Acta Cybernetica **27** (2026) 453–473.

Boundary Approach to Characterize the Inner and Outer Approximation of the Image of a Disk*

Maël Godard[ab], Luc Jaulin[ac], and Damien Massé[de]

Abstract

Calculating directly the inner and outer approximation of the image of a set by a function can be challenging. Then, it is sometimes preferred to compute the image of the boundary of the set instead. However, boundary-based methods are subject to the apparition of fake boundaries in the image set. These fake boundaries add pessimism when characterizing the inner approximation of the image set. This paper then introduces the notion of Box Chains to simplify the detection and the suppression of the fake boundaries. The characterization of the inner and outer approximation of the image set in the case of a function from the unit disk $\mathcal{D}$ to $\mathbb{R}^2$ will be considered, with two examples.

Keywords: fake boundaries, box chains, boundary approach

1 Introduction

Calculating the image of a set by a function has many applications in robotics [7]: Image of a subpaving [9], estimation of the area covered by a sensor [3], state estimation [1, 12]. The interval analysis tools have then been shown to be efficient for roboticians [6, 11].

The methods to calculate the image of a set by a function can be divided into two subclasses. The set-based methods compute directly an outer, and sometimes an inner, approximation of the image of the whole set. They can rely on Interval Analysis for the operations on sets. The boundary-based methods compute instead the image of the boundary of the set, and from this image an inner and outer approximation of the image set can be deduced. These methods are lighter to calculate, but they are subject to the appearance of fake boundaries in the image set.

*This work was supported by the Defense Innovation Agency (AID) and the Brittany Region.
[a]Lab-STICC, ROBEX Team, ENSTA Bretagne, France
[b]E-mail: mael.godard@ensta-bretagne.org, ORCID: 0009-0002-2822-6215
[c]E-mail: luc.jaulin@ensta.fr, ORCID: 0000-0002-0938-0615
[d]Lab-STICC, ROBEX Team, UBO, France
[e]E-mail: damien.masse@univ-brest.fr, ORCID: 0000-0002-4485-8936

DOI: 10.14232/actacyb.314985

These fake boundaries are the result of the difference between the boundary of the image set, and the image of the boundary. These fake boundaries are parts of the image of the boundary that are in facts inside of the image set and should be classified as such. As they add an unwanted pessimism to the estimation of the inner approximation of the image set, these fake boundaries have to be removed.

This problematic has often been addressed as different applications favor different solutions. The papers [2, 16] both expose the fact that the result obtained when using interval methods depends on the way the problem itself is formulated. For example the union of adjacent but non-overlapping sets is subject to the appearance of fake boundaries, whereas a reformulation of the problem can avoid this issue. In [16] a first solution is presented by using a DNF (Disjonctive Normal Form) / CNF (Conjonctive Normal Form) to prevent the fake boundaries from appearing. The paper [2] is a direct continuation of this work and proposes to rely on Karnaugh maps [8] to highlight how an outer approximation of the boundary can be constructed part by part. From this outer approximation of the boundary without the fake boundaries, an inner and outer approximation of the whole set can be obtained.

The problematic of fake boundaries can also be seen from a topological point of view. In [3], the notion of winding number is used to detect the fake boundaries. More precisely, it extends the works from [15] and [5] in order to compute the winding number, i.e. the topological degree, of a whole area. In the context of a coverage measure, the winding number represents how many times a given area was seen. The boundaries then have an interval of winding number, for example $[0, 1]$ if the boundary is between an area seen one time and an unseen area. Fake boundaries can finally be spotted as their winding number does not contain 0.

The method presented in this article is a continuation of these works as the problem considered here can easily be seen as the computation of a covered or visible area without fake boundaries. While the works presented earlier are limited to $\mathbb{R}^2$, the work presented in this article aims to be usable in higher dimensions while remaining as computationally light as possible. To do so a first step is to delete the fake boundaries before computing an inner and an outer approximation of the image set.

Section 2 presents the notions and notations that will be used for the remainder of the article. Section 3 introduces the notion of Box Chains that will be used in the boundary simplification algorithm of Section 4. For this article two examples from the unit disk $\mathcal{D}$ to $\mathbb{R}^2$ will be considered. Finally Section 5 concludes the paper.

