Distributed localization with intervals

L. Jaulin
ENSTA Bretagne, LabSTICC
Présentation Thales
May 16, Brest

Interval analysis

Problem. Given $f:\mathbb{R}^n\rightarrow\mathbb{R}$ and a box $[\mathbf{x}]\subset\mathbb{R}^n$, prove that

$$\forall\mathbf{x}\in[\mathbf{x}],f(\mathbf{x})\geq 0.$$

Interval arithmetic can solve efficiently this problem.

Example. Is the function

$$f(\mathbf{x}) = x_1 x_2 - (x_1 + x_2)\cos x_2 + \sin x_1 \cdot \sin x_2 + 2$$

always positive for $x_1, x_2 \in [-1, 1]$?

Interval arithmetic

$$
\begin{aligned}
[-1,3]+[2,5] &= ?,\\
[-1,3]\cdot[2,5] &= ?,\\
\text{abs}([-7,1]) &= ?
\end{aligned}
$$

Interval arithmetic

$$\begin{aligned} [-1,3]+[2,5] &= [1,8], \\ [-1,3]\cdot[2,5] &= [-5,15], \\ \text{abs}([-7,1]) &= [0,7] \end{aligned}$$

The interval extension of

$$f(x_1, x_2) = x_1 \cdot x_2 - (x_1 + x_2) \cdot \cos x_2 + \sin x_1 \cdot \sin x_2 + 2$$

is

$$\begin{aligned} [f]([x_1], [x_2]) &= [x_1] \cdot [x_2] - ([x_1] + [x_2]) \cdot \cos [x_2] \\ &\quad + \sin [x_1] \cdot \sin [x_2] + 2. \end{aligned}$$

Theorem (Moore, 1970)

$$[f]([\mathbf{x}]) \subset \mathbb{R}^{+} \Rightarrow \forall \mathbf{x} \in [\mathbf{x}], f(\mathbf{x}) \geq 0.$$

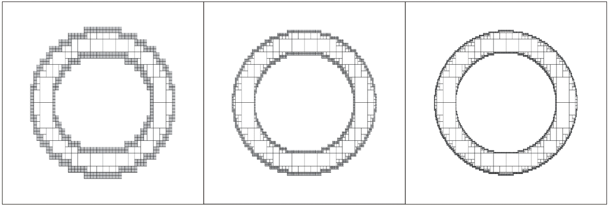
Set Inversion

A subpaving of $\mathbb{R}^n$ is a set of non-overlapping boxes of $\mathbb{R}^n$.
Compact sets $\mathbb{X}$ can be bracketed between inner and outer subpavings:

$$\mathbb{X}^- \subset \mathbb{X} \subset \mathbb{X}^+.$$

Example.

$$\mathbb{X} = \{(x_1, x_2) \mid x_1^2 + x_2^2 \in [1,2]\}.$$



Let $\mathbf{f}: \mathbb{R}^n \to \mathbb{R}^m$ and let $\mathbb{Y}$ be a subset of $\mathbb{R}^m$. Set inversion is the characterization of

$$\mathbb{X} = \{\mathbf{x} \in \mathbb{R}^n \mid \mathbf{f}(\mathbf{x}) \in \mathbb{Y}\} = \mathbf{f}^{-1}(\mathbb{Y}).$$

We shall use the following tests.

$$\begin{array}{lllll} \text{(i)} & [\mathbf{f}]([\mathbf{x}]) \subset \mathbb{Y} & \Rightarrow & [\mathbf{x}] \subset \mathbb{X} \\ \text{(ii)} & [\mathbf{f}]([\mathbf{x}]) \cap \mathbb{Y} = \emptyset & \Rightarrow & [\mathbf{x}] \cap \mathbb{X} = \emptyset. \end{array}$$

Boxes for which these tests failed, will be bisected, except if they are too small.

