PALADYN Journal of Behavioral Robotics

VERSITA

Research Article · DOI: 10.2478/pjbr-2013-0023 JBR · 4(4) · 2013 · 233-244

Cooperative localization of underwater robots with unsynchronized clocks

Aymeric Bethencourt[1]*, **Luc Jaulin**[1]†

1 ENSTA-Bretagne,
Rue Francois Verny, 29200 Brest, France

Received 15-04-2013

Accepted 30-12-2013

Abstract

This paper proposes an interval based approach for localizing a group of underwater robots with unsynchronized clocks. We use sonar communication and consider that the travel time of the sonar waves cannot be neglected, e.g. when the robots are fast and far-spaced. Therefore we cannot suppose that we measure a true distance between robots at the same time, but between robots at different times. Moreover, as the clocks of the robots are unsynchronized, the emitting and receiving times of the sonar waves are uncertain. To solve this problem, we cast the state equations of our robots as a constraint satisfaction problem and consider the sonar measurements as intertemporal constraints on uncertain times. We introduce the notion of interval of function to encompass the robots trajectory and clock with their uncertainty. We then use interval propagation to successively contract these intervals around the true position and clock of the robots. Several test cases are provided, with both simulated and experimental results.

Keywords

Interval analysis · Constraint satisfaction · Localization · Clock synchronization · AUV · Group of underwater robots

1. Introduction

This paper proposes a new approach to localize a group of *Autonomous Underwater Vehicles* (AUVs). Localizing AUVs can be particularly difficult underwater as there is no installed infrastructure. AUVs can only access the GPS on the surface as the electro-magnetic waves can hardly penetrate water. The only tools available underwater are dead-reckoning and acoustical localization systems. To solve this problem, many probabilistic approaches have been studied [24][4], that usually consider the robots close to each other and moving slowly enough so that the displacement of the sonar signal can be consider negligible compared to the precision of the localization [26]. The distance between the robots is measured, and the triangular inequalities are solved to localize the robots [17]. This of course implies that the clocks of each robot have to be perfectly synchronized [18].

In our approach, we consider fast and far-spaced AUVs so that the displacement of the sonar wave can no longer be considered negligible. The measured distance is therefore between the positions of the robots at different times. Moreover we consider the clocks of the robots to be unsynchronized so that the emitting and receiving times are uncertain. We also consider our system to be completely decentralized, meaning that each robot will only localize itself using the data received from the few other robots that are in range of communication.

*E-mail: Aymeric.Bethencourt@ensta-bretagne.fr
†E-mail: Luc.Jaulin@ensta-bretagne.fr

Two approaches can be considered to solve this problem. As presented above, the first approach is probabilistic. If the system is linear, the problem can be solved using for instance a *Kalman Filter*, and if the problem is non-linear, an *Extended Kalman Filter* (EKF) [19] or a *Sequential Monte Carlo* (SMC) method [24] can be considered.

The second approach is ensemblist. When the problem is linear, Ellipsoids and Polytopes [13] can be particularly efficient. However, for non-linear systems, only intervals and sub-pavings have been proven efficient [15]. The principle is to wrap the solution in an interval enclosing its uncertainty, then to successively contract this interval around the solution until a fixed point is reached. For this, we use contractors that are associated to the equations of our system. The advantage of this approach compared to probabilist methods is that the solution is enclosed in an interval. Therefore the results are guaranteed. Moreover Interval analysis is particularly efficient for non-linear systems when the number of equations in far superior to the number of variable, which is often the case in localization applications [14][5].

The originality of this paper is to deal with the inter-temporal aspect of our constraints with uncertain times. To solve this problem, we consider the position and clock of the robots as intervals of trajectories (functions of $\mathbb{R} \rightarrow \mathbb{R}^n$) and cast the equations as a constraint satisfaction problem.

This paper is organized as follows. Section II proposes a simple and new formulation for cooperative localization with inter-temporal constraints on uncertain times. Section III presents the basic notions of constraint propagation and introduces *tubes* (intervals of trajectories), which is used in section IV to solve our problem. Finally section V provides three test cases, with both simulated and experimental results, and Section VI concludes the paper.

2. Problem Statement

Let us first describe our cooperative localization problem. For a robot i in our group or swarm, consider the state equations of the robot as follows:

$$\begin{aligned} \dot{\mathbf{x}}_i &= \mathbf{f}(\mathbf{x}_i, \mathbf{u}_i) + \mathbf{n}_\mathbf{x} \quad \text{(evolution equation)} \\ \mathbf{y}_i &= \mathbf{g}(\mathbf{x}_i) \quad \text{(observation equation)} \end{aligned} \tag{1}$$

where $\mathbf{x}_i$ is the state vector of the robot i, $\mathbf{u}_i$ its inputs or commands, $\mathbf{y}_i$ its outputs or measures, and $\mathbf{f}$ and $\mathbf{g}$ respectively the evolution and observation functions. $\mathbf{n}_\mathbf{x}$ is the state noise. The uncertainty on y_i will be represented by an interval around its unknown value. Notice that the observation function usually depends on the state of the robot at the current time. The originality of our approach is to consider a sonar communication between the robots, which will be represented as an inter-temporal constraint between the states of the robot at different times. We consider that each ping encloses the estimated position box of the emitting robot, along with the interval around the emitting time in the robot's clock.

Formulation. An *intertemporal relation* (or *ping* for short) corresponds to a 4-tuple $\mathbf{p} = (a, b, i, j)$ where $a \in \mathbb{R}$ corresponds to the emission time, $b \in \mathbb{R}$ to the reception time, $i \in \{1, \ldots, m\}$ the emitting robot, and $j \in \{1, \ldots, m\}$ the receiver. Due to the causality, we have $b > a$, or equivalently (a, b) is an element of the t-plane [3]. Denote by $\mathbf{p}(k)$ the k^{th} ping, and by t the true time. We denote by $\tau = h_i(t)$ the clock function [23] which for an absolute time t matches the inner time τ of the robot i. Notice that h_i is strictly increasing when no re-synchronization happens and piecewise increasing when it does. For $i \in \{1, \ldots, m\}$, $t \in \mathbb{R}$ and $k \in \{1, \ldots, k_{\max}\}$, we have:

$$\begin{aligned} &\text{(i)} \quad \dot{\mathbf{x}}_i(t) = \mathbf{f}(\mathbf{x}_i(t), \mathbf{u}_i(t)) + \mathbf{n}_\mathbf{x}(t) \\ &\text{(ii)} \quad g\left(\mathbf{x}_{i(k)}(a(k)), \mathbf{x}_{j(k)}(b(k)), a(k), b(k)\right) = 0 \\ &\text{(iii)} \quad \tilde{a}(k) = h_{i(k)}(a(k)) \\ &\text{(iv)} \quad \tilde{b}(k) = h_{j(k)}(b(k)) \\ &\text{(v)} \quad \dot{h}_i(t) = 1 + n_h(t) \end{aligned} \tag{2}$$

(i) corresponds to the state equations of the i-th robots. The state noise $\mathbf{n}_\mathbf{x}(t)$ is assumed to be bounded.
(ii) is the inter-temporal observation function so that $g : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R} \times \mathbb{R}$ is here:

$$g(\mathbf{x}_1, \mathbf{x}_2, a, b) = \left\|\mathbf{x}_1 - \mathbf{x}_2\right\| - c * (b - a) \tag{3}$$

where c is the speed of sound. Using the Euclidean norm, (ii) can be re-written as

$$c * (b(k) - a(k)) = \sqrt{\begin{array}{c}(x_{i(k)}(a(k)) - x_{j(k)}(b(k)))^2 \\ +(y_{i(k)}(a(k)) - y_{j(k)}(b(k)))^2\end{array}} \tag{4}$$

(iii) $\tilde{a}(k)$ corresponds to the local time of the robot $i(k)$ (i.e. the emitter has emitted the k^{th} ping).
(iv) $\tilde{b}(k)$ corresponds to the local time of the robot $j(k)$ (i.e. the receiver has received the k^{th} ping).
(v) $n_h(t)$ is the clock noise and is assumed to be bounded.
Notice that for all k, we know exactly: $\tilde{a}(k), \tilde{b}(k), i(k), j(k)$.
The advantage of such formalism is that it can encompass many previous formalisms of ensemblist localization [14][5] and also takes into consideration inter-temporal constraints on uncertain times. To our knowledge, such formalism has never been considered before. This formalism can also be applied to *Simultaneous Localization and Mapping* (SLAM) by considering only one robot. Landmarks could be considered as stationary robots. Measurements with the GPS above surface could also be considered as a measurement of the robot to itself.

