Introduction to the Algebra of Separators with Application to Path Planning

Luc Jaulin and Benoît Desrochers
ENSTA-Bretagne, LabSTICC, IHSEV, OSM, 2 rue François Verny, 29806 Brest.
Email : luc.jaulin@ensta-bretagne.fr

Abstract

Contractor algebra is a numerical tool based on interval analysis which makes it possible to solve many nonlinear problems arising in robotics, such as identification, path planning or robust control. This paper presents the new notion of *separators* which is a pair of complementary contractors and presents the corresponding algebra. Using the separator algebra inside a paver will allow us to get an inner and an outer approximation of the solution set in a much simpler way than using any other interval approach. A path planning problem will then be considered in order to illustrate the principle of the approach.

Index Terms

Contractors, interval analysis, path planning, separators, set characterization.

I. Introduction

Many problems in engineering amount to characterizing a set of $\mathbb{R}^n$ defined by constraints (see [35] for a general introduction to set-membership approaches). For instance, the solution set may correspond (i) to the set of all parameters that are consistent with a some interval measurements [23] [11] [20] [12] [4], (ii) to the set of all configuration vectors such that robot does not meet any obstacles [28] [15], (iii) to the set of all parameter vectors of a controller such that the closed loop system is stable [36] [8], (iv) to set all calibration parameters [7] [29], (v) to attractors of dynamical systems [34], ... More formally, the problem to be considered in this paper is to bracket a set $\mathbb{X}$ defined by constraints between two sets $\mathbb{X}^-$ and $\mathbb{X}^+$ such that

$$\mathbb{X}^- \subset \mathbb{X} \subset \mathbb{X}^+. \tag{1}$$

The set $\mathbb{X}$ is assumed to be defined as combinations of atomic sets. These atomic sets may correspond to sets defined by any nonlinear inequalities or geometric sets such as a map. To compute such an approximation of $\mathbb{X}$, two main classes of approaches are considered: symbolic and numerical. The symbolic approach provides a set of methods based on computer algebra that are guaranteed, efficient, but limited to polynomial problems. The numerical approaches are mainly composed with linear methods (for which linear algebra can be used, see, *e.g.*, [30]), convex methods (such as those based on semidefinite programming [14]), Monte-Carlo methods (which can be used for a large class of nonlinear problems [33]) and interval methods [26] [16] which provide algorithms to compute inner and outer subpavings (*i.e.*, union of non overlapping boxes) to approximate $\mathbb{X}$. The principle of interval methods is similar to Monte Carlo except that they compute with boxes (thanks to interval arithmetic [26]) instead of points, so they can guarantee that all the search space has been covered. Interval methods can thus deal with a large class of non-linear problems (larger than for semidefinite or symbolic methods) in a guaranteed way (contrary to Monte-Carlo). The main drawbacks of interval methods is the high complexity with respect to the number of unknown variables and the lack of tools/softwares that make it possible to implement efficiently an interval resolution dedicated to a nonlinear problem. In order to allow a resolution of high dimensional problems and to facilitate the implementation of efficient interval methods, the notion of contractor [5] has recently been introduced. A *contractor* is an operator which is able to contract boxes without removing any solution. Contractor-based techniques [5] combined with a paver (*i.e.,* a bisection algorithm which partitions the research space with boxes) can provide an outer subpaving approximation $\mathbb{X}^+$ of $\mathbb{X}$ in an efficient way. For the inner subpaving $\mathbb{X}^-$, the *De Morgan* rules can be used to express the complementary set $\overline{\mathbb{X}}$ of $\mathbb{X}$. Then basic contractor techniques can be used to get an inner characterization $\mathbb{X}^-$. Now, the task is not so easy and has never been made automatic. The contribution of this paper is to introduce a new mathematical object, named *separator,* which is composed of two complementary contractors: an inner contractor and an outer contractor. These two contractors are computed together by focusing on the boundary of $\mathbb{X}$. An algebra, similar to the algebra developed for contractors [5], will then be developed for these separators. This will make it possible to easily build separators associated with complex sets $\mathbb{X}$. Combined with a paver, separators will then be able to bracket $\mathbb{X}$ between the two subpavings $\mathbb{X}^-$ and $\mathbb{X}^+$, in an easy way and without asking the programmer to build both the inner and the outer contractors.

The paper is organized as follows. Section II defines separators and shows how they can be used inside a paver to characterize subsets of $\mathbb{R}^n$. Section III explains how to extend all basic operations on sets (such as union, intersection, difference, complementary) to separators. The inversion of separators through a vector function is treated in Section IV. Section V presents how atomic separators can be defined. More complex separators will be obtained by compositions of these atomic separators. An application related to path planning is considered in Section VI. Section VII concludes the paper.

II. SEPARATORS

In this section, we first present the notion of contractors [16] that will be needed to define separators. Then, we show how separators can be used by a paver in order to bracket the solution set $\mathbb{X}$ between two subpavings $\mathbb{X}^-$ and $\mathbb{X}^+$.

A. Contractors

An *interval* of $\mathbb{R}$ is a closed connected set of $\mathbb{R}$. A box $[\mathbf{x}]$ of $\mathbb{R}^n$ is the Cartesian product of n intervals. The set of all boxes of $\mathbb{R}^n$ is denoted by $\mathbb{IR}^n$. A *contractor* $\mathcal{C}$ is an operator $\mathbb{IR}^n \mapsto \mathbb{IR}^n$ such that

$$\begin{array}{ll} \mathcal{C}([\mathbf{x}]) \subset [\mathbf{x}] & \text{(contractance)} \\ [\mathbf{x}] \subset [\mathbf{y}] \;\Rightarrow\; \mathcal{C}([\mathbf{x}]) \subset \mathcal{C}([\mathbf{y}]) & \text{(monotonicity)} \end{array} \tag{2}$$

We define the inclusion between two contractors $\mathcal{C}_1$ and $\mathcal{C}_2$ as follows

$$\mathcal{C}_1 \subset \mathcal{C}_2 \Leftrightarrow \forall [\mathbf{x}] \in \mathbb{IR}^n,\, \mathcal{C}_1([\mathbf{x}]) \subset \mathcal{C}_2([\mathbf{x}]). \tag{3}$$

A set $\mathbb{S}$ is *consistent* with the contractor $\mathcal{C}$ (we will write $\mathbb{S} \sim \mathcal{C}$) if for all $[\mathbf{x}]$, we have

$$\mathcal{C}([\mathbf{x}]) \cap \mathbb{S} = [\mathbf{x}] \cap \mathbb{S}. \tag{4}$$

Two contractors $\mathcal{C}$ and $\mathcal{C}_1$ are *consistent* each other (we will write $\mathcal{C} \sim \mathcal{C}_1$) if for any set $\mathbb{S}$, we have

$$\mathbb{S} \sim \mathcal{C} \Leftrightarrow \mathbb{S} \sim \mathcal{C}_1. \tag{5}$$

A contractor $\mathcal{C}$ is *minimal* if for any other contractor $\mathcal{C}_1$, we have the following implication

$$\mathcal{C} \sim \mathcal{C}_1 \Rightarrow \mathcal{C} \subset \mathcal{C}_1. \tag{6}$$

We define the *negation* $\neg\mathcal{C}$ of a contractor $\mathcal{C}$ as follows

$$\neg\mathcal{C}\left([\mathbf{x}]\right) = \left\{\mathbf{x} \in [\mathbf{x}] \mid \mathbf{x} \notin \mathcal{C}\left([\mathbf{x}]\right)\right\}. \tag{7}$$

Note that $\neg\mathcal{C}\left([\mathbf{x}]\right)$ is not a box in general, but a union of boxes.

