

Set-membership Terrain Based Navigation

Simon Rohou

ENSTA Bretagne, Lab-STICC, UMR CNRS 6285, Brest, France

GdR - GT Rob Marine / sous-marine
28th June 2019



Section 1

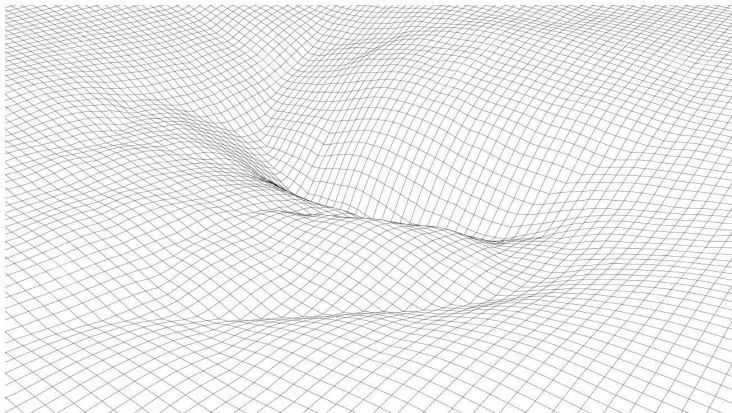
Motivations

Motivations

Problem: wide environment and poor observations

Under the surface:

- ▶ **no satellite** navigation systems available
- ▶ **no seamarks** or points of interest

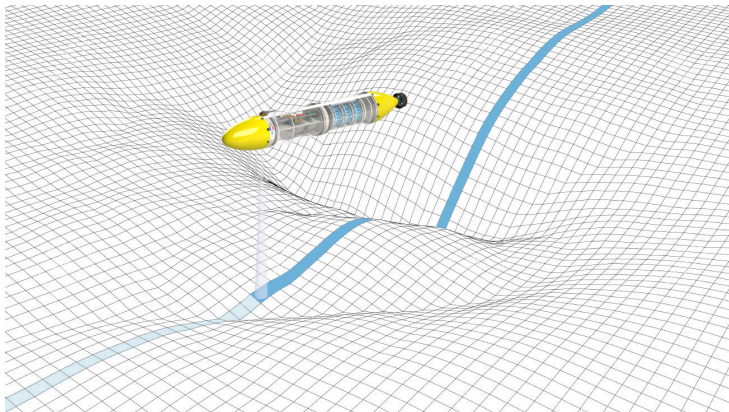


Motivations

Problem: wide environment and poor observations

Available data:

- ▶ **bathymetric** measurements (scalar values)
- ▶ prior knowledge of the environment (embedded map)

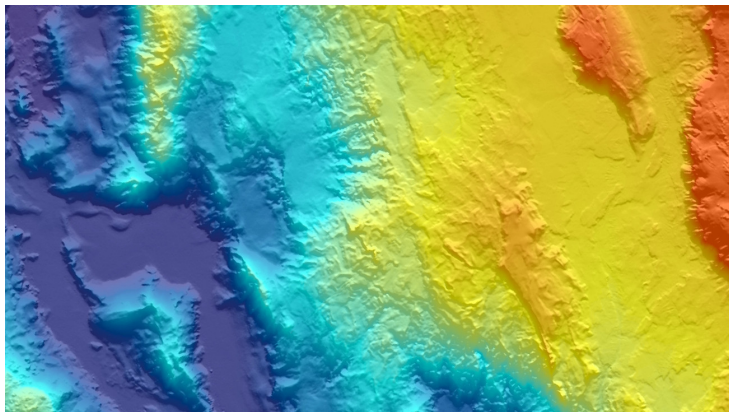


Motivations

Problem: wide environment and poor observations

Available data:

- ▶ **bathymetric** measurements (scalar values)
- ▶ prior knowledge of the environment (embedded map)



Motivations

Available Digital Elevation Models (DEM)

Portail data.shom.fr

- ▶ Shom: Service Hydrographique et Océanographique de la Marine
- ▶ many marine DEM available

SHOM L'océan en référence DATA.SHOM.FR Information géographique maritime et littorale de référence Contact us | Help | FR | EN

Shom catalog

Master data	Oceanographic forecast	Coastal observations
<input checked="" type="checkbox"/> MNT de façade Atlantique		
<input type="checkbox"/> MNT de façade de Clipperton		
<input type="checkbox"/> MNT de façade de Mayotte		
<input type="checkbox"/> MNT de façade de Saint-Martin et Saint-Barthélemy		
<input type="checkbox"/> MNT de façade de la Corse		
<input type="checkbox"/> MNT de façade de la Guadeloupe et de la Martinique		
<input type="checkbox"/> MNT de façade de la Guyane		
<input type="checkbox"/> MNT de façade de la Réunion		
<input type="checkbox"/> MNT de façade du Golfe du Lion - Côte d'Azur		

Vertical Datums >

Sedimentology >

WGS84: 48° 30' 40.13" N, 5° 05' 49.09" W Legal notice and warnings Scale 1 : 433 344

Section 2

Set-membership TBN

Set-membership TBN

Terrain Based Navigation: formalization

Robot localization = state estimation problem.

Classically, we have:

$$\left\{ \begin{array}{l} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \end{array} \right. \quad (\text{navigation})$$

With:

- ▶ \mathbf{x} : state vector (position, bearing, ...)
- ▶ \mathbf{u} : input vector (command)
- ▶ \mathbf{f} : *evolution* function

Set-membership TBN

Terrain Based Navigation: formalization

Robot localization = state estimation problem.

Classically, we have:

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

With:

- ▶ \mathbf{x} : state vector (position, bearing, ...)
- ▶ \mathbf{u} : input vector (command)
- ▶ \mathbf{f} : *evolution* function
- ▶ g : *observation* function
- ▶ z_i : scalar measurements (at t_i)
- ▶ \mathbb{M} : terrain knowledge

Set-membership TBN

Terrain Based Navigation: formalization

Robot localization = state estimation problem.

