbf{p}(\cdot)$, $\mathbf{z}(\cdot)$

Internal variables: $\mathbb{T}^*_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}}$

Constraints:

1. $\mathbb{T}^*_{\mathbf{p}}=\{(t_1,t_2)\mid \mathbf{p}(t_1)=\mathbf{p}(t_2)\}$
2. $\mathbb{T}^*_{\mathbf{z}}=\{(t_1,t_2)\mid \mathbf{z}(t_1)=\mathbf{z}(t_2)\}$
3. $\mathbb{T}^*_{\mathbf{p}}\subset\mathbb{T}^*_{\mathbf{z}}$

Domains: $[\mathbf{p}](\cdot)$, $[\mathbf{z}](\cdot)$, $\mathbb{T}_{\mathbf{p}}$, $\mathbb{T}_{\mathbf{z}}$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: physical interpretation

Variables: $\mathbf{p}(\cdot)$, $\mathbf{z}(\cdot)$
Internal variables: $\mathbb{T}^*_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}}$
Constraints:

1. $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\}$
2. $\mathbb{T}^*_{\mathbf{z}} = \{(t_1, t_2) \mid \mathbf{z}(t_1) = \mathbf{z}(t_2)\}$
3. $\mathbb{T}^*_{\mathbf{p}} \subset \mathbb{T}^*_{\mathbf{z}}$

Domains: $[\mathbf{p}](\cdot)$, $[\mathbf{z}](\cdot)$, $\mathbb{T}_{\mathbf{p}}$, $\mathbb{T}_{\mathbf{z}}$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: physical interpretation

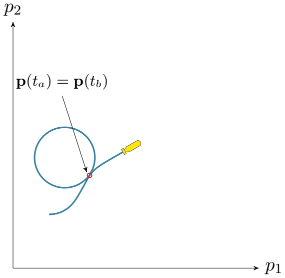
Variables: $\mathbf{p}(\cdot)$, $\mathbf{z}(\cdot)$

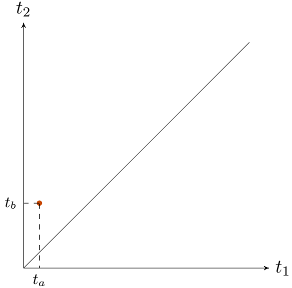
Internal variables: $\mathbb{T}^*_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}}$

Constraints:

1. $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\}$
2. $\mathbb{T}^*_{\mathbf{z}} = \{(t_1, t_2) \mid \mathbf{z}(t_1) = \mathbf{z}(t_2)\}$
3. $\mathbb{T}^*_{\mathbf{p}} \subset \mathbb{T}^*_{\mathbf{z}}$

Domains: $[\mathbf{p}](\cdot)$, $[\mathbf{z}](\cdot)$, $\mathbb{T}_{\mathbf{p}}$, $\mathbb{T}_{\mathbf{z}}$





$\mathbb{T}^*_{\mathbf{p}}$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: physical interpretation

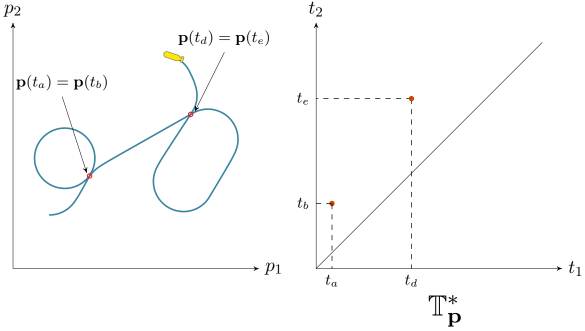
Variables: $\mathbf{p}(\cdot)$, $\mathbf{z}(\cdot)$

Internal variables: $\mathbb{T}^*_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}}$

Constraints:

1. $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\}$
2. $\mathbb{T}^*_{\mathbf{z}} = \{(t_1, t_2) \mid \mathbf{z}(t_1) = \mathbf{z}(t_2)\}$
3. $\mathbb{T}^*_{\mathbf{p}} \subset \mathbb{T}^*_{\mathbf{z}}$

Domains: $[\mathbf{p}](\cdot)$, $[\mathbf{z}](\cdot)$, $\mathbb{T}_{\mathbf{p}}$, $\mathbb{T}_{\mathbf{z}}$



Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: physical interpretation

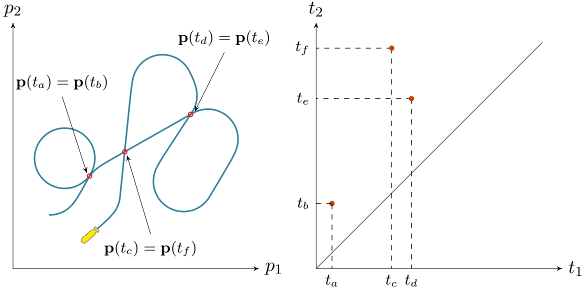
Variables: $\mathbf{p}(\cdot)$, $\mathbf{z}(\cdot)$

Internal variables: $\mathbb{T}^*_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}}$

Constraints:

1. $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\}$
2. $\mathbb{T}^*_{\mathbf{z}} = \{(t_1, t_2) \mid \mathbf{z}(t_1) = \mathbf{z}(t_2)\}$
3. $\mathbb{T}^*_{\mathbf{p}} \subset \mathbb{T}^*_{\mathbf{z}}$

Domains: $[\mathbf{p}](\cdot)$, $[\mathbf{z}](\cdot)$, $\mathbb{T}_{\mathbf{p}}$, $\mathbb{T}_{\mathbf{z}}$



$\mathbb{T}^*_{\mathbf{p}}$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: physical interpretation

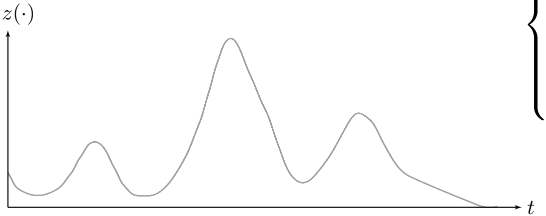
Variables: $\mathbf{p}(\cdot)$, $\mathbf{z}(\cdot)$

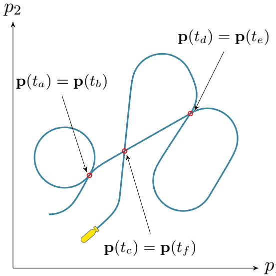
Internal variables: $\mathbb{T}^*_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}}$

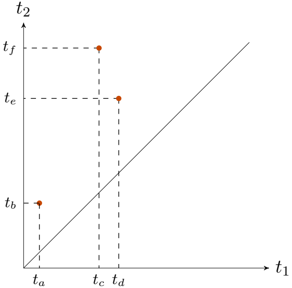
Constraints:

1. $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\}$
2. $\mathbb{T}^*_{\mathbf{z}} = \{(t_1, t_2) \mid \mathbf{z}(t_1) = \mathbf{z}(t_2)\}$
3. $\mathbb{T}^*_{\mathbf{p}} \subset \mathbb{T}^*_{\mathbf{z}}$

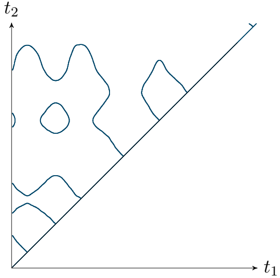
Domains: $[\mathbf{p}](\cdot)$, $[\mathbf{z}](\cdot)$, $\mathbb{T}_{\mathbf{p}}$, $\mathbb{T}_{\mathbf{z}}$







$\mathbb{T}^*_{\mathbf{p}}$



$\mathbb{T}^*_{\mathbf{z}}$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\text{inter}}$: physical interpretation

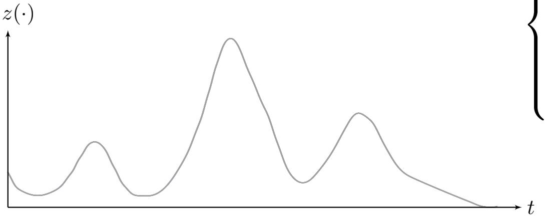
Variables: $\mathbf{p}(\cdot)$, $\mathbf{z}(\cdot)$

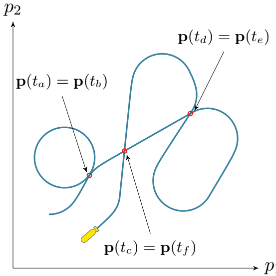
Internal variables: $\mathbb{T}^*_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}}$

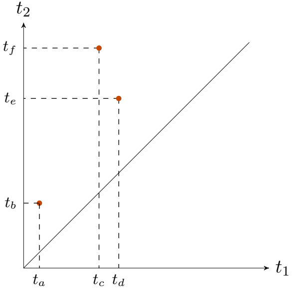
Constraints:

1. $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\}$
2. $\mathbb{T}^*_{\mathbf{z}} = \{(t_1, t_2) \mid \mathbf{z}(t_1) = \mathbf{z}(t_2)\}$
3. $\mathbb{T}^*_{\mathbf{p}} \subset \mathbb{T}^*_{\mathbf{z}}$

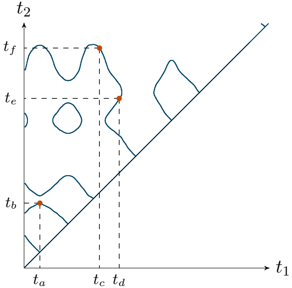
Domains: $[\mathbf{p}](\cdot)$, $[\mathbf{z}](\cdot)$, $\mathbb{T}_{\mathbf{p}}$, $\mathbb{T}_{\mathbf{z}}$







$\mathbb{T}^*_{\mathbf{p}}$

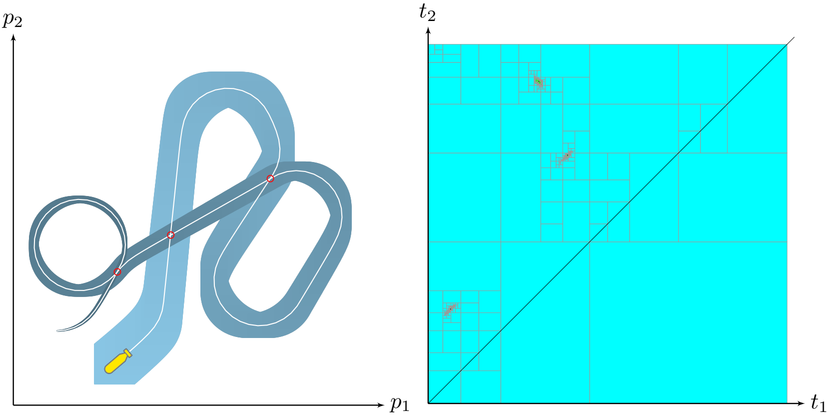


$\mathbb{T}^*_{\mathbf{z}}$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

Bounded-error context

Approximation of the enclosure of t-sets with SIVIA algorithms:

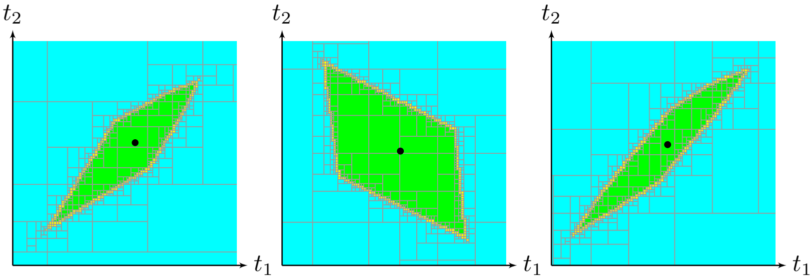


(a) Bounded trajectories (b) Approximation of $\mathbb{T}_{\mathbf{p}}$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

Bounded-error context

Approximation of the enclosure of t-sets with SIVIA algorithms:

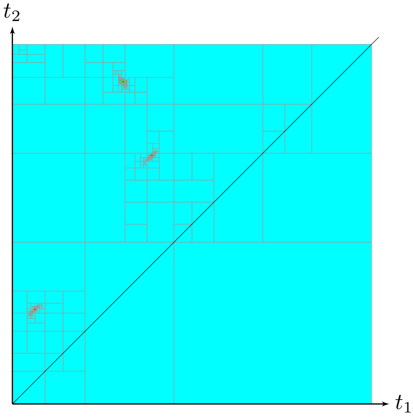


Zoom on the components of $\mathbb{T}_{\mathbf{p}}$

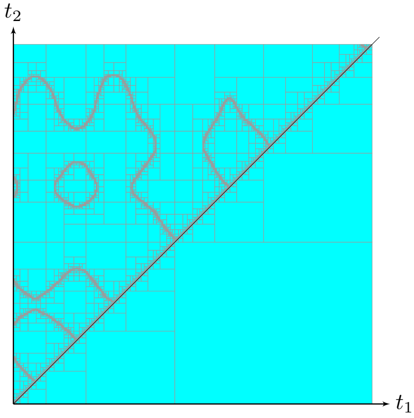
Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

Bounded-error context

Approximation of the enclosure of t-sets with SIVIA algorithms:



(a) Approximation of $\mathbb{T}_{\mathbf{p}}$



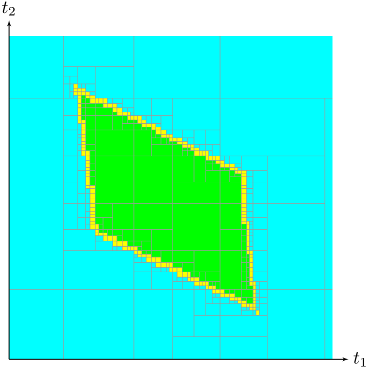
(b) Approximation of $\mathbb{T}_{\mathbf{z}}$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

The $\mathcal{C}_{\text{inter}}$ contractor: t-sets fusion

Constraint:

- $\mathbb{T}^*_{\mathbf{p}} \subset \mathbb{T}^*_{\mathbf{z}}$
- $\mathbb{T}^*_{\mathbf{p}} \in \mathbb{T}_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}} \in \mathbb{T}_{\mathbf{z}}$



Approximation of $\mathbb{T}_{\mathbf{p}}$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

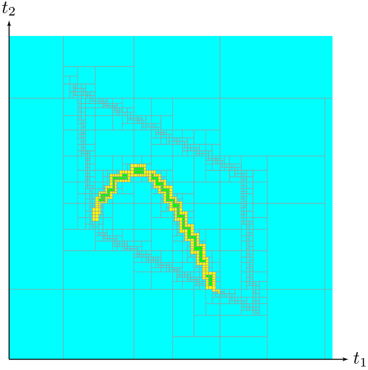
The $\mathcal{C}_{\text{inter}}$ contractor: t-sets fusion

Constraint:

- $\mathbb{T}^*_{\mathbf{p}} \subset \mathbb{T}^*_{\mathbf{z}}$
- $\mathbb{T}^*_{\mathbf{p}} \in \mathbb{T}_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}} \in \mathbb{T}_{\mathbf{z}}$

Contraction:

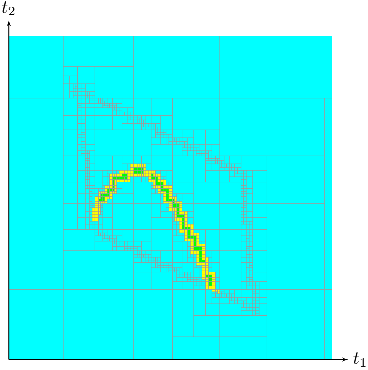
- $\mathbb{T}_{\mathbf{p}} := \mathbb{T}_{\mathbf{p}} \cap \mathbb{T}_{\mathbf{z}}$



Contraction of $\mathbb{T}_{\mathbf{p}}$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

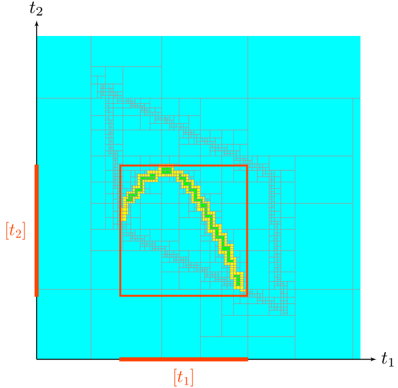
Constraint $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\}$ in backward



Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

Constraint $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\}$ in backward

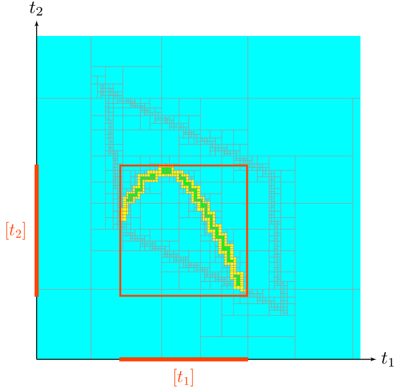
- **time uncertainties:** $[t_1]$, $[t_2]$



Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

Constraint $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\}$ in backward

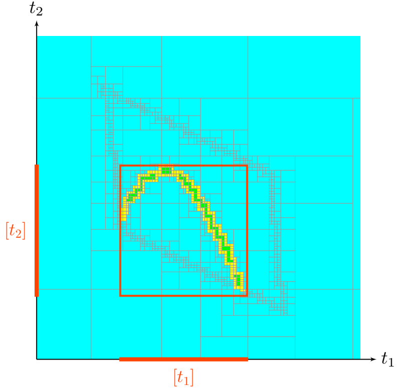
- **time uncertainties:** $[t_1]$, $[t_2]$
- constraint $\mathcal{L}_{t_1,t_2}\big(t_1, t_2, \mathbf{p}(\cdot)\big) : \mathbf{p}(t_1) = \mathbf{p}(t_2)$



Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

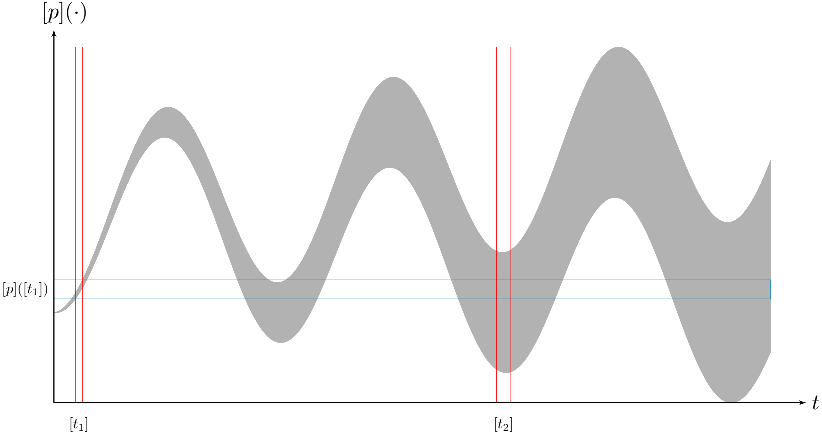
Constraint $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\}$ in backward

- **time uncertainties:** $[t_1]$, $[t_2]$
- constraint $\mathcal{L}_{t_1,t_2}(t_1, t_2, \mathbf{p}(\cdot)) : \mathbf{p}(t_1) = \mathbf{p}(t_2)$
- strong contribution of this thesis:
  - no already existing method
  - study of the $\mathcal{C}_{\text{eval}}$ contractor
  - Reliable non-linear state estimation involving time uncertainties
    Rohou, Jaulin, Mihaylova, Le Bars, Veres
    Automatica, submitted



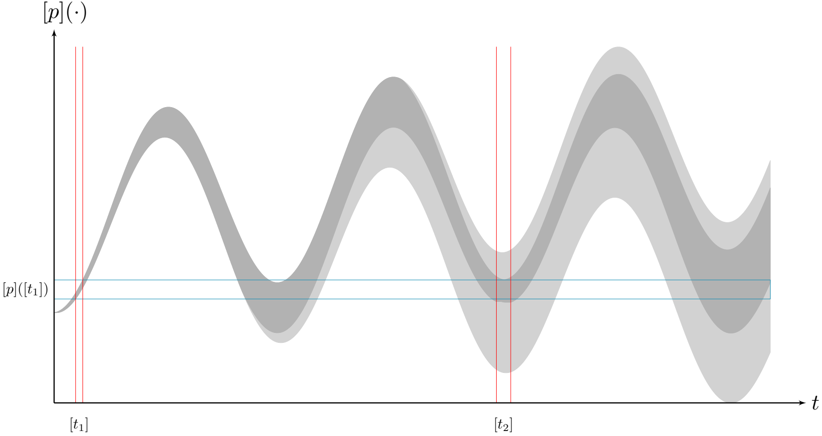
Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

The $\mathcal{C}_{t_1,t_2}\big([t_1],[t_2],[\mathbf{p}](\cdot)\big)$ contractor



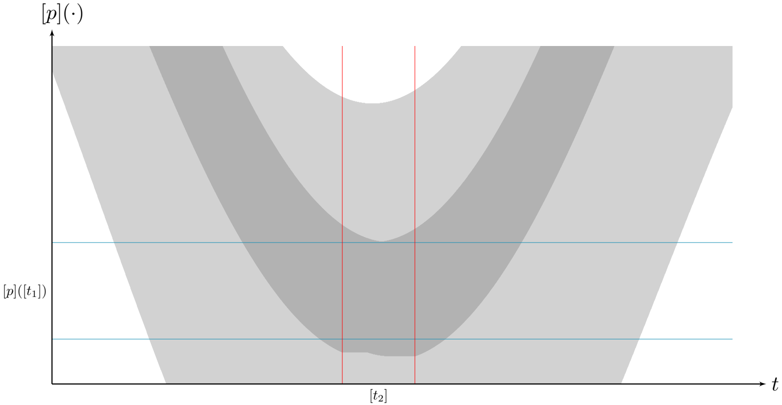
Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