B. Separators

A *separator* $\mathcal{S}$ is pair of contractors $\left\{\mathcal{S}^{\text{in}}, \mathcal{S}^{\text{out}}\right\}$ such that, for all $[\mathbf{x}] \in \mathbb{IR}^n$, we have

$$\mathcal{S}^{\text{in}}([\mathbf{x}]) \cup \mathcal{S}^{\text{out}}([\mathbf{x}]) = [\mathbf{x}] \qquad \text{(complementarity)}. \tag{8}$$

A set $\mathbb{S}$ is *consistent* with the separator $\mathcal{S}$ (we will write $\mathbb{S} \sim \mathcal{S}$), if

$$\mathbb{S} \sim \mathcal{S}^{\text{out}} \text{ and } \overline{\mathbb{S}} \sim \mathcal{S}^{\text{in}}. \tag{9}$$

where $\overline{\mathbb{S}} = \{\mathbf{x} \mid \mathbf{x} \notin \mathbb{S}\}$. We define the *remainder* of a separator $\mathcal{S}$ as

$$\partial\mathcal{S}([\mathbf{x}]) = \mathcal{S}^{\text{in}}([\mathbf{x}]) \cap \mathcal{S}^{\text{out}}([\mathbf{x}]). \tag{10}$$

Note that the remainder is a contractor and not a separator.

Example. Figure 1 represents a set $\mathbb{S}$, an outer contractor $\mathcal{S}^{\text{out}}$, an inner contractor $\mathcal{S}^{\text{in}}$ and a boundary contractor $\partial\mathcal{S}$. The pair $\{\mathcal{S}^{\text{in}}, \mathcal{S}^{\text{out}}\}$ corresponds to a separator associated with $\mathbb{S}$ (painted grey on the figure). Note that $\mathcal{S}^{\text{in}}$ is only allowed to eliminate the part of the research space which is inside $\mathbb{S}$ whereas $\mathcal{S}^{\text{out}}$ only eliminates the part which is outside $\mathbb{S}$, *i.e.*,

$$\left\{\begin{array}{lll} \mathcal{S}^{\text{in}}\left([\mathbf{x}]\right) \cap \overline{\mathbb{S}} & = & [\mathbf{x}] \cap \overline{\mathbb{S}} \\ \mathcal{S}^{\text{out}}\left([\mathbf{x}]\right) \cap \mathbb{S} & = & [\mathbf{x}] \cap \mathbb{S}. \end{array}\right.$$

For a given box $[\mathbf{x}]$, it is trivial to show that $\neg\mathcal{S}^{\text{in}}([\mathbf{x}])$, $\neg\mathcal{S}^{\text{out}}([\mathbf{x}])$ and $\partial\mathcal{S}([\mathbf{x}])$ cover $[\mathbf{x}]$, i.e.,

$$\neg\mathcal{S}^{\text{in}}([\mathbf{x}]) \cup \neg\mathcal{S}^{\text{out}}([\mathbf{x}]) \cup \partial\mathcal{S}([\mathbf{x}]) = [\mathbf{x}]\,. \tag{11}$$

Moreover, they do not overlap each other. By applying the De Morgan rule on the complementarity property (8), we get that $\neg\mathcal{S}^{\text{in}}([\mathbf{x}]) \cap \neg\mathcal{S}^{\text{out}}([\mathbf{x}]) = \emptyset$. The action of a separator will thus be depicted using the ($\neg\mathcal{S}^{\text{in}}$, $\neg\mathcal{S}^{\text{out}}, \partial\mathcal{S}$) representation as illustrated by Figure 2. But, for simplicity, the computations will be done using a ($\mathcal{S}^{\text{in}}, \mathcal{S}^{\text{out}}$) representation.

Consider two *separators* $\mathcal{S}_1$ and $\mathcal{S}_2$. We define the inclusion between separators as follows

$$\mathcal{S}_1 \subset \mathcal{S}_2 \Leftrightarrow \mathcal{S}_1^{\text{in}} \subset \mathcal{S}_2^{\text{in}} \text{ and } \mathcal{S}_1^{\text{out}} \subset \mathcal{S}_2^{\text{out}}. \tag{12}$$

A separator $\mathcal{S}$ is *minimal* if

$$\mathcal{S}_1 \subset \mathcal{S} \Rightarrow \mathcal{S}_1 = \mathcal{S}. \tag{13}$$

It is trivial to check that $\mathcal{S}$ is minimal if and only if the two contractors $\mathcal{S}^{\text{in}}$ and $\mathcal{S}^{\text{out}}$ are both minimal.

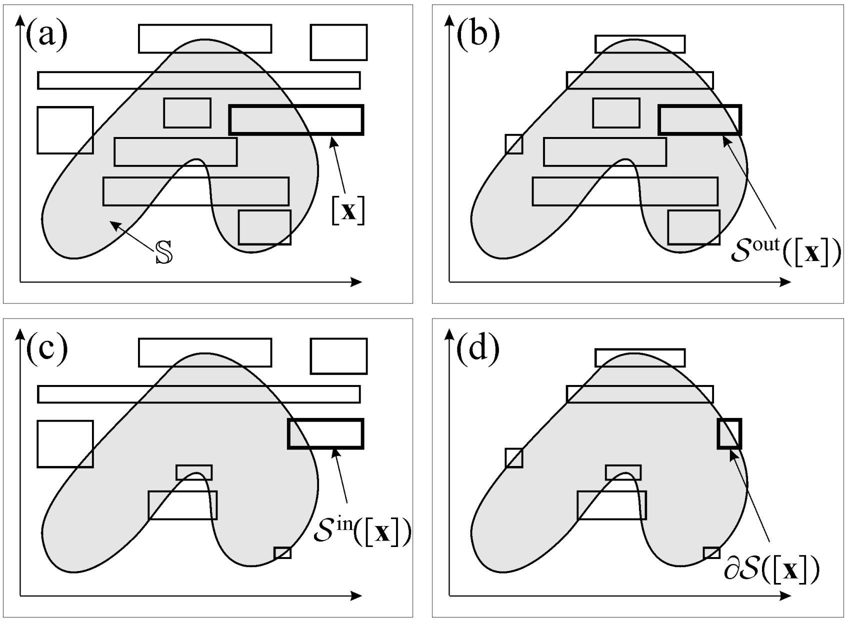
(a)
[x]
𝕊
(b)
$\mathcal{S}^{\text{out}}([\mathbf{x}])$
(c)
$\mathcal{S}^{\text{in}}([\mathbf{x}])$
(d)
$\partial\mathcal{S}([\mathbf{x}])$

Fig. 1. A separator $\{\mathcal{S}^{\text{in}}, \mathcal{S}^{\text{out}}\}$ is a pair of two contractors.