Dynamical localization

Contractors

The operator $\mathscr{C} : \mathbb{IR}^n \to \mathbb{IR}^n$ is a *contractor* [4] for the equation $f(\mathbf{x}) = 0$, if

$$\begin{cases} \mathscr{C}([\mathbf{x}]) \subset [\mathbf{x}] & \text{(contractance)} \\ \mathbf{x} \in [\mathbf{x}] \text{ and } f(\mathbf{x}) = 0 \Rightarrow \mathbf{x} \in \mathscr{C}([\mathbf{x}]) & \text{(consistence)} \end{cases}$$

Building contractors

Consider the primitive equation

$$x_1 + x_2 = x_3$$

with $x_1 \in [x_1]$, $x_2 \in [x_2]$, $x_3 \in [x_3]$.

We have

$$\begin{aligned} x_3 = x_1 + x_2 \Rightarrow \quad x_3 &\in \quad [x_3] \cap ([x_1] + [x_2]) \\ x_1 = x_3 - x_2 \Rightarrow \quad x_1 &\in \quad [x_1] \cap ([x_3] - [x_2]) \\ x_2 = x_3 - x_1 \Rightarrow \quad x_2 &\in \quad [x_2] \cap ([x_3] - [x_1]) \end{aligned}$$

The contractor associated with $x_1 + x_2 = x_3$ is thus

$$\mathscr{C}\begin{pmatrix} [x_1] \\ [x_2] \\ [x_3] \end{pmatrix} = \begin{pmatrix} [x_1] \cap ([x_3] - [x_2]) \\ [x_2] \cap ([x_3] - [x_1]) \\ [x_3] \cap ([x_1] + [x_2]) \end{pmatrix}$$

Tubes

A trajectory is a function $\mathbf{f}:\mathbb{R}\to\mathbb{R}^n$. [6, 5]. For instance

$$\mathbf{f}(t)=\left(\begin{array}{c}\cos t\\ \sin t\end{array}\right)$$

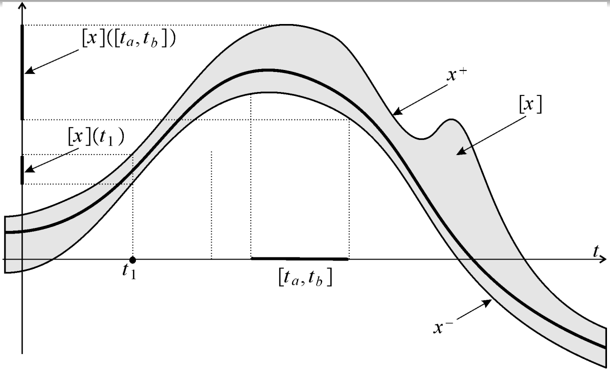
is a trajectory.

Order relation

$$\mathbf{f} \leq \mathbf{g} \Leftrightarrow \forall t, \forall i, f_i(t) \leq g_i(t).$$

We have

$$\mathbf{h} = \mathbf{f} \wedge \mathbf{g} \Leftrightarrow \forall t, \forall i, h_i(t) = \min(f_i(t), g_i(t)),$$
$$\mathbf{h} = \mathbf{f} \vee \mathbf{g} \Leftrightarrow \forall t, \forall i, h_i(t) = \max(f_i(t), g_i(t)).$$



The set of trajectories is a lattice. Interval of trajectories (tubes) can be defined.

Example.

$$[\mathbf{f}](t) = \begin{pmatrix} \cos t + [0, t^2] \\ \sin t + [-1, 1] \end{pmatrix}$$

is an interval trajectory (or tube).

Tube arithmetics

If $[x]$ and $[y]$ are two scalar tubes [1], we have

$$
\begin{array}{ll}
[z]=[x]+[y]\Rightarrow[z](t)=[x](t)+[y](t) & \text{(sum)}\\
[z]=\text{shift}_a([x])\Rightarrow[z](t)=[x](t+a) & \text{(shift)}\\
[z]=[x]\circ[y]\Rightarrow[z](t)=[x]([y](t)) & \text{(composition)}\\
[z]=\int[x]\Rightarrow[z](t)=\left[\int_0^t x^-(\tau)\,d\tau,\int_0^t x^+(\tau)\,d\tau\right] & \text{(integral)}
\end{array}
$$

Tube Contractors

Tube arithmetic allows us to build contractors [3].