2 Problem presentation

2.1 Notations and definitions

Let $\mathcal{D}$ be the unit disk with $\mathcal{S}^1$, the unit circle, its contour. As computing the image of the whole disk $\mathcal{D}$ can give a result too pessimistic to be usable, it is often

prefered to compute the image of its boundary $\mathcal{S}^1$ instead. For this article we will consider functions of the form $\mathbf{f} : \mathcal{D} \to \mathbb{R}^2$ that satisfies :

- $\mathbf{f}$ is $\mathcal{C}^1$, we will note $\mathbf{J_f}$ its Jacobian function.
- $\mathbf{f}$ has no singularity on $\mathcal{D}$, i.e. the determinant of its Jacobian is never null.

Thanks to these assumptions, cases where the image of the set contains a cusp or a fold, as depicted Figure 1, will not be considered.

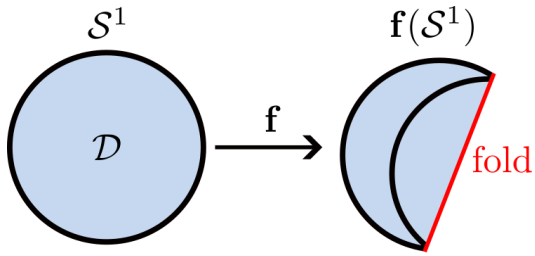


Figure 1: Fold in the image of the set

Considering a set X, we will further note ∂X its boundary. For the unit disk $\mathcal{D}$, its boundary $\partial\mathcal{D}$ is the unit circle $\mathcal{S}^1$.

We will then denote $\mathbf{f}(\mathcal{D})$ the image of the unit disk by the function $\mathbf{f}$ and $\partial\mathbf{f}(\mathcal{D})$ its boundary. Similarly, $\mathbf{f}(\mathcal{S}^1)$ will denote the image of the unit circle by the function $\mathbf{f}$.

As depicted in Figure 2a, we then have:

$$\partial\mathbf{f}(\mathcal{D}) \subseteq \mathbf{f}(\mathcal{S}^1) \tag{1}$$

Proof. Consider a function $\mathbf{f} : \mathcal{D} \to \mathbb{R}^2$. Under the assumptions cited earlier:

- $\mathbf{f}$ is $\mathcal{C}^1$.
- The determinant of its Jacobian is never null.

We can say that $\mathbf{f}$ is locally bijective at every point of $\mathcal{D}$ and is an open mapping. This mean that the image of an open set by $\mathbf{f}$ is an open set:

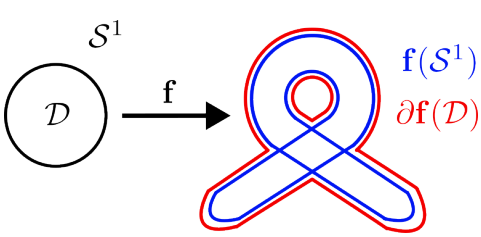
$$\forall \mathbf{x} \in \mathcal{D}, \mathbf{x} \notin \mathcal{S}^1 \implies \mathbf{f}(\mathbf{x}) \notin \partial\mathbf{f}(\mathcal{D}) \tag{2}$$

Then,

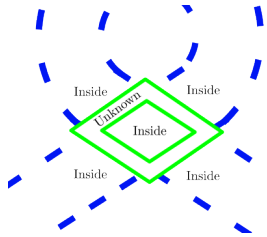
$$\forall \mathbf{y} \in \mathbb{R}^2, \mathbf{y} \in \partial\mathbf{f}(\mathcal{D}) \implies \exists \mathbf{x} \in \mathcal{S}^1, \mathbf{y} = \mathbf{f}(\mathbf{x}) \tag{3}$$

Finally, all the points of $\partial\mathbf{f}(\mathcal{D})$ are points of $\mathbf{f}(\mathcal{S}^1)$. □

Calculating the image of $\mathcal{S}^1$ instead of the image of $\mathcal{D}$ makes fake boundaries appear. These fake boundaries are defined by:



2(a) Fake boundaries



2(b) Unknown area due to the fake boundaries

Figure 2: Pessimism induced by the fake boundaries

$$FB_{\mathbf{f}}(\mathcal{D}) = \mathbf{f}(\mathcal{S}^1) \setminus \partial\mathbf{f}(\mathcal{D}) \tag{4}$$

When calculating the image of a disk, or any other closed outline, we often want to compute the inner and the outer approximation of the image set to verify if they satisfy a given condition (e.g. obstacle avoidance [4, 10]).

Given $\partial\mathbf{f}(\mathcal{D})$ it is possible to compute the inner and the outer approximation of the image set. However using $\mathbf{f}(\mathcal{S}^1)$ will give a more pessimistic result as the neighborhood of $FB_{\mathbf{f}}(\mathcal{D})$ will be considered as part of the boundary as depicted on Figure 2b.