Classically, we have:

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

With:

- ▶ \mathbf{x} : state vector (position, bearing, ...)
- ▶ \mathbf{u} : input vector (command)
- ▶ \mathbf{f} : *evolution* function
- ▶ g : *observation* function
- ▶ z_i : scalar measurements (at t_i) (ex: scalar bathymetric values)
- ▶ \mathbb{M} : terrain knowledge (ex: digital elevation map)

Set-membership TBN

Terrain Based Navigation: formalization

Robot localization = state estimation problem.

Classically, we have:

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) & \text{(navigation)} \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) & \text{(measurements)} \end{cases}$$

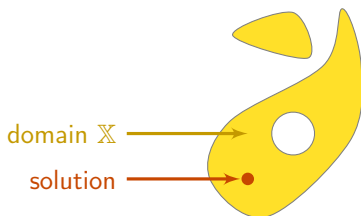
With:

- ▶ \mathbf{x} : state vector (position, bearing, ...)
- ▶ \mathbf{u} : input vector (command)
- ▶ \mathbf{f} : *evolution* function
- ▶ g : *observation* function
- ▶ z_i : scalar measurements (at t_i) (ex: scalar bathymetric values)
- ▶ \mathbb{M} : terrain knowledge (ex: digital elevation map)

Set-membership TBN

Constraint programming: overall concept

- ▶ system described by a *constraint network*
- ▶ **variables** belonging to **domains** \mathbb{X}



Constraint network:

Variables: x **Constraints:****Domains:** \mathbb{X}

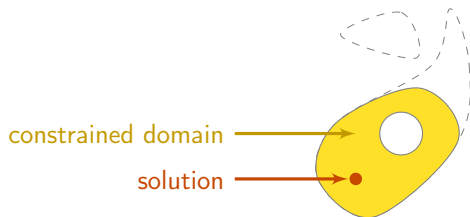
■ Contractor Programming

Chabert, Jaulin *Artificial Intelligence*, 2009

Set-membership TBN

Constraint programming: overall concept

- ▶ system described by a *constraint network*
- ▶ **variables** 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}

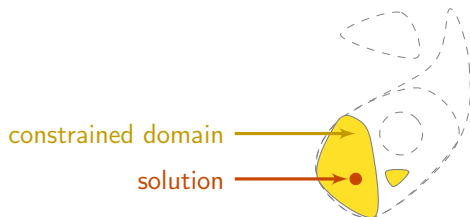
■ Contractor Programming

Chabert, Jaulin *Artificial Intelligence*, 2009

Set-membership TBN

Constraint programming: overall concept

- ▶ system described by a *constraint network*
- ▶ **variables** belonging to **domains** \mathbb{X}
- ▶ continuous **constraints** \mathcal{L} : non-linear equations, inequalities, ...



Constraint network:

Variables: \mathbf{x}

Constraints:

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

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

Domains: \mathbb{X}

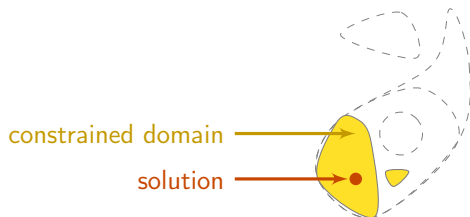
■ Contractor Programming

Chabert, Jaulin *Artificial Intelligence*, 2009

Set-membership TBN

Constraint programming: overall concept

- ▶ system described by a *constraint network*
- ▶ **variables** 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}

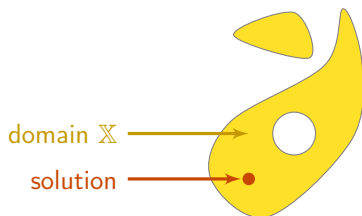
■ Contractor Programming

Chabert, Jaulin *Artificial Intelligence*, 2009

Set-membership TBN

Constraint programming: overall concept

- ▶ system described by a *constraint network*
- ▶ **variables** 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. $\mathcal{L}_2(\mathbf{x})$

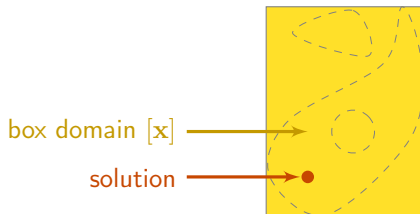
3. ...

Domains: \mathbb{X}

Set-membership TBN

Constraint programming: overall concept

- ▶ system described by a *constraint network*
- ▶ **variables** belonging to **domains** \mathbb{X}
- ▶ continuous **constraints** \mathcal{L} : non-linear equations, inequalities, ...
- ▶ representable domains: e.g. boxes $[\mathbf{x}]$



Constraint network:

Variables: \mathbf{x}

Constraints:

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

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

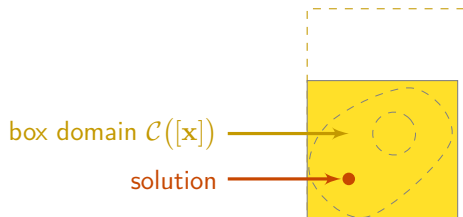
3. ...

Domains: $[\mathbf{x}]$

Set-membership TBN

Constraint programming: overall concept

- ▶ system described by a *constraint network*
- ▶ **variables** belonging to **domains** \mathbb{X}
- ▶ continuous **constraints** \mathcal{L} : non-linear equations, inequalities, ...
- ▶ representable domains: e.g. boxes $[\mathbf{x}]$
- ▶ resolution by **contractors**, $\mathcal{C}_{\mathcal{L}}([\mathbf{x}])$



Constraint network:

Variables: \mathbf{x} **Constraints:**1. $\mathcal{L}_1(\mathbf{x})$ 2. $\mathcal{L}_2(\mathbf{x})$

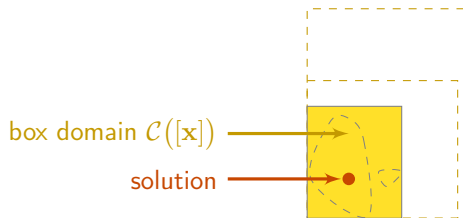
3. ...