Consider for instance the differential constraint

$$\begin{array}{rcl} \dot{x}(t) & = & x(t+\tau)\cdot u(t), \\ x(t) & \in & [x](t), \dot{x}(t) \in [\dot{x}](t), u(t) \in [u](t), \tau \in [\tau] \end{array}$$

We decompose as follows

$$\left\{ \begin{array}{rcl} x(t) & = & x(0) + \int_0^t y(\tau)\, d\tau \\ y(t) & = & a(t)\cdot u(t). \\ a(t) & = & x(t+\tau) \end{array} \right.$$

Possible contractors are

$$\left\{\begin{array}{lll}
[x](t) & = & [x](t)\cap\left([x](0)+\int_0^t [y](\tau)\,d\tau\right)\\
[y](t) & = & [y](t)\cap[a](t)\cdot[u](t)\\
[u](t) & = & [u](t)\cap\frac{[y](t)}{[a](t)}\\
[a](t) & = & [a](t)\cap\frac{[y](t)}{[u](t)}\\
[a](t) & = & [a](t)\cap[x](t+[\tau])\\
[x](t) & = & [x](t)\cap[a](t-[\tau])\\
[\tau] & = & [\tau](t)\cap\dots
\end{array}\right.$$

Example. Consider $x(t) \in [x](t)$ with the constraint

$$\forall t,\ x(t) = x(t+1)$$

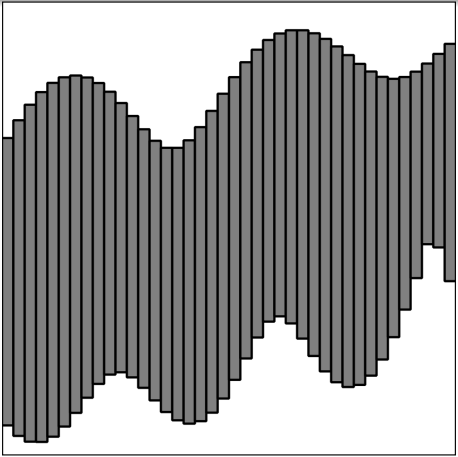
Contract the tube $[x](t)$.

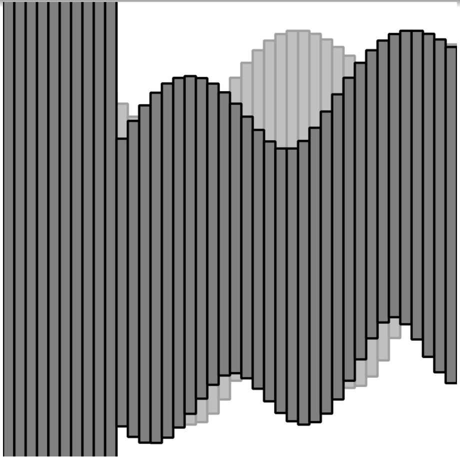
We first decompose into primitive trajectory constraints

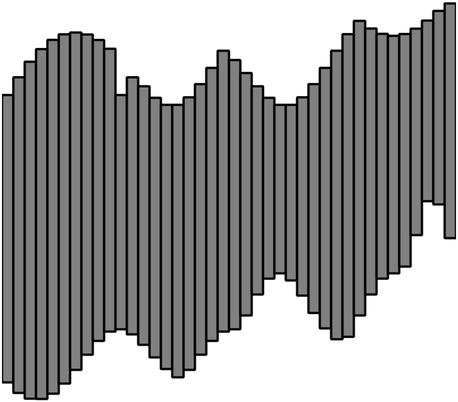
$$\begin{aligned} x(t) &= a(t+1) \\ x(t) &= a(t). \end{aligned}$$