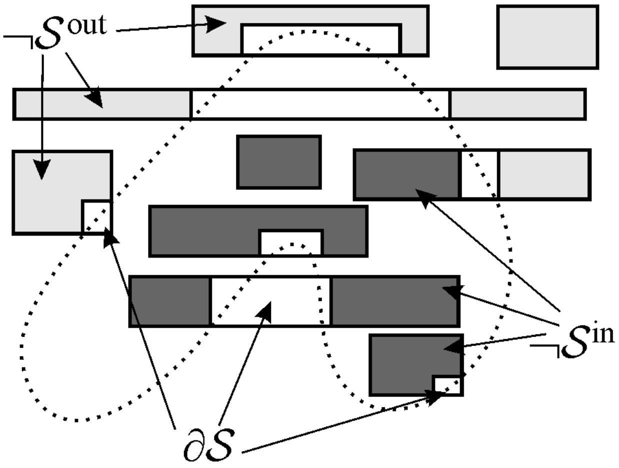

Fig. 2. Illustration of a separator. The light (resp. dark) grey area corresponds to the part of the initial box that is eliminated by $\mathcal{S}^{\text{out}}$ (resp. $\mathcal{S}^{\text{in}}$).

C. Paver

A *paver* is a branch-and-bound algorithm which calls the separator $\mathcal{S}$ to classify part of the search space inside or outside the solution set $\mathbb{X}$ associated with $\mathcal{S}$. The algorithm is given in the table below. Step 1 initializes a list $\mathcal{L}$ containing all boxes to be studied. Step 2 takes one box $[\mathbf{x}]$ in $\mathcal{L}$. At Step 3, the separator $\mathcal{S}$ is then called to contract $[\mathbf{x}]$ into two boxes $[\mathbf{x}^{\text{in}}]$ and $[\mathbf{x}^{\text{out}}]$. Step 4 stores $\neg\mathcal{S}^{\text{in}}([\mathbf{x}])$, the part of $[\mathbf{x}]$ is proved to be inside $\mathbb{X}$, into $\mathbb{X}^-$ and also into $\mathbb{X}^+$. Step 5 computes $\partial\mathcal{S}([\mathbf{x}])$ by intersecting $[\mathbf{x}^{\text{out}}]$ and $[\mathbf{x}^{\text{in}}]$. If this box is too small (i.e. with a width smaller than ε), it is store inside $\mathbb{X}^+$ and will not be studied anymore. Otherwise, it is bisected at Step 7 and stored into $\mathcal{L}$ waiting to be processed. After completion

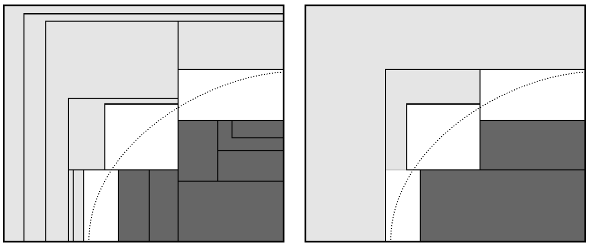

Fig. 3. Simplication of the paving generated by the paver; left: before simplication, right: after.

of the algorithm, we have the enclosure $\mathbb{X}^- \subset \mathbb{X} \subset \mathbb{X}^+$.

```
Algorithm PAVER(in: $[\mathbf{x}], \mathcal{S}$; out: $\mathbb{X}^-$, $\mathbb{X}^+$)
1  $\mathcal{L} := \{[\mathbf{x}]\}$;
2  Pull $[\mathbf{x}]$ from $\mathcal{L}$;
3  $\{[\mathbf{x}^{\text{in}}], [\mathbf{x}^{\text{out}}]\} = \mathcal{S}([\mathbf{x}])$;
4  Store $[\mathbf{x}] \backslash [\mathbf{x}^{\text{in}}]$ into $\mathbb{X}^-$ and also into $\mathbb{X}^+$;
5  $[\mathbf{x}] = [\mathbf{x}^{\text{in}}] \cap [\mathbf{x}^{\text{out}}]$;
6  If $w([\mathbf{x}]) < \varepsilon$, then store $[\mathbf{x}]$ in $\mathbb{X}^+$,
7  Else bisect $[\mathbf{x}]$ and push into $\mathcal{L}$ the two resulting boxes
8  If $\mathcal{L} \neq \emptyset$, go to 2.
```

Remark. For the implementation, the resulting paving can be represented by a binary tree (i.e., each node has two sons or it is a leaf). The binary tree is said to be *minimal* if for any node i_1 (not the root) with brother i_2 and father j, we have

$$\begin{cases} \text{(i)} & [\mathbf{x}^{\text{in}}](i_1) \neq \emptyset,\ [\mathbf{x}^{\text{out}}](i_1) \neq \emptyset \\ \text{(ii)} & [\mathbf{x}^{\text{in}}](j) \cap [\mathbf{x}^{\text{out}}](j) = \left([\mathbf{x}^{\text{in}}](i_1) \cap [\mathbf{x}^{\text{out}}](i_1)\right) \sqcup \left([\mathbf{x}^{\text{in}}](i_2) \cap [\mathbf{x}^{\text{out}}](i_2)\right) \end{cases} \tag{14}$$

where $\sqcup$ denotes the *interval hull* (i.e., the smallest box which encloses the union). From a given tree generated by the paver, it is possible to simplify it into an minimal tree without changing the approximation for $\mathbb{X}$. This simplification can be done by scanning all nodes of the tree upward from the leaves to the root. If for a given node, properties (i) or (ii) of (14) is not satisfied, then the two brother nodes i_1, i_2 with the father j are reunited into a single node. The procedure has a complexity which is linear with respect to the number of nodes of the tree. It will allow us to drastically reduce the number of elements of the subpaving. An illustration is provided on Figure 3.

III. ALGEBRA

The algebra for separators is a direct extension of contractor algebra [5]. The main difference is that contractor algebra does not allow any non monotonic (or decreasing) operation. It means that if a contractor $\mathcal{C}$ is defined by an expression $\mathcal{E}$ of other contractors $\mathcal{C}_i$ then we always have

$$\forall i, \mathcal{C}_i \subset \mathcal{C}_i' \Rightarrow \mathcal{E}\left(\mathcal{C}_1, \mathcal{C}_2, \dots\right) \subset \mathcal{E}\left(\mathcal{C}_1', \mathcal{C}_2', \dots\right). \tag{15}$$

As a consequence the complementary $\overline{\mathcal{C}}$ of a contractor $\mathcal{C}$ or the restriction $\mathcal{C}_1 \backslash \mathcal{C}_2$ of two contractors $\mathcal{C}_1, \mathcal{C}_2$ (which both correspond to non monotonic operations) cannot be defined. The main advantage of separators is that it extends the operations allowed for contractors to non monotonic expressions. Let us now define some operations for separator. If $\mathcal{S} = \left\{\mathcal{S}^{\text{in}}, \mathcal{S}^{\text{out}}\right\}$ is a separator, we define the *complement* as

$$\overline{\mathcal{S}} = \left\{\mathcal{S}^{\text{out}}, \mathcal{S}^{\text{in}}\right\}. \tag{16}$$