Domains: $[\mathbf{x}]$

Set-membership TBN

Constraint programming: overall concept

- ▶ system described by a *constraint network*
- ▶ **variables** belonging to **domains** \mathbb{X}
- ▶ continuous **constraints** \mathcal{L} : non-linear equations, inequalities, ...
- ▶ representable domains: e.g. boxes $[\mathbf{x}]$
- ▶ resolution by **contractors**, $\mathcal{C}_{\mathcal{L}}([\mathbf{x}])$



Constraint network:

Variables: \mathbf{x} **Constraints:**1. $\mathcal{L}_1(\mathbf{x})$ 2. $\mathcal{L}_2(\mathbf{x})$

3. ...

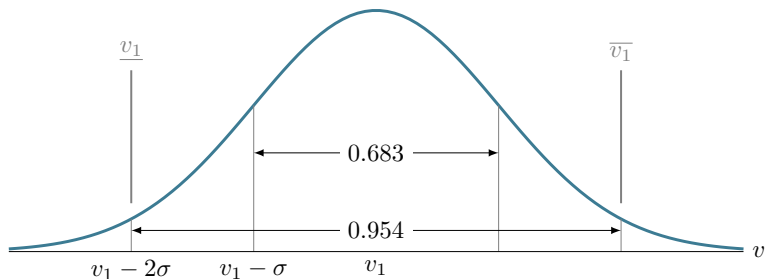
Domains: $[\mathbf{x}]$

Set-membership TBN

Domains from sensor data

Uncertainties:

- ▶ datasheets \implies standard deviation σ for each sensor
- ▶ 95% confidence rate: $v_1^* \in [v_1] = [v_1 - 2\sigma, v_1 + 2\sigma]$

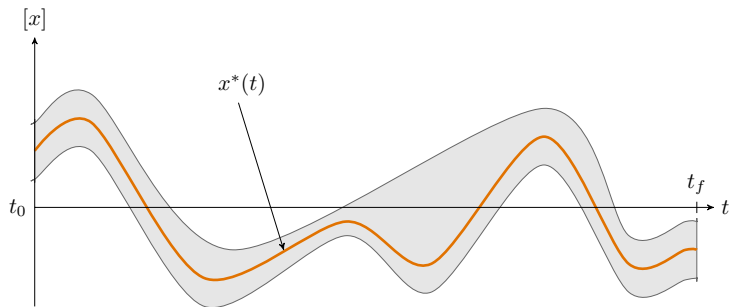


- ▶ uncertainties then reliably propagated in the system

Set-membership TBN

Tubes: domains for trajectories

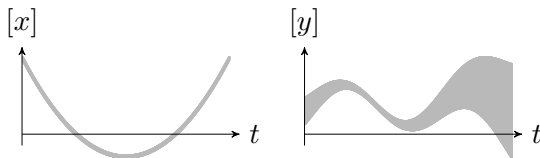
Tube $[x](\cdot)$: interval of trajectories $[x^-(\cdot), x^+(\cdot)]$
such that $\forall t \in \mathbb{R}, x^-(t) \leq x^+(t)$



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

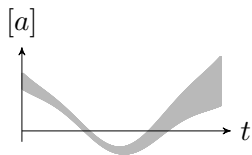
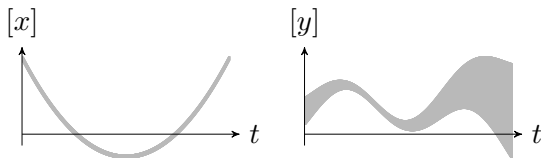
Set-membership TBN

Tubes arithmetic

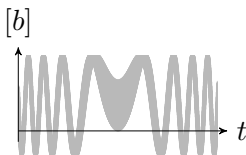


Set-membership TBN

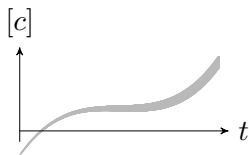
Tubes arithmetic



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



$$[b](\cdot) = \sin([x](\cdot))$$



$$[c](\cdot) = \int_0^{\cdot} [x](\tau) d\tau$$

Set-membership TBN

Constraint programming for TBN

Using intervals and tubes to enclose our solutions:

Variables:

- ▶ reals: z_i
 - ▶ trajectories: $\mathbf{x}(\cdot)$
-

Set-membership TBN

Constraint programming for TBN

Using intervals and tubes to enclose our solutions:

Variables:

- ▶ reals: z_i
- ▶ trajectories: $\mathbf{x}(\cdot)$

Numerical domains of the variables:

- ▶ intervals: $[z_i]$
- ▶ tubes: $[\mathbf{x}](\cdot)$

Set-membership TBN

Constraint programming for TBN

Using contractors to reduce the domains of the variables:

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Set-membership TBN

Constraint programming for TBN

Using contractors to reduce the domains of the variables:

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Elementary constraints decomposition:

► $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot))$

Set-membership TBN

Constraint programming for TBN

Using contractors to reduce the domains of the variables:

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Elementary constraints decomposition:

- ▶ $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot))$
- ▶ $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$

Set-membership TBN

Constraint programming for TBN

Using contractors to reduce the domains of the variables:

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Elementary constraints decomposition:

- ▶ $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot))$
- ▶ $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$
- ▶ $\mathbf{p}_i = \mathbf{x}(t_i)$

Set-membership TBN

Constraint programming for TBN

Using contractors to reduce the domains of the variables:

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Elementary constraints decomposition:

- ▶ $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot))$
- ▶ $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$
- ▶ $\mathbf{p}_i = \mathbf{x}(t_i)$
- ▶ $z_i = \mathbb{M}(\mathbf{p}_i)$

Set-membership TBN

Constraint programming for TBN

Using contractors to reduce the domains of the variables:

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Elementary constraints decomposition:

- ▶ $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \rightarrow$ algebraic constraint
- ▶ $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$
- ▶ $\mathbf{p}_i = \mathbf{x}(t_i)$
- ▶ $z_i = \mathbb{M}(\mathbf{p}_i)$

Set-membership TBN

Constraint programming for TBN

Using contractors to reduce the domains of the variables:

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Elementary constraints decomposition:

- ▶ $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \rightarrow$ algebraic constraint
- ▶ $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot) \rightarrow$ derivative constraint
- ▶ $\mathbf{p}_i = \mathbf{x}(t_i)$
- ▶ $z_i = \mathbb{M}(\mathbf{p}_i)$

