A temporal approach for the SLAM problem

Simon Rohou

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

LS2N Seminar
5$^{\mathrm{th}}$ April 2018

LABORATOIRE DES SCIENCES DU NUMÉRIQUE DE NANTES

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Contredo project

AGENCE NATIONALE DE LA RECHERCHE

Outline

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)

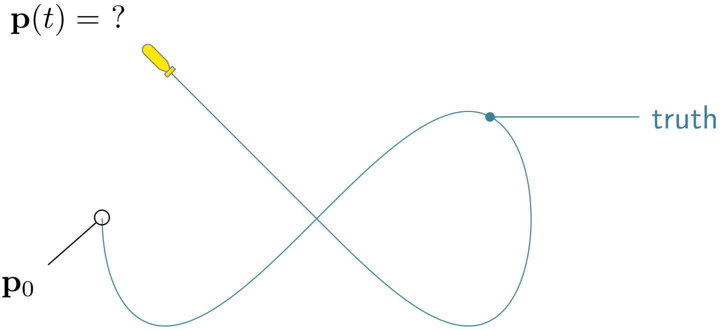
Titanic wreck: 3821m deep

Lost MH370 aircraft: up to 6000m deep

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

Simple solution, **dead-reckoning**:

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

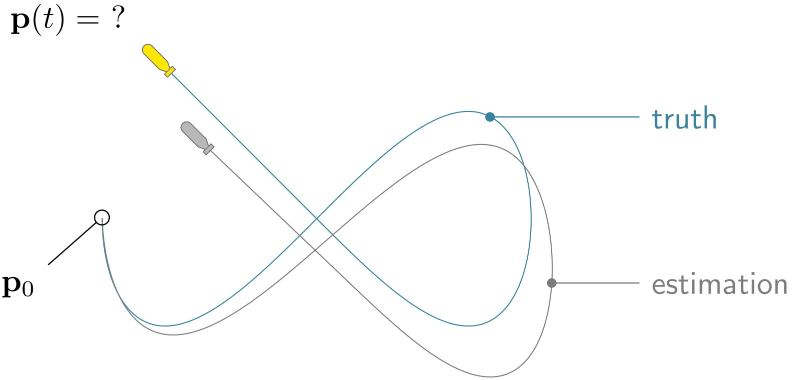


Motivations

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

Simple solution, **dead-reckoning**:

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

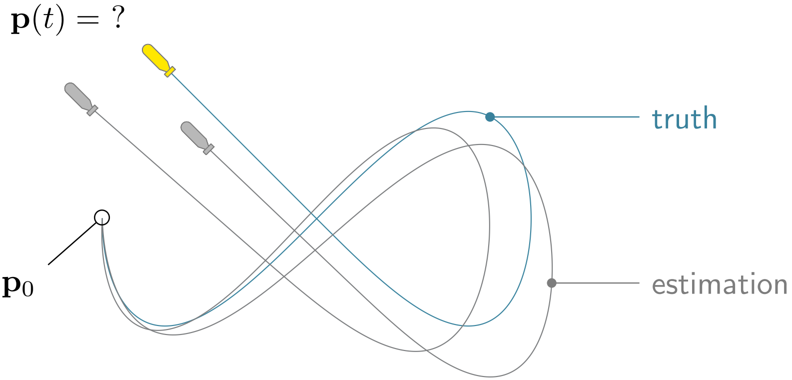


Motivations

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

Simple solution, **dead-reckoning**:

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

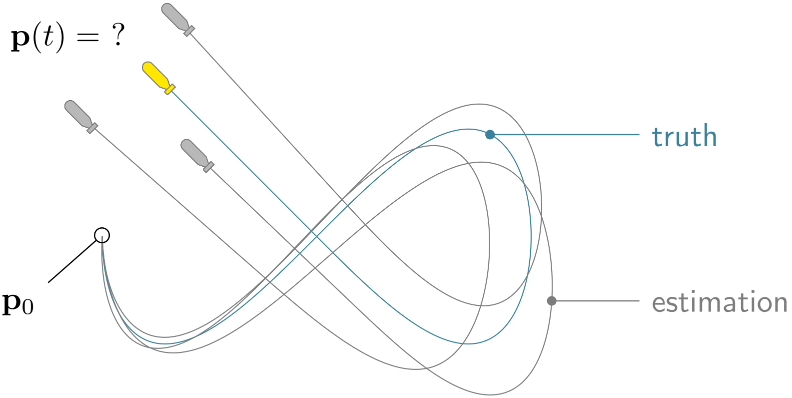


Motivations

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

Simple solution, **dead-reckoning**:

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

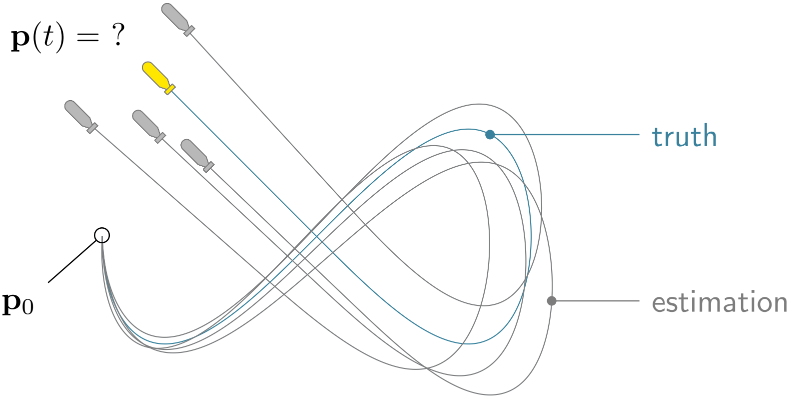


Motivations

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

Simple solution, **dead-reckoning**:

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

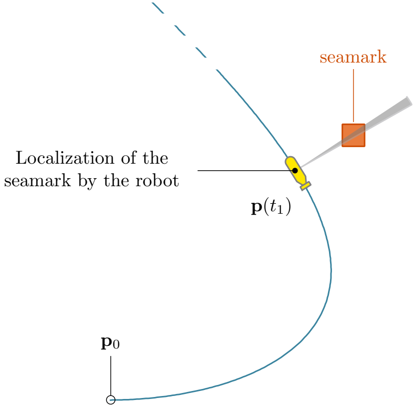


Motivations

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

Exploration solution, **SLAM**:

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

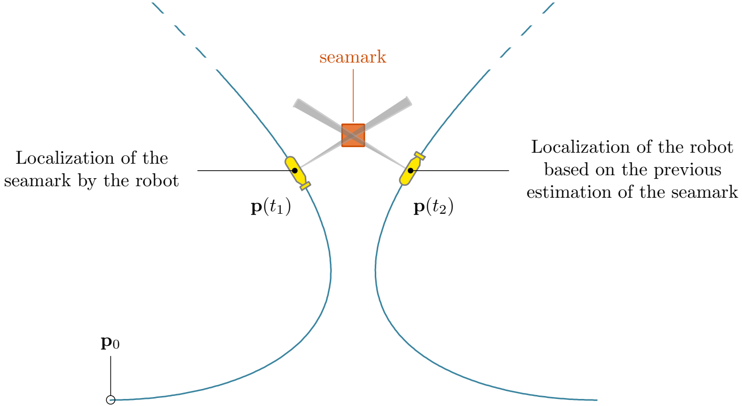


Motivations

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

Exploration solution, **SLAM**:

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

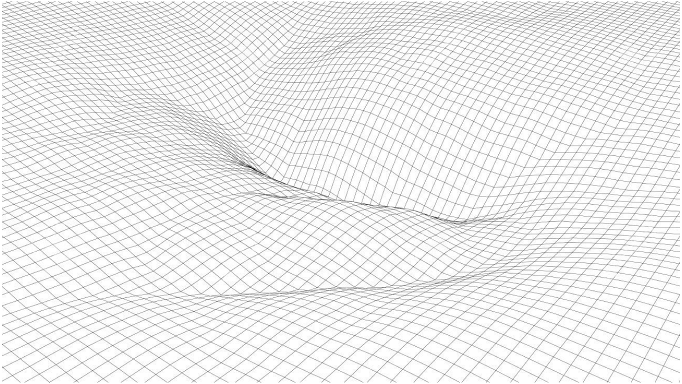


Motivations

Problem: homogeneous environments

Under the surface:

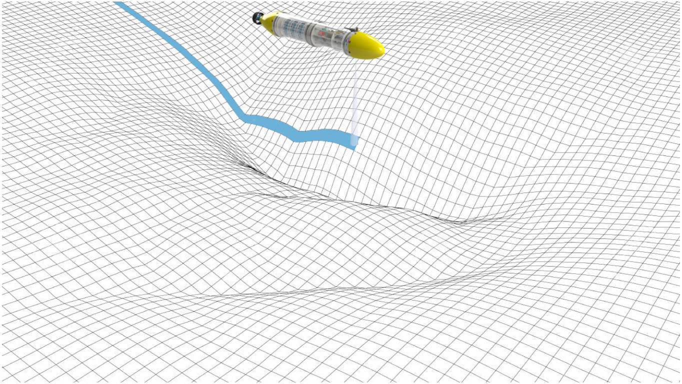
- **no seamarks** or points of interest
- usual SLAM methods do not apply



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:

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

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:

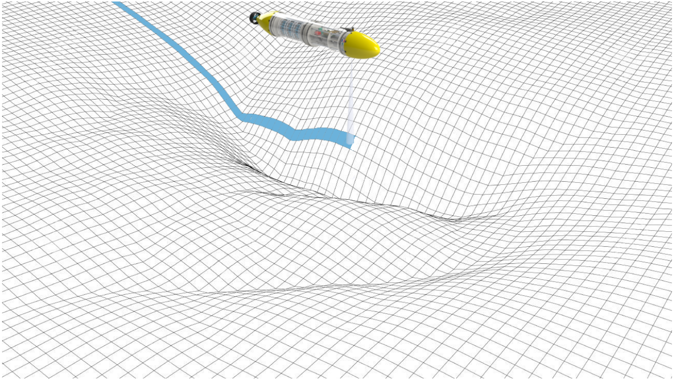
- $\mathbf{x} \in \mathbb{R}^n$ is the state vector (position, bearing, ...)
- $\mathbf{u} \in \mathbb{R}^m$ is the input vector (command)
- $\mathbf{f} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is the *evolution* function
- $\mathbf{g} : \mathbb{R}^n \to \mathbb{R}^p$ is the *observation*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

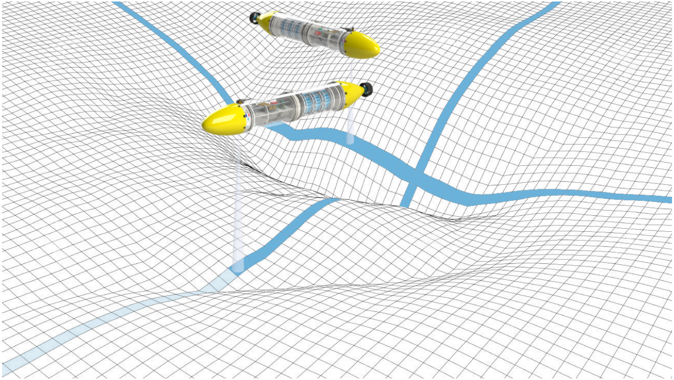


SLAM formalization

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:

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

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:

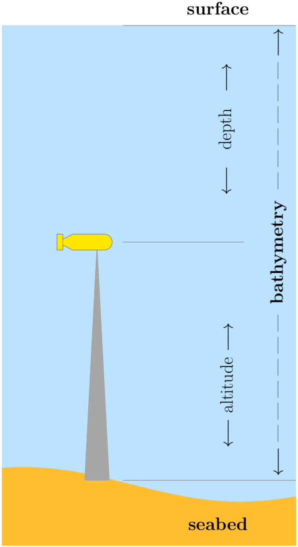
$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) & \text{(navigation)} \\ \cancel{\mathbf{z}(t) = \mathbf{g}(\mathbf{x}(t))} & \text{(observation)} \\ \underbrace{\mathbf{h}\big(\mathbf{x}(t_1)\big) = \mathbf{h}\big(\mathbf{x}(t_2)\big)}_{\text{same \textbf{state configurations}}} \implies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

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:

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

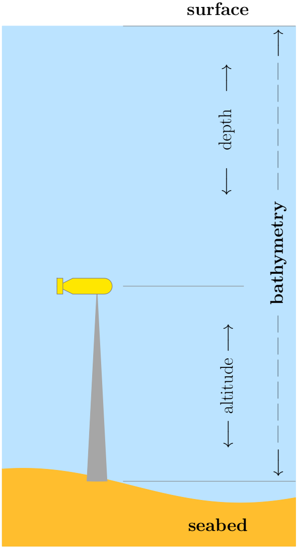
In the case of **bathymetric SLAM**:

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

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

SLAM formalization

New SLAM formalism: inter-temporal measurements



Inter-temporal configuration:

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

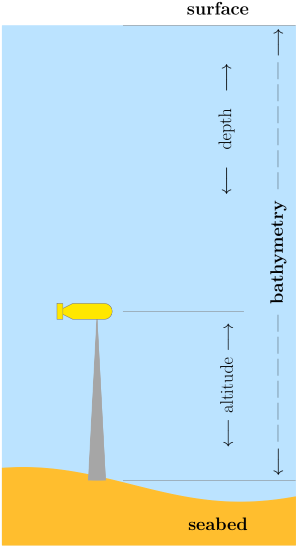
In the case of **bathymetric SLAM**:

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

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

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:

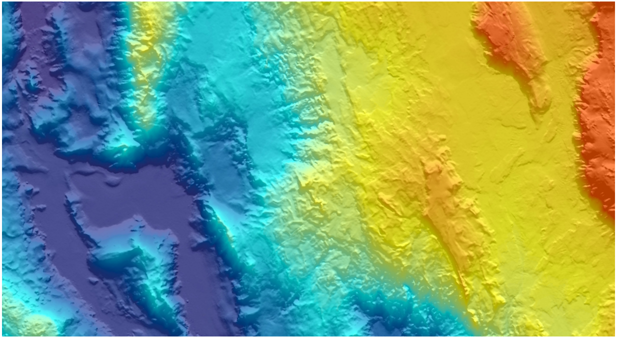
- bounded error context
- no unpredictable change in the environment

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

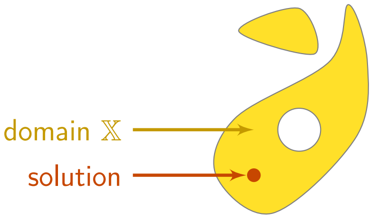
Constraint programming

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:

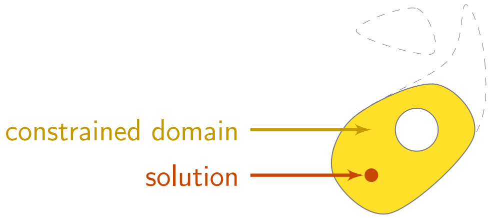
$$\left\{\begin{array}{l}\textbf{Variables: } \mathbf{x}\\ \textbf{Constraints:}\\ \\ \\ \\ \textbf{Domains: } \mathbb{X}\end{array}\right.$$

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, ...



Constraint network:

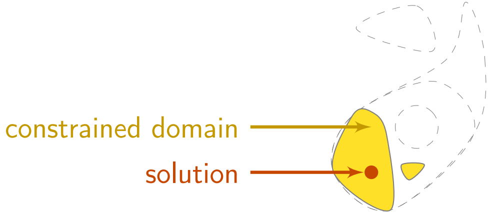
$$\left\{\begin{array}{l}\textbf{Variables: } \mathbf{x} \\ \textbf{Constraints:} \\ \quad 1.\ \mathcal{L}_1(\mathbf{x}) \\ \\ \textbf{Domains: } \mathbb{X}\end{array}\right.$$

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, ...



Constraint network:

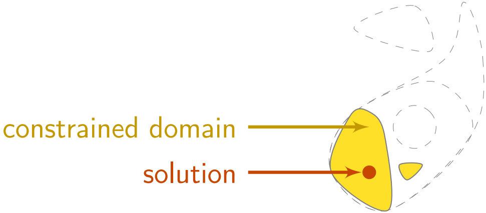
$$\left\{\begin{array}{l}\textbf{Variables: } \mathbf{x} \\ \textbf{Constraints:} \\ \quad 1.\ \mathcal{L}_1(\mathbf{x}) \\ \quad 2.\ \mathcal{L}_2(\mathbf{x}) \\ \\ \textbf{Domains: } \mathbb{X}\end{array}\right.$$

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, ...



Constraint network:

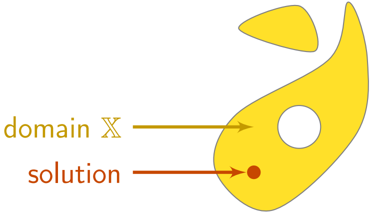
$$\begin{cases} \textbf{Variables: } \mathbf{x} \\ \textbf{Constraints:} \\ \quad 1.\ \mathcal{L}_1(\mathbf{x}) \\ \quad 2.\ \mathcal{L}_2(\mathbf{x}) \\ \quad 3.\ \dots \\ \textbf{Domains: } \mathbb{X} \end{cases}$$

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. ...