The $\mathcal{C}_{t_1,t_2}\big([t_1],[t_2],[\mathbf{p}](\cdot)\big)$ contractor



Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

The $\mathcal{C}_{t_1,t_2}\big([t_1],[t_2],[\mathbf{p}](\cdot)\big)$ contractor

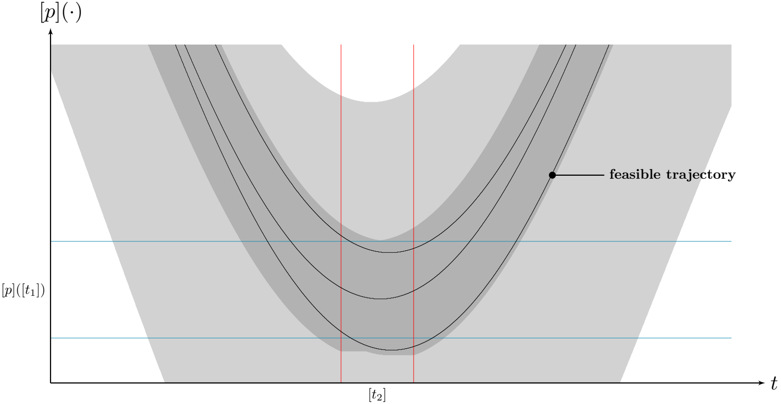


Importance of the **derivative** $\mathbf{w}(\cdot)$

$$\mathcal{L}_{t_1,t_2}\big(t_1,t_2,\mathbf{p}(\cdot),\mathbf{w}(\cdot)\big): \quad \begin{cases} \mathbf{p}(t_1) = \mathbf{p}(t_2) \\ \dot{\mathbf{p}}(\cdot) = \mathbf{w}(\cdot) \end{cases}$$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

The $\mathcal{C}_{t_1,t_2}\big([t_1],[t_2],[\mathbf{p}](\cdot)\big)$ contractor



Importance of the **derivative** $\mathbf{w}(\cdot)$

$$\mathcal{L}_{t_1,t_2}\big(t_1,t_2,\mathbf{p}(\cdot),\mathbf{w}(\cdot)\big): \quad \begin{cases} \mathbf{p}(t_1) = \mathbf{p}(t_2) \\ \dot{\mathbf{p}}(\cdot) = \mathbf{w}(\cdot) \end{cases}$$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

Summary

$$\mathcal{L}_{\text{inter}}\big(\mathbf{p}(\cdot), \mathbf{z}(\cdot) \qquad \big) : \left\{ \begin{array}{l} \mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \\ \\ \end{array} \right.$$

Constraint $\mathcal{L}_{\text{inter}}$: $\mathbf{p}(t_1)=\mathbf{p}(t_2) \implies \mathbf{z}(t_1)=\mathbf{z}(t_2)$

Summary

$$\mathcal{L}_{\text{inter}}\big(\mathbf{p}(\cdot),\mathbf{z}(\cdot),\mathbf{w}(\cdot)\big) : \begin{cases} \mathbf{p}(t_1)=\mathbf{p}(t_2) \implies \mathbf{z}(t_1)=\mathbf{z}(t_2) \\ \dot{\mathbf{p}}(\cdot)=\mathbf{w}(\cdot) \end{cases}$$

Section 5

Bathymetric SLAM

Bathymetric SLAM

Contractor programming

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM: constraint problem over trajectories

$\mathrm{SLAM}:$

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

Constraints:

1. Evolution constraint:
   - $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot))$
   - $\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot), \mathbf{v}(\cdot)\big)$
2. Inter-temporal constraint:
   - $\mathbf{p}(\cdot) = \mathbf{h}(\mathbf{x}(\cdot))$
   - $\mathcal{L}_{\mathrm{inter}}\big(\mathbf{p}(\cdot), \mathbf{z}(\cdot)\big)$

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

Bathymetric SLAM

Contractor programming

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM: constraint problem over trajectories

SLAM :

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

Constraints:

1. Evolution constraint:
   - $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot))$
   - $\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot), \mathbf{v}(\cdot)\big)$
2. Inter-temporal constraint:
   - $\mathbf{p}(\cdot) = \mathbf{h}(\mathbf{x}(\cdot))$
   - $\mathcal{L}_{\text{inter}}\big(\mathbf{p}(\cdot), \mathbf{w}(\cdot), \mathbf{z}(\cdot)\big)$
   - $\mathbf{w}(\cdot) = \dfrac{d\mathbf{h}}{d\mathbf{x}(\cdot)} \cdot \mathbf{v}(\cdot) = \mathbf{h}(\mathbf{v}(\cdot))$

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

Bathymetric SLAM

Contractor programming

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM constraints:

1. Evolution constraint:
   - $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot))$
   - $\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot), \mathbf{v}(\cdot)\big)$
2. Inter-temporal constraint:
   - $\mathbf{p}(\cdot) = \mathbf{h}(\mathbf{x}(\cdot))$
   - $\mathbf{w}(\cdot) = \mathbf{h}(\mathbf{v}(\cdot))$
   - $\mathcal{L}_{\text{inter}}\big(\mathbf{p}(\cdot), \mathbf{w}(\cdot), \mathbf{z}(\cdot)\big)$

Bathymetric SLAM

Contractor programming

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM constraints:

1. Evolution constraint:
   - $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot))$
   - $\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot), \mathbf{v}(\cdot)\big)$
2. Inter-temporal constraint:
   - $\mathbf{p}(\cdot) = \mathbf{h}(\mathbf{x}(\cdot))$
   - $\mathbf{w}(\cdot) = \mathbf{h}(\mathbf{v}(\cdot))$
   - $\mathcal{L}_{\text{inter}}\big(\mathbf{p}(\cdot), \mathbf{w}(\cdot), \mathbf{z}(\cdot)\big)$

SLAM algorithm:

1: $\mathcal{C}_{\mathbf{f}}\left([\mathbf{v}](\cdot), [\mathbf{x}](\cdot)\right)$

2: $\mathcal{C}_{\frac{d}{dt}}\left([\mathbf{x}](\cdot), [\mathbf{v}](\cdot)\right)$

3: $\mathcal{C}_{\mathbf{h}}\left([\mathbf{p}](\cdot), [\mathbf{x}](\cdot)\right)$

4: $\mathcal{C}_{\mathbf{h}}\left([\mathbf{w}](\cdot), [\mathbf{v}](\cdot)\right)$

5: $\mathcal{C}_{\text{inter}}\left([\mathbf{p}](\cdot), [\mathbf{w}](\cdot), [\mathbf{z}](\cdot), \varepsilon\right)$

Bathymetric SLAM

Contractor programming

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ \mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \end{cases}$$

SLAM constraints:

1. Evolution constraint:
   - $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot))$
   - $\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot), \mathbf{v}(\cdot)\big)$
2. Inter-temporal constraint:
   - $\mathbf{p}(\cdot) = \mathbf{h}(\mathbf{x}(\cdot))$
   - $\mathbf{w}(\cdot) = \mathbf{h}(\mathbf{v}(\cdot))$
   - $\mathcal{L}_{\text{inter}}\big(\mathbf{p}(\cdot), \mathbf{w}(\cdot), \mathbf{z}(\cdot)\big)$

SLAM algorithm:

1: $\mathcal{C}_{\mathbf{f}}\left([\mathbf{v}](\cdot), [\mathbf{x}](\cdot)\right)$

2: $\mathcal{C}_{\frac{d}{dt}}\left([\mathbf{x}](\cdot), [\mathbf{v}](\cdot)\right)$

3: $\mathcal{C}_{\mathbf{h}}\left([\mathbf{p}](\cdot), [\mathbf{x}](\cdot)\right)$

4: $\mathcal{C}_{\mathbf{h}}\left([\mathbf{w}](\cdot), [\mathbf{v}](\cdot)\right)$

5: $\mathcal{C}_{\text{inter}}\left([\mathbf{p}](\cdot), [\mathbf{w}](\cdot), [\mathbf{z}](\cdot), \varepsilon\right)$

Only **one parameter** to set:

- ε, precision of the approximation of temporal spaces

Bathymetric SLAM

Experimental mission with the Daurade AUV

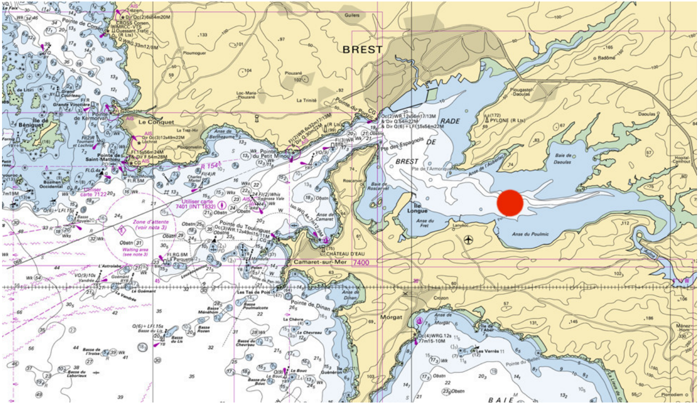
- Daurade: Autonomous Underwater Vehicle
- weight: 1010kg – length: 5m – max depth: 300m

Special thanks to DGA-TN Brest (formerly GESMA)

Bathymetric SLAM

Experimental mission with the Daurade AUV

- 2 hours experimental mission
- in the *Rade de Brest*, Brittany

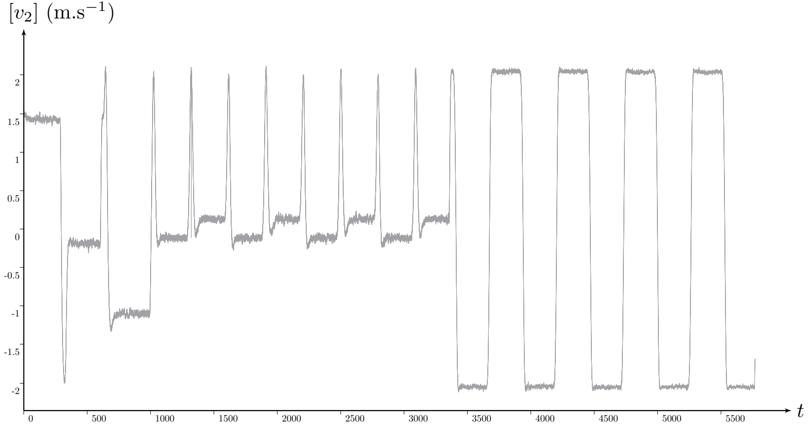


Location: *Polygone de Rascas* – Credits: SHOM

Bathymetric SLAM

Evolution measurements

- velocity measurements obtained with a DVL
- considering uncertainties, building a tube $[\mathbf{v}](\cdot)$

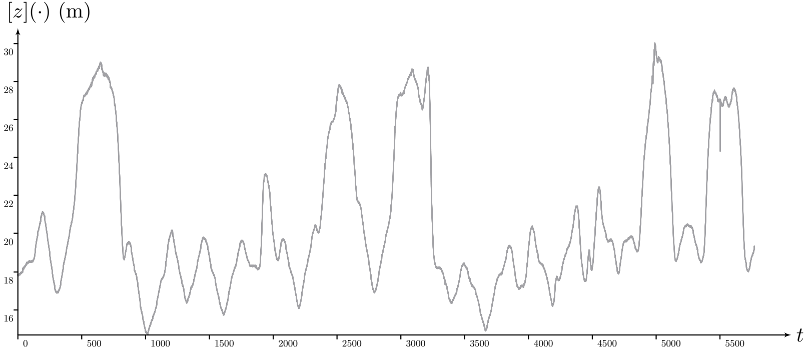


North speed velocity tube $[v_2](\cdot)$

Bathymetric SLAM

Observations measurements: bathymetric values

- DVL, same sensor, can provide **altitude measurements** z_{alt}
- pressure sensor: depth values z_{depth}
- time-dependent measurements, use of **tide models**
- $z = z_{\text{alt}} + z_{\text{depth}} + z_{\text{tide}}$