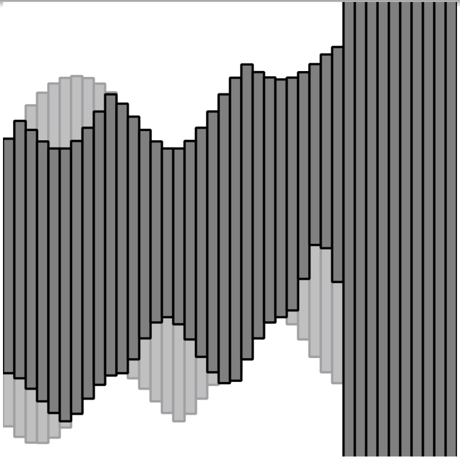
Contractors

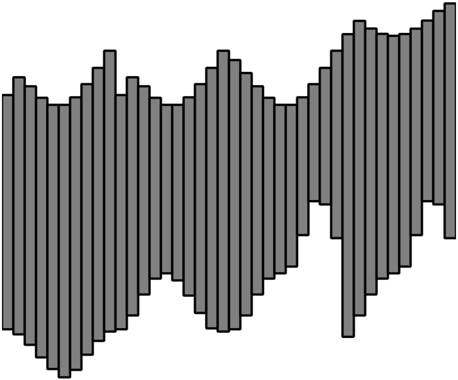
$$
\begin{aligned}
[x](t) &:= [x](t) \cap [a](t+1) \\
[a](t) &:= [a](t) \cap [x](t-1) \\
[x](t) &:= [x](t) \cap [a](t) \\
[a](t) &:= [a](t) \cap [x](t)
\end{aligned}
$$

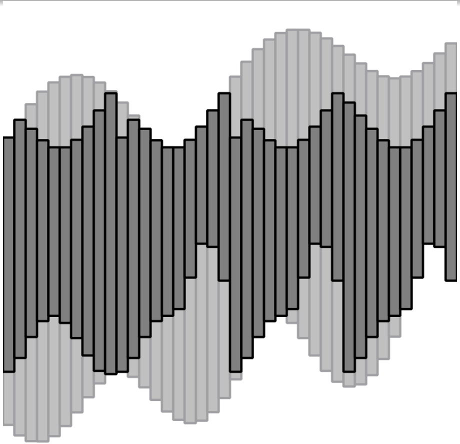


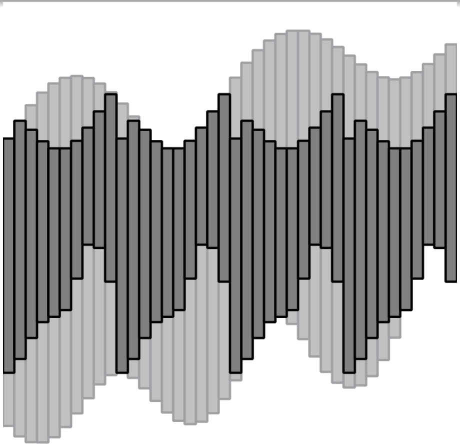


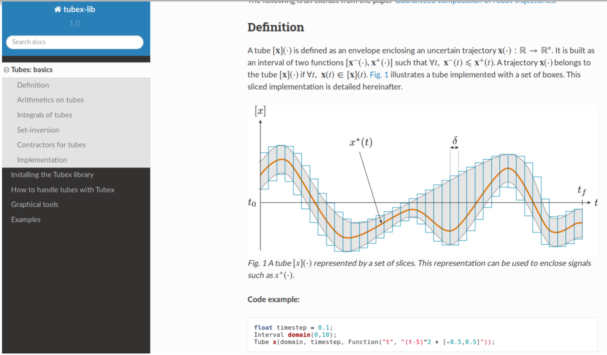












tubex-lib
1.0
Search docs

⊟ Tubes: basics
- Definition
- Arithmetics on tubes
- Integrals of tubes
- Set-inversion
- Contractors for tubes
- Implementation

Installing the Tubex library
How to handle tubes with Tubex
Graphical tools
Examples

The following is an extract from the paper Guaranteed computation of robot trajectories.