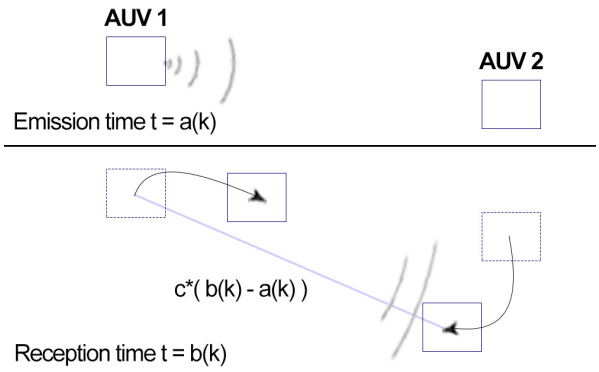

Figure 1. When robot 1 sends a sonar signal, it is received by robot 2 after $b(k) - a(k)$ seconds. Both robots have moved during this time. Therefore the measurement is the distance from robot 1 at $a(k)$ to robot 2 at $b(k)$.

When an AUV send a sonar ping k at $t = a(k)$ received by another AUV at time $t = b(k)$, the distance between the two AUV can be measured as $c * (b(k) - a(k))$ where c is the celerity of the sound. Therefore the measurement between the two robots is inter-temporal. It is between the position of AUV 1 at $a(k)$ and the position of AUV 2 at $b(k)$. Figure 1 illustrates the principle. Notice that these times are uncertain, as the clocks of the robots might not be synchronized. We initially only know an interval (which may be very large) around the clock function h of each robot and are going to contract it to re-synchronize the clock. To our knowledge, this problem cannot be easily solved using standard Monte Carlo methods.

3. Constraint propagation

The domains we consider in our problem are sets enclosing the true value for the variables. Depending on the nature of the set $\mathbb{Z}$ to which an unknown variable z belongs, the domain can have different representations. For instance, if $\mathbb{Z}$ is finite, the domain for z can be described by extension. When $\mathbb{Z}$ is a lattice, the domain for z can be represented by intervals or by a union of intervals (also called sub-paving) [22]. In our context, the trajectories $\mathbf{x}$ and h belong to a lattice and their domains will be represented by tubes (interval of trajectories). To contract these tubes, we use constraint satisfaction, a numerical method to solve non-linear problems such as SLAM [5][7]. This section presents the principle of constraint propagation and extends the technique to tubes.

3.1. Lattices

A *lattice* $(\mathcal{E}, \leq)$ is a partially ordered set, closed under least upper and greatest lower bounds (see [11], for more details). The least upper bound (or infimum) of x and y is called the *join* and is denoted by $x \vee y$. The greatest lower bound (or *supremum*) is called the *meet* and is written as $x \wedge y$.

Example: The set $\mathbb{R}^n$ is a lattice with respect to the partial order relation given by $\mathbf{x} \leq \mathbf{y} \Leftrightarrow \forall i \in \{1, \ldots, n\}, x_i \leq y_i$. We have

$$\mathbf{x} \wedge \mathbf{y} = (x_1 \wedge y_1, \ldots, x_n \wedge y_n) \text{ and}$$
$$\mathbf{x} \vee \mathbf{y} = (x_1 \vee y_1, \ldots, x_n \vee y_n)$$

where $x_i \wedge y_i = \min(x_i, y_i)$ and $x_i \vee y_i = \max(x_i, y_i)$.
A lattice $\mathcal{E}$ is ***complete*** if for all (finite of infinite) subsets $\mathcal{A}$ of $\mathcal{E}$, the least upper bound (denoted $\wedge\mathcal{A}$) and the greatest lower bound (denoted $\vee\mathcal{A}$) belong to $\mathcal{A}$. When a lattice $\mathcal{E}$ is not complete, it is possible to add new elements (corresponding to the supremum or infimum of infinite subsets of $\mathcal{E}$ that do not belong to $\mathcal{E}$) to make it complete. For instance, the set $\mathbb{R}$ is not a complete lattice whereas $\overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty, \infty\}$ is. By convention, for the empty set, we set $\wedge\emptyset = \vee\mathcal{E}$ and $\vee\emptyset = \wedge\mathcal{E}$. The product of two lattices $(\mathcal{E}_1, \leq_1)$ and $(\mathcal{E}_2, \leq_2)$ is the lattice $(\mathcal{E}, \leq)$ defined as the set of all $(a_1, a_2) \in \mathcal{E}_1 \times \mathcal{E}_2$ with the order relation $(a_1, a_2) \leq (b_1, b_2) \Leftrightarrow ((a_1 \leq_1 b_1) \text{ and } (a_2 \leq_2 b_2))$.

3.2. Intervals

An ***interval*** $[x]$ is defined as the set of real numbers x between the lower bound x and upper bound $\bar{x}$.

$$[x] = [\underline{x}, \bar{x}] = \{x \in \mathbb{R}, \underline{x} \leq x \leq \bar{x}\}$$

This representation has several advantages: It allows us to represent random variables with imprecise probability density functions, deal with uncertainties in a reliable way and last but foremost, it is possible to contract the interval around all feasible values given a set of constraints (i.e. equations or inequalities).
A ***closed interval*** (or ***interval*** for short) $[x]$ of a complete lattice $\mathcal{E}$ is a subset of $\mathcal{E}$ which satisfies $[x] = \{x \in \mathcal{E} \mid \wedge[x] \leq x \leq \vee[x]\}$. Both $\emptyset$ and $\mathcal{E}$ are intervals of $\mathcal{E}$. An interval is a sub-lattice of $\mathcal{E}$. $\mathbb{I}\mathcal{E}$ denotes the set of all intervals of $\mathcal{E}$. An interval $[x]$ of $\mathcal{E}$ will also be denoted by $[x] = [\wedge[x], \vee[x]]_{\mathcal{E}}$. For example, the sets $\emptyset = [\infty, -\infty]_{\mathbb{R}}$; $\mathbb{R} = [-\infty, \infty]_{\mathbb{R}}$; $[0, 1]_{\mathbb{R}}$ and $[0, \infty]_{\mathbb{R}}$ are intervals of $\mathbb{R}$, the set $\{2, 3, 4, 5\} = [2, 5]_{\mathbb{N}}$ is an interval of the set of integers $\mathbb{N}$ and the set $\{4, 6, 8, 10\} = [4, 10]_{2\mathbb{N}}$ is an interval of $2\mathbb{N}$.
The ***intersection*** of two non-empty closed intervals $[x]$ and $[y]$ satisfies:

$$[x] \cap [y] = \begin{cases} [\max\{\underline{x}, \underline{y}\}, \min\{\bar{x}, \bar{y}\}] & \text{if } \max\{\underline{x}, \underline{y}\} \leq \min\{\bar{x}, \bar{y}\} \\ \emptyset & \text{otherwise} \end{cases}$$

Example: $[1, 3] \cap [2, 5] = [2, 3]$
The ***union*** of two non-empty closed intervals $[x]$ and $[y]$ satisfies:

$$[x] \sqcup [y] = [\min\{\underline{x}, \underline{y}\}, \max\{\bar{x}, \bar{y}\}]$$

Example: $[1, 3] \sqcup [5, 7] = [1, 7]$

3.3. Intervals Arithmetic

We can apply arithmetic operators and functions to intervals to obtain all feasible values of the variable. For example, consider a real α and $[x]$ a non-empty interval. Then

$$\alpha[x] = \begin{cases} [\alpha\underline{x}, \alpha\bar{x}] & \text{if } \alpha \geq 0 \\ [\alpha\bar{x}, \alpha\underline{x}] & \text{if } \alpha \leq 0 \end{cases}$$

For two intervals $[x]$ and $[y]$ and an operator $\diamond \in \{+, -, *, /\}$, we define $[x] \diamond [y]$ as the smallest interval containing all feasible values for $x \diamond y$ when $x \in [x]$ and $y \in [y]$ or

$$[x] \diamond [y] = [\{x \diamond y \in \mathbb{R} | x \in [x], y \in [y]\}]$$

In the case of closed intervals, we have

$$[x] + [y] = [\underline{x} + \underline{y}, \bar{x} - \bar{y}]$$
$$[x] - [y] = [\underline{x} - \bar{y}, \bar{x} - \underline{y}]$$
$$[x] * [y] = [\min\{\underline{x}\underline{y}, \underline{x}\bar{y}, \bar{x}\underline{y}, \bar{x}\bar{y}\}, \max\{\underline{x}\underline{y}, \underline{x}\bar{y}, \bar{x}\underline{y}, \bar{x}\bar{y}\}]$$