Remark. As $\mathbf{f}$ is continuous, the image of $\mathcal{S}^1$ by $\mathbf{f}$ is a closed outline in $\mathbb{R}^2$.

For practical reasons we define the function displayed Figure 3 $\phi : [0, 2\pi] \to \mathcal{S}^1$ by :

$$\forall t \in [0, 2\pi], \phi(t) = \begin{pmatrix} cos(t) \\ sin(t) \end{pmatrix} \tag{5}$$

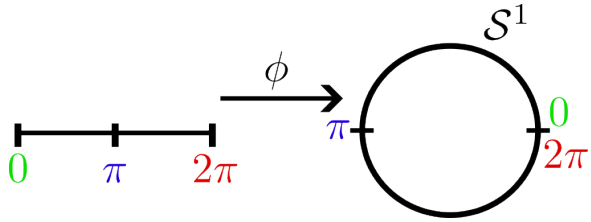


Figure 3: Function ϕ from $[0, 2\pi]$ to $\mathcal{S}^1$.

Remark. This function ϕ is bijective over $[0, 2\pi[$ and $\phi(0) = \phi(2\pi)$. This means that ϕ is bijective over any strict subset of $[0, 2\pi]$.

For the sake of simplicity, let us denote $\mathbf{g}$ the function $\mathbf{f} \circ \phi$, function from $[0, 2\pi]$ to $\mathbb{R}^2$. We then have:

$$\forall t \in [0, 2\pi], \mathbf{g}(t) = \mathbf{f}(\phi(t)) \tag{6}$$

Considering an interval $[t] \subset [0, 2\pi]$, studying the properties of $\mathbf{g}$ over $[t]$ or the properties of $\mathbf{f}$ over $\phi([t])$ will then give the same result as ϕ is bijective on this interval (see Remark 2.1). For instance, if $\mathbf{g}$ is injective over $[t]$, then $\mathbf{f}$ is injective over $\phi([t])$. For the remaining of this article the focus will then be on the function $\mathbf{g}$ to detect and remove the fake boundaries in the image of the unit disk $\mathcal{D}$ by $\mathbf{f}$. Note that as for $\mathbf{f}$, $\mathbf{g}$ is a $\mathcal{C}^1$ function and we note $\mathbf{J_g}$ its Jacobian.

Definition. *Consider a set X. A collection of sets $\{U_\alpha\}_{\alpha \in A}$ is a cover of X if*

$$X \subset \bigcup_{\alpha \in A} U_\alpha \tag{7}$$

As computing the exact image of $[0, 2\pi]$ by $\mathbf{g}$ is not possible, in this article we will consider a cover C of $[0, 2\pi]$ that does not contain $[0, 2\pi]$, and compute $[\mathbf{g}([t])]$ for each interval $[t] \in C$. The result is a set of boxes that contains the true boundary, image of $[0, 2\pi]$ by $\mathbf{g}$. Figure 4 shows the computation of the image of $[0, 2\pi]$ by $\mathbf{g}$ in the case where $\mathbf{f}$ is the identity. In this example all the intervals of C have a width of 0.1.

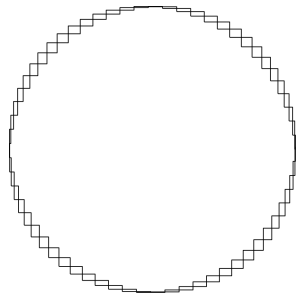


Figure 4: Image of $[0, 2\pi]$ by $\mathbf{g}$ when $\mathbf{f}$ is the identity.

2.2 Problem formulation

The objective of this article is to compute the inner and the outer approximation of the image of the unit disk $\mathcal{D}$ by a function $\mathbf{f}$ using a boundary approach. We will limit our study to the cases where the function $\mathbf{f}$ respects the assumption presented in Subsection 2.1.

As suggested in Subsection 2.1, the study of the boundary will rely on the function $\mathbf{g} : [0, 2\pi] \to \mathbb{R}^2$ defined by $\mathbf{g} = \mathbf{f} \circ \phi$, ϕ being the function defined by Equation (5).

As we are using a boundary approach, the method presented here is subject to the appearance of fake boundaries. As these fake boundaries add an unwanted pessimism to the computation of the image set, the first step is to detect and remove them.

Section 3 introduces the notion of Box Chains to decompose $[0, 2\pi]$ into subsets where $\mathbf{g}$ is injective in order to facilitate the detection of self-intersections in the

boundary. This definition will later be used in the boundary simplification algorithm presented in Section 4. Once the fake boundaries have been removed, the computation of the inner and outer approximation of the image set can be done with less pessimism.