We define the *exponentiation* of a separator $\mathcal{S} = \left\{\mathcal{S}^{\text{in}}, \mathcal{S}^{\text{out}}\right\}$ by induction as follows:

$$\begin{array}{ll} \mathcal{S}^0 & = \{\top, \top\} \\ \mathcal{S}^{k+1} & = \left\{\neg \mathcal{S}^{k\ \text{out}} \sqcup (\mathcal{S}^{\text{in}} \circ \partial \mathcal{S}^k), \neg \mathcal{S}^{k\ \text{in}} \sqcup (\mathcal{S}^{\text{out}} \circ \partial \mathcal{S}^k)\right\}. \end{array} \tag{17}$$

The negation $\neg$ is defined by (7), and $\top$ is the identity contractor (i.e. $\top([\mathbf{x}]) = [\mathbf{x}]$). Note that

$$\begin{array}{rcl} \mathcal{S}^1 & = & \{ \quad \neg(\mathcal{S}^0)^{\text{out}} \sqcup \mathcal{S}^{\text{in}} \circ \left(\left(\mathcal{S}^0\right)^{\text{in}} \cap \left(\mathcal{S}^0\right)^{\text{out}}\right), \\ & & \quad \neg(\mathcal{S}^0)^{\text{in}} \sqcup \mathcal{S}^{\text{out}} \circ \left(\left(\mathcal{S}^0\right)^{\text{in}} \cap \left(\mathcal{S}^0\right)^{\text{out}}\right) \quad \} \\ & = & \left\{\neg \top \sqcup \mathcal{S}^{\text{in}} \circ \top \cap \top), \neg \top \sqcup \mathcal{S}^{\text{out}} \circ \top \cap \top)\right\} \\ & = & \left\{\mathcal{S}^{\text{in}}, \mathcal{S}^{\text{out}})\right\} = \mathcal{S}. \end{array}$$

The role of the exponentiation is to produce more efficient separators by using fixed point methods. If $\mathcal{S}_i = \left\{\mathcal{S}_i^{\text{in}}, \mathcal{S}_i^{\text{out}}\right\}, i \in \{1, 2, \dots, m\}$ are m separators, we define

$$\begin{array}{rcll}
\mathcal{S}_1 \cap \mathcal{S}_2 & = & \left\{\mathcal{S}_1^{\text{in}} \cup \mathcal{S}_2^{\text{in}}, \mathcal{S}_1^{\text{out}} \cap \mathcal{S}_2^{\text{out}}\right\} & \text{(intersection)} \\
\mathcal{S}_1 \cup \mathcal{S}_2 & = & \left\{\mathcal{S}_1^{\text{in}} \cap \mathcal{S}_2^{\text{in}}, \mathcal{S}_1^{\text{out}} \cup \mathcal{S}_2^{\text{out}}\right\} & \text{(union)} \\
\bigcap^{\{q\}} \mathcal{S}_i & = & \left\{ \bigcap^{\{m-q-1\}} \mathcal{S}_i^{\text{in}}, \bigcap^{\{q\}} \mathcal{S}_i^{\text{out}} \right\} & \text{(relaxed intersection)} \\
\mathcal{S}_1 \backslash \mathcal{S}_2 & = & \mathcal{S}_1 \cap \overline{\mathcal{S}_2}. & \text{(difference)}
\end{array} \tag{18}$$

The q-relaxed intersection [17] of m sets corresponds to the set of all elements which belongs to at least $m - q$ of these sets. It is used for robust bounded-error estimation [6], for localization [21], for probabilistic estimation [18] or for certified calibration of robots [9].

Theorem 1. If $\mathbb{S}_i$ are m sets of $\mathbb{R}^n$, we have

$$\begin{array}{cccc}
\text{(i)} & \mathbb{S}_1 \cap \mathbb{S}_2 & \sim & \mathcal{S}_1 \cap \mathcal{S}_2 \\
\text{(ii)} & \mathbb{S}_1 \cup \mathbb{S}_2 & \sim & \mathcal{S}_1 \cup \mathcal{S}_2 \\
\text{(iii)} & \overline{\mathbb{S}_i} & \sim & \overline{\mathcal{S}_i} \\
\text{(iv)} & \mathbb{S}_i & \sim & \mathcal{S}_i^k,\ k \geq 0 \\
\text{(v)} & \bigcap^{\{q\}} \mathbb{S}_i & \sim & \bigcap^{\{q\}} \mathcal{S}_i \\
\text{(vi)} & \mathbb{S}_1 \backslash \mathbb{S}_2 & \sim & \mathcal{S}_1 \backslash \mathcal{S}_2.
\end{array} \tag{19}$$

Proof. Let us prove property (i). We have

$$\begin{array}{cll}
 & \mathbb{S}_1 \sim \{\mathcal{S}_1^{\text{in}}, \mathcal{S}_1^{\text{out}}\} \text{ and } \mathbb{S}_2 \sim \{\mathcal{S}_2^{\text{in}}, \mathcal{S}_2^{\text{out}}\} & \\
\Leftrightarrow & \mathbb{S}_1 \sim \mathcal{S}_1^{\text{out}},\ \overline{\mathbb{S}_1} \sim \mathcal{S}_1^{\text{in}}, \mathbb{S}_2 \sim \mathcal{S}_2^{\text{out}},\ \overline{\mathbb{S}_2} \sim \mathcal{S}_2^{\text{in}} & \text{(see 9)} \\
\Rightarrow & \mathbb{S}_1 \cap \mathbb{S}_2 \sim \mathcal{S}_1^{\text{out}} \cap \mathcal{S}_2^{\text{out}},\ \overline{\mathbb{S}_1} \cup \overline{\mathbb{S}_2} \sim \mathcal{S}_1^{\text{in}} \cup \mathcal{S}_2^{\text{in}} & \text{(see [5])} \\
\Leftrightarrow & \mathbb{S}_1 \cap \mathbb{S}_2 \sim \mathcal{S}_1^{\text{out}} \cap \mathcal{S}_2^{\text{out}},\ \overline{\mathbb{S}_1 \cap \mathbb{S}_2} \sim \mathcal{S}_1^{\text{in}} \cup \mathcal{S}_2^{\text{in}} & \text{(De Morgan)} \\
\Leftrightarrow & \mathbb{S}_1 \cap \mathbb{S}_2 \sim \left\{\mathcal{S}_1^{\text{in}} \cup \mathcal{S}_2^{\text{in}}, \mathcal{S}_1^{\text{out}} \cap \mathcal{S}_2^{\text{out}}\right\} & \text{(see (9))} \\
\Leftrightarrow & \mathbb{S}_1 \cap \mathbb{S}_2 \sim \mathcal{S}_1 \cap \mathcal{S}_2 & \text{(see 18)}
\end{array}$$