The inversion is given by

$$1/[y] = \begin{cases} \emptyset & \text{if } [y] = [0, 0] \\ [1/\bar{y}, 1/\underline{y}] & \text{if } 0 \notin [y] \\ [1/\bar{y}, \infty[& \text{if } \underline{y} = 0 \text{ and } \bar{y} > 0 \\ ]-\infty, 1/\bar{y}] & \text{if } \underline{y} < 0 \text{ and } \bar{y} = 0 \\ ]-\infty, \infty[& \text{if } \underline{y} < 0 \text{ and } \bar{y} > 0 \end{cases}$$

and the division by

$$[x]/[y] = [x] * (1/[y])$$

These rules are simplified for punctual intervals, in which case we return to the rules of arithmetic on reals. Therefore Interval arithmetic can be considered as an extension of the arithmetic on reals.
For example:

$$[-1, 3] + [2, 7] = [1, 10]$$
$$[-1, 3] - [2, 7] = [-8, 1]$$
$$[-1, 3].[2, 7] = [-7, 21]$$
$$[-1, 3]/[2, 7] = [-1/2, 3/2]$$

The image of $f([x])$ of an interval by a function f is $f([x]) = \{f(x) | x \in [x]\}$. This image might not be an interval. Indeed, if f is not continuous $f([x])$ is a union of intervals. The interval extension is defined as the function returning the interval hull $[f]([x]) = [\{f(x) | x \in [x]\}]$.
The interval extension of elementary functions can be directly written in accordance to its bounds. For example, for a non-empty interval $[x]$, the exponential function $[\exp]([x]) = [\exp \underline{x}, \exp \bar{x}]$.
In case of non-monotonic functions, the situation is a little bit more complicated. Indeed $[\cos]([-\pi, \pi]) = [-1, 1]$ differs from $[\cos(-\pi), \cos(\pi)] = [-1, -1]$. Specific algorithms can be constructed for the interval evaluation of such functions. [8]

3.4. Contractors

Consider n_x real variables $x_i \in \mathbb{R}, i \in \{1, \ldots, n_f\}$ linked by n_f relations (or constraints) of the form :

$$f_j(x_1, x_2, \ldots, x_{n_x}) = 0, j \in \{1, \ldots, n_f\} \quad (5)$$

whereby f_j denotes the function for each coordinate j. We know that each variable x_i belong to a domain $\mathbb{X}_i$. To simplify we consider the domains as intervals, noted $[x_i]$. We define $\mathbf{x} = (x_1, x_2, ..., x_{n_x})^T$ and the domain for $\mathbf{x}$ as $[\mathbf{x}] = [x_1].[x_2]...[x_{n_x}]$. We also note $\mathbf{f}$ the function which coordinates functions are the f_j. We can therefore re-write (5) in a vector form $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ and this represents a ***constraint satisfaction problem*** (CSP) that we can call $\mathfrak{S}$ and write

$$\mathfrak{S} : (\mathbf{f}(\mathbf{x}) = 0, x \in [x])$$

Therefore a CSP is composed of a set of variables, domains containing these variables, and constraints. The solution $\mathbb{S}$ of $\mathfrak{S}$ is defined as

$$\mathbb{S} = \{\mathbf{x} \in [\mathbf{x}] | \mathbf{f}(\mathbf{x}) = \mathbf{0}\}$$

Contracting a CSP $\mathfrak{S}$ consists in replacing the domain $[\mathbf{x}]$ with a smaller domain $[\mathbf{x}']$ without changing the solution set. We have $\mathbb{S} \subset [\mathbf{x}'] \subset [\mathbf{x}]$. We define the optimal contraction of $\mathfrak{S}$ as the operation replacing $[x]$ by the smallest box containing $\mathbb{S}$. An operator that allows the contraction of $\mathfrak{S}$ is called a ***contractor***.
Many problems of estimation, control, robotics, ...etc can be represented as ***constraint satisfaction problems*** (CSP) [2],[9] and many contractors (called minimal) can be applied to optimally contract the domains depending on the class of the problem [10]. Several minimal contractors have been developed over years – Gauss elimination, Gauss-Seidel Algorithm, Krawczyk method, Newton algorithm, ...etc[15] The one we are going to use in our localization problem is the forward-backward contractor [6] which contracts the domains of the CSP $\mathfrak{S} : (\mathbf{f}(\mathbf{x}) = 0, x \in [x])$ by isolating each constraint separately. We suppose that each constraint has a form $f_j(x_1, x_2, ..., x_{n_x}) = 0$, and that the function f_j can be broken down to a series of operations involving operators and elementary functions such as $+, -, *, /, sin, cos, exp, ...etc$ Then we can break down the constraint into primary constraints.
In our example of localization with the distance measurement (4) between AUVs, the associated constraint can be written as

$$d = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \tag{6}$$

so we can break it down to primary constraints by introducing intermediate variables :

$$\begin{aligned} i_1 &= -x_j \\ i_2 &= x_i + i_1 \\ i_3 &= i_2^2 \\ i_4 &= -y_j \\ i_5 &= y_i + i_4 \\ i_6 &= i_5^2 \\ i_7 &= i_3 + i_6 \\ d &= \sqrt{i_7} \end{aligned} \tag{7}$$

The initial domains associated with the intermediate variables i_k are $]-\infty;\infty[$. A method to contract $\mathfrak{S}$ with the constraint is to contract each primitive constraint until the contractor reaches a fixed point. This is the principle of constraint propagation introduced by Waltz [25]. For constraints involving two variables and a function, such as the square root, two steps of contraction are made by rewriting the constraint: one from the direct image of the function and the other from the inverse. Therefore, in our example, the constraint $d = \sqrt{i_7}$ can be re-written in two forms :

$$\begin{aligned} d &= \sqrt{i_7} \\ i_7 &= d^2 \end{aligned}$$

and then the contractions are :

$$\begin{aligned} [d] &= [d] \cap \sqrt{[i_7]} \\ [i_7] &= [i_7] \cap [d^2] \end{aligned}$$

For constraints linking three variables with a binary operation such as addition, there are three ways to rewrite the constraint. Let's consider the constraint $i_7 = i_3 + i_6$ with for example the initial intervals $[i_3] = [-\infty, 2], [i_6] = [-\infty, 3]$ and $[i_7] = [4, \infty]$. We can easily contract these intervals without removing any feasible value:

$$\begin{aligned} i_7 = i_3 + i_6 \rightarrow i_7 &\in [4, \infty] \cap ([-\infty, 2] + [-\infty, 3]) \\ &= [4, \infty] \cap [-\infty, 5] = [4, 5] \\ i_3 = i_7 - i_6 \rightarrow i_3 &\in [-\infty, 2] \cap ([4, \infty] - [-\infty, 3]) \\ &= [-\infty, 2] \cap [1, \infty] = [1, 2] \\ i_6 = i_7 - i_3 \rightarrow z &\in [-\infty, 3] \cap ([4, \infty] + [-\infty, 2]) \\ &= [-\infty, 3] \cap [2, \infty] = [2, 3] \end{aligned}$$

We obtain much smaller intervals for $[i_3] = [1, 2], [i_6] = [2, 3]$ and $[i_7] = [4, 5]$.
The same principle is applied for all primary constraints (7) in order to contract the intervals around the values of (6). The sequence of contractions made by the forward-backward algorithm is optimal to maximize the contraction. The forward-backward algorithm presented in Table 1 will be used for every distance measurement, i.e. sonar reception of an AUV.

3.5. Tubes

In our system, we consider the distance measurements to be intertemporal, i.e. between a emission time by AUV1, and a different reception time by AUV2. This means that we need to enclose the position and clock of the robots at the reception time but also at the emission time. We shall now present ***tubes*** which are an ensemblist vision of a random signal. They will enclose the whole history of the AUVs trajectory and clock function.
Tube. The set $\mathcal{F}^n$ of all functions from $\mathbb{R}$ to $\mathbb{R}^n$ is a complete lattice with the following partial order $\mathbf{x} \leq \mathbf{y} \Leftrightarrow \forall t \in \mathbb{R}, \mathbf{x}(t) \leq \mathbf{y}(t)$. A ***tube*** $[\mathbf{x}]$ (see, e.g., [16], [20]) is an interval $[\mathbf{x}^-, \mathbf{x}^+]$ of $\mathcal{F}^n$, i.e., a pair of two functions $\mathbf{x}^-, \mathbf{x}^+$ such that for all t, $\mathbf{x}^-(t) \leq \mathbf{x}^+(t)$. The set of all tubes of $\mathcal{F}^n$ is denoted by $\mathbb{I}\mathcal{F}^n$.
An element $\mathbf{x}$ of $\mathcal{F}^n$ belongs to the tube $[\mathbf{x}]$ if $\forall t, \mathbf{x}(t) \in [\mathbf{x}](t)$. Figure 2 presents a function $x \in \mathcal{F}^1$ which is inside the tube $[x]$. This tube gives us information related to the unknown function x.