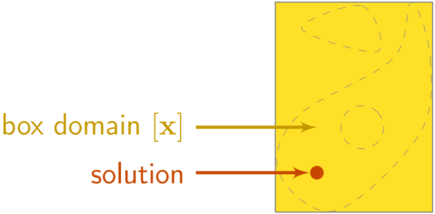
Domains: $\mathbb{X}$

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, ...
- 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. ...

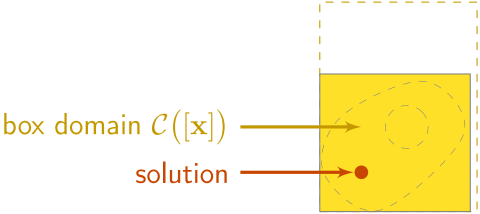
Domains: $[\mathbf{x}]$

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:

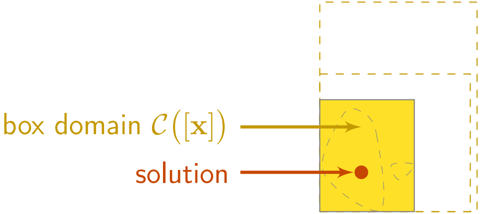
$$\begin{cases} \textbf{Variables: } \mathbf{x} \\ \textbf{Constraints:} \\ \quad 1.\ \mathcal{L}_1(\mathbf{x}) \\ \quad 2.\ \mathcal{L}_2(\mathbf{x}) \\ \quad 3.\ \dots \\ \textbf{Domains: } [\mathbf{x}] \end{cases}$$

Constraint programming

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

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

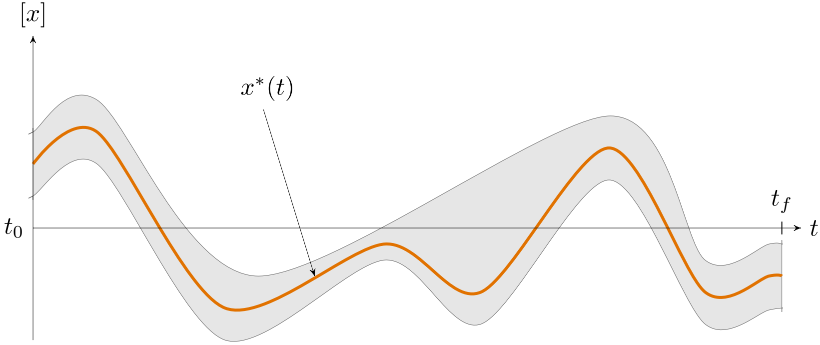
Objectives:

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

Constraint programming

Tubes

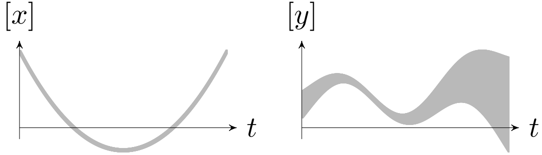
Tube $[x](\cdot)$: interval of trajectories $[x^-(\cdot), x^+(\cdot)]$
such that $\forall t \in \mathbb{R},\ x^-(t) \leqslant x^+(t)$



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

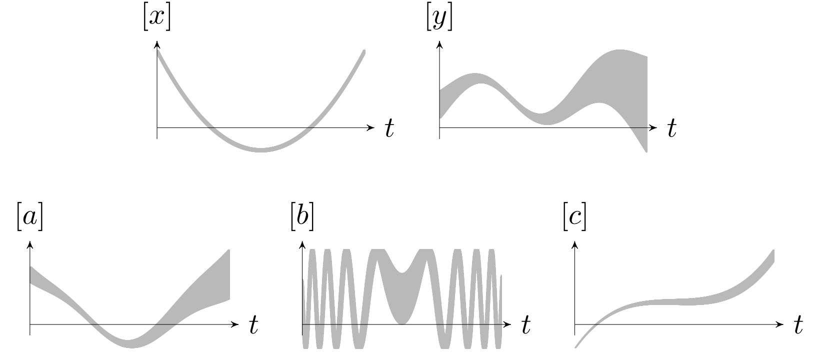
Constraint programming

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

$$\text{SLAM}: \begin{cases} \textbf{Variables:} \\ \textbf{Constraints:} \\ \\ \\ \\ \\ \textbf{Domains:} \end{cases}$$

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

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 constraints:
     - $\mathbf{v}(\cdot) = \mathbf{f}\big(\mathbf{x}(\cdot)\big)$
- **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

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

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

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

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

SLAM :

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

Constraints:

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

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

Constraint programming

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 constraints:
   - $\mathbf{v}(\cdot) = \mathbf{f}\big(\mathbf{x}(\cdot)\big)$ $\longleftarrow$ algebraic constraint
   - $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$
2. Inter-temporal constraints:
   - $\mathbf{p}(\cdot) = \mathbf{h}\big(\mathbf{x}(\cdot)\big)$ $\longleftarrow$ 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 constraints:
   - $\mathbf{v}(\cdot) = \mathbf{f}\big(\mathbf{x}(\cdot)\big)$ $\longleftarrow$ 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 constraints:
   - $\mathbf{p}(\cdot) = \mathbf{h}\big(\mathbf{x}(\cdot)\big)$ $\longleftarrow$ 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

$$\text{SLAM} : \begin{cases} \textbf{Variables: } \mathbf{x}(\cdot),\ \mathbf{z}(\cdot),\ \mathbf{v}(\cdot),\ \mathbf{p}(\cdot) \\ \textbf{Constraints:} \\ \quad 1.\ \text{Evolution constraints:} \\ \qquad \blacktriangleright\ \mathbf{v}(\cdot) = \mathbf{f}\big(\mathbf{x}(\cdot)\big) \qquad \longleftarrow \text{algebraic constraint} \\ \qquad \blacktriangleright\ \mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot), \mathbf{v}(\cdot)\big) \text{: } \dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot) \\ \quad 2.\ \text{Inter-temporal constraints:} \\ \qquad \blacktriangleright\ \mathbf{p}(\cdot) = \mathbf{h}\big(\mathbf{x}(\cdot)\big) \qquad \longleftarrow \text{algebraic constraint} \\ \qquad \blacktriangleright\ \mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}\big(\mathbf{p}(\cdot), \mathbf{z}(\cdot)\big) \text{: } \mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2) \\ \textbf{Domains: } [\mathbf{x}](\cdot),\ [\mathbf{z}](\cdot),\ [\mathbf{v}](\cdot),\ [\mathbf{p}](\cdot) \end{cases}$$

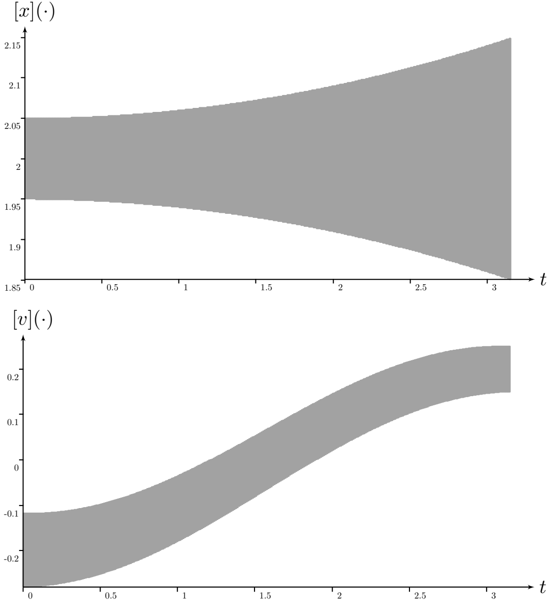
Constraint programming
$$\mathcal{L}_{\frac{d}{dt}}\Big(\mathbf{x}(\cdot),\mathbf{v}(\cdot)\Big)$$

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

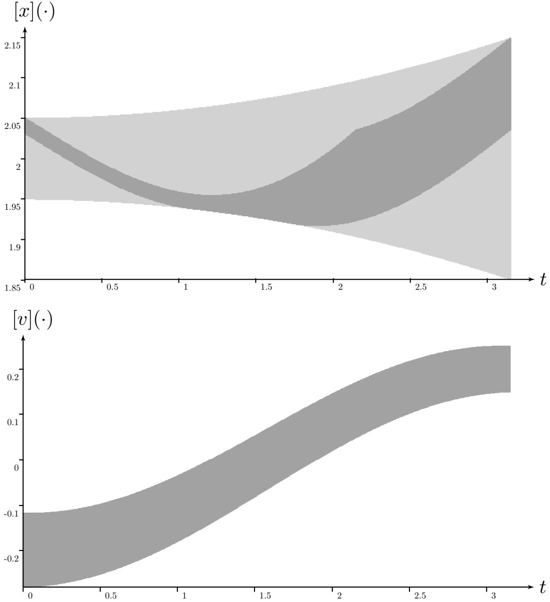
$$\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot),\mathbf{v}(\cdot)\big)$$