The proof of other all other properties are similar. ■

IV. Inversion of separators

The inverse of a set $\mathbb{Y} \subset \mathbb{R}^n$ by a function $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ is defined as

$$\mathbb{X} = \mathbf{f}^{-1}(\mathbb{Y}) = \{\mathbf{x} \mid \mathbf{f}(\mathbf{x}) \in \mathbb{Y}\}. \tag{20}$$

The function $\mathbf{f}$ can be a translation, rotation, homothety, projection, or any other function. If $\mathcal{C}_\mathbb{Y}$ is a contractor for $\mathbb{Y}$, a contractor $\mathcal{C}_\mathbb{X}$ for $\mathbb{X}$ can be defined using a generalization of the forward-backward contractor as shown in [3]. The contractor $\mathcal{C}_\mathbb{X}$ is called the inverse of $\mathcal{C}_\mathbb{Y}$ by $\mathbf{f}$ and we write $\mathcal{C}_\mathbb{X} = \mathbf{f}^{-1}(\mathcal{C}_\mathbb{Y})$. If $\mathcal{S}_\mathbb{Y}$ is a separator associated with a set $\mathbb{Y}$, we define the inverse of the separator $\mathcal{S}_\mathbb{Y}$ as follows

$$\mathbf{f}^{-1}(\mathcal{S}_\mathbb{Y}) = \left\{\mathbf{f}^{-1}(\mathcal{S}_\mathbb{Y}^{\text{in}}), \mathbf{f}^{-1}(\mathcal{S}_\mathbb{Y}^{\text{out}})\right\}. \tag{21}$$

Theorem 2. The separator $\mathbf{f}^{-1}(\mathcal{S}_\mathbb{Y})$ is a separator associated with the set $\mathbb{X} = \mathbf{f}^{-1}(\mathbb{Y})$, i.e.,

$$\mathbf{f}^{-1}(\mathbb{Y}) \sim \mathbf{f}^{-1}(\mathcal{S}_\mathbb{Y}). \tag{22}$$

Proof. We have

$$\begin{array}{cll}
 & \mathbb{Y} \sim \{\mathcal{S}_\mathbb{Y}^{\text{in}}, \mathcal{S}_\mathbb{Y}^{\text{out}}\} & \\
\Leftrightarrow & \mathbb{Y} \sim \mathcal{S}_\mathbb{Y}^{\text{out}},\ \overline{\mathbb{Y}} \sim \mathcal{S}_\mathbb{Y}^{\text{in}} & \text{(see 9)} \\
\Rightarrow & \mathbf{f}^{-1}(\mathbb{Y}) \sim \mathbf{f}^{-1}(\mathcal{S}_\mathbb{Y}^{\text{out}}),\ \mathbf{f}^{-1}(\overline{\mathbb{Y}}) \sim \mathbf{f}^{-1}(\mathcal{S}_\mathbb{Y}^{\text{in}}) & \text{(see [3])} \\
\Leftrightarrow & \mathbf{f}^{-1}(\mathbb{Y}) \sim \mathbf{f}^{-1}(\mathcal{S}_\mathbb{Y}^{\text{out}}),\ \overline{\mathbf{f}^{-1}(\mathbb{Y})} \sim \mathbf{f}^{-1}(\mathcal{S}_\mathbb{Y}^{\text{in}}) & \\
\Leftrightarrow & \mathbf{f}^{-1}(\mathbb{Y}) \sim \left\{\mathbf{f}^{-1}(\mathcal{S}_\mathbb{Y}^{\text{in}}), \mathbf{f}^{-1}(\mathcal{S}_\mathbb{Y}^{\text{out}})\right\} & \text{(see (9))} \\
\Leftrightarrow & \mathbf{f}^{-1}(\mathbb{Y}) \sim \mathbf{f}^{-1}(\mathcal{S}_\mathbb{Y}) & \text{(see 22)}
\end{array} \tag{23}$$

which terminates the proof. ■

Example. Consider the following function

$$f(x_1, x_2) = x_1 + 2x_2. \tag{24}$$

Here, the function is linear, but it is not required by the method. The separator $f^{-1}(\mathcal{S}_{\mathbb{Y}})$ corresponds to the following algorithm.

```
Separator $\mathcal{S}_{\mathbb{X}}$(in: $[\mathbf{x}], \mathcal{S}_{\mathbb{Y}}$; out: $\{[\mathbf{x}^{\text{in}}], [\mathbf{x}^{\text{out}}]\}$)
1   $[y] = [x_1] + 2[x_2]$;
2   $[y^{\text{in}}] = \mathcal{S}_{\mathbb{Y}}^{\text{in}}([y])$;  $[y^{\text{out}}] = \mathcal{S}_{\mathbb{Y}}^{\text{out}}([y])$;
3   $[\mathbf{x}^{\text{in}}] = [\mathbf{x}]; [\mathbf{x}^{\text{out}}] = [\mathbf{x}]$;
4   $[x_1^{\text{in}}] = [x_1^{\text{in}}] \cap [y^{\text{in}}] - 2[x_2^{\text{in}}]$;
5   $[x_2^{\text{in}}] = [x_2^{\text{in}}] \cap \frac{1}{2}\left([y^{\text{in}}] - [x_1^{\text{in}}]\right)$;
6   $[x_1^{\text{out}}] = [x_1^{\text{out}}] \cap [y^{\text{out}}] - 2[x_2^{\text{out}}]$;
7   $[x_2^{\text{out}}] = [x_2^{\text{out}}] \cap \frac{1}{2}\left([y^{\text{out}}] - [x_1^{\text{out}}]\right)$.
```

The inputs of the algorithm are the box $[\mathbf{x}]$ and the separator $\mathcal{S}_{\mathbb{Y}}$. Steps 1 corresponds to the forward propagation and computes an interval $[y]$. At Step 2, the separator $\mathcal{S}_{\mathbb{Y}}$ contracts $[y]$ twice with respect to the inner and the outer contractors associated with $\mathbb{Y}$. Step 3 initializes the boxes to be returned by the algorithm. Steps 4 and 5 correspond to the backward propagation associated with Step 1 from $[\mathbf{y}^{\text{in}}]$ toward $[\mathbf{x}^{\text{in}}]$. Step 6 and 7 are also associated with Step 1 but now from $[\mathbf{y}^{\text{out}}]$ toward $[\mathbf{x}^{\text{out}}]$. Note that here, we have only one forward propagation (Step 1) and two backward propagations (Step 4 to 7). Using a classical contractor technique, two forward propagations would have been needed: one for the inner contractor and one for the outer contractor. Note also that we could have a single backward propagation by limiting the backward propagation to the boundary box $[\partial y] = [y^{\text{in}}] \cap [y^{\text{out}}]$. Now, it remains to rebuild the two boxes $[\mathbf{x}^{\text{in}}], [\mathbf{x}^{\text{out}}]$ from the unique box $[\partial \mathbf{x}]$ that has been obtained after the backward propagation.

V. Atomic separators

Complex separators are built from operations and compositions of atomic separators. Two different types of atomic separators can be built: those that are associated with a mathematical equation and those that are associated with a database.