Definition

A tube $[\mathbf{x}](\cdot)$ is defined as an envelope enclosing an uncertain trajectory $\mathbf{x}(\cdot) : \mathbb{R} \rightarrow \mathbb{R}^n$. It is built as an interval of two functions $[\mathbf{x}^-(\cdot), \mathbf{x}^+(\cdot)]$ such that $\forall t,\ \mathbf{x}^-(t) \leqslant \mathbf{x}^+(t)$. A trajectory $\mathbf{x}(\cdot)$ belongs to the tube $[\mathbf{x}](\cdot)$ if $\forall t,\ \mathbf{x}(t) \in [\mathbf{x}](t)$. Fig. 1 illustrates a tube implemented with a set of boxes. This sliced implementation is detailed hereinafter.

Fig. 1 A tube $[x](\cdot)$ represented by a set of slices. This representation can be used to enclose signals such as $x^*(\cdot)$.

Code example:

```
float timestep = 0.1;
Interval domain(0,10);
Tube x(domain, timestep, Function("t", "(t-5)^2 + [-0.5,0.5]"));
```

http://www.simon-rohou.fr/research/tubex-lib/ [6]

Time-space estimation

Classical state estimation

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t),\mathbf{u}(t)) & t \in \mathbb{R} \\ \mathbf{0} = \mathbf{g}(\mathbf{x}(t),t) & t \in \mathbb{T} \subset \mathbb{R}. \end{cases}$$

Space constraint $\mathbf{g}(\mathbf{x}(t),t) = 0$.

Example.

$$\begin{cases} \dot{x}_1 = x_3 \cos x_4 \\ \dot{x}_2 = x_3 \cos x_4 \\ \dot{x}_3 = u_1 \\ \dot{x}_4 = u_2 \\ (x_1(5) - 1)^2 + (x_2(5) - 2)^2 - 4 = 0 \\ (x_1(7) - 1)^2 + (x_2(7) - 2)^2 - 9 = 0 \end{cases}$$

With time-space constraints

$$\begin{cases} \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t),\mathbf{u}(t)) & t \in \mathbb{R} \\ \mathbf{0} = \mathbf{g}(\mathbf{x}(t),\mathbf{x}(t'),t,t') & (t,t') \in \mathbb{T} \subset \mathbb{R} \times \mathbb{R}. \end{cases}$$

Example. An ultrasonic underwater robot with state

$$\mathbf{x} = (x_1, x_2, \dots) = (x, y, \theta, v, \dots)$$

At time t the robot emits an omnidirectional sound. At time t' it receives it

$$\left(x_1 - x_1^{'}\right)^2 + \left(x_2 - x_2^{'}\right)^2 - c\left(t - t'\right)^2 = 0.$$

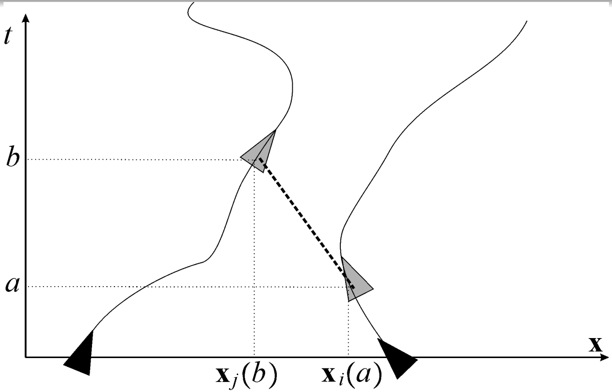
Swarm localization

Consider n robots $\mathscr{R}_1, \dots, \mathscr{R}_n$ described by

$$\dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i), \mathbf{u}_i \in [\mathbf{u}_i].$$

Omnidirectional sounds are emitted and received.