Observation tube $[z](\cdot)$

Bathymetric SLAM

Dead-reckoning

Actual trajectory:

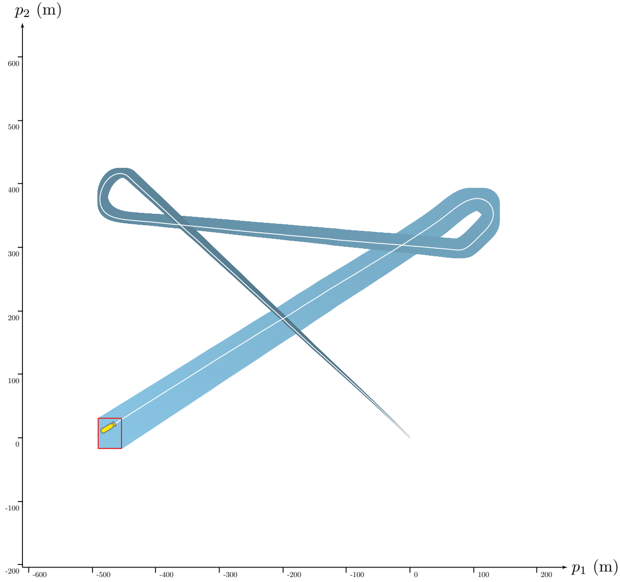
- white

Tube of positions:

- blue

Last position box:

- red



Bathymetric SLAM

Dead-reckoning

Actual trajectory:

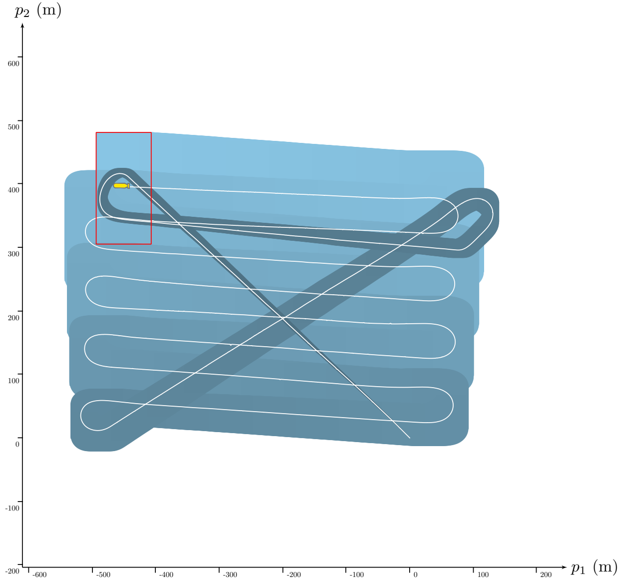
- white

Tube of positions:

- blue

Last position box:

- red



Bathymetric SLAM

Dead-reckoning

Actual trajectory:

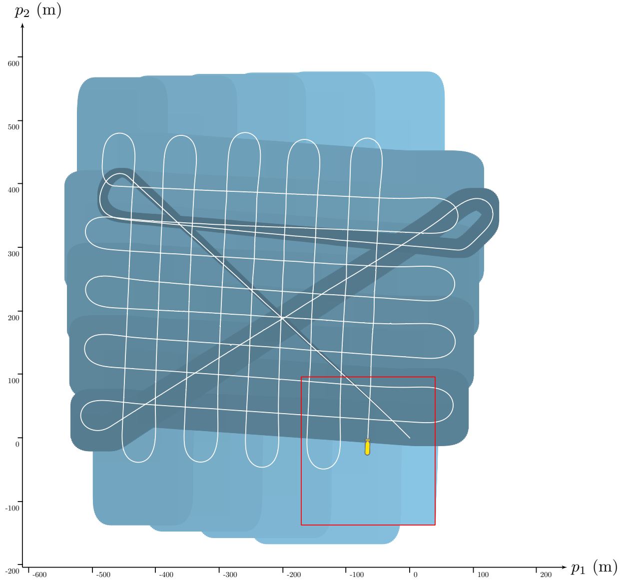
- white

Tube of positions:

- blue

Last position box:

- red



Bathymetric SLAM

SLAM results

Actual trajectory:

- white

Tube of positions:

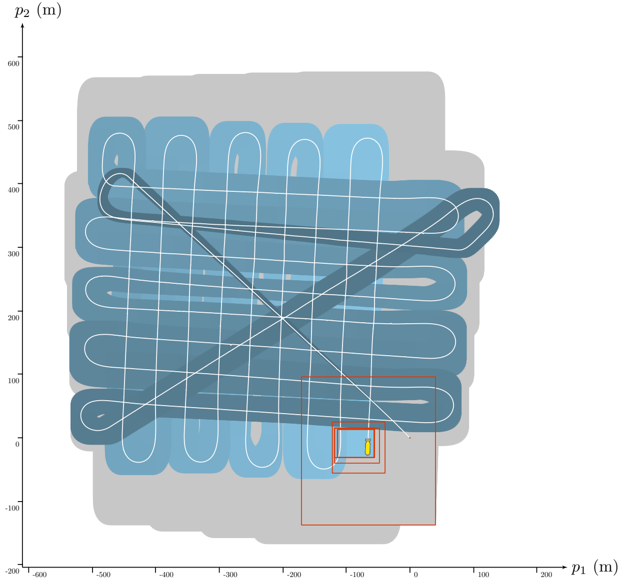
- blue

Last position box:

- red

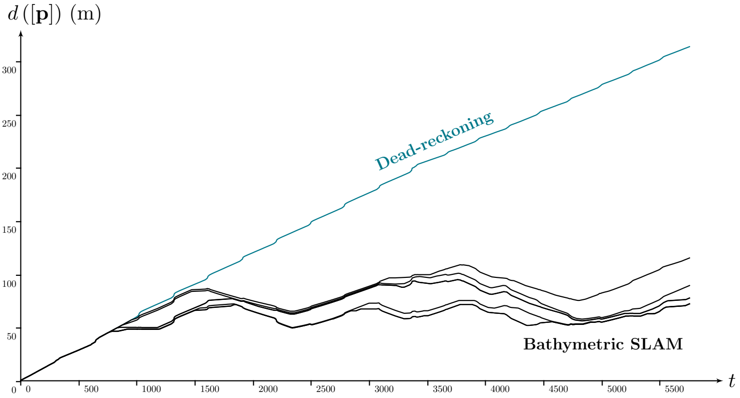
Contracted parts:

- gray



Bathymetric SLAM

SLAM results



Thicknesses of the tube $[\mathbf{p}](\cdot)$

Localization:

- dead-reckoning: linear drift
- SLAM: no cumulated drift

Constraint method:

- iterative resolution
- reliable outputs, pessimism

Section 6

Conclusions

Conclusions

Originality of this work

- localization even in case of **unknown observation function** g
  inter-temporal measurements

Conclusions

Originality of this work

- localization even in case of **unknown observation function** $\mathbf{g}$
  inter-temporal measurements
- consideration of any kind of **time-invariant measurements**
  for instance: temperatures, radioactivity, electric fields

Conclusions

Originality of this work

- localization even in case of **unknown observation function** $\mathbf{g}$
  inter-temporal measurements
- consideration of any kind of **time-invariant measurements**
  for instance: temperatures, radioactivity, electric fields
- **temporal resolution**
  approximation of time references

Conclusions

Originality of this work

- localization even in case of **unknown observation function** $\mathbf{g}$
  inter-temporal measurements
- consideration of any kind of **time-invariant measurements**
  for instance: temperatures, radioactivity, electric fields
- **temporal resolution**
  approximation of time references
- **constraint programming approach**
  simplicity, genericity, few configurations

Conclusions

Originality of this work

- localization even in case of **unknown observation function** $\mathbf{g}$
  inter-temporal measurements
- consideration of any kind of **time-invariant measurements**
  for instance: temperatures, radioactivity, electric fields
- **temporal resolution**
  approximation of time references
- **constraint programming approach**
  simplicity, genericity, few configurations
- study of new **constraints over dynamical systems**
  $\mathcal{L}_{\frac{d}{dt}}$, $\mathcal{L}_{t_1,t_2}$, $\mathcal{L}_{\text{inter}}$, $\dots$

Conclusions

Prospects

- **new experiments** based on a single-beam echosounder
  really convincing results expected

Conclusions

Prospects

- **new experiments** based on a single-beam echosounder
  really convincing results expected
- **merge the approach** with usual probabilistic methods
  e.g. reduce the domain of particle filters

Conclusions

Prospects

- **new experiments** based on a single-beam echosounder
  really convincing results expected
- **merge the approach** with usual probabilistic methods
  e.g. reduce the domain of particle filters
- towards an **active SLAM** approach
  integrate robot control to induce relevant loops

Conclusions

Prospects

- **new experiments** based on a single-beam echosounder
  really convincing results expected
- **merge the approach** with usual probabilistic methods
  e.g. reduce the domain of particle filters
- towards an **active SLAM** approach
  integrate robot control to induce relevant loops
- study a **complementary** constraint $\mathcal{L}_{\overline{t_1,t_2}} : \mathbf{p}(t_1) \neq \mathbf{p}(t_2)$
  and benefit from unconsidered information in temporal sets $\mathbb{T}_{\mathbf{p}}$, $\mathbb{T}_{\mathbf{z}}$

Reliable robot localization:
a constraint programming approach
over dynamical systems

Reliable robot localization: a constraint programming approach over dynamical systems

— thank you for your attention —

Bibliography

- **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

- **Loop detection of mobile robots using interval analysis**
C. Aubry, R. Desmare, L. Jaulin. *Automatica*, 2013

- **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*, submitted

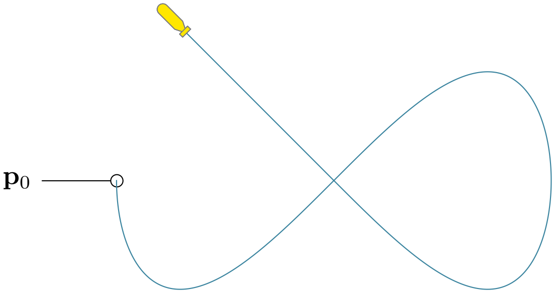
- **Proving the existence of loops in robot trajectories**
S. Rohou, P. Franek, C. Aubry, L. Jaulin. *International Journal of Robotics Research*, submitted