Table 1. Forward-backward algorithm applied to distance measurement (6).

	Algorithm C_{FB} (in : box, inout : $[x],[\dot{x}]$)
	// *Forward steps*
1	$[i_1] := -[x_j]$
2	$[i_2] := [x_i] + [i_1]$
3	$[i_3] := [i_2^2]$
4	$[i_4] := [-y_j]$
5	$[i_5] := [y_i] + [i_4]$
6	$[i_6] := [i_5^2]$
7	$[i_7] := [i_3] + [i_6]$
8	$[d] := [d] \cap \sqrt{[i_7]}$
	// *Bacward steps*
9	$[i_7] := [i_7] \cap [d]^2$
10	$[i_3] := [i_3] \cap ([i_7] - [i_6])$
11	$[i_6] := [i_6] \cap ([i_7] - [i_3])$
12	$[i_5] := [i_5] \cap (sqr^{-1}[i_6])$
13	$[y_i] := [y_i] \cap ([i_5] - [i_4])$
14	$[i_4] := [i_4] \cap ([i_5] - [y])$
15	$[y_j] := [y_j] \cap -[i_4]$
16	$[i_2] := [i_2] \cap (sqr^{-1}[i_3])$
17	$[x_i] := [x_i] \cap ([i_2] - [i_1])$
18	$[i_4] := [i_4] \cap ([i_2] - [x_i])$
19	$[x_j] := [x_j] \cap -[i_1]$

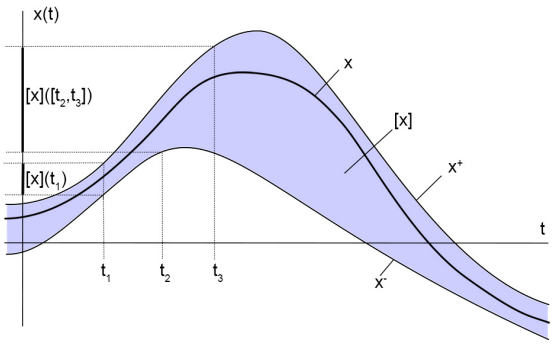

Figure 2. A tube $[x]$ of $\mathbb{R}$ which encloses the function x

Interval evaluation of a tube. If $\mathbf{x}$ is a function from $\mathbb{R}$ to $\mathbb{R}^n$ (i.e., $\mathbf{x} \in \mathcal{F}^n$), we define

$$\mathbf{x}([t]) = \{\mathbf{x}(t), t \in [t]\}.$$

The interval evaluation of a tube $[\mathbf{x}]$ is defined by

$$[\mathbf{x}]([t]) = \bigsqcup_{t \in [t]} [\mathbf{x}](t),$$

i.e., $[\mathbf{x}]([t])$ is the smallest box which encloses all boxes $[\mathbf{x}](t), t \in [t]$. It is easy to prove that

$$\mathbf{x} \in [\mathbf{x}], t \in [t] \Rightarrow \mathbf{x}(t) \in [\mathbf{x}]([t]),$$

and that no box smaller than $[\mathbf{x}]([t])$ satisfies this property.

Tube arithmetic. We can extend some classical operations we have on functions of $\mathcal{F}^n$, such as sums, multiplication, image of a function, ...etc to tubes. It suffices to use the classical notion of interval arithmetic and inclusion functions [21] for all t. An arithmetic on tubes is thus a direct extension of interval arithmetic. As in the case of interval computation, the result of an operation on tubes contains all results of the same operation performed on the enclosed elements of $\mathcal{F}^n$.

Integral. Consider two numbers t_1, t_2 such that $t_2 \geq t_1 \geq 0$. The integral of a tube $[\mathbf{x}]$ over an interval $[t_1, t_2]$ is defined by

$$\int_{t_1}^{t_2} [\mathbf{x}](\tau)\, d\tau = \left\{ \int_{t_1}^{t_2} \mathbf{x}(\tau)\, d\tau \text{ such that } \mathbf{x} \in [\mathbf{x}] \right\}.$$

Since $t_2 \geq t_1$, we deduce from the monotonicity of the integral operator that

$$\int_{t_1}^{t_2} [\mathbf{x}](\tau)\, d\tau = \left[\int_{t_1}^{t_2} \mathbf{x}^-(\tau)\, d\tau, \int_{t_1}^{t_2} \mathbf{x}^+(\tau)\, d\tau \right].$$

From the definition of tube integrals, we have

$$\mathbf{x} \in [\mathbf{x}] \Rightarrow \int_{t_1}^{t_2} \mathbf{x}(\tau)\, d\tau \in \int_{t_1}^{t_2} [\mathbf{x}](\tau)\, d\tau.$$

Moreover, the *interval primitive* defined by $\int_0^t [\mathbf{x}](\tau)\, d\tau$ defines a tube that vanishes for $t = 0$.

3.6. Constraint propagation on tubes

Tube contractor. Consider a constraint on trajectories of the form $\mathcal{L}(x)$. A contractor associated with the constraint $\mathcal{L}$ is an operator

$$[x] \xrightarrow{C_{\mathcal{L}}} [y]$$

where $[x]$ and $[y]$ are tubes, such that

$$\begin{pmatrix} \forall t, [x](t) \subset [y](t) & \text{(contraction)} \\ \mathcal{L}(x) \\ x \in [x] \end{pmatrix} \Longrightarrow x \in [y] \quad \text{(completeness)}$$

We call C^* is the *minimal contractor* for $\mathcal{L}$ that returns the smallest tubes $[y]$ that are consistent with the constraint $\mathcal{L}$.

Minimal tube contractor. Contractors on tubes are built in a similar way to what is done for its interval counterpart. The constraint $\mathcal{L}(x_1, ..., x_p)$ with $x_1, ..., x_p$ in $\mathbb{I}\mathcal{F}^n$ is first rewritten as p equivalent form $x_1 = f_1(x_2, ..., x_p) \iff x_2 = f_2(x_1, x_3, ..., x_p) \iff ...$ (in a similar way to what is done for constraints involving real numbers). The tube arithmetic is then used to automatically generate the contractors.

Proposition 1. The minimal contractor associated with the constraint $x \leqslant y$ or $\forall t \in [0, \infty[, x(t) \leqslant y(t)$ is

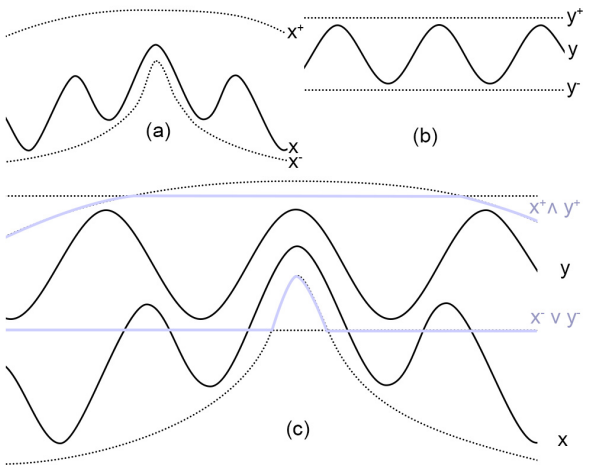

Figure 3. (a) represents the trajectory x and its tube $[x]$, (b) the trajectory y and its tube $[y]$, and (c) illustrates the meet and join of them.

$$C_{\leqslant}\left(\begin{array}{c}[x]\\[y]\end{array}\right)=\left(\begin{array}{c}[x^-,x^+\wedge y^+]\\[x^-\vee y^-,y^+]\end{array}\right)$$

Proof. $x \leqslant y$ so according to the independence theorem, we can fin a such that $y = x + a$ with $a \geqslant 0$. Therefore we can contract the associated tube

$$\begin{aligned}&[y]=[y]\cap([x]+[a])\\\iff&[y]=[y^-\vee(x^-+a^-),y^+\wedge(x^++a^+)]\\\iff&[y]=[y^-\vee(x^-+0),y^+\wedge(x^++\infty)]\\\iff&[y]=[y^-\vee x^-,y^+]\end{aligned}$$

since $[a] = [0, \infty[$. By considering $x = y + a$ with $a \leqslant 0$, we can prove that $[x] = [x^-, x^+ \wedge y^+]$ the same way.
Figure 3 illustrates the contractions.

Proposition 2. The minimal contractor associated with the constraint $x = y$ or $\forall t \in [0, \infty[, x(t) = y(t)$ is

$$C_{=}\left(\begin{array}{c}[x]\\[y]\end{array}\right)=\left(\begin{array}{c}[x^-\vee y^-,x^+\wedge y^+]\\[x^-\vee y^-,x^+\wedge y^+]\end{array}\right)$$

Proof. As $x = y$, any given elements of x and y cannot exceed each other's range. Both variables will share a new interval equal to the intersection of their respective intervals

$$\begin{aligned}&[x]=[x]\cap[y]\\\iff&[x]=[x^-\vee y^-,x^+\wedge y^+]\end{aligned}$$

The same is proven for y.