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

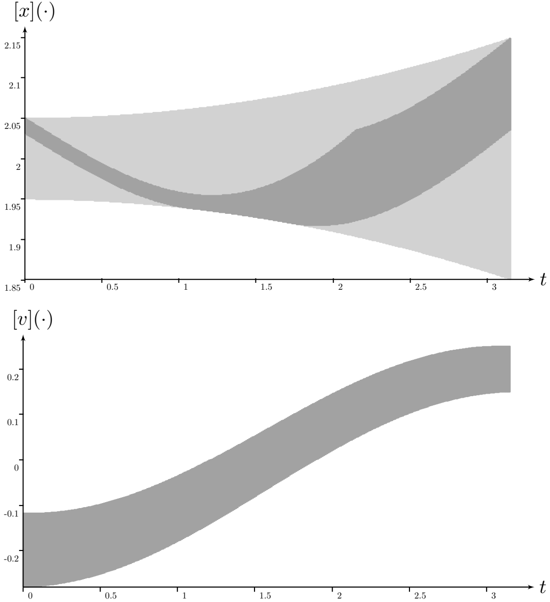
Related contractor $\mathcal{C}_{\frac{d}{dt}}$:

- 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

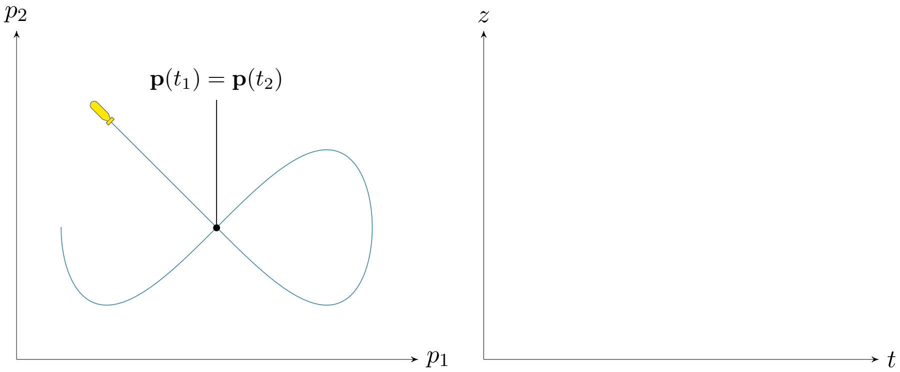
Constraint $\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$:

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

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

$\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: physical interpretation

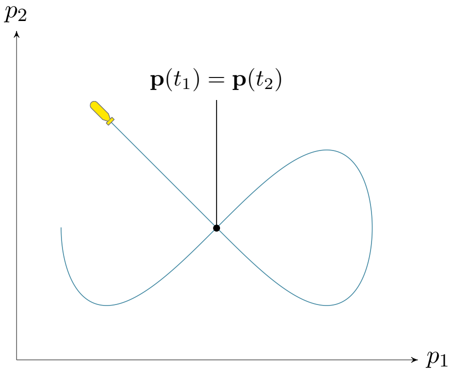
A robot coming back to a previous position $\mathbf{p}$ should sense the same observation $\mathbf{z}$.



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

$\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: physical interpretation

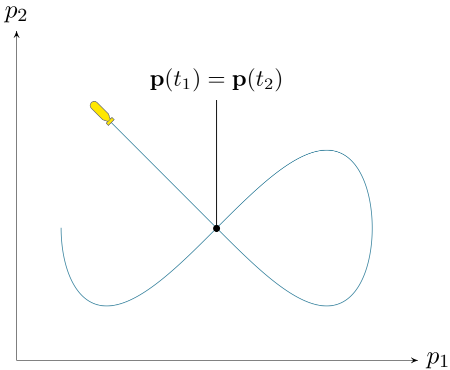
A robot coming back to a previous position $\mathbf{p}$ should sense the same observation $\mathbf{z}$.

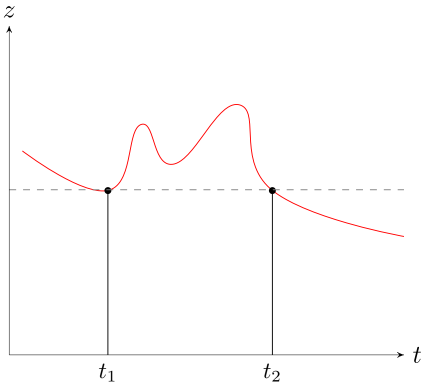


Constraint $\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: physical interpretation

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

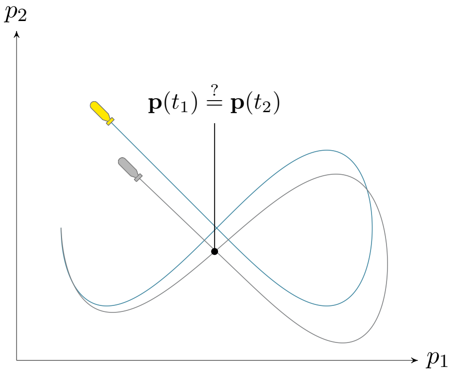


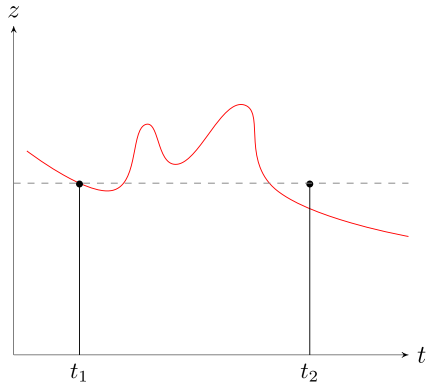


Constraint $\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: physical interpretation

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

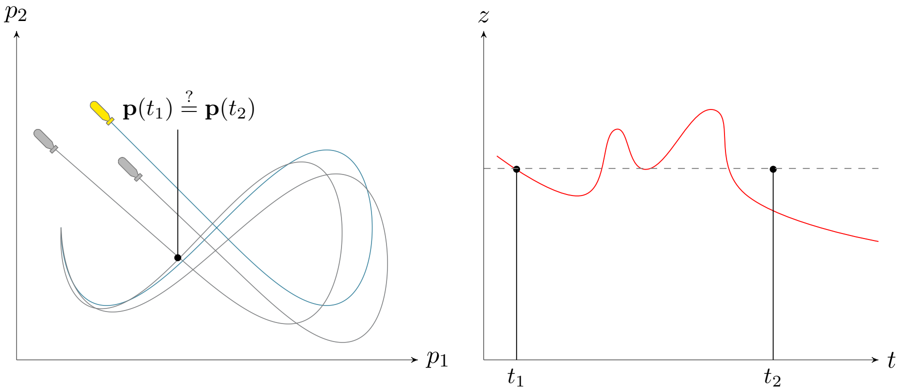




Constraint $\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

$\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: physical interpretation

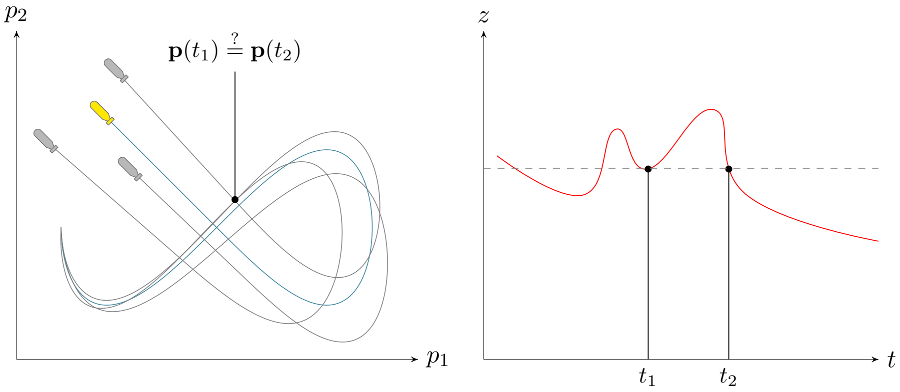
A robot coming back to a previous position $\mathbf{p}$ should sense the same observation $\mathbf{z}$.