Section 7

Appendices

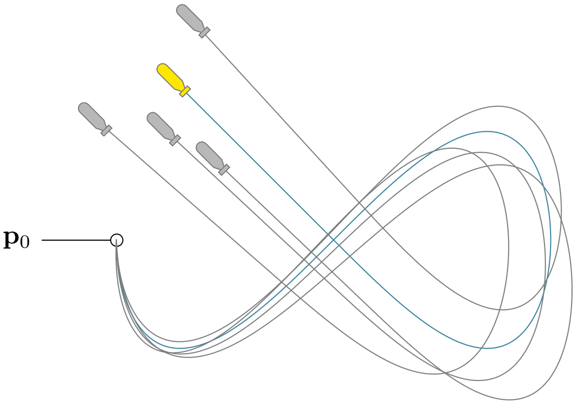
Appendices

Uncertain trajectories



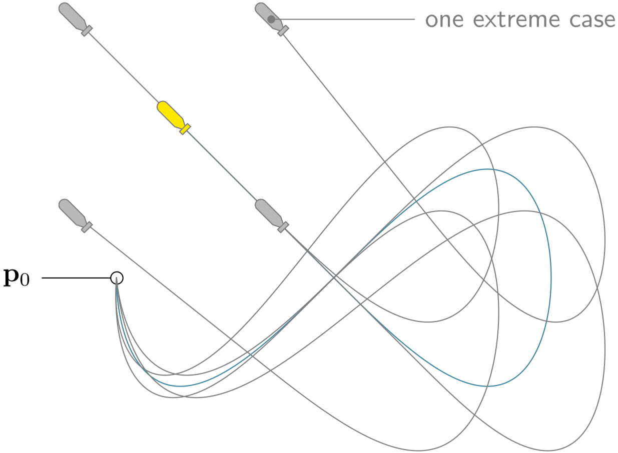
Appendices

Uncertain trajectories



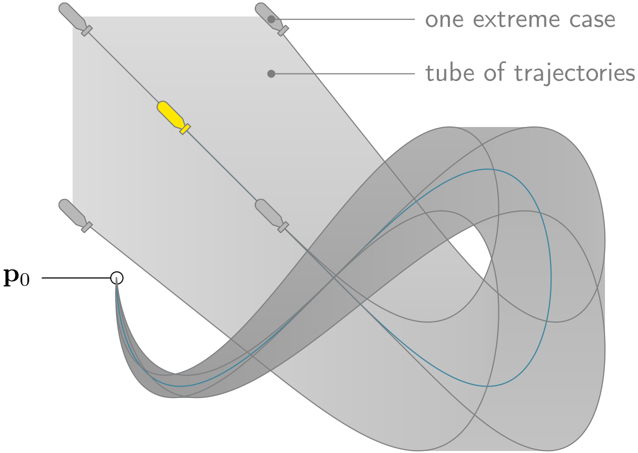
Appendices

Uncertain trajectories



Appendices

Uncertain trajectories



Appendices

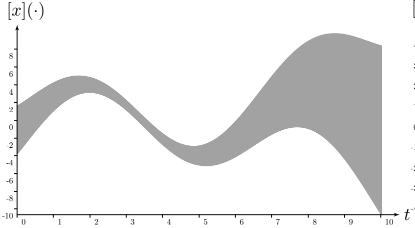
Differential constraint $\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot),\mathbf{v}(\cdot)\big):\dot{\mathbf{x}}(\cdot)=\mathbf{v}(\cdot)$

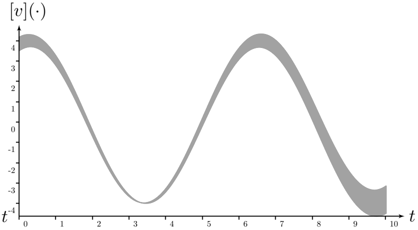
Proposition: contractor $\mathcal{C}_{\frac{d}{dt}}$ defined as

$$\begin{pmatrix}[x](t)\\ [v](t)\end{pmatrix} \xmapsto{\mathcal{C}_{\frac{d}{dt}}} \begin{pmatrix} \bigcap\limits_{t_1=t_0}^{t_f}\left([x](t_1)+\int_{t_1}^{t}[v](\tau)d\tau\right) \\ [v](t)\end{pmatrix}$$

Appendices

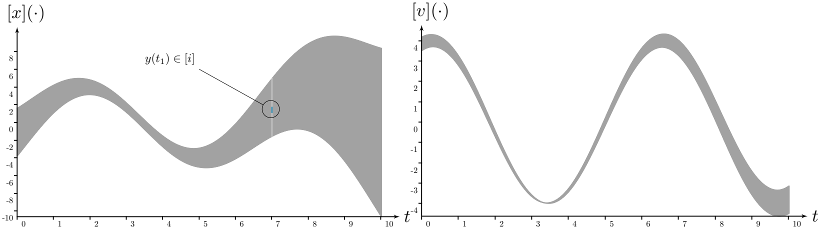
Differential constraint $\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot),\mathbf{v}(\cdot)\big) : \dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$





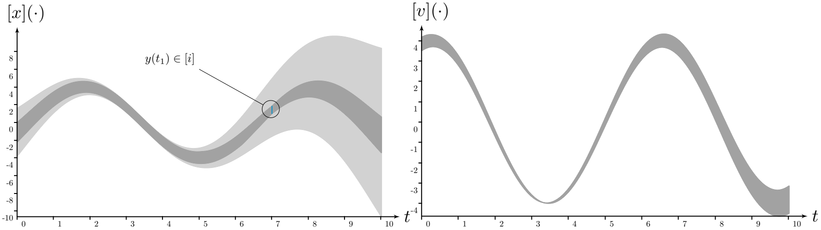
Appendices

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



Appendices

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



Appendices

Evaluation constraint $\mathcal{L}_{\text{eval}}\big(t, \mathbf{z}, \mathbf{y}(\cdot), \mathbf{w}(\cdot)\big)$

$$\mathcal{L}_{\text{eval}}\big(t, \mathbf{z}, \mathbf{y}(\cdot), \mathbf{w}(\cdot)\big) : \begin{cases} \mathbf{z} = \mathbf{y}(t) \\ \dot{\mathbf{y}}(\cdot) = \mathbf{w}(\cdot) \end{cases}$$

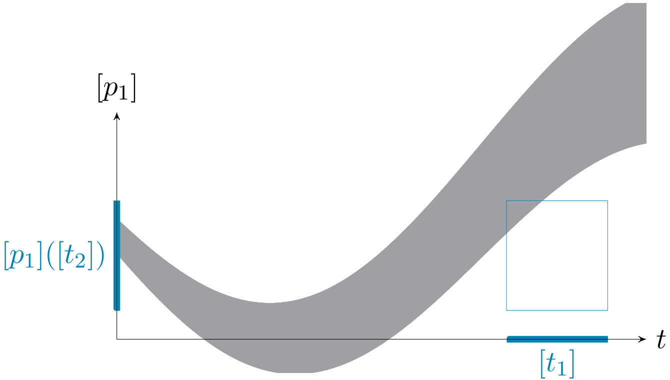
Proposition: contractor $\mathcal{C}_{\text{eval}}$ defined as

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

Appendices

Evaluation constraint $\mathcal{L}_{\text{eval}}\big(t, \mathbf{z}, \mathbf{y}(\cdot), \mathbf{w}(\cdot)\big)$

$$\mathcal{C}_{\text{eval}}\big([t_1], [p_1]([t_2]), [p_1](\cdot), [v_1](\cdot)\big)$$



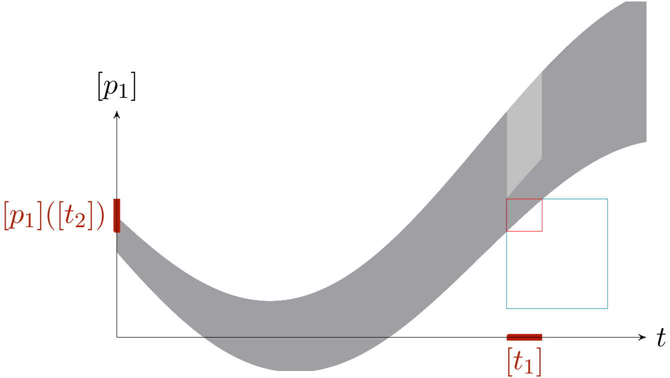
tube $[p_1](\cdot)$ before contraction

Reliable non-linear state estimation involving time uncertainties
Rohou, Jaulin, Mihaylova, Le Bars, Veres
Automatica, submitted

Appendices

Evaluation constraint $\mathcal{L}_{\text{eval}}\big(t, \mathbf{z}, \mathbf{y}(\cdot), \mathbf{w}(\cdot)\big)$

$$\mathcal{C}_{\text{eval}}\big([t_1], [p_1]([t_2]), [p_1](\cdot), [v_1](\cdot)\big)$$



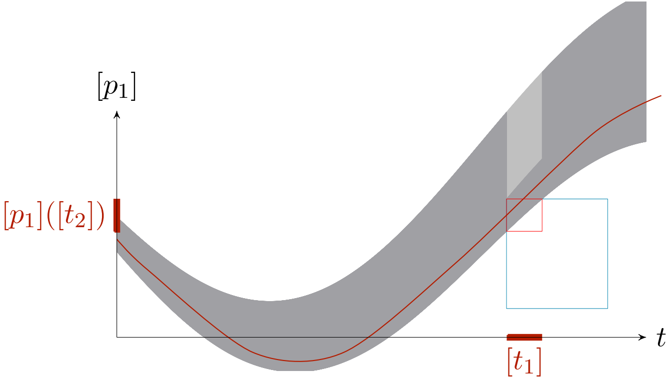
contraction of tube $[p_1](\cdot)$ and both $[p_1]([t_2])$ and $[t_1]$

Reliable non-linear state estimation involving time uncertainties
Rohou, Jaulin, Mihaylova, Le Bars, Veres
Automatica, submitted

Appendices

Evaluation constraint $\mathcal{L}_{\mathrm{eval}}\big(t, \mathbf{z}, \mathbf{y}(\cdot), \mathbf{w}(\cdot)\big)$

$$\mathcal{C}_{\mathrm{eval}}\big([t_1], [p_1]([t_2]), [p_1](\cdot), [v_1](\cdot)\big)$$



contraction of tube $[p_1](\cdot)$ and both $[p_1]([t_2])$ and $[t_1]$

Reliable non-linear state estimation involving time uncertainties

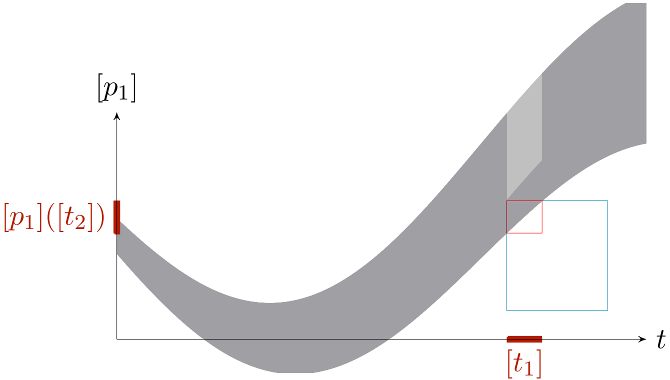
Rohou, Jaulin, Mihaylova, Le Bars, Veres

Automatica, submitted

Appendices

Evaluation constraint $\mathcal{L}_{\mathrm{eval}}\big(t, \mathbf{z}, \mathbf{y}(\cdot), \mathbf{w}(\cdot)\big)$

$$\mathcal{C}_{\mathrm{eval}}\big([t_1], [p_1]([t_2]), [p_1](\cdot), [v_1](\cdot)\big)$$