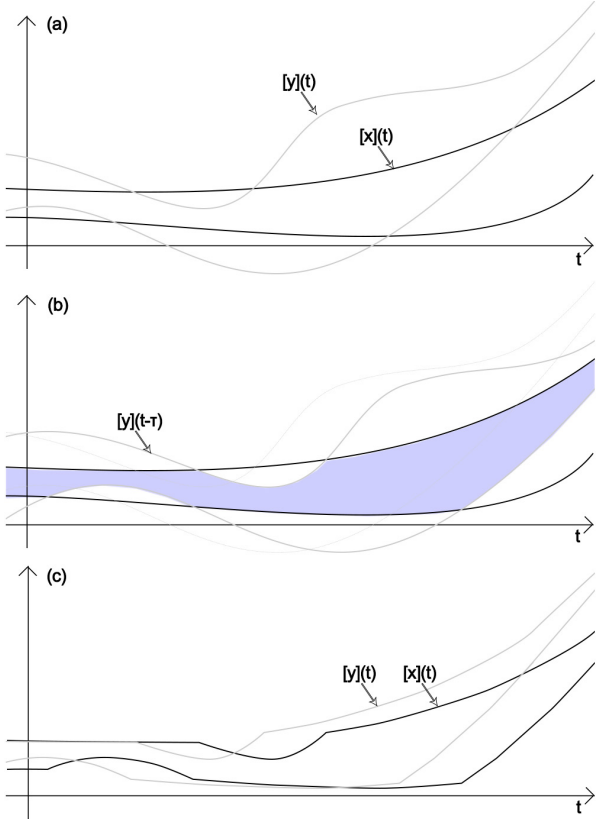

Figure 4. Inter-temporal constraint propagation on tubes. The purple area is the contracted tube.

Proposition 3. The minimal contractor associated with a translation (delay) $\forall t \in [0, \infty[, x(t) = y(t - T)$ is

$$C_{delay}\left(\begin{array}{c}[x]\\[y]\end{array}\right)=\left(\begin{array}{c}[x^-(t)\vee y^-(t-\tau),x^+(t)\wedge y^+(t-\tau)]\\[x^-(t+\tau)\vee y^-(t),x^+(t+\tau)\wedge y^+(t)]\end{array}\right)$$

Proof. The proof is immediate considering Proposition 2.
Fig. 4 illustrates this notion. The purple area represents the tube $[x^-(t) \vee y^-(t - \tau), x^+(t) \wedge y^+(t - \tau)]$

4. Cooperative localization as a constraint satisfaction problem

The cooperative localization problem we described in equation 2 can be considered as a constraint satisfaction problem on tubes and thus we shall apply the contractors presented in section III.

Let's consider a group of multiple AUVs. For any AUV j in our group, we can contract the tubes $[\mathbf{x}_j]$ and $[h_j]$ using the integral contractor from

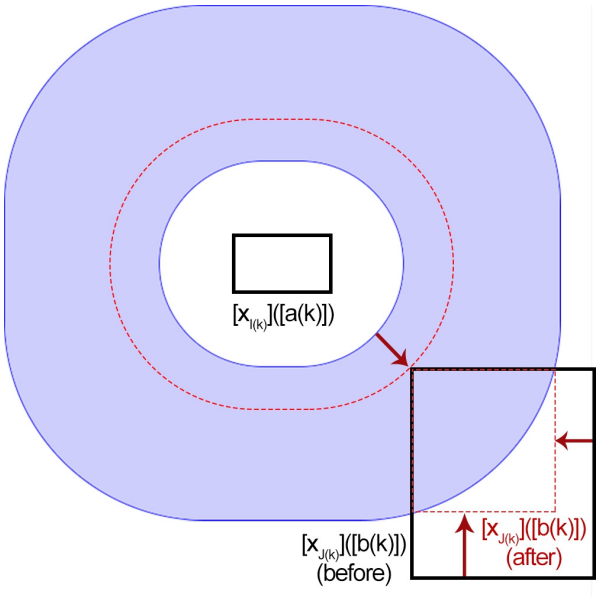

Figure 5. Illustration of the contraction on the position box and on the clock interval.

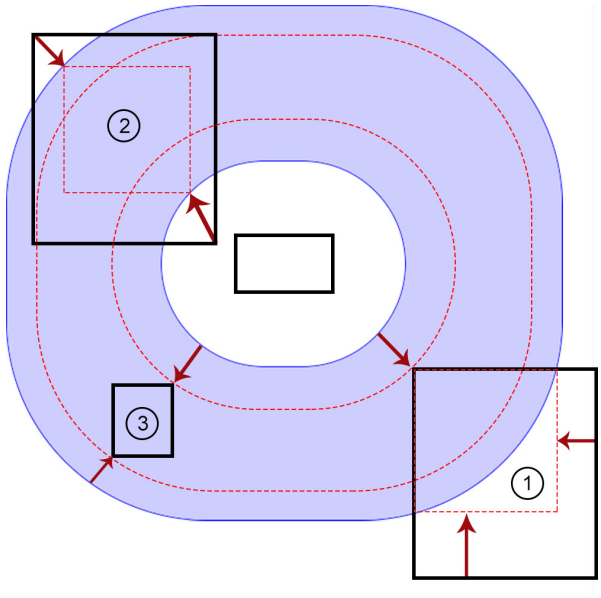

Figure 6. 1. Both the position and the clock are contracted. 2. Only the position is contracted. 3. Only the clock is contracted.

constraints (2i) and (2v):

$$[\mathbf{x}_j](t) = [\mathbf{x}_j](t) \cap \int_0^t ([\mathbf{f}]([\mathbf{u}_j(\tau)],[\mathbf{x}_j(\tau)]) + [\mathbf{n}_x(\tau)])d\tau \quad (8)$$

$$[h_j](t) = [h_j](t) \cap \int_0^t (1 + [n_h](\tau))d\tau \quad (9)$$

These contractors are available for any t; and because of the state noise and clock noise, the width of tubes $[\mathbf{x}_j]$ and $[h_j]$ will keep on growing if nothing is done. However when the robot receives a sonar communication (called ping), equations (2ii), (2iii) and (2iv) will be available as constraints on $[\mathbf{x}_j]$ and $[h_j]$ and enable us to contract them punctually at each ping reception.

Let's consider a robot i emitting a ping received by robot j. Each ping contains the estimated position box $[\mathbf{x}_i]$ of the emitting robot, along with the interval around the time $[a(k)]$ robot i sent the ping k in its own clock; so each time j receives a sonar ping from a robot i, the receiving robot j has data on $[a(k)]$, $[b(k)]$, $[x_{i(k)}](a(k))$, $[x_{j(k)}](b(k))$, $[y_{i(k)}](a(k))$, $[y_{j(k)}](b(k))$ and it can apply the forward-backward algorithm presented in section III on equation 4 to contract these intervals.

Fig. 5 illustrates the contractions made by the forward-backward algorithm. The purple area represents the interval enclosing the solution $\mathbf{x}_{j(k)}(b(k))$. Before receiving the ping, an interval $[\mathbf{x}_{j(k)}]([b(k)])$ $(before)$ is known from the contracted tube $[\mathbf{x}_i]$ (contractor 8). This interval can be contracted to $[\mathbf{x}_{j(k)}]([b(k)])$ $(after)$ by intersecting it with the solutions of (2ii), which contracts the position of the robot. Notice that the purple area can also be contracted to the red doted form. This translates in the forward-backward algorithm as a contraction of $[c * (b(k) - a(k))]$ thus $[b(k)]$ and the clock as illustrated in section V. Fig. 14 also illustrates different cases of contraction.

5. Test cases

In all the test cases, we consider fast and far-spaced AUVs defined by the following evolution equation:

$$\dot{\mathbf{x}}_i(t) = \begin{pmatrix} u_{i1}(t).\cos(u_{i2}(t)) \\ u_{i1}(t).\sin(u_{i2}(t)) \end{pmatrix} + \mathbf{n}_\mathbf{x}(t)$$

where $u_{i1}(t)$ and $u_{i2}(t)$ are the components of vector $\mathbf{u}_i(t)$. They are respectively the given speed and steering command to robot i. The vector $\mathbf{x}_i$ contains the abscissa x_i and ordinate y_i of the robot i. $\mathbf{n}_\mathbf{x}(t)$ is the state noise.