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

$\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: physical interpretation

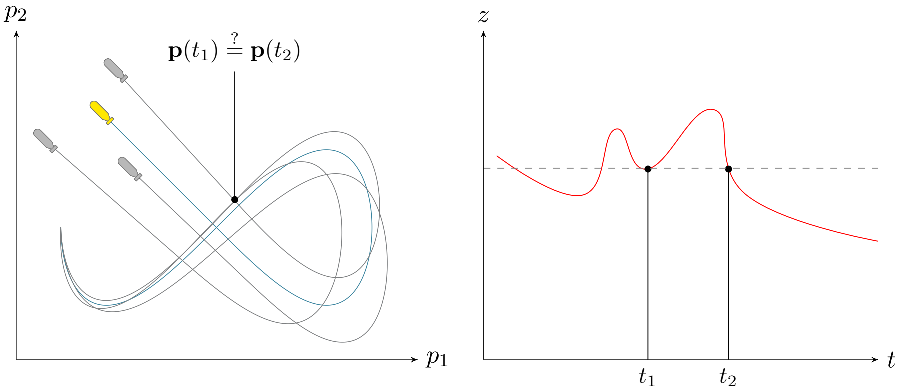
A robot coming back to a previous position $\mathbf{p}$ should sense the same observation $\mathbf{z}$.



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

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

Temporal decomposition

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

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

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: $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2) \, , \, t_1 < t_2\}$

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

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: $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\ ,\ t_1 < t_2\}$
- the effect ②: $\mathbb{T}^*_{\mathbf{z}} = \{(t_1, t_2) \mid \mathbf{z}(t_1) = \mathbf{z}(t_2)\ ,\ t_1 < t_2\}$

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

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: $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\ ,\ t_1 < t_2\}$
- the effect ②: $\mathbb{T}^*_{\mathbf{z}} = \{(t_1, t_2) \mid \mathbf{z}(t_1) = \mathbf{z}(t_2)\ ,\ t_1 < t_2\}$

From the implication ① $\implies$ ②:

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

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

$\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: constraint network

$\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$:

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

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

$\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: physical interpretation

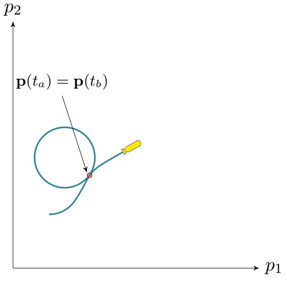
Variables: $\mathbf{p}(\cdot)$, $\mathbf{z}(\cdot)$

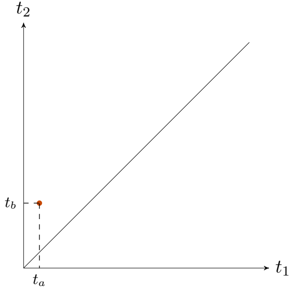
Internal variables: $\mathbb{T}^*_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}}$

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

$\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: physical interpretation

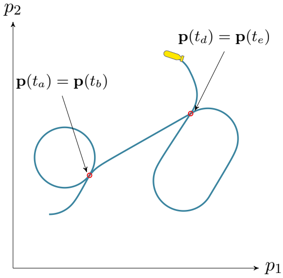
Variables: $\mathbf{p}(\cdot)$, $\mathbf{z}(\cdot)$

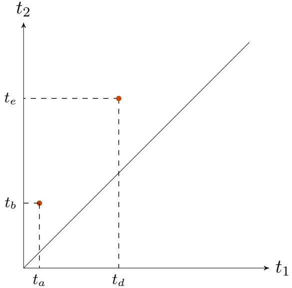
Internal variables: $\mathbb{T}^*_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}}$

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

$\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: physical interpretation

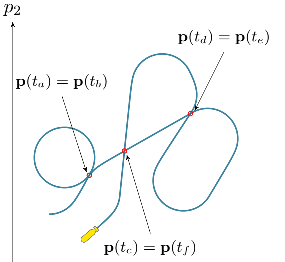
Variables: $\mathbf{p}(\cdot)$, $\mathbf{z}(\cdot)$

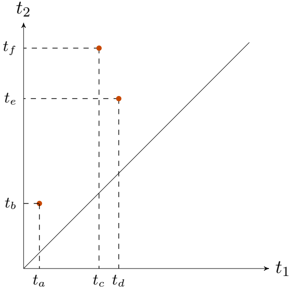
Internal variables: $\mathbb{T}^*_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}}$

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

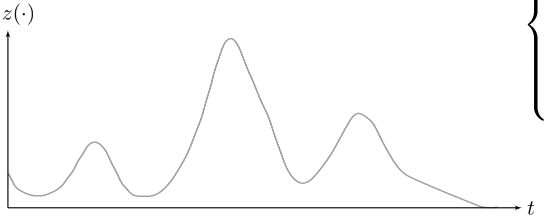




$\mathbb{T}^*_{\mathbf{p}}$

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

$\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: physical interpretation



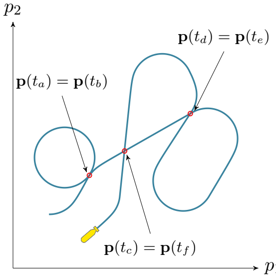
Variables: $\mathbf{p}(\cdot)$, $\mathbf{z}(\cdot)$

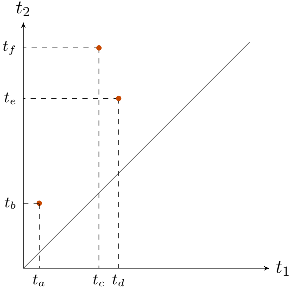
Internal variables: $\mathbb{T}^*_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}}$

Constraints:

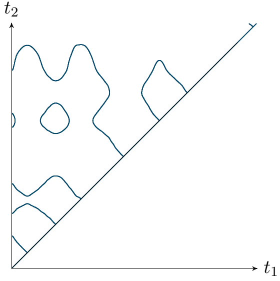
1. $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\}$
2. $\mathbb{T}^*_{\mathbf{z}} = \{(t_1, t_2) \mid \mathbf{z}(t_1) = \mathbf{z}(t_2)\}$
3. $\mathbb{T}^*_{\mathbf{p}} \subset \mathbb{T}^*_{\mathbf{z}}$

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





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



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

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

$\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: physical interpretation

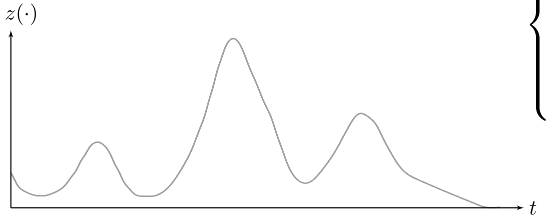
Variables: $\mathbf{p}(\cdot)$, $\mathbf{z}(\cdot)$

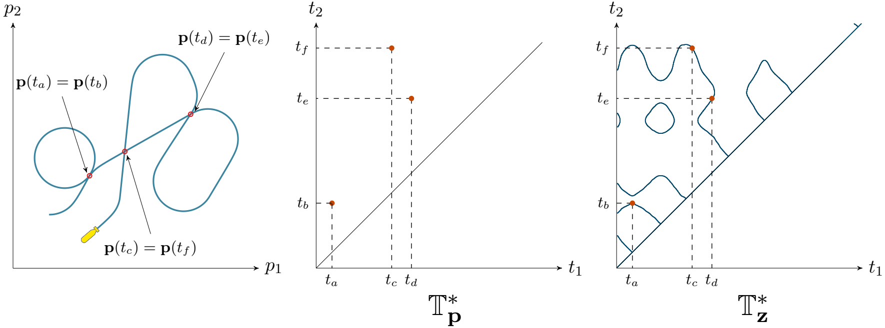
Internal variables: $\mathbb{T}^*_{\mathbf{p}}$, $\mathbb{T}^*_{\mathbf{z}}$

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

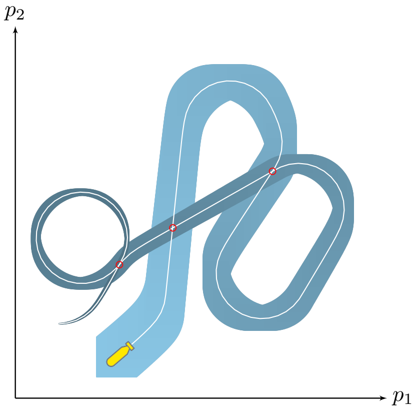




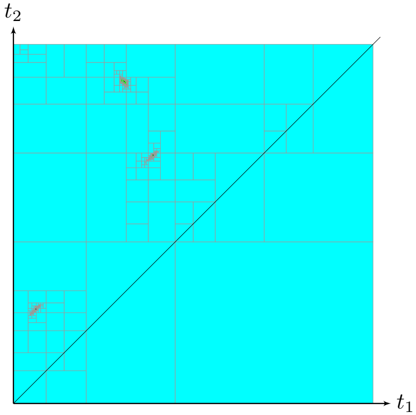
Constraint $\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