Set-membership TBN

Constraint programming for TBN

Using contractors to reduce the domains of the variables:

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Elementary constraints decomposition:

- ▶ $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \rightarrow$ algebraic constraint
- ▶ $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot) \rightarrow$ derivative constraint $\rightarrow \mathcal{L}_{\frac{d}{dt}}(\mathbf{x}(\cdot), \mathbf{v}(\cdot))$
- ▶ $\mathbf{p}_i = \mathbf{x}(t_i)$
- ▶ $z_i = \mathbb{M}(\mathbf{p}_i)$

Set-membership TBN

Constraint programming for TBN

Using contractors to reduce the domains of the variables:

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Elementary constraints decomposition:

- ▶ $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \rightarrow$ algebraic constraint
- ▶ $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot) \rightarrow$ derivative constraint $\rightarrow \mathcal{L}_{\frac{d}{dt}}(\mathbf{x}(\cdot), \mathbf{v}(\cdot))$
- ▶ $\mathbf{p}_i = \mathbf{x}(t_i) \rightarrow$ evaluation constraint
- ▶ $z_i = \mathbb{M}(\mathbf{p}_i)$

Set-membership TBN

Constraint programming for TBN

Using contractors to reduce the domains of the variables:

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Elementary constraints decomposition:

- ▶ $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \rightarrow$ algebraic constraint
- ▶ $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot) \rightarrow$ derivative constraint $\rightarrow \mathcal{L}_{\frac{d}{dt}}(\mathbf{x}(\cdot), \mathbf{v}(\cdot))$
- ▶ $\mathbf{p}_i = \mathbf{x}(t_i) \rightarrow$ evaluation constraint $\rightarrow \mathcal{L}_{\text{eval}}(t_i, \mathbf{p}_i, \mathbf{x}(\cdot))$
- ▶ $z_i = \mathbb{M}(\mathbf{p}_i)$

Set-membership TBN

Constraint programming for TBN

Using contractors to reduce the domains of the variables:

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Elementary constraints decomposition:

- ▶ $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \rightarrow$ algebraic constraint
- ▶ $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot) \rightarrow$ derivative constraint $\rightarrow \mathcal{L}_{\frac{d}{dt}}(\mathbf{x}(\cdot), \mathbf{v}(\cdot))$
- ▶ $\mathbf{p}_i = \mathbf{x}(t_i) \rightarrow$ evaluation constraint $\rightarrow \mathcal{L}_{\text{eval}}(t_i, \mathbf{p}_i, \mathbf{x}(\cdot))$
- ▶ $z_i = \mathbb{M}(\mathbf{p}_i) \rightarrow$ map constraint $\mathcal{L}_{\mathbb{M}}$

Set-membership TBN

Constraint programming for TBN

Using contractors to reduce the domains of the variables:

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Elementary constraints decomposition:

- ▶ $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \rightarrow$ algebraic constraint
- ▶ $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot) \rightarrow$ derivative constraint $\rightarrow \mathcal{L}_{\frac{d}{dt}}(\mathbf{x}(\cdot), \mathbf{v}(\cdot))$
- ▶ $\mathbf{p}_i = \mathbf{x}(t_i) \rightarrow$ evaluation constraint $\rightarrow \mathcal{L}_{\text{eval}}(t_i, \mathbf{p}_i, \mathbf{x}(\cdot))$
- ▶ $z_i = \mathbb{M}(\mathbf{p}_i) \rightarrow$ map constraint $\mathcal{L}_{\mathbb{M}}$

Note: a solver could break down such problem automatically

Set-membership TBN

Algebraic constraints on trajectories

Example 1: consider the constraint $a(\cdot) = x(\cdot) + y(\cdot)$

A minimal **contractor** to apply this constraint is:

$$\begin{pmatrix} [a](\cdot) \\ [x](\cdot) \\ [y](\cdot) \end{pmatrix} \xrightarrow{\mathcal{C}_+} \begin{pmatrix} [a](\cdot) \cap ([x](\cdot) + [y](\cdot)) \\ [x](\cdot) \cap ([a](\cdot) - [y](\cdot)) \\ [y](\cdot) \cap ([a](\cdot) - [x](\cdot)) \end{pmatrix}$$

Contractor programming: $\mathcal{C}_+([a](\cdot), [x](\cdot), [y](\cdot))$

Set-membership TBN

Algebraic constraints on trajectories

Example 1: consider the constraint $a(\cdot) = x(\cdot) + y(\cdot)$

A minimal **contractor** to apply this constraint is:

$$\begin{pmatrix} [a](\cdot) \\ [x](\cdot) \\ [y](\cdot) \end{pmatrix} \xrightarrow{\mathcal{C}_+} \begin{pmatrix} [a](\cdot) \cap ([x](\cdot) + [y](\cdot)) \\ [x](\cdot) \cap ([a](\cdot) - [y](\cdot)) \\ [y](\cdot) \cap ([a](\cdot) - [x](\cdot)) \end{pmatrix}$$

Contractor programming: $\mathcal{C}_+([a](\cdot), [x](\cdot), [y](\cdot))$

Example 2: consider the constraint $c(\cdot) = \int_0^{\cdot} x(\tau) d\tau$

A non-minimal **contractor** to apply this constraint is:

$$\begin{pmatrix} [x](\cdot) \\ [c](\cdot) \end{pmatrix} \xrightarrow{\mathcal{C}_f} \begin{pmatrix} [x](\cdot) \\ [c](\cdot) \cap \int_0^{\cdot} [x](\tau) d\tau \end{pmatrix}$$