A *ping* is a 4-uple (a, b, i, j) where a is the emission time, b is the reception time, i is the emitting robot and j the receiver.


t
b
a
x
$\mathbf{x}_j(b)$
$\mathbf{x}_i(a)$

With the time space constraint

$$\begin{array}{l} \dot{\mathbf{x}}_i = \mathbf{f}\left(\mathbf{x}_i, \mathbf{u}_i\right), \mathbf{u}_i \in [\mathbf{u}_i]. \\ g\left(\mathbf{x}_{i(k)}\left(a\left(k\right)\right), \mathbf{x}_{j(k)}\left(b\left(k\right)\right), a\left(k\right), b\left(k\right)\right) = 0 \end{array}$$

where

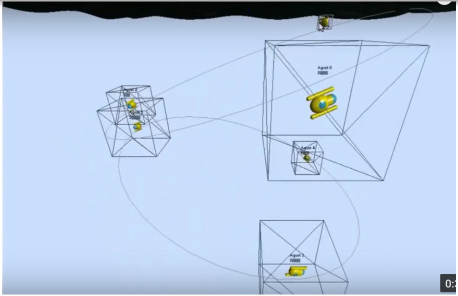
$$g\left(\mathbf{x}_i, \mathbf{x}_j, a, b\right) = \|x_1 - x_2\| - c\left(b - a\right).$$

Clocks are uncertain. We only have measurements $\tilde{a}(k), \tilde{b}(k)$ of $a(k), b(k)$ thanks to clocks h_i. Thus

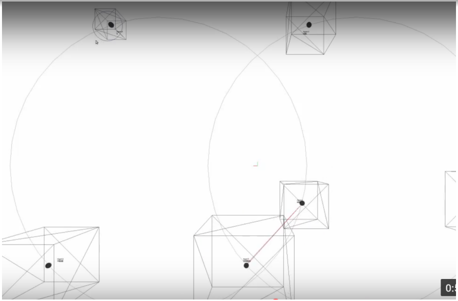
$$
\begin{array}{l}
\dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i), \mathbf{u}_i \in [\mathbf{u}_i]. \\
g\left(\mathbf{x}_{i(k)}(a(k)), \mathbf{x}_{j(k)}(b(k)), a(k), b(k)\right) = 0 \\
\tilde{a}(k) = h_{i(k)}(a(k)) \\
\tilde{b}(k) = h_{j(k)}(b(k))
\end{array}
$$

The drift of the clocks is bounded

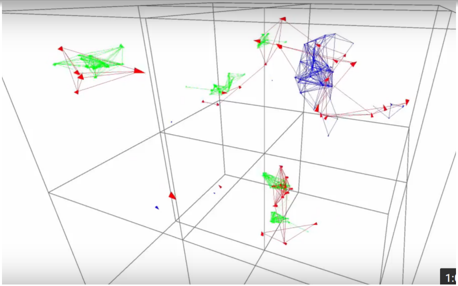
$$\begin{array}{l}
\dot{\mathbf{x}}_i = \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i), \mathbf{u}_i \in [\mathbf{u}_i]. \\
g\left(\mathbf{x}_{i(k)}(a(k)), \mathbf{x}_{j(k)}(b(k)), a(k), b(k)\right) = 0 \\
\tilde{a}(k) = h_{i(k)}(a(k)) \\
\tilde{b}(k) = h_{j(k)}(b(k)) \\
\dot{h}_i = 1 + n_h,\ n_h \in [n_h]
\end{array}$$



https://youtu.be/j-ERcoXF1Ks [2]



https://youtu.be/jr8xKIe0Nds



https://youtu.be/GycJxGFvYE8

https://youtu.be/GVGTwnJ_dpQ

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

A. Bethencourt and L. Jaulin.
Cooperative localization of underwater robots with unsynchronized clocks.
Journal of Behavioral Robotics, 4(4):233–244, 2013.

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

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

S. Rohou.

Reliable robot localization: a constraint programming approach over dynamical systems.
PhD dissertation, Université de Bretagne Occidentale, ENSTA-Bretagne, France, december 2017.

S. Rohou, L. Jaulin, M. Mihaylova, F. Le Bars, and S. Veres.
Guaranteed Computation of Robots Trajectories.
Robotics and Autonomous Systems, 93:76–84, 2017.