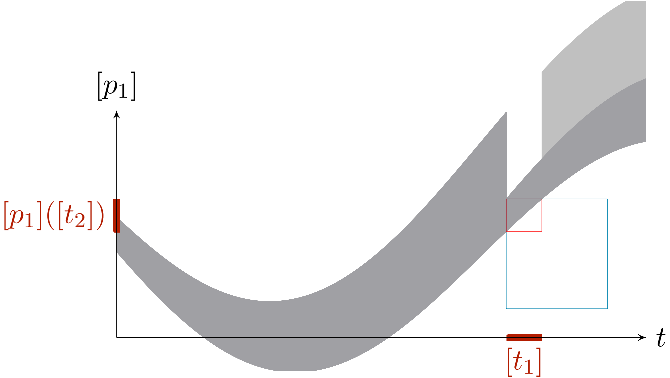
contraction of tube $[p_1](\cdot)$ and both $[p_1]([t_2])$ and $[t_1]$

Reliable non-linear state estimation involving time uncertainties
Rohou, Jaulin, Mihaylova, Le Bars, Veres
Automatica, submitted

Appendices

Evaluation constraint $\mathcal{L}_{\text{eval}}\left(t, \mathbf{z}, \mathbf{y}(\cdot), \mathbf{w}(\cdot)\right)$

$$\mathcal{C}_{\text{eval}}\left([t_1], [p_1]([t_2]), [p_1](\cdot), [v_1](\cdot)\right)$$



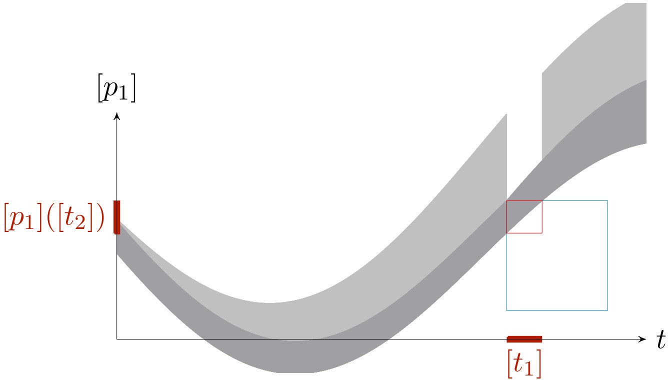
tube contraction in forward

Reliable non-linear state estimation involving time uncertainties
Rohou, Jaulin, Mihaylova, Le Bars, Veres
Automatica, submitted

Appendices

Evaluation constraint $\mathcal{L}_{\text{eval}}\big(t, \mathbf{z}, \mathbf{y}(\cdot), \mathbf{w}(\cdot)\big)$

$$\mathcal{C}_{\text{eval}}\big([t_1], [p_1]([t_2]), [p_1](\cdot), [v_1](\cdot)\big)$$

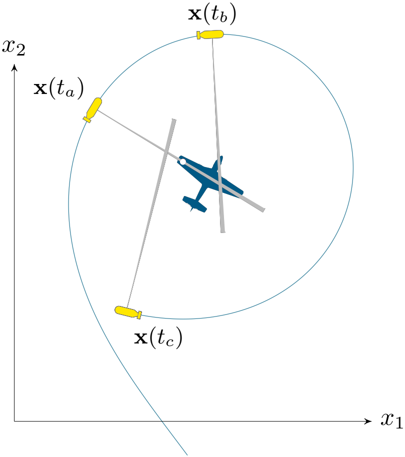


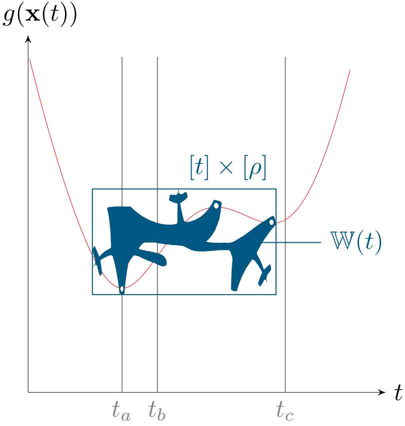
tube contraction in forward/backward

Reliable non-linear state estimation involving time uncertainties
Rohou, Jaulin, Mihaylova, Le Bars, Veres
Automatica, submitted

Appendices

Wreck-based localization method





Appendices

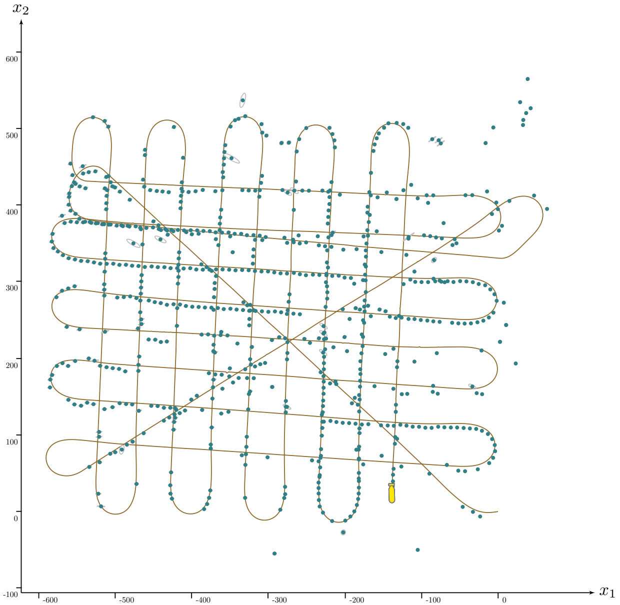
USBL

Appendices

USBL

Appendices

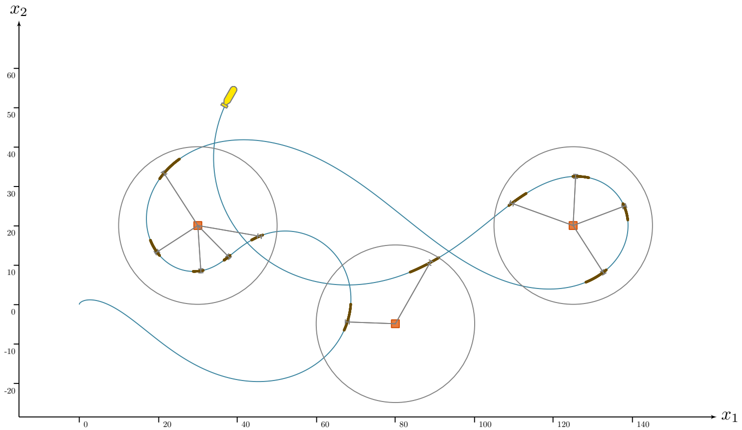
USBL



Appendices

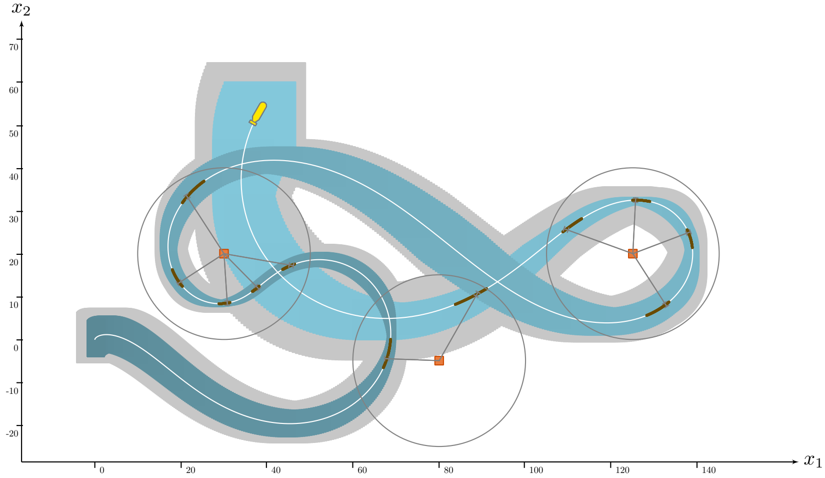
State estimation: mobile robotics

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot), \mathbf{u}(\cdot)) \\ z_i = g(\mathbf{x}(t_i)) \end{cases}$$



Appendices

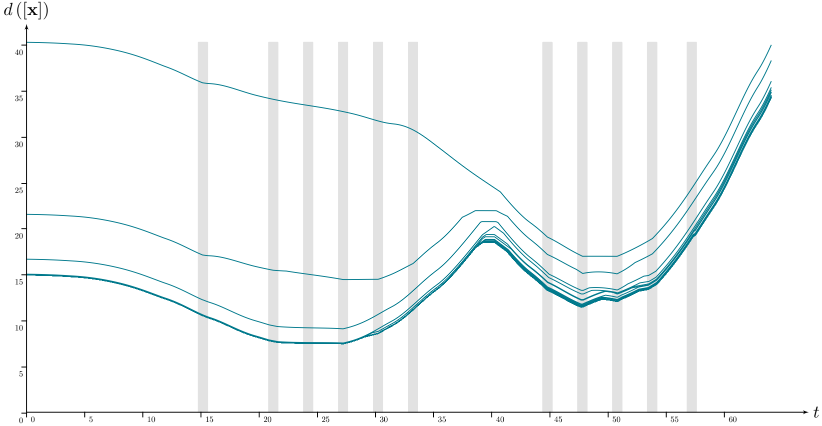
State estimation: mobile robotics



A mobile robot evolving among beacons – bounded error context

Appendices

State estimation: mobile robotics



Drifting curves – thicknesses of positioning tubes

Appendices

The $\mathcal{L}_{t_1,t_2}$ constraint: decomposition

$\mathcal{L}_{t_1,t_2}$, not canonic, amounts to the following composition:

Variables: t_1, t_2, $\mathbf{y}(\cdot)$

Constraints:

- $\mathbf{y}(t_1) = \mathbf{y}(t_2)$

Domains: $[t_1]$, $[t_2]$, $[\mathbf{y}](\cdot)$

Appendices

The $\mathcal{L}_{t_1,t_2}$ constraint: decomposition

$\mathcal{L}_{t_1,t_2}$, not canonic, amounts to the following composition:

$$\left\{\begin{array}{l}\textbf{Variables: } t_1,\ t_2,\ \mathbf{y}(\cdot)\\[1em] \textbf{Constraints:}\\[1em] \quad \blacktriangleright\ \mathbf{y}(t_1)=\mathbf{y}(t_2) \iff \left\{\begin{array}{l}\mathbf{a}=\mathbf{y}(t_1)\\ \mathbf{b}=\mathbf{y}(t_2)\\ \mathbf{a}=\mathbf{b}\end{array}\right.\\[2em] \textbf{Domains: } [t_1],\ [t_2],\ [\mathbf{y}](\cdot)\end{array}\right.$$

Appendices

The $\mathcal{L}_{t_1,t_2}$ constraint: decomposition

$\mathcal{L}_{t_1,t_2}$, not canonic, amounts to the following composition:

$$\left\{\begin{array}{l}
\textbf{Variables: } t_1,\ t_2,\ \mathbf{y}(\cdot) \\
\textbf{Constraints:} \\
\quad \blacktriangleright\ \mathbf{y}(t_1)=\mathbf{y}(t_2) \Longleftrightarrow \left\{\begin{array}{l}\mathbf{a}=\mathbf{y}(t_1)\\ \mathbf{b}=\mathbf{y}(t_2)\\ \mathbf{a}=\mathbf{b}\end{array}\right. \Longleftrightarrow \left\{\begin{array}{l}\mathcal{L}_{\text{eval}}\left(t_1,\mathbf{a},\mathbf{y}(\cdot),\mathbf{w}(\cdot)\right)\\ \mathcal{L}_{\text{eval}}\left(t_2,\mathbf{b},\mathbf{y}(\cdot),\mathbf{w}(\cdot)\right)\\ \mathbf{a}=\mathbf{b}\end{array}\right. \\
\textbf{Domains: } [t_1],\ [t_2],\ [\mathbf{y}](\cdot)
\end{array}\right.$$