5.1. Simple example with 2 AUVs.

Let us first demonstrate the contractions with a simple example involving only two AUVs. The AUVs follow a circular trajectory such that :

$$\mathbf{x}_1(t) = 10\begin{pmatrix} \sin t \\ \cos t \end{pmatrix}$$
$$\mathbf{x}_2(t) = 10\begin{pmatrix} \sin(t+\pi) \\ \cos(t+\pi) \end{pmatrix}$$

The state and clock noises are considered higher for the receiving robot than for the emitting robot to clearly illustrate the contraction.

$$\text{For AUV 1, } \mathbf{n}_\mathbf{x}(t) = 0.1t \text{ and } n_h(t) = 0.01t$$
$$\text{For AUV 2, } \mathbf{n}_\mathbf{x}(t) = t \text{ and } n_h(t) = 0.1t$$

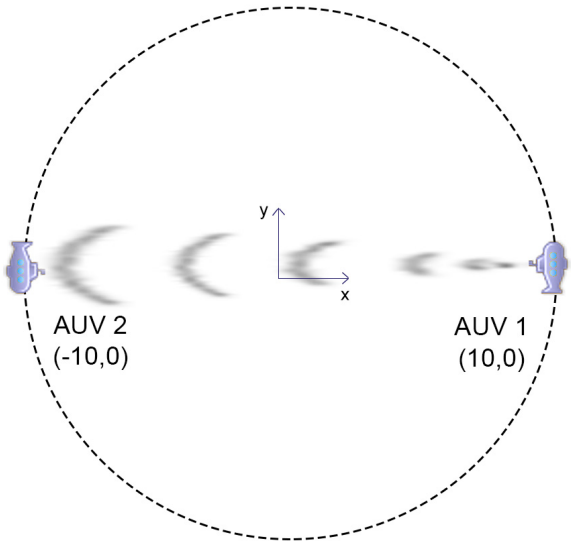

Figure 7. Simple example with two AUV moving on a circular trajectory. AUV 1 emits, AUV 2 receives.

AUV 1 starts at $(10,0)^T$ and AUV 2 at $(-10,0)^T$. Fig. 7 illustrates this example. We consider the celerity of the sound $c = 100$ m/s to simplify.
Using (8) and (9) the AUVs can estimate their position and clock over time thanks to the tubes enclosing the real value. However, the width of the tubes will increase indefinitely if nothing is done.
AUV 1 sends a sonar ping $k = 0$ at what it thinks is $t = 1$ s in its own clock. This is what we note $\tilde{a}(0) = 1$ s which is not the real time $a(0)$. However using the inverse tube of $[h_1]$, we can get an interval $[a(0)]$ enclosing the real emitting time. Using the tube $[x_1]$ and $[y_1]$, we can then get intervals $[x_1]([a(0)])$ and $[y_1]([a(0)])$ enclosing the position of AUV 1 when emitting the signal.
In our example, we obtained:

$\tilde{a}(0)$	$[a(0)]$	$[x_1]([a(0)])$	$[y_1]([a(0)])$
1.00	$[0.99, 1.01]$	$[5.21, 5.59]$	$[8.26, 8.57]$

AUV 2 receives the transmission from AUV 1 at what it thinks is $t = 1.37$ s. This is what we note $\tilde{b}(0) = 1.37$ s which is not the real time $b(0)$. However using the inverse tube of $[h_2]$, we can get an interval $[b(0)]$ enclosing the real emitting time. Using the tube $[x_2]$ and $[y_2]$, we can then get intervals $[x_2]([b(0)])$ and $[y_2]([b(0)])$ enclosing the position of AUV 2 when receiving the signal.
In our example, we obtained:

$\tilde{b}(0)$	$[b(0)]$	$[x_2]([b(0)])$	$[y_2]([b(0)])$
1.37	$[1.22, 1.51]$	$[-4.67, 0.90]$	$[-11.49, -8.16]$

We can then apply our forward-backward algorithm on (4) to contract the intervals of AUV 2. The output is as follow (the contracted values have been emphasized):

$\tilde{b}(0)$	$[b(0)]$	$[x_2]([b(0)])$	$[y_2]([b(0)])$
1.23	$[1.22, \mathbf{1.23}]$	$[-4.67, \mathbf{-1.25}]$	$[-11.49, \mathbf{-9.98}]$

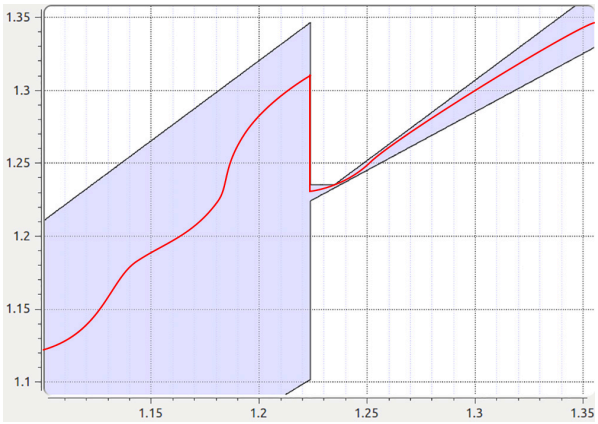

Figure 8. Clock re-synchronization. The purple area represents the tube $[h](t)$ obtained for AUV 2 in Ex.1. The red line represents an hypothetical clock function $\tau = h(t)$ (chosen arbitrarily) enclosed in it. When the tube is contracted, the function is punctually re-synchronized to the center of the tube.

We notice that the box around the position on AUV 2 at the reception of the signal has been properly contracted. The interval enclosing the real time of reception was also contracted which allow us to re-synschronize the clock from 1.37 s to 1.23 s at the reception of the signal. Fig. 8 illustrates the principle of clock re-synchronization. We then use the integral contractors (8) and (9) to propagate the constraint to the rest of the tube. All the results of this simulation will be shown in Fig. 14 and the procedure will be explained step by step at the end of this article.

5.2. Full simulation with 6 AUVs.

Let us now consider a group a 6 simulated AUVs following two circular trajectories:

$$\forall i \in \{1,2,3\}, \mathbf{x}_i(t) = 10\begin{pmatrix}\sin t - 10\\ \cos t\end{pmatrix}$$
$$\forall i \in \{4,5,6\}, \mathbf{x}_i(t) = 10\begin{pmatrix}\sin t\\ \cos t\end{pmatrix}$$

All other parameters are the same than in Example 1. Each robot is equipped with a simulated data pinger with a range of $5m$ emitting its position box and clock interval every 10 seconds. Any robot at a distance more than this range will not receive the ping. We suppose that AUVs $i = 1, 2$ and 3 go to the surface when $x_i(t) < -9m$. Therefore they can access the GPS to contract their position box and clock interval. AUV $4, 5$ and 6 however never go to the surface and have to localize themselves using only the received pings from other AUVs. The simulation is presented in Fig. 9 and results for AUV 4 in Fig. 10-(2).
We illustrated the method with online propagation only, meaning that we contracted the tubes only in the direction of time increasing. However it is possible to consider offline propagation as well, in which we retro-propagate the contractions in decreasing time once the simulation or experiment is done. Fig. 10-(3) illustrates the principle. Offline localization allow us to contract the tubes of the AUVs with much better accuracy but is only useful once the experiment is over.
This simulation runs in realtime on a 3.2 GHz dual core processor.

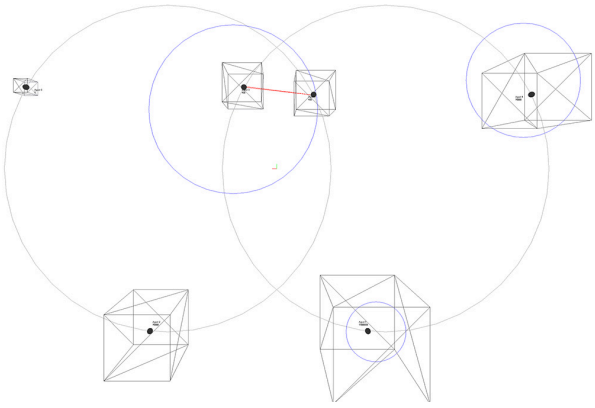

Figure 9. 3D simulation of the robots. The position of the AUV is represented by the boxes. The red line shows robots that are in range of communication and the blue circles represent the displacement of the sonar wave. A video of this simulation is available on http://aymericbethencourt.com/swarm/

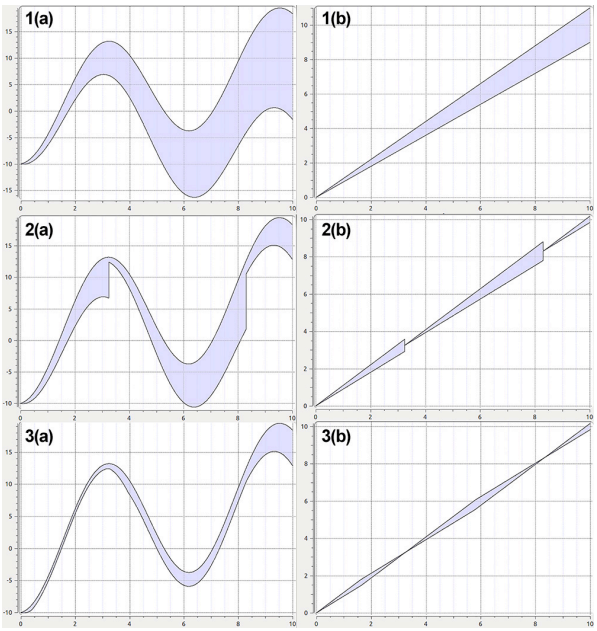

Figure 10. Contracted tubes for (a) $[h_4]$ and (b) $[x_4]$ if (1) there is no ping, (2) two pings with online propagation, (3) two pings with offline propagation.