A. *Equation-based separators*

A typical situation where we can take advantage of contractor techniques is when we want to built a separator associated with a set defined by a mathematical expression. For instance, if

$$\mathbb{X} = \{f(\mathbf{x}) \leq 0\}, \tag{25}$$

we can easily build a contractor $\mathcal{S}^{\text{out}}$ associated with the constraint $f(\mathbf{x}) \leq 0$ and a contractor $\mathcal{S}^{\text{in}}$ associated with the constraint $f(\mathbf{x}) \geq 0$. The pair $\{\mathcal{S}^{\text{in}}, \mathcal{S}^{\text{out}}\}$ is then a separator associated with $\mathbb{X}$.

B. *Database-based separators*

A contractor represents an information. This information can be an equation or an inequality, but it can also come from a database (for instance it can be the map of the surrounding environment or the shape of an object). In this situation, it is often more convenient to represent the corresponding set by its boundary. We now show on an example that the boundary representation may help to build minimal separators. The principle can be interpreted as the separator extension of the *map_contract* algorithm [32], [13], [10] that has been developed for contractors. Figure 4 represents a map (painted gray). The corresponding set $\mathbb{M}$ corresponding to the free space is painted gray. On subfigure (a), a box $[\mathbf{x}]$ is waiting for contractions. Denote by Δ_i the segments and arcs that make up the boundary $\partial\mathbb{M}$ of $\mathbb{M}$. Since the union of minimal contractors is minimal (see [31], Section 4.6.3), the minimal contractor for $\partial\mathbb{M}$ is

$$\partial\mathcal{S}([\mathbf{x}]) = \bigsqcup_i [[\mathbf{x}] \cap \Delta_i], \tag{26}$$

where $[[\mathbf{x}] \cap \Delta_i]$ is the smallest box enclosing $[\mathbf{x}] \cap \Delta_i$. Subfigure (b) represents the boxes $[[\mathbf{x}] \cap \Delta_i]$ and the hull of these boxes are on Subfigure (c). As a consequence, as illustrated by subfigure (d), we get the minimal separator by testing two points (for instance the two corners of $[\mathbf{x}]$ represented on Subfigure (c)) that have been eliminated by $\partial\mathcal{S}$.

Remark. If we use a pure contractor approach to compute $\mathcal{S}^{\text{in}}([\mathbf{x}])$ and $\mathcal{S}^{\text{out}}([\mathbf{x}])$ by taking into account that the corresponding set $\mathbb{S}$ can be defined by inequalities, the approximation will be very poor. It is also possible to build specific contractors focusing on the boundary but the computation of the boundary intersecting $[\mathbf{x}]$ will have to be performed twice: once for $\mathcal{S}^{\text{in}}$ and once for $\mathcal{S}^{\text{out}}$. Contrary to a contractors, a separator approach considers that a large part of the computation made by $\mathcal{S}^{\text{in}}$ and $\mathcal{S}^{\text{out}}$ are identical and should be factorized. ■

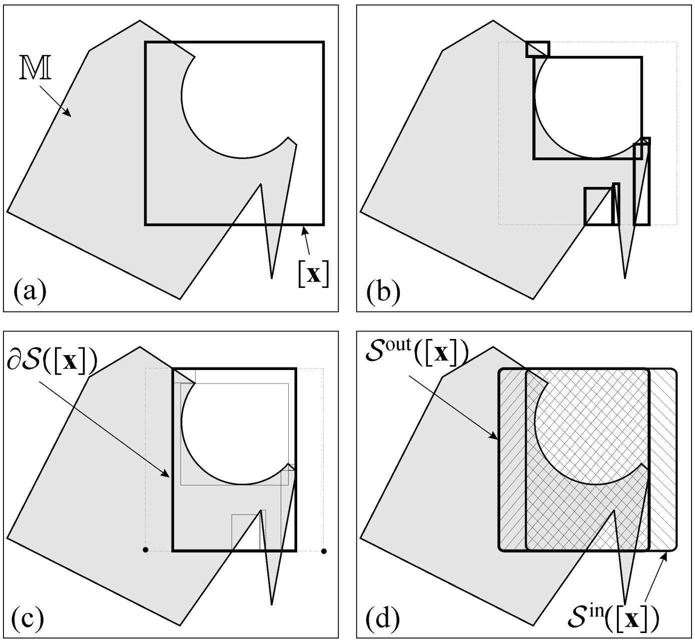
$\mathbb{M}$
$[\mathbf{x}]$
(a)
(b)
$\partial\mathcal{S}([\mathbf{x}])$
(c)
$\mathcal{S}^{\text{out}}([\mathbf{x}])$
(d)
$\mathcal{S}^{\text{in}}([\mathbf{x}])$

Fig. 4. The minimal separator associated with a map (painted gray) can be obtain using the contractor associated with the boundary of this map.

VI. Application to path planning

The goal of path planning is to find a collision-free path for a robot in a given space with obstacles. The issue of path planning in a known environment has been addressed since many years (see, *e.g.*, [22] [24] [27] [19]) and can easily be combined with set-membership techniques to take into account some uncertainties [2]. Most approaches are based on the concept of configuration space (*C-space*). Each coordinate of the C-space represents a degree of freedom of the object. An example of such robots are industrial robots which are kinematic chains in which adjacent links are connected by n prismatic or rotary joints, each with one degree of freedom. The positions and orientations of each link of the industrial robot can be characterized by n real numbers, which are the coordinates of a single n-dimensional point in the C-space [25]. The *feasible configuration space* $\mathbb{M}$ is the subset of the C-space corresponding to feasible configurations of the robot. Partitioning the C-space with subpavings makes it possible to solve the problem using graph algorithms [1], [15]. The objective of this section is to show how separators could solve the path planning problem on a simple example which is a 2D version of the wire loop game. This game involves a metal loop on a handle and a length of curved wire (see Figure 5). The player holds the loop in one hand and attempts to guide it along the curved wire without touching the loop to the wire. In our 2D version of this game the player is an articulated robot with two rotary joints and the loop is a segment. The curved wire corresponds to the boundary a set $\mathbb{Y}$ with an inside (grey in the figure) and an outside part. The length of the first and second arms are 4 and 2, respectively. The length of the loop is 1. The feasible configuration space is

$$\mathbb{M} = \{(x_1, x_2) \mid \mathbf{f}_2(\mathbf{x}) \in \mathbb{Y} \text{ and } \mathbf{f}_3(\mathbf{x}) \notin \mathbb{Y}\} = \mathbf{f}_2^{-1}(\mathbb{Y}) \cap \mathbf{f}_3^{-1}\left(\overline{\mathbb{Y}}\right),$$

where

$$\mathbf{f}_\ell(\mathbf{x}) = 4\begin{pmatrix} \cos x_1 \\ \sin x_1 \end{pmatrix} + \ell \begin{pmatrix} \cos(x_1 + x_2) \\ \sin(x_1 + x_2) \end{pmatrix}, \quad \ell \in \{2, 3\}. \tag{27}$$

If $\mathcal{S}_\mathbb{Y}$ is a separator associated with $\mathbb{Y}$ (built as explained in Section V-B), then a separator for $\mathbb{M}$ is

$$\mathcal{S}_\mathbb{M} = \mathbf{f}_2^{-1}(\mathcal{S}_\mathbb{Y}) \cap \mathbf{f}_3^{-1}\left(\overline{\mathcal{S}_\mathbb{Y}}\right). \tag{28}$$

The paver is able to approximate the feasible configuration space as illustrated by Figure 6. A graph-based method is able to compute a feasible path (see the black path of the figure) which corresponds to a solution of our wire loop game. The minimality of the resulting subpaving is illustrated by the fact that each patch of the subpaving intersects the boundary of the solution set. The corresponding motion in the world space is depicted on Figure 7.