Appendices

The $\mathcal{L}_{t_1,t_2}$ constraint: decomposition

$\mathcal{L}_{t_1,t_2}$, not canonic, amounts to the following composition:

$$\left\{\begin{array}{l}
\textbf{Variables: } t_1,\ t_2,\ \mathbf{y}(\cdot),\ \mathbf{w}(\cdot) \\
\textbf{Constraints:} \\
\quad \blacktriangleright\ \mathbf{y}(t_1)=\mathbf{y}(t_2) \iff \left\{\begin{array}{l}\mathbf{a}=\mathbf{y}(t_1)\\ \mathbf{b}=\mathbf{y}(t_2)\\ \mathbf{a}=\mathbf{b}\end{array}\right. \iff \left\{\begin{array}{l}\mathcal{L}_{\text{eval}}\left(t_1,\mathbf{a},\mathbf{y}(\cdot),\mathbf{w}(\cdot)\right)\\ \mathcal{L}_{\text{eval}}\left(t_2,\mathbf{b},\mathbf{y}(\cdot),\mathbf{w}(\cdot)\right)\\ \mathbf{a}=\mathbf{b}\end{array}\right. \\
\quad \blacktriangleright\ \dot{\mathbf{y}}(\cdot)=\mathbf{w}(\cdot) \\
\textbf{Domains: } [t_1],\ [t_2],\ [\mathbf{y}](\cdot),\ [\mathbf{w}](\cdot)
\end{array}\right.$$

Appendices

The $\mathcal{L}_{t_1,t_2}$ constraint: decomposition

$\mathcal{L}_{t_1,t_2}$, not canonic, amounts to the following composition:

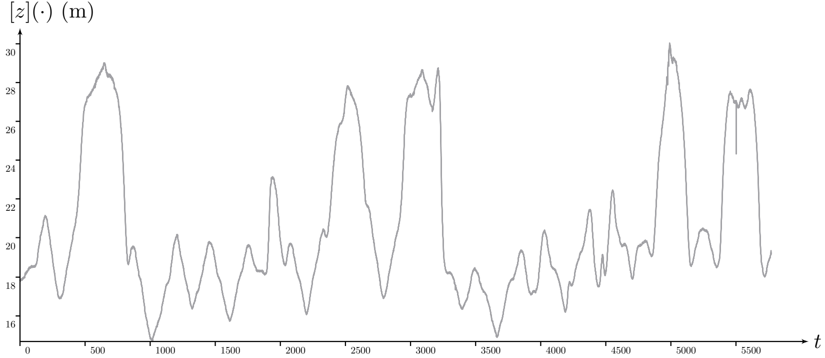
$$\left\{\begin{array}{l}
\textbf{Variables: } t_1,\ t_2,\ \mathbf{y}(\cdot),\ \mathbf{w}(\cdot) \\
\textbf{Constraints:} \\
\quad \blacktriangleright\ \mathbf{y}(t_1)=\mathbf{y}(t_2) \iff \left\{\begin{array}{l}\mathbf{a}=\mathbf{y}(t_1)\\ \mathbf{b}=\mathbf{y}(t_2)\\ \mathbf{a}=\mathbf{b}\end{array}\right. \iff \left\{\begin{array}{l}\mathcal{L}_{\text{eval}}\left(t_1,\mathbf{a},\mathbf{y}(\cdot),\mathbf{w}(\cdot)\right)\\ \mathcal{L}_{\text{eval}}\left(t_2,\mathbf{b},\mathbf{y}(\cdot),\mathbf{w}(\cdot)\right)\\ \mathbf{a}=\mathbf{b}\end{array}\right. \\
\quad \blacktriangleright\ \dot{\mathbf{y}}(\cdot)=\mathbf{w}(\cdot) \\
\textbf{Domains: } [t_1],\ [t_2],\ [\mathbf{y}](\cdot),\ [\mathbf{w}](\cdot)
\end{array}\right.$$

$\mathcal{L}_{\text{eval}}$ constraint:

- $\mathcal{L}_{\text{eval}}\left(t,\mathbf{z},\mathbf{y}(\cdot),\mathbf{w}(\cdot)\right):\left\{\begin{array}{l}\mathbf{z}=\mathbf{y}(t)\\ \dot{\mathbf{y}}(\cdot)=\mathbf{w}(\cdot)\end{array}\right.$

Appendices

Daurade mission: 20/10/2015 11h



Observation tube $[z](\cdot)$: bathymetric measurements

Appendices

Daurade mission: 20/10/2015 11h

Actual trajectory:

- white

Tube of positions:

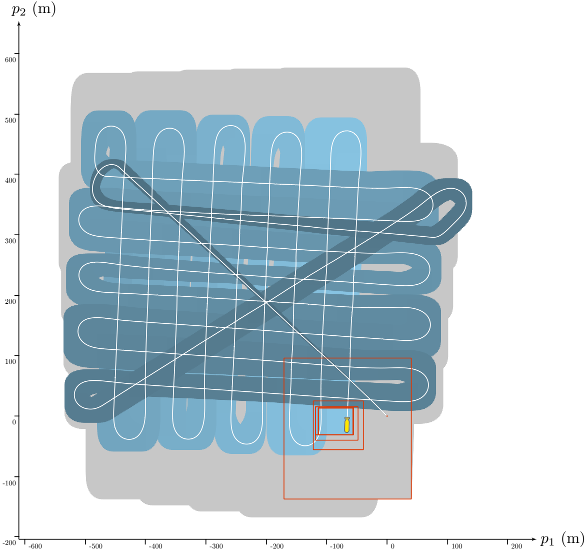
- blue

Last position box:

- red

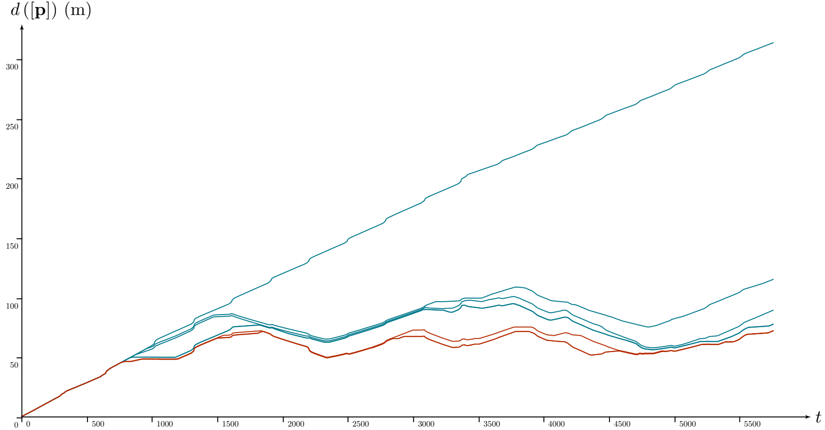
Contracted parts:

- gray



Appendices

Daurade mission: 20/10/2015 11h



Thicknesses of the tube $[\mathbf{p}](\cdot)$

Appendices

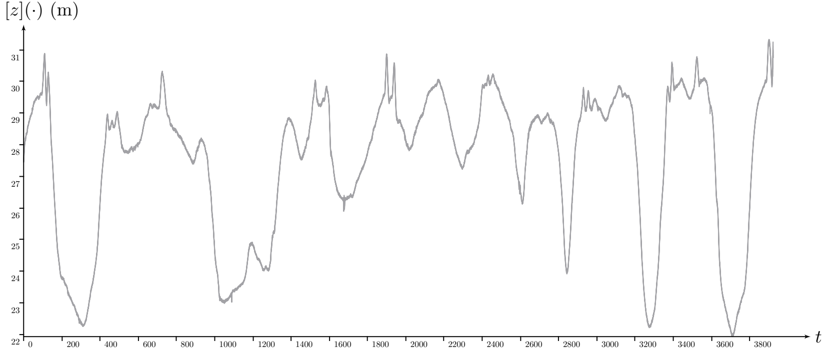
Daurade mission: 20/10/2015 11h

SLAM iterations on *Daurade*'s experiment:

i	loop detections	loop proofs	computation time	cumulated comp. time	$[\mathbf{p}](t_f)$ contraction	SLAM algorithm
1	122	104	259s	259s	63.22%	fast
2	128	112	192s	451s	71.46%	fast
3	128	112	172s	623s	75.17%	fast
4	129	115	180s	803s	75.22%	fast
5	129	115	182s	985s (0h16)	**75.22%**	fast
fixed point						
6	129	115	2708s (0h45)	3693s (1h02)	76.91%	accurate
7	129	115	2506s (0h41)	6199s (1h43)	76.96%	accurate
8	129	115	2391s (0h40)	8590s (2h23)	**76.96%**	accurate
fixed point						

Appendices

Daurade mission: 19/10/2015 10h



Observation tube $[z](\cdot)$: bathymetric measurements

Appendices

Daurade mission: 19/10/2015 10h

Actual trajectory:

- white

Tube of positions:

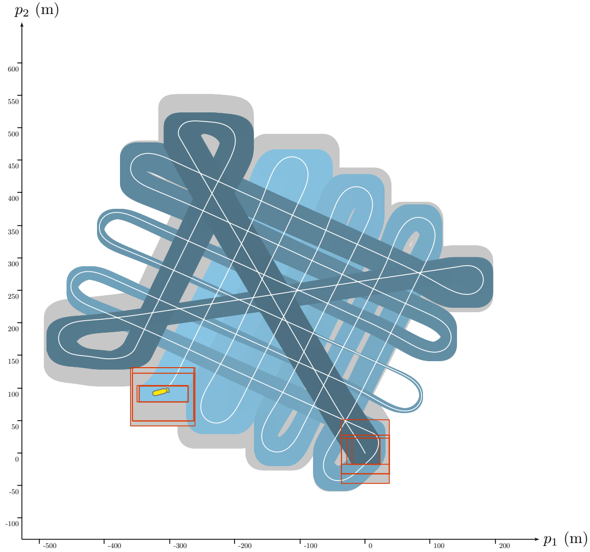
- blue

Last position box:

- red

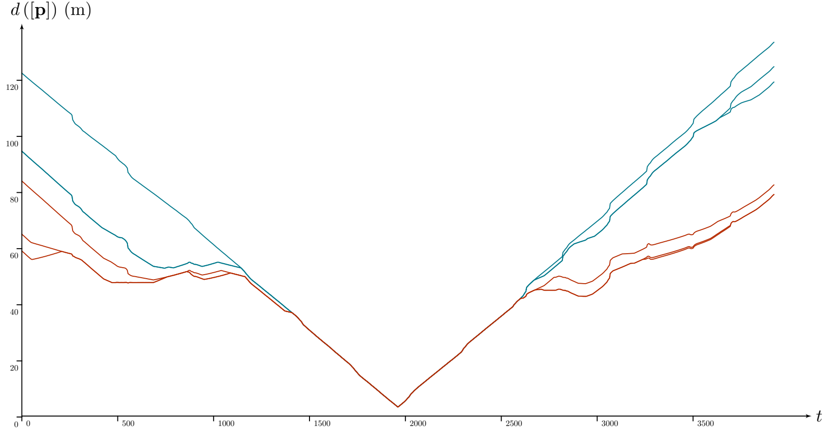
Contracted parts:

- gray



Appendices

Daurade mission: 19/10/2015 10h



Thicknesses of the tube $[\mathbf{p}](\cdot)$

Appendices

Daurade mission: 19/10/2015 10h

SLAM iterations on *Daurade*'s experiment:

i	loop detections	loop proofs	computation time	cumulated comp. time	$[\mathbf{p}](t_0)$ contraction	SLAM algorithm
1	76	65	93s	93s	22.76%	fast
2	78	67	90s	183s	22.76%	fast
3	78	67	108s	291s (0h05)	**22.76%**	fast
fixed point						
4	78	67	1726s (0h29)	2017s (0h34)	31.47%	accurate
5	77	67	1392s (0h23)	3409s (0h57)	46.96%	accurate
6	77	67	1424s (0h24)	4833s (1h21)	51.85%	accurate
7	77	68	1470s (0h24)	6303s (1h45)	**51.85%**	accurate
fixed point						

Appendices

Dynamical constraints

SLAM problem was an opportunity to study the following elementary constraints:

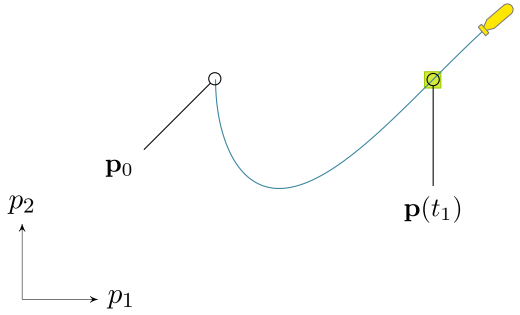
1. Evolution constraint
   $\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot),\mathbf{v}(\cdot)\big):\dot{\mathbf{x}}(\cdot)=\mathbf{v}(\cdot)$
2. Evaluation constraint
   $\mathcal{L}_{\mathrm{eval}}\big(t,\mathbf{z},\mathbf{y}(\cdot),\mathbf{w}(\cdot)\big):\begin{cases}\mathbf{z}=\mathbf{y}(t)\\ \dot{\mathbf{y}}(\cdot)=\mathbf{w}(\cdot)\end{cases}$
3. Inter-temporal evaluation constraint
   $\mathcal{L}_{t_1,t_2}\big(t_1,t_2,\mathbf{y}(\cdot),\mathbf{w}(\cdot)\big):\begin{cases}\mathbf{y}(t_1)=\mathbf{y}(t_2)\\ \dot{\mathbf{y}}(\cdot)=\mathbf{w}(\cdot)\end{cases}$
4. Inter-temporal implication constraint
   $\mathcal{L}_{\mathrm{inter}}\big(\mathbf{y}(\cdot),\mathbf{w}(\cdot),\mathbf{z}(\cdot)\big):\begin{cases}\mathbf{y}(t_1)=\mathbf{y}(t_2)\implies\mathbf{z}(t_1)=\mathbf{z}(t_2)\\ \dot{\mathbf{y}}(\cdot)=\mathbf{w}(\cdot)\end{cases}$

Appendices

Dynamical constraints

Example:

- $\mathbf{x} = (p_1, p_2, p_3, \theta)^\intercal \in \mathbb{R}^4$
- $\mathbf{p}(t) = \mathbf{h}(\mathbf{x}(t)) = \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix}$

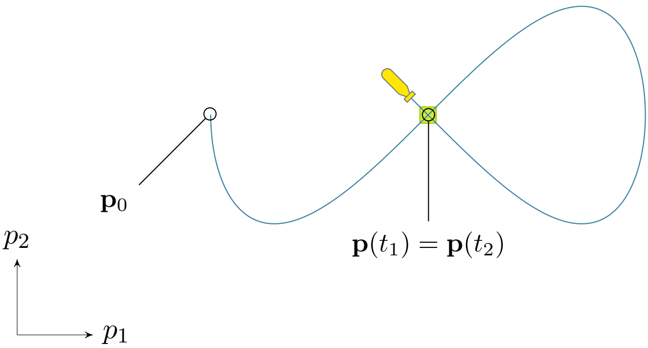


Appendices

Dynamical constraints

Example:

- $\mathbf{x} = (p_1, p_2, p_3, \theta)^\intercal \in \mathbb{R}^4$
- $\mathbf{p}(t) = \mathbf{h}(\mathbf{x}(t)) = \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix}$

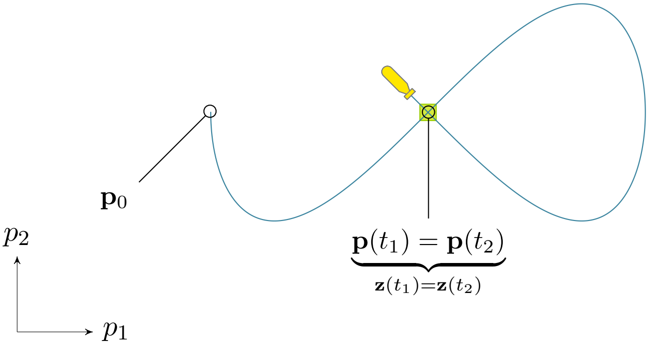


Appendices

Dynamical constraints

Example:

- $\mathbf{x} = (p_1, p_2, p_3, \theta)^\intercal \in \mathbb{R}^4$
- $\mathbf{p}(t) = \mathbf{h}(\mathbf{x}(t)) = \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix}$



Appendices

Constraints: decomposition

Complex constraints can be broken down.
Example, observation function for range-only state estimation:

$$\mathcal{L}_{\mathbf{g}}\big(\rho, \mathbf{a}, \mathbf{b}\big) : \rho = \sqrt{\left(a_1 - b_1\right)^2 + \left(a_2 - b_2\right)^2} \Longleftrightarrow \left\{ \begin{array}{ccl} c & = & a_1 - b_1 \\ d & = & a_2 - b_2 \\ i & = & c^2 \\ j & = & d^2 \\ l & = & i + j \\ \rho & = & \sqrt{l} \end{array} \right.$$

- $c, d, \dots, l$: intermediate variables used for ease of decomposition
- network of **elementary constraints**: $\mathcal{L}_-$, $\mathcal{L}_+$, $\mathcal{L}_{(\cdot)^2}$, $\mathcal{L}_{\sqrt{\cdot}}$

Appendices

Constraints: application

Each elementary constraint $\mathcal{L}$ is applied by an operator:

- a **contractor** $\mathcal{C}_{\mathcal{L}} : \mathbb{IR}^n \to \mathbb{IR}^n$
- example, $\mathcal{C}_+$:

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

Contractor programming: Chabert and Jaulin 2009

- contractor seen as a subset of $\mathbb{R}^n$

$\Longrightarrow$ operations on sets applicable on contractors: $\cup$, $\cap$, ...

$\Longrightarrow$ simple **combinations** of primitive contractors

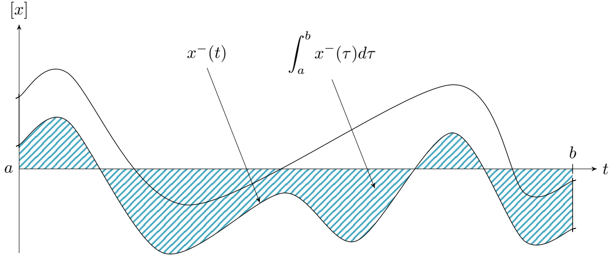
Appendices

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]



blue area: lower bound of the tube's integral

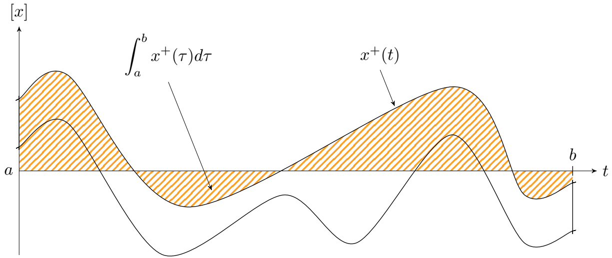
Appendices

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]



orange area: upper bound of the tube's integral

Appendices

Back to the trajectories space

At this point:

- temporal set $\mathbb{T}_{\mathbf{p}}$ contracted,
- it remains to contract the positions tube $[\mathbf{p}](\cdot)$

Constraint of interest:

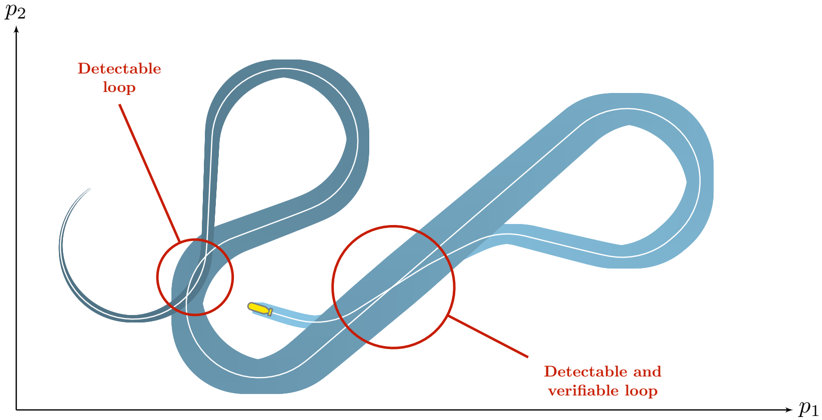
- $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\}$
- *backward way*: from the set $\mathbb{T}^*_{\mathbf{p}}$ to the trajectory $\mathbf{p}(\cdot)$

However:

- pessimistic enclosure $[\mathbf{p}](\cdot)$: $\mathbb{T}_{\mathbf{p}}$ may not contain a solution
  $\implies$ **risk of false contraction**
- before contracting $[\mathbf{p}](\cdot)$, need to prove that $\exists \mathbf{t} \in \mathbb{T}_{\mathbf{p}} \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)$
- physically: we need to **prove loops** along the trajectory $\mathbf{p}(\cdot)$

Appendices

Proving the existence of loops

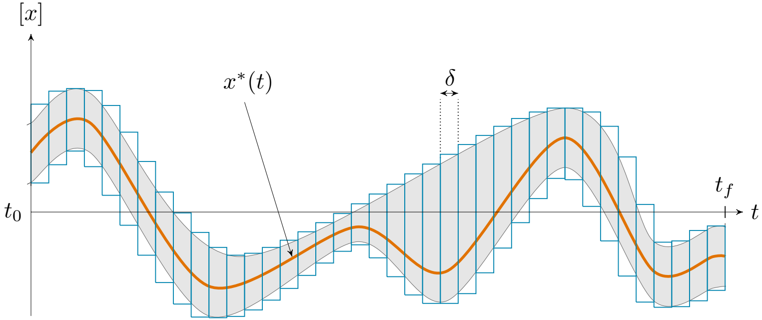


Proving the existence of loops in robot trajectories
S. Rohou, P. Franek, C. Aubry, L. Jaulin. *International Journal of Robotics Research*, submitted

Appendices

Tubex library

An open-source C++ library providing tools to guarantee computations over sets of trajectories.



http://www.simon-rohou.fr/research/tubex-lib/