Bounded-error context

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



(a) Bounded trajectories

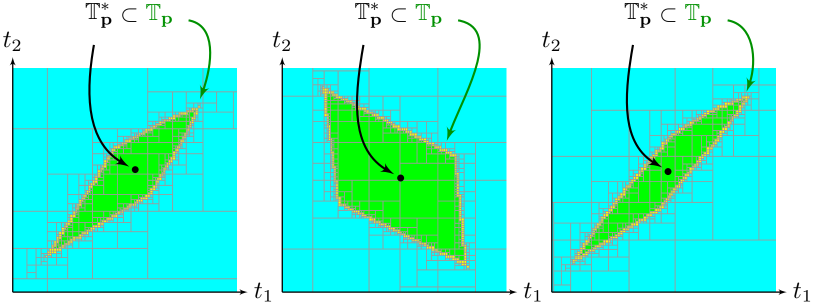


(b) Approximation of $\mathbb{T}_{\mathbf{p}}$

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

Bounded-error context

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

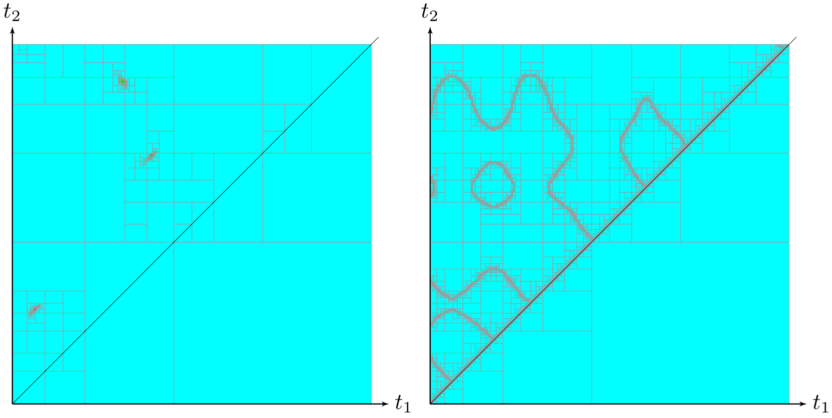


Zoom on the components of $\mathbb{T}_{\mathbf{p}}$

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

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

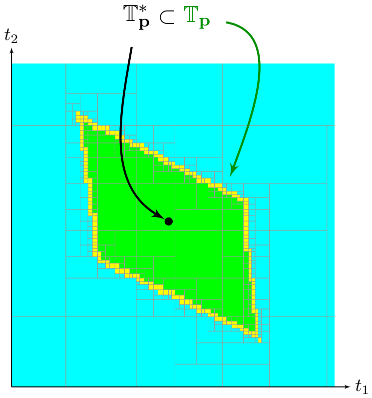
The $\mathcal{C}_{\mathbf{p}\Rightarrow\mathbf{z}}$ contractor: t-sets fusion

Constraint:

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

Domains $\mathbb{T}_{\mathbf{p}}$, $\mathbb{T}_{\mathbf{z}}$:

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



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

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

The $\mathcal{C}_{\mathbf{p}\Rightarrow\mathbf{z}}$ contractor: t-sets fusion

Constraint:

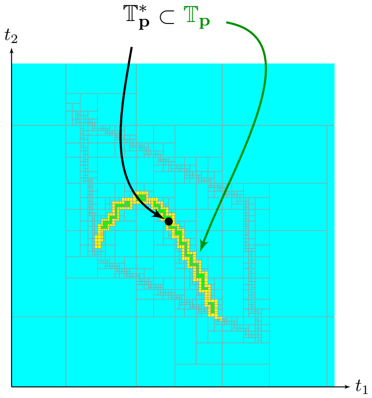
- $\mathbb{T}^*_{\mathbf{p}} \subset \mathbb{T}^*_{\mathbf{z}}$

Domains $\mathbb{T}_{\mathbf{p}}$, $\mathbb{T}_{\mathbf{z}}$:

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

Contraction:

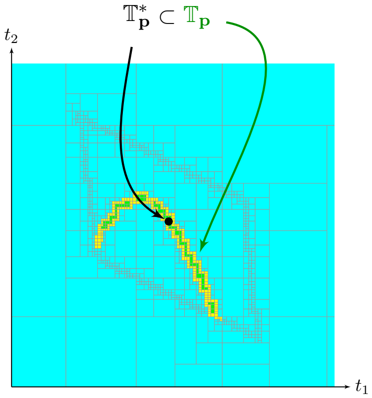
- $\mathbb{T}_{\mathbf{p}} := \mathbb{T}_{\mathbf{p}} \cap \mathbb{T}_{\mathbf{z}}$



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

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

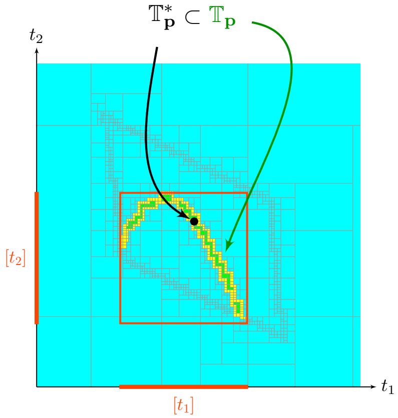
Constraint $\mathbb{T}^*_{\mathbf{p}} = \{(t_1, t_2) \mid \mathbf{p}(t_1) = \mathbf{p}(t_2)\}$ in backward



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

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

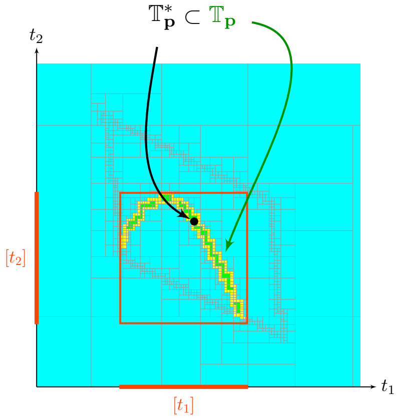
- **time uncertainties:** $[t_1]$, $[t_2]$



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

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

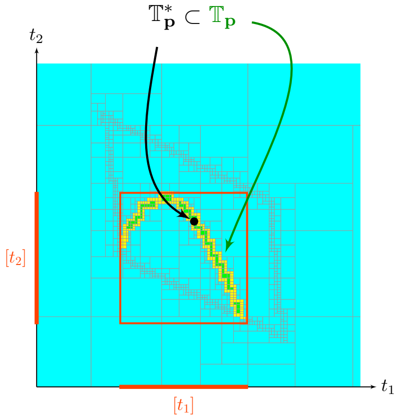
- **time uncertainties:** $[t_1]$, $[t_2]$
- constraint $\mathcal{L}_{t_1,t_2}\big(t_1, t_2, \mathbf{p}(\cdot)\big) : \mathbf{p}(t_1) = \mathbf{p}(t_2)$



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

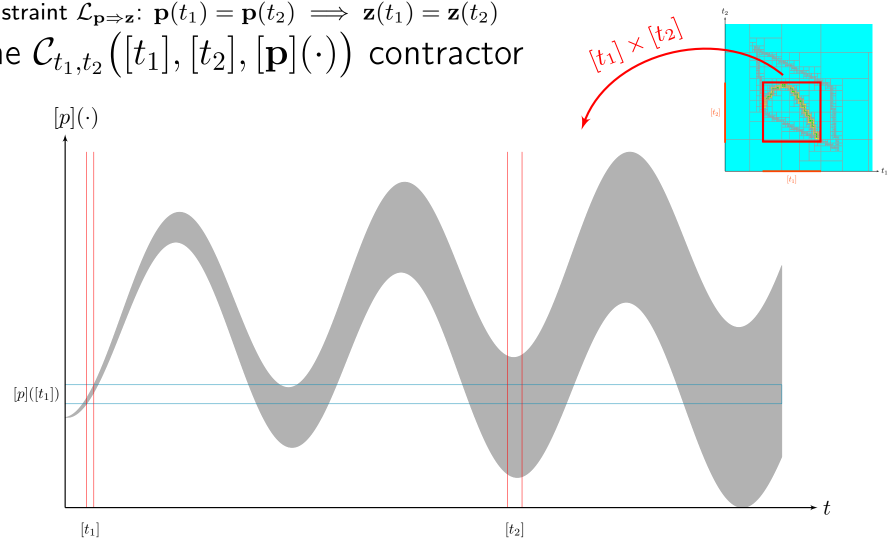
Constraint $\mathbb{T}^*_{\mathbf{p}} = \{(t_1,t_2) \mid \mathbf{p}(t_1)=\mathbf{p}(t_2)\}$ in backward