5.3. Sea testing with 2 real AUVs.

Let us now demonstrate the capacity on our algorithm at sea with the two AUVs from the company *CISCREA* presented in Fig. 11. As no regulator was implemented onboard, we had to command the AUVs with a joystick, make them follow approximate circles and reconstruct their trajectories by measuring the command $\mathbf{u}_i$ from the joystick every 0.001 s. AUV1 was staying at the surface so it could continuously use its GPS to contract its position and synchronize its clock. AUV 2 was staying underwater communicating with AUV 1 through a data pinger.

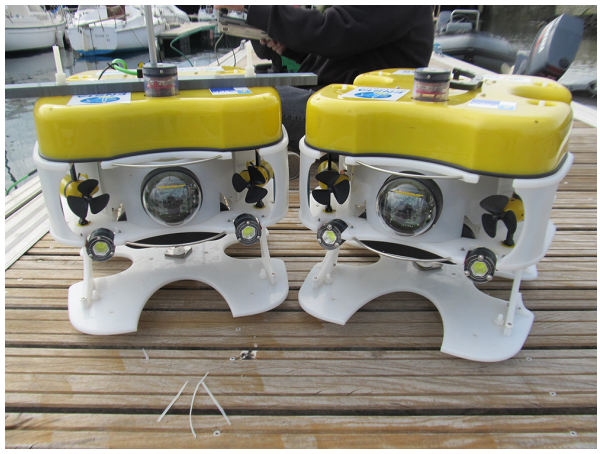

Figure 11. The two CISCREA AUVs used for sea testing.

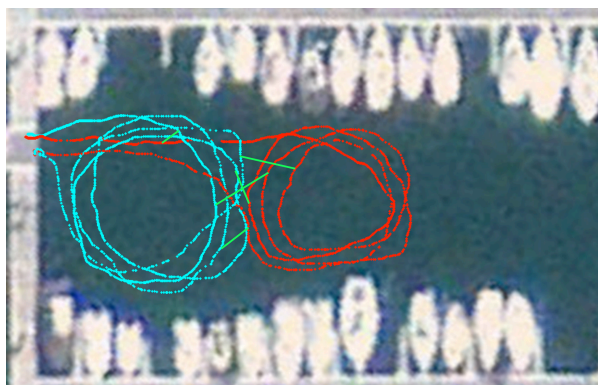

Figure 12. Sea testing in the port of Brest, France. The blue trajectory represents AUV 1 staying at the surface. The red trajectory represents AUV 2 staying underwater. Both AUV were commanded with a joystick. The green lines represent sonar pings when the AUV are in range.

Before putting the robots in the water, we purposefully unsynchronized the clock of AUV 2 so it would be drifted 2 seconds forward. The trajectories of the AUVs and sonar pings are represented in Fig. 12 and the result of the contractions on the position and clock of AUV 2 are presented in Fig. 13.

We notice that on the first ping received by AUV 2, its position is contracted and its clock re-synchronized to be only 0.1 s apart from AUV 1. However the following pings only contracted the position but not the clock any more as the state noise was too high compared to the clock drift. This shows the limits of our algorithm as the two first test cases were considering a high clock drift of 0.1 s per second. In actual AUVs, the clock only drifts a few picoseconds per second. Therefore, on small length missions like presented in these test cases, there is no effective contractions on actual clocks drift, except if the the clocks are initially drifted for a few seconds before the start of the mission. We could also imagine long term missions of several months, or consider AUVs staying asleep at the bottom of the ocean for several months before awakening with their clock drifted for a few seconds.

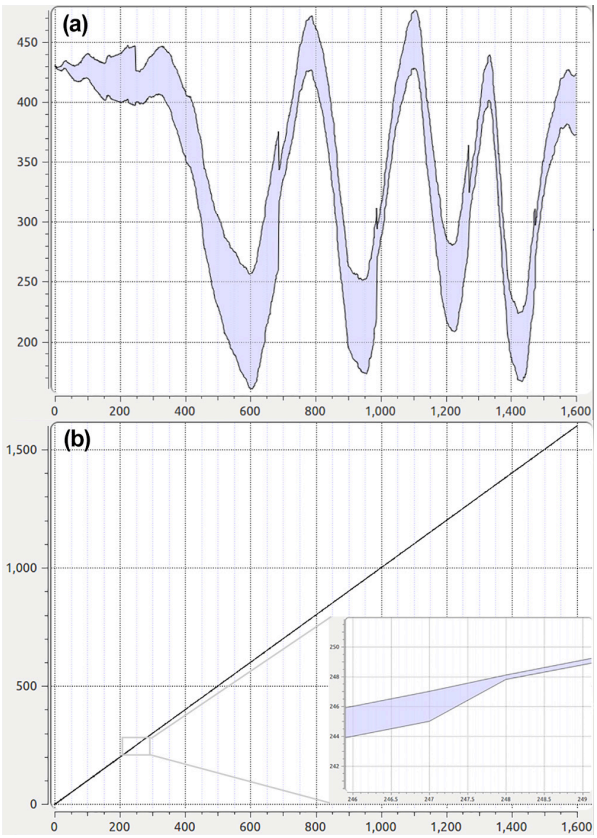

Figure 13. Result of the contracted tubes for (a) the ordinate of AUV 2 and (b) its clock (with a zoom on the important area).

6. Conclusion

Localizing a group of AUVs while synchronizing their clocks is a difficult problem, mainly due to the inter-temporal constraints on uncertain times. To solve the problem, this paper has first introduced the notion of tube, which encompasses the information needed to guarantee possible associations upon trajectories. Then, an arithmetic has been developed around this notion, and some minimal contractors have been proposed.

Then we treated our cooperative localization problem as a constraint satisfaction problem and contracted the boxes around the positions of the AUVs and their clock using a forward-backward algorithm on the inter-temporal measurements with sonar pings. Several test cases were provided, proving the efficiency of the algorithm when the uncertainty on the position and the clock drift had the same order of magnitude. However we also shown that the algorithm was ineffective at contracting the clock when the orders of magnitude were too far apart, especially on short term missions with accurate clocks.

As with most interval-based methods, the proposed approach could be combined with probabilistic methods [1] and made robust with respect to outliers by relaxing a given number of constraints [12].

Finally, in order to share our research with the community, we integrated Tubes, their properties, operators and minimal contractors presented in this article to *IBEX* (Interval-Based EXplorer), a powerful library for interval computation. You can download IBEX and our tube integration on http://www.emn.fr/z-info/ibex/ The source codes for the test cases are available on http://AymericBethencourt.com/swarm/ along with videos of the simulation and experiment.

References

[1] F. Abdallah, A. Gning, and P. Bonnifait. Box particle filtering for nonlinear state estimation using interval analysis. *Automatica*, 44(3):807–815, 2008.

[2] I. Araya, B. Neveu, and G. Trombettoni. Exploiting Common Subexpressions in Numerical CSPs. In *Proc. CP, Constraint Programming*, pages 342–357, LNCS 5202, 2008.

[3] Clement Aubry, Rozenn Desmare, and Luc Jaulin. Loop detection of mobile robots using interval analysis. *Automatica*, 49(2):463–470, 2013.

[4] Alexander Bahr, John J. Leonard, Alexander Bahr, and John J. Leonard. Cooperative localization for autonomous underwater vehicles. In *IN PROC. 10TH INTERNATIONAL SYMPOSIUM ON EXPERIMENTAL ROBOTICS (ISER), RIO DE*, 2006.

[5] F. Le bars, A. Bertholom, J. Sliwka, and L. Jaulin. Interval slam for underwater robots; a new experiment. In *NOLCOS 2010*, Italy, 2010.