3 Box Chains

3.1 Neighborhood relation

To define the notion of Box Chain we first need to introduce the relation of neighborhood.

Definition. *Let* $[t_i] \in \mathbb{IR}$ *and* $[t_j] \in \mathbb{IR}$ *be two intervals. We define the neighborhood relation noted* $\mathcal{R}_n$ *between* $[t_i]$ *and* $[t_j]$ *as :*

$$[t_i] \, \mathcal{R}_n \, [t_j] \Leftrightarrow [t_i] \cap [t_j] \neq \emptyset \tag{8}$$

This relation can be represented for real-value intervals in the t-plane [14] or by its logical matrix as shown Figure 5.

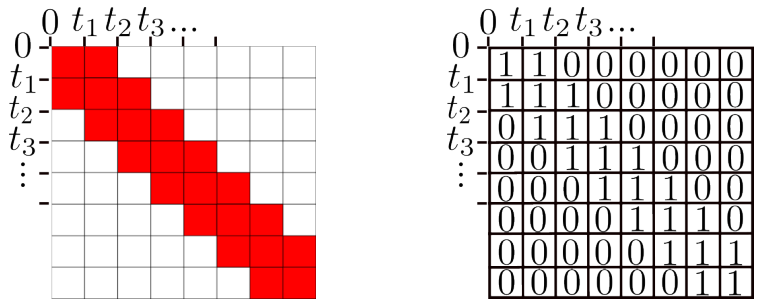


Figure 5: Neighborhood Relation in the t-plane and logical matrix.

The t-plane represents real-value intervals $[0, t_1], [t_1, t_2], ...$ on the abscissa and on the ordinate. The grid is then colored where the corresponding intervals verify the relation, and is left blank otherwise. The resulting grid can be interpreted as a logical matrix, say M, where the blank boxes are null, and the colored boxes are ones. This graphical representation of the relation highlights its properties:

- **Reflexivity** As the main diagonal of the grid has no blank box, i.e. the identity matrix is included in M, it means that the relation is reflexive.
- **Symmetry** As M is symmetric, the relation itself is symmetric.

However we can see that $(t_i \cap t_j \neq \emptyset) \wedge (t_j \cap t_k \neq \emptyset) \not\Longrightarrow t_i \cap t_k \neq \emptyset$, meaning that this relation is not transitive.

3.2 Box Chain relation

Definition. *Let* $[t_i] \in \mathbb{IR}$ *and* $[t_j] \in \mathbb{IR}$ *be two intervals and* $\mathbf{g} : \mathbb{R} \to \mathbb{R}^2$. *We define the Box Chain relation for* $\mathbf{g}$, *noted* $\mathcal{R}_{BC,\mathbf{g}}$ *between* $[t_i]$ *and* $[t_k]$, *as:*

$$[t_i]\,\mathcal{R}_{BC,\mathbf{g}}\,[t_k] \iff \exists m \in \mathbb{N}, ([t_{j_1}], \ldots, [t_{j_m}]) \in \mathbb{IR}^m, \begin{cases} [t_i]\,\mathcal{R}_n\,[t_{j_1}] \wedge \cdots \wedge [t_{j_m}]\,\mathcal{R}_n\,[t_k] \\ \mathbf{g}_{|[t_i] \cup [t_{j_1}] \cup \ldots \cup [t_{j_m}] \cup [t_k]} \text{ is injective} \end{cases} \tag{9}$$

Equivalently, $[t_i]$ *and* $[t_k]$ *are in Box Chain relation for* $\mathbf{g}$ *if a trajectory from neighbor to neighbor exists between* $[t_i]$ *and* $[t_k]$ *on which* $\mathbf{g}$ *is injective.*

The t-plane representation and the logical matrix of the Box Chain relation depends on both the expression of $\mathbf{g}$ and the chosen $t_1, t_2, \cdots$. However this relation is always **symmetric** as the neighborhood relation and the union of sets are symmetric.

3.3 Box Chain decomposition

As depicted in Figure 2a, fake boundaries appear when the considered contour crosses itself. This means that the considered function is not globally injective as two distincts inputs give the same output.

As mentioned in Section 2, the boundary we are dealing with is not a line, but a set of boxes. Finding the self intersections in the contour can then be hard as each box of the contour crosses at least two other boxes as shown Figure 6. This is the result of the continuity of the function $\mathbf{g}$.