- **time uncertainties:** $[t_1]$, $[t_2]$
- constraint $\mathcal{L}_{t_1,t_2}\big(t_1,t_2,\mathbf{p}(\cdot)\big) : \mathbf{p}(t_1)=\mathbf{p}(t_2)$
- strong contribution of this work:
  - 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, 2018



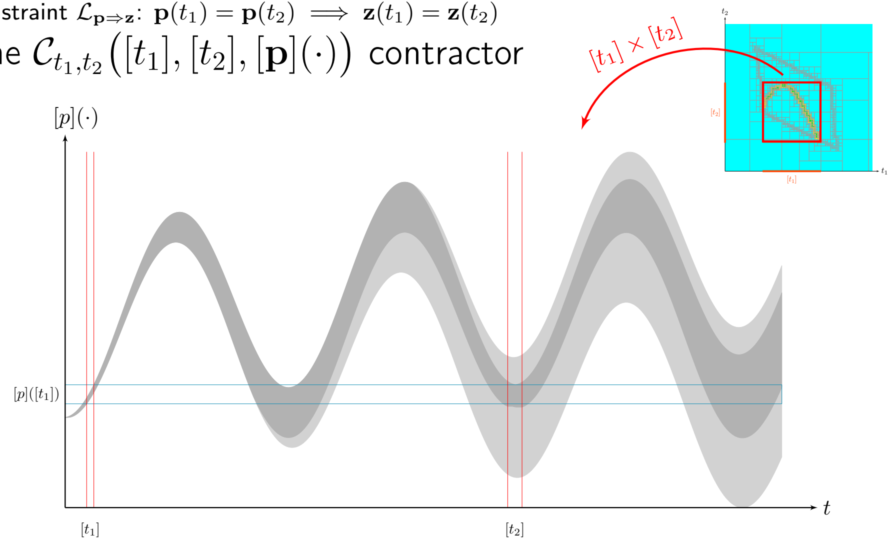
Constraint $\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: $\mathbf{p}(t_1) = \mathbf{p}(t_2) \implies \mathbf{z}(t_1) = \mathbf{z}(t_2)$

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



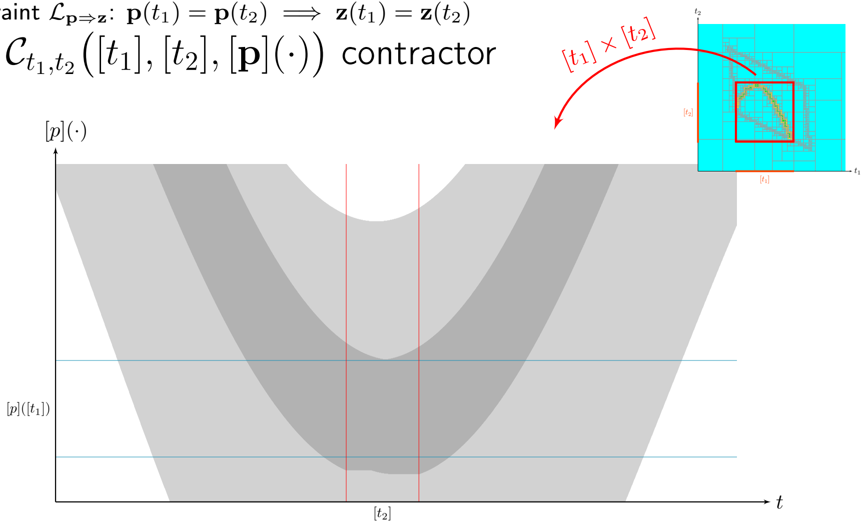
Constraint $\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$: $\mathbf{p}(t_1)=\mathbf{p}(t_2) \implies \mathbf{z}(t_1)=\mathbf{z}(t_2)$

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



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

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

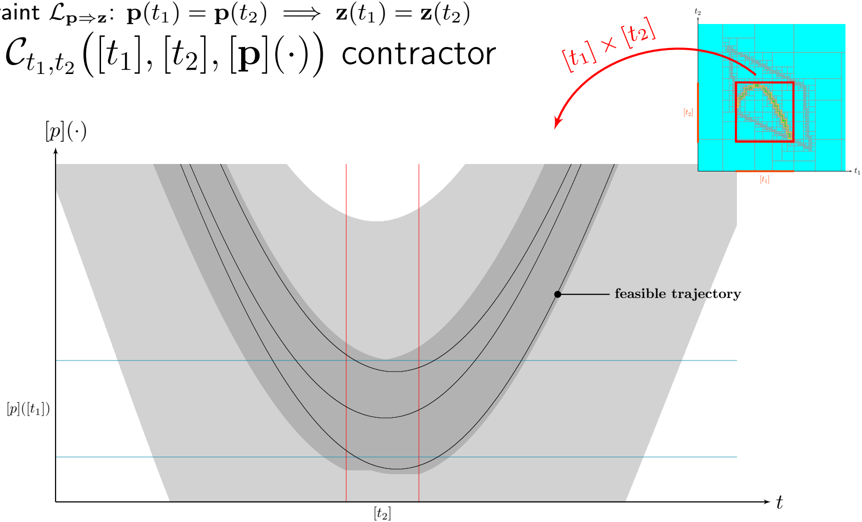


Importance of the **derivative** $\mathbf{w}(\cdot)$

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

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

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



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

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

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

Summary

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

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

Summary

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

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

SLAM :

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

Constraints:

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

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

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:

- Evolution constraints:
  1: $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot))$
  2: $\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot), \mathbf{v}(\cdot)\big)$
- Inter-temporal constraints:
  3: $\mathbf{p}(\cdot) = \mathbf{h}(\mathbf{x}(\cdot))$
  4: $\mathbf{w}(\cdot) = \mathbf{h}(\mathbf{v}(\cdot))$
  5: $\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}\big(\mathbf{p}(\cdot), \mathbf{w}(\cdot), \mathbf{z}(\cdot)\big)$

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:

- Evolution constraints:
  1: $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot))$
  2: $\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot), \mathbf{v}(\cdot)\big)$
- Inter-temporal constraints:
  3: $\mathbf{p}(\cdot) = \mathbf{h}(\mathbf{x}(\cdot))$
  4: $\mathbf{w}(\cdot) = \mathbf{h}(\mathbf{v}(\cdot))$
  5: $\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}\big(\mathbf{p}(\cdot), \mathbf{w}(\cdot), \mathbf{z}(\cdot)\big)$

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

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:

- Evolution constraints:
  1: $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot))$
  2: $\mathcal{L}_{\frac{d}{dt}}\big(\mathbf{x}(\cdot), \mathbf{v}(\cdot)\big)$
- Inter-temporal constraints:
  3: $\mathbf{p}(\cdot) = \mathbf{h}(\mathbf{x}(\cdot))$
  4: $\mathbf{w}(\cdot) = \mathbf{h}(\mathbf{v}(\cdot))$
  5: $\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}\big(\mathbf{p}(\cdot), \mathbf{w}(\cdot), \mathbf{z}(\cdot)\big)$

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

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

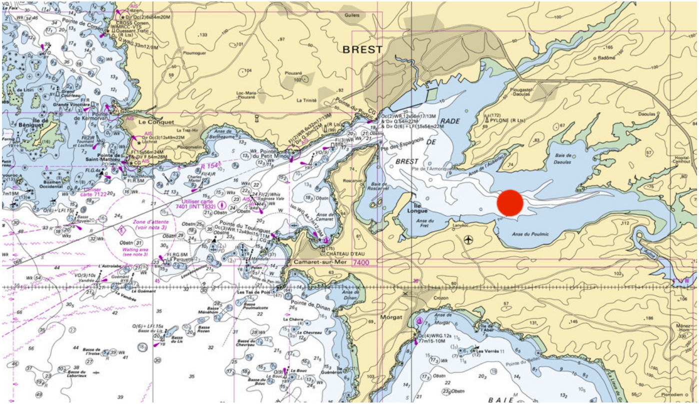
- 2 hours experimental mission
- in the *Rade de Brest*, Brittany