Remark. For our example, the set to be characterized is given by $\mathbb{M} = \mathbf{f}_2^{-1}(\mathbb{Y}) \cap \mathbf{f}_3^{-1}\left(\overline{\mathbb{Y}}\right)$ and the separator is obtained by the same expression $\mathcal{S}_\mathbb{M} = \mathbf{f}_2^{-1}(\mathcal{S}_\mathbb{Y}) \cap \mathbf{f}_3^{-1}\left(\overline{\mathcal{S}_\mathbb{Y}}\right)$. If we separate this expression we get the contractor counterpart

$$\begin{cases} \mathcal{S}_\mathbb{M}^{\text{in}} & = & \mathbf{f}_2^{-1}(\mathcal{S}_\mathbb{Y}^{\text{in}}) \cup \mathbf{f}_3^{-1}(\mathcal{S}_\mathbb{Y}^{\text{out}}) \\ \mathcal{S}_\mathbb{M}^{\text{out}} & = & \mathbf{f}_2^{-1}(\mathcal{S}_\mathbb{Y}^{\text{out}}) \cap \mathbf{f}_3^{-1}(\mathcal{S}_\mathbb{Y}^{\text{in}}). \end{cases}$$

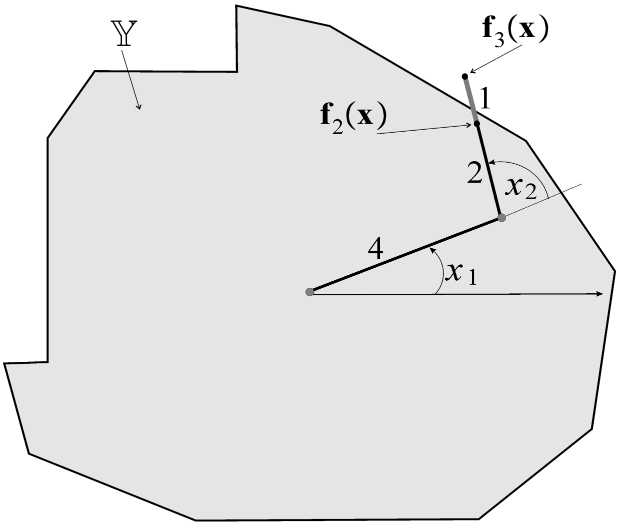

Fig. 5. Two-dimensional wire loop game. For a feasible configuration (x_1, x_2), as represented here, the gray segment should cross the wire. Is it possible to perform to complete feasible circular path?

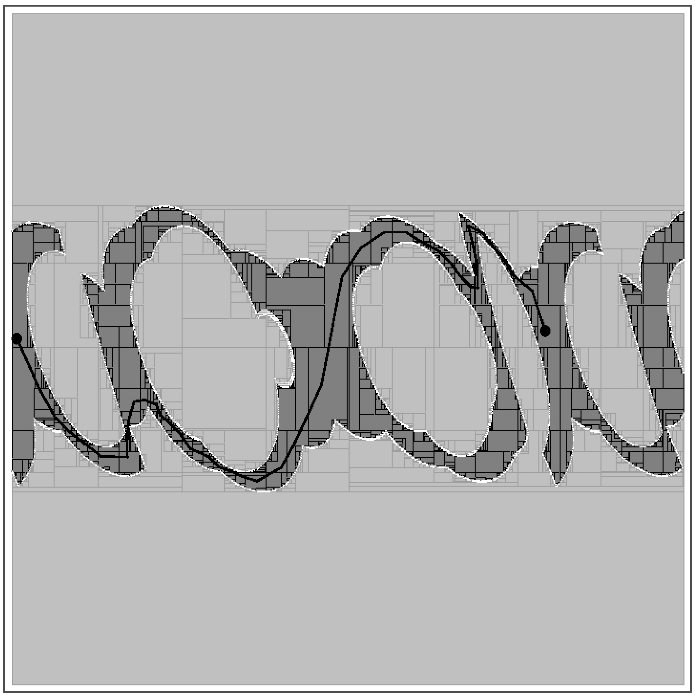

Fig. 6. Approximation of the feasible configuration space. A path, corresponding to one solution of the wire loop game, is represented.

The separator formalism thus makes it possible get directly the separator from the expression of the set we want to characterize. This was not possible using contractor algebra which does not allow any decreasing operation. With contractors, the user has to rewrite the complementary expression. Moreover, the separator algebra requires less computation since a part of the common computation that is made for $\mathcal{S}^{\text{in}}_{\mathbb{M}}$ and $\mathcal{S}^{\text{out}}_{\mathbb{M}}$ is factorized.■

VII. Conclusion

Contractor algebra is an efficient tool to compute with subsets of $\mathbb{R}^n$. To compute the union, the intersection or any other monotonic operations between sets, it suffices to apply the same operations on the contractors. Then, a paver with the corresponding contractor will provide a guaranteed approximation of the solution set. Now, due the external representation of sets they have, contractors cannot deal with decreasing operations such as the complement or the set difference. The main contribution of this paper is the introduction of *separators* which is a pair of two complementary contractors. Since the complementary operator or any other decreasing operation are now available, this contribution allows us to compute with sets in

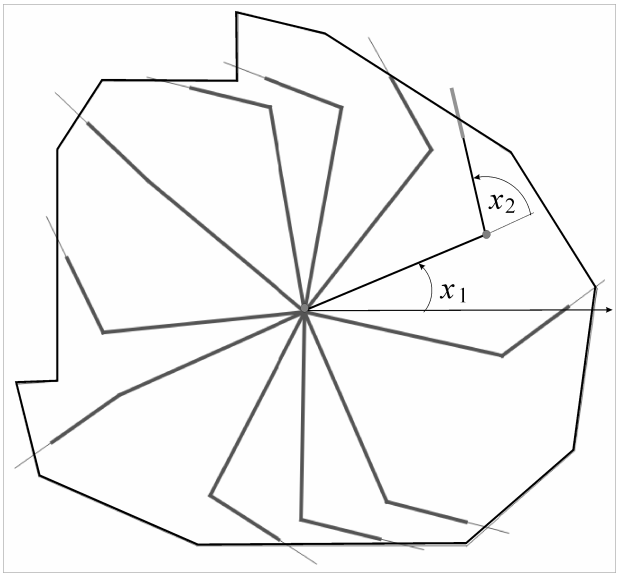

Fig. 7. Path of the robot in the world frame

a much more general way. The application to robotics is direct as soon as we want to compute inner and outer approximations of the solution set. As an illustration, we have considered a path planning problem where the inner and the outer approximations of the feasible configuration space are needed to find a feasible path.