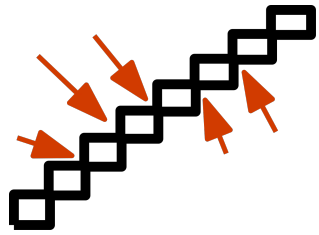


Figure 6: Intersections between the boxes.

3.3.1 Decomposition algorithm

To make this detection easier, Box Chains can be used to decompose the domain of any function into subsets on which the function is locally injective. Looking for the self-intersections in the boundary will then come down to looking for the intersections between different Box Chains. Algorithm 1 is suggested to perform this decomposition. It takes three inputs :

- The studied function $\mathbf{g}$

- An injectivity criterion h
- A cover of $[0, 2\pi]$, C

The injectivity criterion is defined over the powerset of $[0, 2\pi]$, $\mathcal{P}([0, 2\pi])$. For a given interval in $[0, 2\pi]$ it outputs 1 if **g** is injective over this interval, 0 otherwise. This criterion must be sufficient, but does not need to be necessary. An example of injectivity criterion will be given in Section 3.3.3.

The algorithm takes the element of the cover C one by one and try to group them into Box Chains. As soon as an element can not be added to the Box Chain without loosing the injectivity of **g** on it, a new Box Chain is created. Finally the algorithm sorts the intervals of the cover C into a list of Box Chains.

Algorithm 1 Box Chain decomposition.

Input : a function $\mathbf{g} : [0, 2\pi] \to \mathbb{R}^2$,
an injectivity criterion $h : \mathcal{P}([0, 2\pi]) \to \{0, 1\}$ and C a cover of $[0, 2\pi]$
Output : $\mathcal{L}_{BC}$ a list of Box Chains

Set the working list $\mathcal{L}_W := C$ and the final list $\mathcal{L}_{BC} := \{\}$
Pop E from $\mathcal{L}_W$
while ($\mathcal{L}_W \neq \emptyset$) **do**
  Initialize the Box Chain $\mathcal{L}_B := \{E\}$
  while ($\mathcal{L}_W \neq \emptyset$) **do**
    Pop E from $\mathcal{L}_W$
    if $h(\mathcal{L}_B \cup E)$ **and** $\mathcal{L}_B \mathcal{R}_n E$ **then** *injectivity criterion*
      Store E in $\mathcal{L}_B$
    else
      break
    end if
  end while
  Store $\mathcal{L}_B$ in $\mathcal{L}_{BC}$
end while
return $\mathcal{L}_{BC}$

3.3.2 Complexity

To get the best result possible, i.e. as few Box Chains as possible, the cover C will be sorted in increasing order of lower bound, noted lb below, and the popped element will be the first of the list. If we consider a cover of n elements $C = \{[c_i]\}_{i \in [\![1,n]\!]}$, we then have :

$$\forall (i, j) \in [\![1, n]\!]^2, i < j \implies lb([c_i]) \leq lb([c_j]) \tag{10}$$

This sorting and popping process allows to have successive elements in neighborhood relationship.

In terms of complexity the tests $h(\mathcal{L}_B \cup E)$ and $\mathcal{L}_B \mathcal{R}_n E$ require the computation of m jacobians and $m-1$ unions and intersections, m being the number of elements in the list $\mathcal{L}_B$. These tests then have a complexity in $\mathcal{O}(m)$.

The worse case happens when all the elements can be added to the same Box Chain except the last one. The tests mentionned earlier are then runned with $m \in [\![1, n-1]\!]$ elements. Finally, the complexity of the algorithm in the worse case is $\mathcal{O}(n^2)$.

3.3.3 Injectivity criterion

An example of injectivity criterion could rely on the tangent to the contour. The Jacobian of a function represents this tangent when defined, see Figure 7. Note that as the domain of the studied function $\mathbf{g}$ is $\mathbb{R}$, its Jacobian is a vector.

Theorem. *Let $\mathbb{T}$ be a subset of $\mathbb{R}$, $\mathbf{g} : \mathbb{R} \to \mathbb{R}^2$ be a $\mathcal{C}^1$ function and $\mathbf{J_g} : \mathbb{R} \to \mathbb{R}^2$ its Jacobian. If $0_{\mathbb{R}^2} \notin \left[\bigcup_{t \in \mathbb{T}} \mathbf{J_g}(t)\right]$, then $\mathbf{g}$ is injective over $\mathbb{T}$.*

Proof. Let us demonstrate the contrapositive.