Location: *Polygone de Rascas* – Credits: SHOM

Bathymetric SLAM

Experimental mission with the Daurade AUV

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

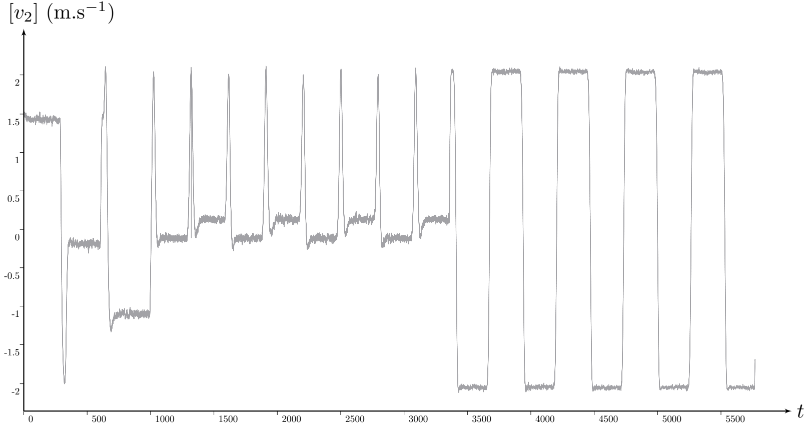


Location: *Polygone de Rascas* – Credits: SHOM

Bathymetric SLAM

Evolution measurements

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

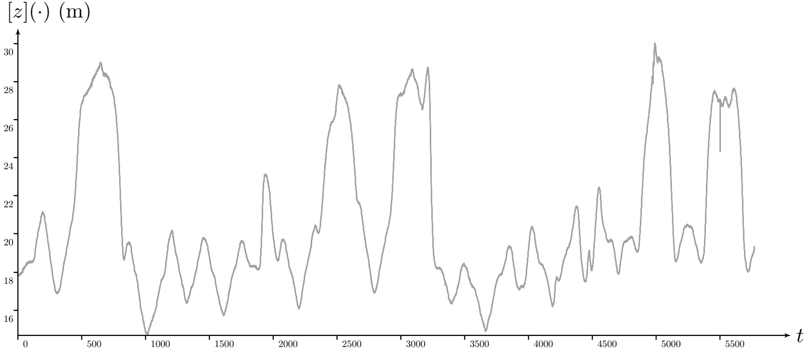


North speed velocity tube $[v_2](\cdot)$

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:

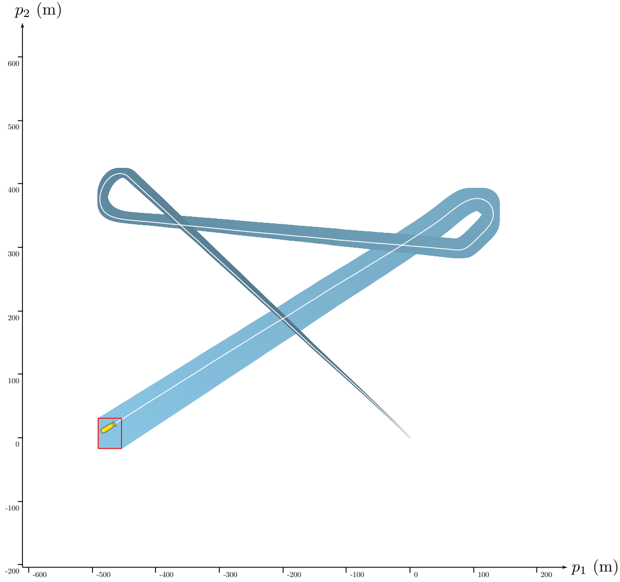
- white

Tube of positions:

- blue

Last position box:

- red



Bathymetric SLAM

Dead-reckoning

Actual trajectory:

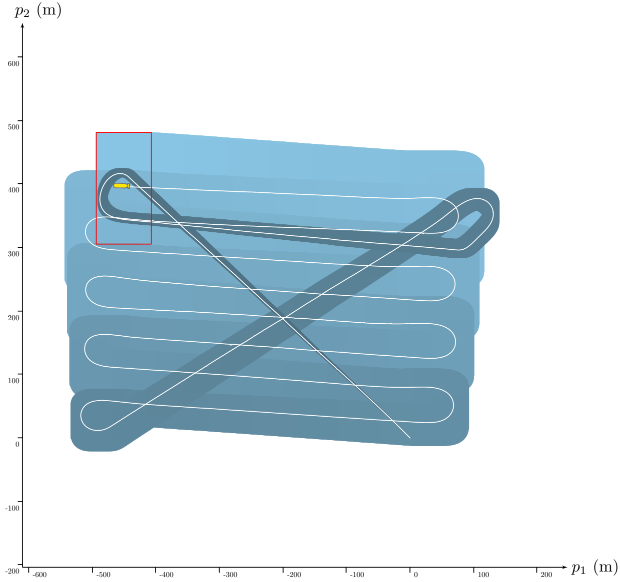
- white

Tube of positions:

- blue

Last position box:

- red



Bathymetric SLAM

Dead-reckoning

Actual trajectory:

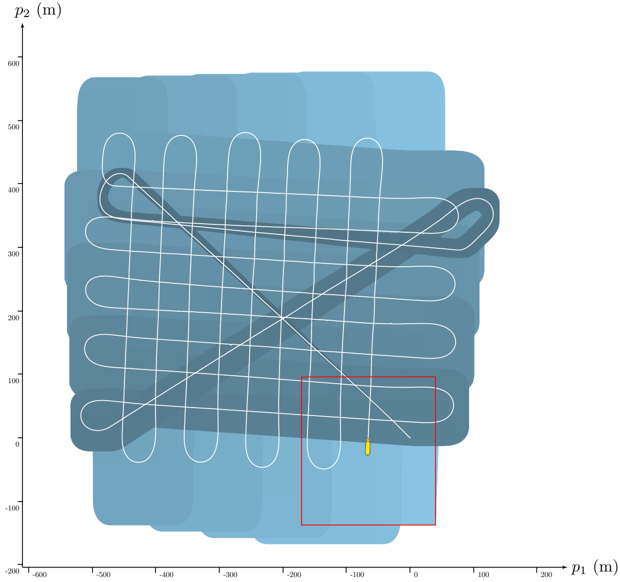
- white

Tube of positions:

- blue

Last position box:

- red



Bathymetric SLAM

SLAM results

Actual trajectory:

- white

Tube of positions:

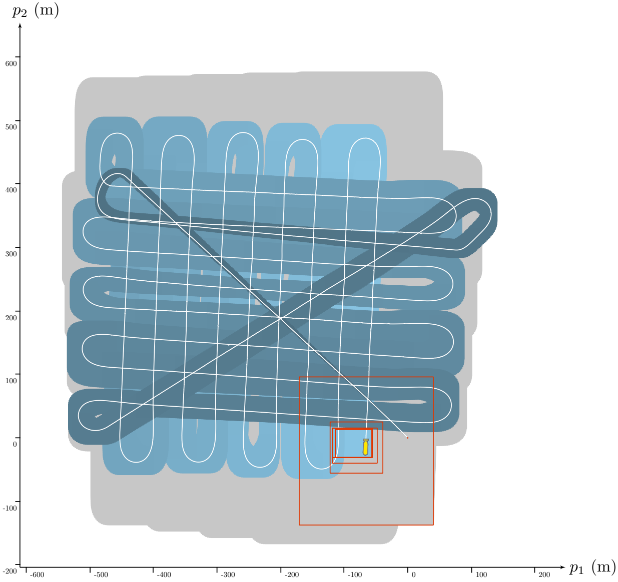
- blue

Last position box:

- red

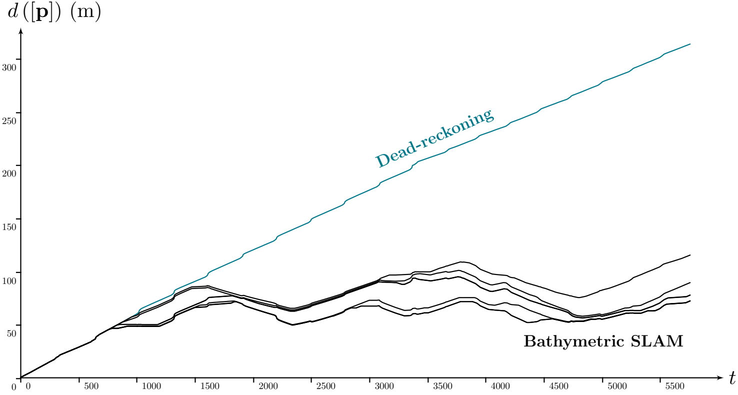
Contracted parts:

- 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** $\mathbf{g}$
  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** 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**
  $\mathcal{L}_{\frac{d}{dt}}$, $\mathcal{L}_{t_1,t_2}$, $\mathcal{L}_{\mathbf{p}\Rightarrow\mathbf{z}}$, ...

A temporal approach for the SLAM problem

— 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*, 2018

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