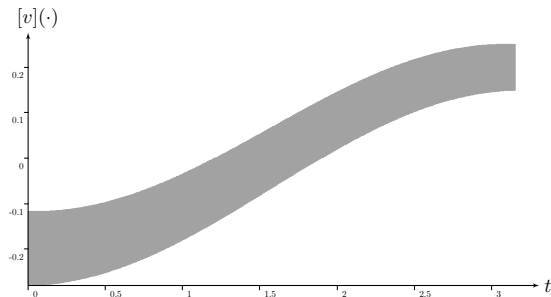
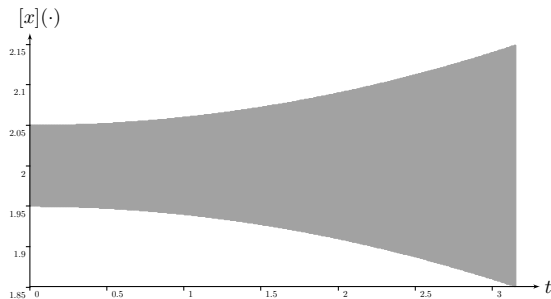
Contractor programming: $\mathcal{C}_f([x](\cdot), [c](\cdot))$

Set-membership TBN

Derivative constraint

Differential constraint:

- ▶ $\dot{x}(\cdot) = v(\cdot)$
- ▶ one trajectory and its derivative



Set-membership TBN

Derivative constraint

Differential constraint:

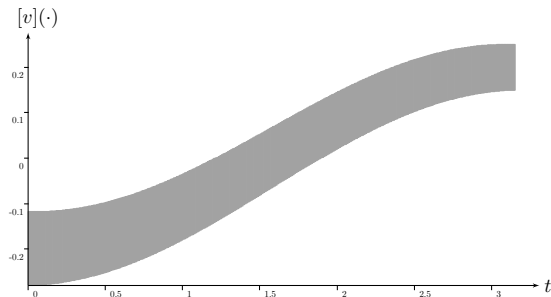
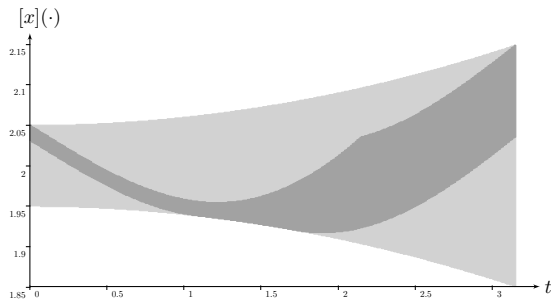
- ▶ $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot)$
- ▶ one trajectory and its derivative

Contractor programming:

- ▶ $\mathcal{C}_{\frac{d}{dt}}([\mathbf{x}(\cdot), [\mathbf{v}(\cdot))$

■ Guaranteed computation of robot trajectories

Rohou, Jaulin, Mihaylova, Le Bars, Veres
Robotics and Autonomous Systems, 2017



Set-membership TBN

Evaluation constraint

$$\text{Trajectory evaluation} \left\{ \begin{array}{l} \mathbf{z} = \mathbf{y}(t) \end{array} \right.$$

■ Reliable non-linear state estimation involving time uncertainties

Rohou, Jaulin, Mihaylova, Le Bars, Veres *Automatica*, 2018

Set-membership TBN

Evaluation constraint

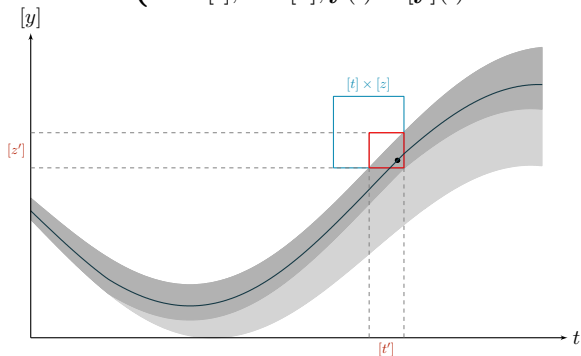
$$\text{Trajectory evaluation} \left\{ \begin{array}{l} \mathbf{z} = \mathbf{y}(t) \\ t \in [t], \mathbf{z} \in [\mathbf{z}], \mathbf{y}(\cdot) \in [\mathbf{y}](\cdot) \end{array} \right.$$

■ Reliable non-linear state estimation involving time uncertainties
Rohou, Jaulin, Mihaylova, Le Bars, Veres *Automatica*, 2018

Set-membership TBN

Evaluation constraint

$$\text{Trajectory evaluation} \begin{cases} \mathbf{z} = \mathbf{y}(t) \\ t \in [t], \mathbf{z} \in [\mathbf{z}], \mathbf{y}(\cdot) \in [\mathbf{y}](\cdot) \end{cases}$$



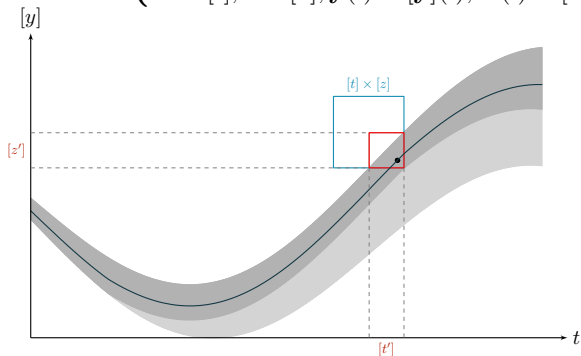
■ Reliable non-linear state estimation involving time uncertainties

Rohou, Jaulin, Mihaylova, Le Bars, Veres *Automatica*, 2018

Set-membership TBN

Evaluation constraint

$$\text{Trajectory evaluation} \begin{cases} \mathbf{z} = \mathbf{y}(t) \\ \dot{\mathbf{y}}(\cdot) = \mathbf{w}(\cdot) \\ t \in [t], \mathbf{z} \in [\mathbf{z}], \mathbf{y}(\cdot) \in [\mathbf{y}](\cdot), \mathbf{w}(\cdot) \in [\mathbf{w}](\cdot) \end{cases}$$



Contractor programming: $\mathcal{C}_{\text{eval}}([t], [\mathbf{z}], [\mathbf{y}](\cdot), [\mathbf{w}](\cdot))$

■ Reliable non-linear state estimation involving time uncertainties

Rohou, Jaulin, Mihaylova, Le Bars, Veres *Automatica*, 2018

Set-membership TBN

Constraint programming for TBN

Using constraints to reduce the domains of the variables:

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Elementary constraints decomposition:

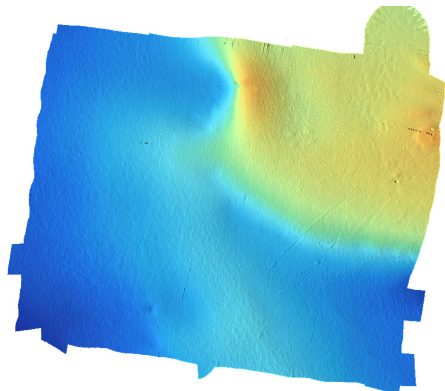
- ▶ $\mathbf{v}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \rightarrow$ algebraic constraint ✓
- ▶ $\dot{\mathbf{x}}(\cdot) = \mathbf{v}(\cdot) \rightarrow$ derivative constraint $\rightarrow \mathcal{L}_{\frac{d}{dt}}(\mathbf{x}(\cdot), \mathbf{v}(\cdot))$ ✓
- ▶ $\mathbf{p}_i = \mathbf{x}(t_i) \rightarrow$ evaluation constraint $\rightarrow \mathcal{L}_{\text{eval}}(t_i, \mathbf{p}_i, \mathbf{x}(\cdot))$ ✓
- ▶ $z_i = \mathbb{M}(\mathbf{p}_i) \rightarrow \mathcal{L}_{\mathbb{M}}$, related contractor $\mathcal{C}_{\mathbb{M}}$ to be defined

Set-membership TBN

The map constraint $\mathcal{L}_{\mathbb{M}}(z_i, \mathbf{p}_i) : z_i = \mathbb{M}(\mathbf{p}_i)$

Definition of the related operator $\mathcal{C}_{\mathbb{M}}([z_i], [\mathbf{p}_i])$:

$$\begin{pmatrix} [z_i] \\ [\mathbf{p}_i] \end{pmatrix} \mapsto \begin{pmatrix} [z_i] \cap \mathbb{M}([\mathbf{p}_i]) \\ \sqcup (\mathbf{p}_i \in [\mathbf{p}_i] \mid \mathbb{M}(\mathbf{p}_i) \in [z_i]) \end{pmatrix} \quad (1)$$

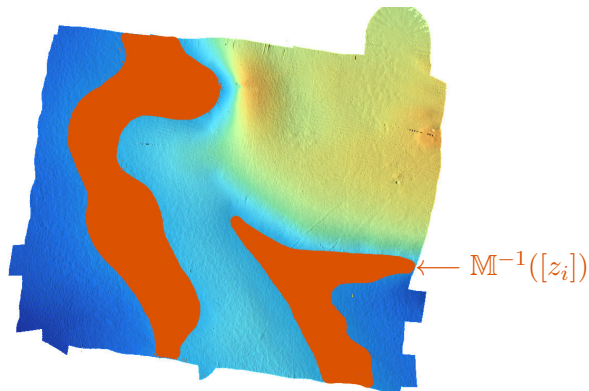


Set-membership TBN

The map constraint $\mathcal{L}_{\mathbb{M}}(z_i, \mathbf{p}_i) : z_i = \mathbb{M}(\mathbf{p}_i)$

Definition of the related operator $\mathcal{C}_{\mathbb{M}}([z_i], [\mathbf{p}_i])$:

$$\begin{pmatrix} [z_i] \\ [\mathbf{p}_i] \end{pmatrix} \mapsto \begin{pmatrix} [z_i] \cap \mathbb{M}([\mathbf{p}_i]) \\ \sqcup (\mathbf{p}_i \in [\mathbf{p}_i] \mid \mathbb{M}(\mathbf{p}_i) \in [z_i]) \end{pmatrix} \quad (1)$$

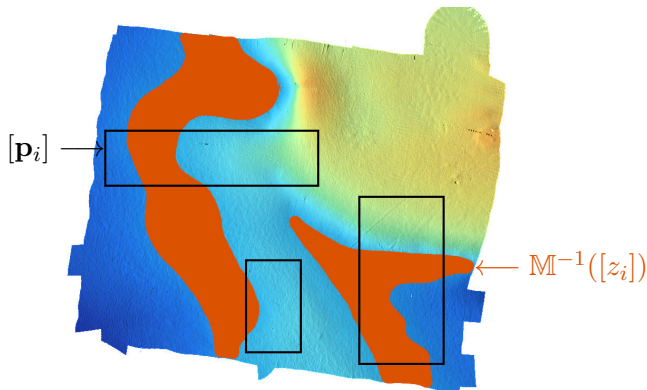


Set-membership TBN

The map constraint $\mathcal{L}_{\mathbb{M}}(z_i, \mathbf{p}_i) : z_i = \mathbb{M}(\mathbf{p}_i)$

Definition of the related operator $\mathcal{C}_{\mathbb{M}}([z_i], [\mathbf{p}_i])$:

$$\begin{pmatrix} [z_i] \\ [\mathbf{p}_i] \end{pmatrix} \mapsto \begin{pmatrix} [z_i] \cap \mathbb{M}([\mathbf{p}_i]) \\ \sqcup (\mathbf{p}_i \in [\mathbf{p}_i] \mid \mathbb{M}(\mathbf{p}_i) \in [z_i]) \end{pmatrix} \quad (1)$$

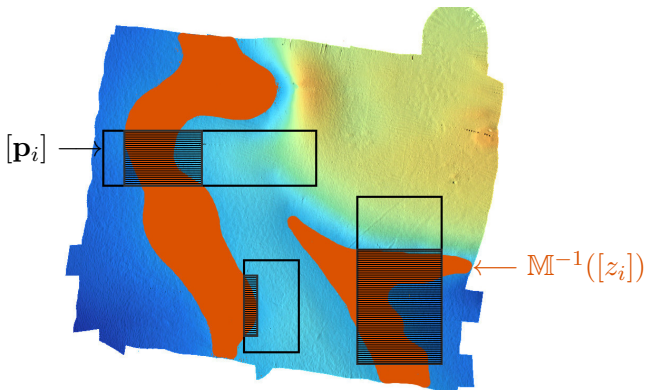


Set-membership TBN

The map constraint $\mathcal{L}_{\mathbb{M}}(z_i, \mathbf{p}_i) : z_i = \mathbb{M}(\mathbf{p}_i)$