If $\mathbf{g}$ is not injective, $\exists (t_1, t_2) \in \mathbb{T}^2$, $t_1 \neq t_2$ so that $\mathbf{g}(t_1) = \mathbf{g}(t_2)$ as shown Figure 7. Let us denote $g_1 : \mathbb{R} \to \mathbb{R}$ and $g_2 : \mathbb{R} \to \mathbb{R}$ the functions that give respectively the first and second component of $\mathbf{g}$. t_1 and t_2 then verify

$$\forall i \in \{1, 2\}, g_i(t_1) = g_i(t_2) \tag{11}$$

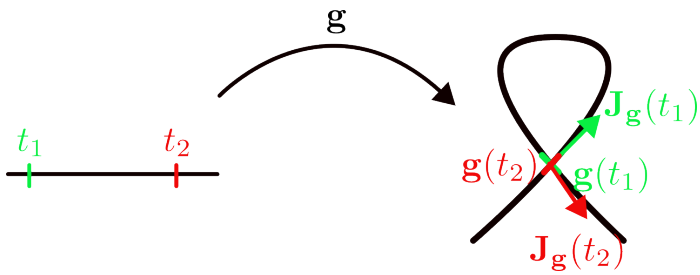


Figure 7: Loop in $\mathbb{R}^2$, $\mathbf{g}$ is not injective.

The mean value theorem can then be applied to each of the g_i functions, giving

$$\exists \tau_i \in [t_1, t_2], J_{g,i}(\tau_i) = 0 \tag{12}$$

Meaning that

$$\forall i \in \{1, 2\}, 0 \in J_{g,i}([t_1, t_2]) \tag{13}$$

Then,

$$0_{\mathbb{R}^2} \in J_{g,1}([t_1, t_2]) \times J_{g,2}([t_1, t_2]) \tag{14}$$

Finally per definition of the cartesian product and as shown Figure 8,

$$J_{g,1}([t_1,t_2]) \times J_{g,2}([t_1,t_2]) = \left[\bigcup_{t\in[t_1,t_2]} \mathbf{J_g}(t) \right] \tag{15}$$

As $(t_1,t_2) \in \mathbb{T}^2$, $[t_1,t_2] \subset \mathbb{T}$. Then $0_{\mathbb{R}^2} \in \left[\bigcup_{t\in\mathbb{T}} \mathbf{J_g}(t) \right]$.

Finally, we get that if $\mathbf{g}$ is not injective over $\mathbb{T}$, $0_{\mathbb{R}^2} \in \left[\bigcup_{t\in\mathbb{T}} \mathbf{J_g}(t) \right]$.

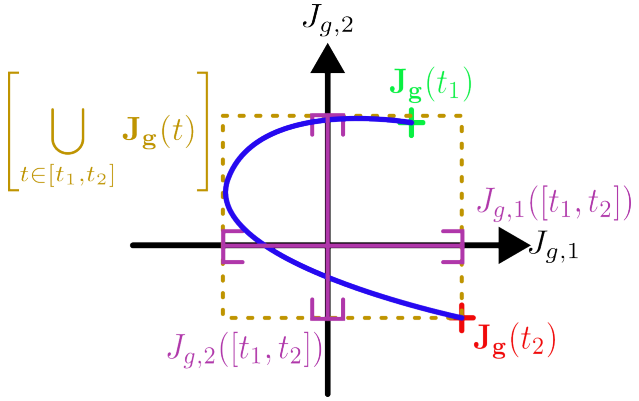


Figure 8: The brown box contains the origin, we can't conclude on the injectivity.

□

4 Determination of the inner and outer approximation of the image a disk

To limit the pessimism while determining the inner and outer approximation of the image of the whole set, the first step is to remove the fake boundaries. To do so, Box Chains presented in the previous section will be used to detect the boxes that belong to the self intersections in the boundary. Once these boxes have been set aside, the fake boundary can be removed before computing the inner and outer approximation of the image set.

4.1 Boundary Simplification algorithm

In this section, the function $\mathbf{f}$ defined in Equation (16) will be considered for the illustrations. However the algorithm presented works with any function as long as the assumptions from Section 2 are satisfied.

$$\forall \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \in \mathcal{D}, \mathbf{f}(\mathbf{x}) = \begin{pmatrix} x_1^2 - x_2^2 + x_1 \\ 2x_1x_2 + x_2 \end{pmatrix} \tag{16}$$

As explained in Section 2, when studying this function on its boundary $\mathcal{S}^1$, we will work with the function $\mathbf{g} : [0, 2\pi] \to \mathbb{R}^2$ for illustration purposes. It is defined by:

$$\forall t \in [0, 2\pi], \mathbf{g}(t) = \begin{pmatrix} cos(t)^2 - sin(t)^2 + cos(t) \\ 2cos(t)sin(t) + sin(t) \end{pmatrix} \tag{17}$$

If we consider a cover C of $[0, 2\pi]$ where each element has a diameter of 0.01, Figure 9 shows the image of C by the function $\mathbf{g}$ of Equation (17) with and without fake boundaries. As mentioned earlier, the boundary is here a set of boxes that constitutes an over-approximation of the real boundary.