[6] F. Benhamou, F. Goualard, L. Granvilliers, and J. F. Puget. Revising hull and box consistency. In *Proceedings of the International Conference on Logic Programming*, pages 230–244, Las Cruces, NM, 1999.

[7] Aymeric Bethencourt and Luc Jaulin. 3d reconstruction using interval analysis on the kinect device coupled with an imu. *International Journal of Advanced Robotic Systems*, 10, 2013.

[8] P. Bouron. *Méthodes ensemblistes pour le diagnostic, l'estimation d'état et la fusion de données temporelles*. PhD dissertation, Université de Compiègne, Compiègne, France, Juillet 2002.

[9] M. Ceberio and L. Granvilliers. Solving nonlinear systems by constraint inversion and interval arithmetic. In *Artificial Intelligence and Symbolic Computation*, volume 1930, pages 127–141, LNCS 5202, 2001.

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

[11] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, (ISBN 0521784514), 2002.

[12] V. Drevelle and P. Bonnifait. High integrity gnss location zone characterization using interval analysis. In *ION GNSS*, 2009.

[13] S. Gollamudi, S. Nagaraj, S. Kapoor, and Y.-F. Huang. Set-membership state estimation with optimal bounding ellipsoids. In *Int. Symposium on Information Theory and its Applications*, 1996.

[14] L. Jaulin. Range-only SLAM with occupancy maps; A set-membership approach. *IEEE Transaction on Robotics*, 27(5):1004–1010, 2011.

[15] L. Jaulin, M. Kieffer, O. Didrit, and E. Walter. Applied interval analysis. *Springer-Verlag*, 2001.

[16] A. Kurzhanski and I. Valyi. *Ellipsoidal Calculus for Estimation and Control*. Birkhäuser, Boston, MA, 1997.

[17] J.J. Leonard and H.F. Durrant-Whyte. Dynamic Map Building for an Autonomous Mobile Robot. *International Journal of Robotics Research*, 11(4), 1992.

[18] Yuanqing Lin, Paul Vernaza, Jihun Ham, and D.D. Lee. Cooperative relative robot localization with audible acoustic sensing. In *Intelligent Robots and Systems, 2005. (IROS 2005). 2005*

IEEE/RSJ International Conference on, pages 3764–3769, 2005.

[19] L. Ljung. Asymptotic behavior of the extended kalman filter as a parameter estimator for linear systems. *IEEE Transactions on Automatic Systems*, 24:36–50, 1979.

[20] M. Milanese, J. Norton, H. Piet-Lahanier, and E. Walter, editors. *Bounding Approaches to System Identification*. Plenum Press, New York, NY, 1996.

[21] R. E. Moore. *Methods and Applications of Interval Analysis*. SIAM, Philadelphia, PA, 1979.

[22] Ralf Sainudiin. Machine interval experiments: Accounting for the physical limits on empirical and numerical resolutions. *LAP Academic Publishers, Kln, Germany*, 2010.

[23] T. K. Srikanth and S. Toueg. Optimal clock synchronization. *Journal of the Association for Computing Machinery*, 34(3):626–645, 1987.

[24] S. Thrun, W. Bugard, and D. Fox. *Probabilistic Robotics*. MIT Press, Cambridge, M.A., 2005.

[25] D.L. Waltz. Generating semantic descriptions from drawings of scenes with shadows. In P. H. Winston, editor, *The Psychology of Computer Vision*, pages 19–91. McGraw-Hill, New York, NY, 1975.

[26] Li-chuan Zhang, De-min Xu, Ming-yong Liu, and Wei-sheng Yan. Cooperative navigation and localization for multiple UUVs. *Journal of Marine Science and Application*, 8(3):216–221, 2009.

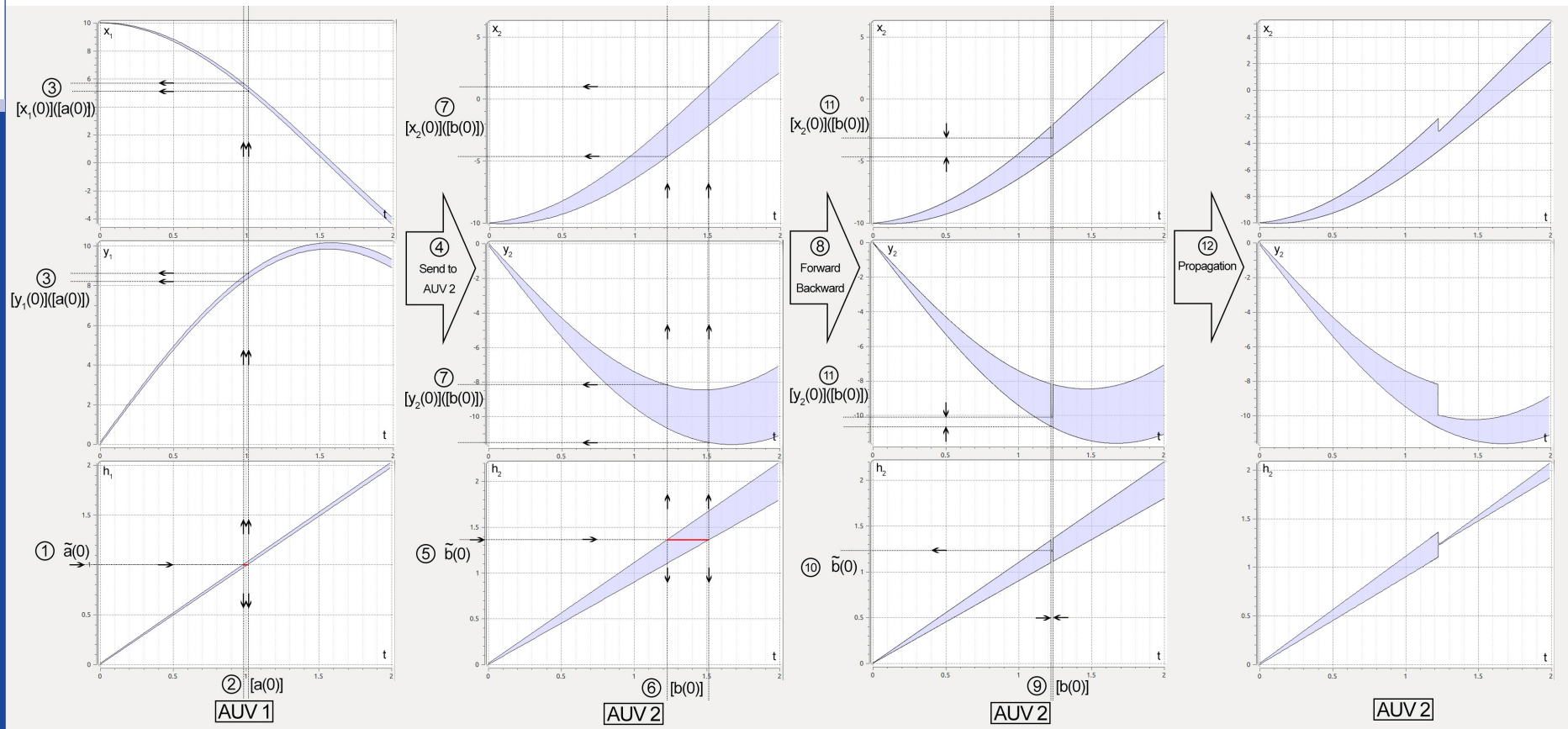

Figure 14. 1. AUV 1 sends a ping at what he thinks is $\tilde{a}(0)$ in its clock. 2. We use the tube inverse of $[h_1]$ to compute that the ping was actually sent at a time enclosed in $[a(0)]$. 3. We use the tubes $[x_1]$ and $[y_1]$ to compute the position of AUV 1 when sending the ping. 4. The ping sent to AUV 2 contains the data $[a(0)]$, $[y1(0)]([a(0)])$ and $[x1(0)]([a(0)])$. 5. AUV 2 receive the ping at what he thinks is $\tilde{b}(0)$ in its clock. 6. We use the tube inverse of $[h_2]$ to compute that the ping was actually received at a time enclosed in $[b(0)]$. 7. We use the tubes $[x_2]$ and $[y_2]$ to compute the position of AUV 2 when receiving the ping. 8. We apply the forward backward algorithm presented in section III, which contract the intervals $[b(0)]$, $[y2(0)]([b(0)])$ and $[x2(0)]([b(0)])$. 9. We apply the contraction to the tube $[h_2]$ which has the effect of re-synchronizing the clock if not included in the contracted tube. Fig.8 illustrates this step. 10. We can compute a supposed receiving time in the robot's clock if it was synchronized. 11. We apply the contractions to the tubes $[x_2]$ and $[y_2]$. 12. We use the integral contractors presented in section IV to propagate the constraint to the rest the tube.