Definition of the related operator $\mathcal{C}_{\mathbb{M}}([z_i], [\mathbf{p}_i])$:

$$\begin{pmatrix} [z_i] \\ [\mathbf{p}_i] \end{pmatrix} \mapsto \begin{pmatrix} [z_i] \cap \mathbb{M}([\mathbf{p}_i]) \\ \sqcup (\mathbf{p}_i \in [\mathbf{p}_i] \mid \mathbb{M}(\mathbf{p}_i) \in [z_i]) \end{pmatrix} \quad (1)$$



Set-membership TBN

TBN: resolution method

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Set-membership TBN

TBN: resolution method

System:

$$\begin{cases} \dot{\mathbf{x}}(\cdot) = \mathbf{f}(\mathbf{x}(\cdot)) \\ z_i = \mathbb{M}(\mathbf{x}(t_i)) \end{cases}$$

Contractor programming:

- ▶ $\mathcal{C}_{\mathbf{f}}([\mathbf{x}](\cdot), [\mathbf{v}](\cdot))$
- ▶ $\mathcal{C}_{\frac{d}{dt}}([\mathbf{x}](\cdot), [\mathbf{v}](\cdot))$
- ▶ $\mathcal{C}_{\text{eval}}([t_i], [\mathbf{p}_i], [\mathbf{x}](\cdot))$
- ▶ $\mathcal{C}_{\mathbb{M}}([z_i], [\mathbf{p}_i])$

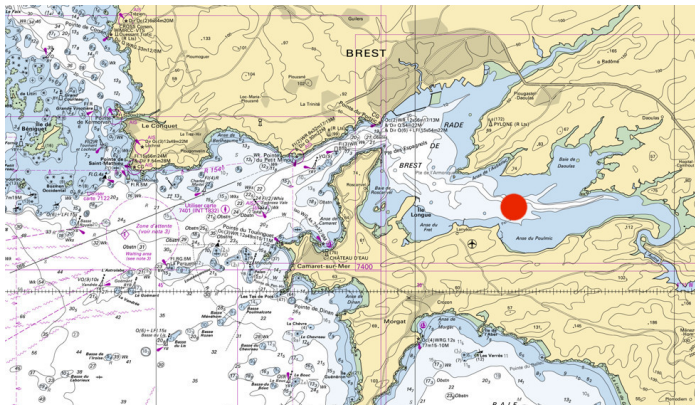
Section 3

Application

Application

Experimental mission with the Daurade AUV

- ▶ 2 hours experimental mission
- ▶ *Rade de Brest*, Brittany



Location: *Polygone de Rascas* – Credits: Shom

Application

Experimental mission with the Daurade AUV

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



Special thanks to DGA-TN Brest (formerly GESMA)

Application

Experimental mission with the Daurade AUV

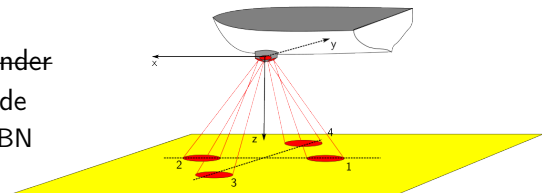
Mission data

Proprioceptive sensors:

- ▶ Inertial Navigation System (INS): Euler angles, accelerations
- ▶ Doppler Velocity Log (DVL): absolute velocities

Exteroceptive sensors:

- ▶ ~~single beam echo sounder~~
- ▶ DVL → vehicle altitude
 - ▶ not suitable for TBN

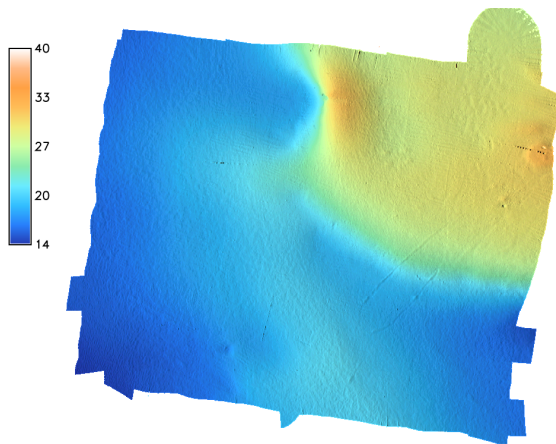


Double Janus DVL with 4 beams (image: Michel Legris)

Application

Map \mathbb{M} of the environment

Prior knowledge: bathymetric map of Rascas, denoted \mathbb{M}

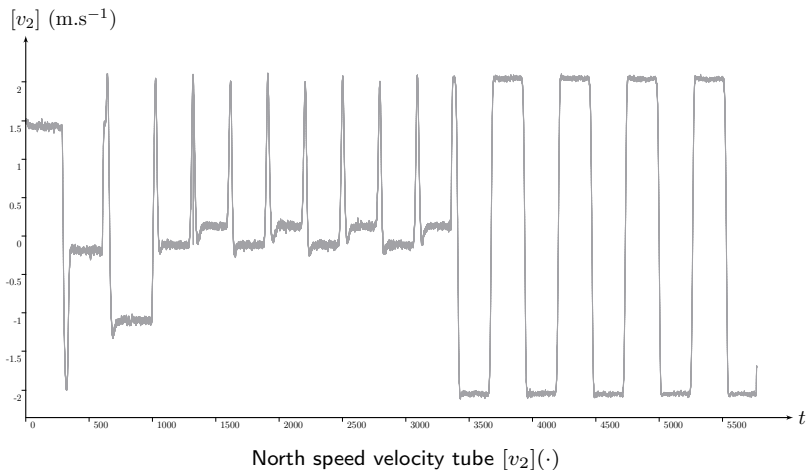


Thanks to Irène Mopin, Pierre Simon,
Romain Schwab, Michel Legris, Rodéric Moitié