All C++ codes associated with the test-case can be found at `www.ensta-bretagne.fr/jaulin/seppath.html`.

REFERENCES

[1] R. A. BROOKS AND T. LOZANO-PÉREZ. A subdivision algorithm in configuration space for findpath with rotation. *"IEEE Transactions on Systems Man and Cybernetics"* **15** (1985).

[2] N. CECCARELLI, M. D. MARCO, A. GARULLI, A. GIANNITRAPANI, AND A. VICINO. Set membership localization and map building for mobile robots. *Current Trends in Nonlinear Systems and Control Systems and Control: Foundations and Applications* pp. 289–308 (2006).

[3] G. CHABERT. A simple slam example with ibex. In "SWIM13", Brest (2013).

[4] G. CHABERT AND L. JAULIN. A Priori Error Analysis with Intervals. *SIAM Journal on Scientific Computing* **31**(3), 2214–2230 (2009).

[5] G. CHABERT AND L. JAULIN. Contractor Programming. *Artificial Intelligence* **173**, 1079–1100 (2009).

[6] E. COLLE AND GALERNE. Mobile robot localization by multiangulation using set inversion. *Robotics and Autonomous Systems* **61**(1), 39–48 (2013).

[7] D. DANEY, N. ANDREFF, G. CHABERT, AND Y. PAPEGAY. Interval Method for Calibration of Parallel Robots : Vision-based Experiments. *Mechanism and Machine Theory, Elsevier* **41**, 926–944 (2006).

[8] O. DIDRIT, L. JAULIN, AND E. WALTER. Guaranteed analysis and optimization of parametric systems, with application to their stability degree. In "Proceedings of 3rd European Control Conference", pp. 1412–1417, Rome (1995).

[9] J. A. DIT SANDRETTO, G. TROMBETTONI, D. DANEY, AND G. CHABERT. Certified calibration of a cable-driven robot using interval contractor programming. In F. THOMAS AND A. P. GRACIA, editors, "Computational Kinematics, Mechanisms and Machine Science", Spinger (2014).

[10] V. DREVELLE AND P. BONNIFAIT. Reliable positioning domain computation for urban navigation. *IEEE Intelligent Transportation Systems Magazine* **5**(3), 21–29 (2013).

[11] A. GNING AND P. BONNIFAIT. Constraints propagation techniques on intervals for a guaranteed localization using redundant data. *Automatica* **42**(7), 1167–1175 (2006).

[12] A. GNING, B. RISTIC, L. MIHAYLOVA, AND F. ABDALLAH. An Introduction to Box Particle Filtering. *IEEE Signal Processing Magazine* **30**(1), 166–171 (2013).

[13] R. GUYONNEAU, S. LAGRANGE, AND L. HARDOUIN. A visibility information for multi-robot localization. In "IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)" (2013).

[14] D. HENRION, J. LASSERRE, AND J. LOFBERG. Gloptipoly 3: moments, optimization and semidefinite programming. *Optimization Methods and Software* **24**(4–5), 761–779 (2009).

[15] L. JAULIN. Path Planning Using Intervals and Graphs. *Reliable Computing* **7**(1), 1–15 (2001).

[16] L. JAULIN, M. KIEFFER, O. DIDRIT, AND E. WALTER. "Applied Interval Analysis, with Examples in Parameter and State Estimation, Robust Control and Robotics". Springer-Verlag, London (2001).

[17] L. JAULIN AND E. WALTER. Guaranteed robust nonlinear minimax estimation. *IEEE Transaction on Automatic Control* **47**(11), 1857–1864 (2002).

[18] M. KIEFFER AND E. WALTER. Guaranteed Characterization of Exact non-Asymptotic Confidence Regions as Defined by LSCR and SPS. *Automatica* **50**(2), 507–512 (2014).

[19] D. KODITSCHEK. Exact robot navigation by means of potential functions: some topological considerations. In ""Proceedings of the IEEE International Conference on Robotics and Automation"", pp. 1–6, Raleigh (1987).

[20] V. KREINOVICH, A. LAKEYEV, J. ROHN, AND P. KAHL. Computational complexity and feasibility of data processing and interval computations. *Reliable Computing* **4**(4), 405–409 (1997).

[21] M. LANGERWISCH AND B. WAGNER. Guaranteed mobile robot tracking using robust interval constraint propagation. *Intelligent Robotics and Applications* (2012).

[22] J. P. LAUMOND. Feasible trajectories for mobile robots with kinematic and environment constraints. In "Proceedings of the International Conference on Intelligent Autonomous Systems", pp. 246–354, Amsterdam, the Netherlands (1986).

[23] O. Lévêque, L. Jaulin, D. Meizel, and E. Walter. Vehicule localization from inaccurate telemetric data: a set inversion approach. In “Proceedings of 5th IFAC Symposium on Robot Control SY.RO.CO.’97”, vol. 1, pp. 179–186, Nantes, France (1997).

[24] T. Lozano-Pérez. Automatic planning of manipulator transfer movements. *"IEEE Transactions on Systems Man and Cybernetics"* **11**(10), 681–698 (1981).

[25] T. Lozano-Pérez. Spatial planning: A configuration space approach. *IEEE Transactions on Computers* **32**(2), 108–120 (1983).

[26] R. E. Moore. “Interval Analysis”. Prentice-Hall, Englewood Cliffs, NJ (1966).

[27] C. O’Dunlaing and C. K. Yap. A retraction method for planning the motion of a disc. *Journal of Algorithms* **6**(1), 104–111 (1982).

[28] J. Porta, J. Cortes, L. Ros, and F. Thomas. A Space Decomposition Method for Path Planning of Loop Linkages. In “Proceedings of International Conference on Intelligent Robots and Systems, IROS”, pp. 1882–1888 (2007).

[29] N. Ramdani and P.Poignet. Robust dynamic experimental identification of robots with set membership uncertainty. *IEEE/ASME Transactions on Mechatronics* **10(2)**, 253–256 (2005).

[30] D. Rokityanski and S. Veres. Application of ellipsoidal estimation to satellite control. *Mathematical and Computer Modelling of Dynamical Systems* **11**(2), 239–249 (2005).

[31] J. Sliwka. “Using set membership methods for robust underwater robot localization.” PhD dissertation, Universitée de Bretagne occidentale, Brest, France (2011).

[32] J. Sliwka, F. L. Bars, O. Reynet, and L. Jaulin. Using interval methods in the context of robust localization of underwater robots. In “NAFIPS 2011”, El Paso, USA (2011).

[33] S. Thrun, W. Bugard, and D. Fox. “Probabilistic Robotics”. MIT Press, Cambridge, M.A. (2005).

[34] W. Tucker. The Lorenz Attractor Exists. *Comptes Rendus de l’Académie des Sciences* **328**(12), 1197–1202 (1999).

[35] S. Veres and D. Wall. “Synergy and Duality of Identification and Control”. Taylor & Francis - Technology & Engineering (2000).

[36] J. Wan, J. Vehi, and N. Luo. A numerical approach to design control invariant sets for constrained nonlinear discrete-time systems with guaranteed optimality. *"Journal of Global Optimization"* (2009).