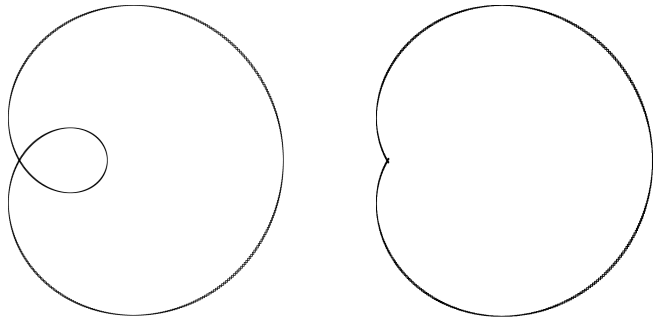


Figure 9: Boundary with (left) and without (right) fake boundaries.

The different steps to eliminate the fake boundaries in the image of C by $\mathbf{g}$ are described below.

4.1.1 Step 1: Box Chain decomposition

The first step before removing the fake boundaries is to look for the self-intersections in the boundary. Indeed the boxes that belong to the intersections have to be kept in the resulting boundary.

The first step is then to decompose the cover C into Box Chains as suggested in Subsection 3.3. This decomposition will make the detection of the fake boundaries easier. Algorithm 1 can be used to this end and Theorem 3.3.3 gives an injectivity criterion h:

$$\forall [t] \subset [0, 2\pi], h([t]) = \begin{cases} 0 \; if \; \mathbf{0}_{\mathbb{R}^2} \in [\mathbf{J}_\mathbf{g}([t])] \\ 1 \; otherwise \end{cases} \tag{18}$$

C can then be decomposed into Box Chains thanks to Algorithm 1 to get the result shown on Figure 10. As expected the self intersections in the boundary appear between different Box Chains.

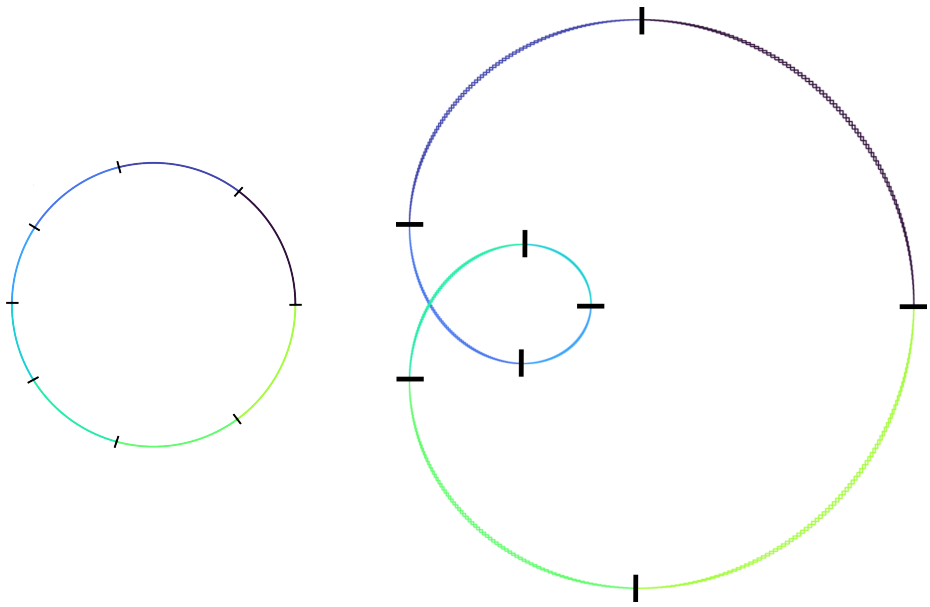


Figure 10: Eight Box Chains in $\mathcal{S}^1$ (left) and their image by $\mathbf{g}$ (right).

4.1.2 Step 2: Finding the self intersections

Once the Box Chain decomposition of C has been done, we can look for intersections between the images of the Box Chains by $\mathbf{g}$. Two cases of intersections can be observed Figure 11.

- If two Box Chains are in neighborhood relation, their images by $\mathbf{g}$ intersect each other at their junction point.
- If fake boundaries exist, the boundary crosses itself and the image of two different Box Chains intersect each other.