Application

Evolution measurements

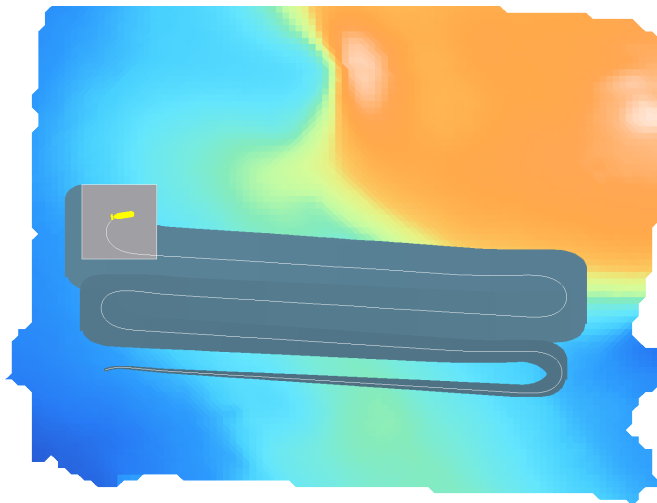
- ▶ velocity measurements obtained with a DVL
- ▶ considering uncertainties, building a tube $[\mathbf{v}](\cdot)$



Application

Dead reckoning from evolution measurements only

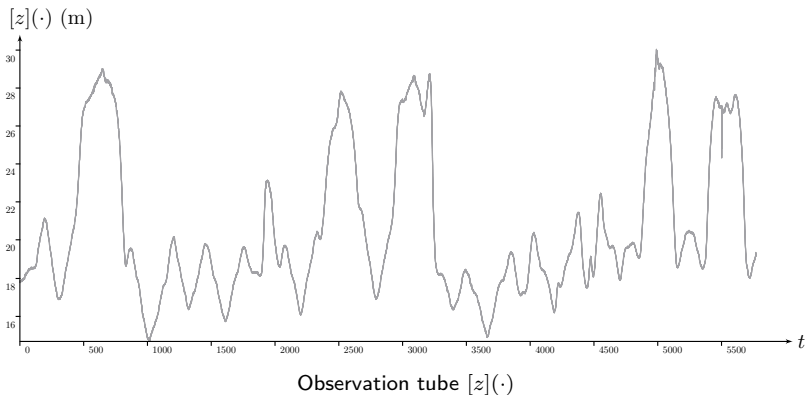
Video



Application

Observations measurements: bathymetric values

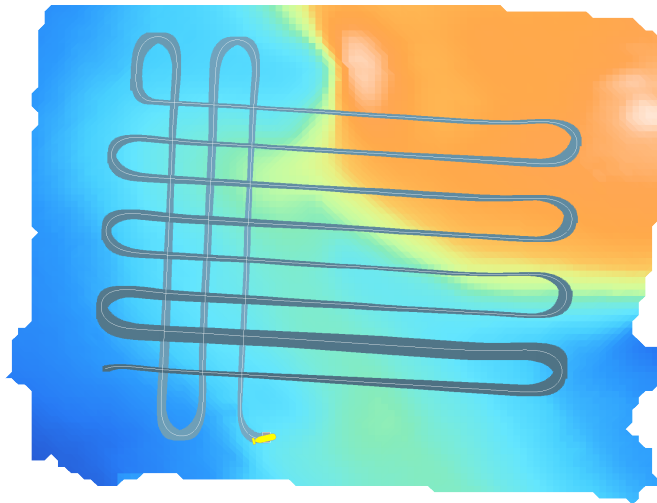
- ▶ DVL, same sensor, can provide **altitude measurements** z_{alt}
- ▶ pressure sensor: depth values z_{depth}
- ▶ time-dependent measurements, use of **tide models**
- ▶ $z = z_{\text{alt}} + z_{\text{depth}} + z_{\text{tide}}$



Application

Full example of TBN

Video



Section 4

Conclusions

Conclusions

Assets of constraint programming

- ▶ **simplicity** of the approach
transparent application of contractors on elementary constraints

Conclusions

Assets of constraint programming

- ▶ **simplicity** of the approach
transparent application of contractors on elementary constraints
- ▶ **reliability** of the results: no solution can be lost
useful for proof purposes and the safety of systems

Conclusions

Assets of constraint programming

- ▶ **simplicity** of the approach
transparent application of contractors on elementary constraints
- ▶ **reliability** of the results: no solution can be lost
useful for proof purposes and the safety of systems
- ▶ focus on **the *what* instead of the *how***
no expertise required on how to solve a problem

Conclusions

Assets of constraint programming

- ▶ **simplicity** of the approach
transparent application of contractors on elementary constraints
- ▶ **reliability** of the results: no solution can be lost
useful for proof purposes and the safety of systems
- ▶ focus on **the *what* instead of the *how***
no expertise required on how to solve a problem
- ▶ **complex systems** easily handled
non-linearities, differential equations, values from datasets

Conclusions

Assets of constraint programming

- ▶ **simplicity** of the approach
transparent application of contractors on elementary constraints
- ▶ **reliability** of the results: no solution can be lost
useful for proof purposes and the safety of systems
- ▶ focus on **the *what* instead of the *how***
no expertise required on how to solve a problem
- ▶ **complex systems** easily handled
non-linearities, differential equations, values from datasets

Tubex library: open-source library providing tools for constraint programming over dynamical systems

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

Conclusions

About this TBN method

- ▶ **fast implementation** with constraint programming
use of already existing contractors, one more to be defined

Conclusions

About this TBN method

- ▶ **fast implementation** with constraint programming
use of already existing contractors, one more to be defined
- ▶ **reliability** of the results
guaranteed enclosure of the actual trajectory

Conclusions

About this TBN method

- ▶ **fast implementation** with constraint programming
use of already existing contractors, one more to be defined
- ▶ **reliability** of the results
guaranteed enclosure of the actual trajectory
- ▶ based on **poor observations**
scalar bathymetric values, e.g. from a single beam echo sounder

Conclusions

About this TBN method

- ▶ **fast implementation** with constraint programming
use of already existing contractors, one more to be defined
- ▶ **reliability** of the results
guaranteed enclosure of the actual trajectory
- ▶ based on **poor observations**
scalar bathymetric values, e.g. from a single beam echo sounder
- ▶ **no initial condition** required
the constraints apply in any order up to a fixed point