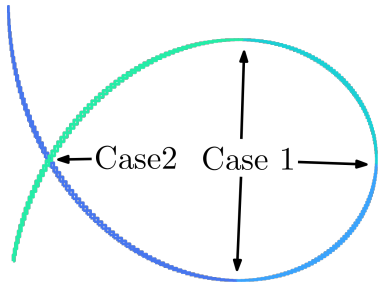


Figure 11: Two cases of intersections between the images of two Box Chains.

Remark. The neighborhood relation can be extended to Box Chains : Let there be two Box Chains B_1 and B_2, $B_1 \mathcal{R}_n B_2 \Leftrightarrow \exists([t_1],[t_2]) \in B_1 \times B_2, [t_1]\mathcal{R}_n[t_2]$

The injectivity criterion h can be used to distinguish these two cases. Indeed in the first case the function **g** is injective around the junction point between the two Box Chains, which is not the case when the boundary really crosses itself. Algorithm 2 sums up this step. When applied, this algorithm outputs a list of intervals where the studied function **g** is not injective. The corresponding intersections can be then computed as visible in red on Figure 12.

Algorithm 2 Finding self intersections.

Input : a function $\mathbf{g} : [0, 2\pi] \to \mathbb{R}^2$, a list of Box Chains $\mathcal{L}_{BC}$ and an injectivity criterion $h : \mathcal{P}([0, 2\pi]) \to \{0, 1\}$

Output : a list of intervals $\mathcal{L}_I$

Set the working list $\mathcal{L}_I := \{\}$.
for $i = 1$ to $len(\mathcal{L}_{BC})$ **do**
  for $j = i + 1$ to $len(\mathcal{L}_{BC})$ **do**
    for t_i in $(\mathcal{L}_{BC}[i])$ **do**
      for t_j in $(\mathcal{L}_{BC}[j])$ **do**
        if $\mathbf{g}(t_i) \cap \mathbf{g}(t_j) \neq \emptyset$ **and not** $h(t_i \cup t_j)$ **then** *self intersection*
          Store individually t_i and t_j in $\mathcal{L}_I$
        end if
      end for
    end for
  end for
end for
return $\mathcal{L}_I$

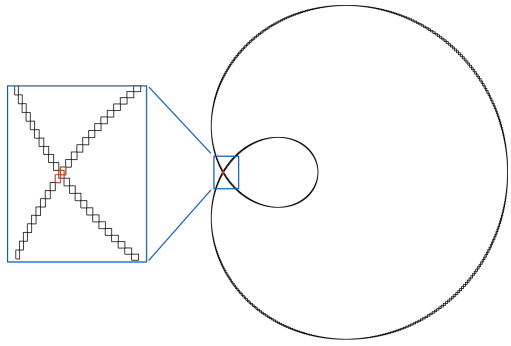


Figure 12: Self intersection in the boundary (in red).

4.1.3 Step 3: Determination of the inner areas

To determine the inner approximation of the image set an initial paving of the set $S = \bigcup_{[t]\in C} [\mathbf{g}([t])]$ is needed. To achieve this the SIVIA algorithm [7] was used with an accuracy of 0.04 to get the result shown on Figure 13. By doing so, the union of the yellow boxes is an outer approximation of S, and each yellow box has a width smaller than 0.04.

The blue boxes then represent the complementary of the yellow boxes. Their union is an inner approximation of the complementary of S. They can be divided into connected subsets as shown on Figure 13. For this work we will rely on the connected subset decomposition implemented in the Codac library [13].

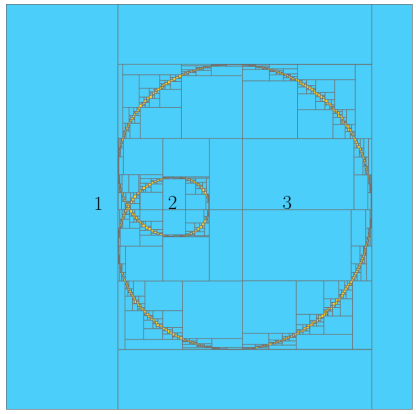


Figure 13: Paving with three connected subsets.

Then instead of qualifying each box individually as inside or outside the image set, we can qualify the whole connected subset. This means that once a box is proven to be inside, the corresponding connected subset can directly be classified as inside.

A solution to distinguish the boxes that are inside and outside is to look at the normal to the boundary. As depicted Figure 15 the normal of a real boundary points towards the exterior. This means that the opposite corner is inside.

Remark. Note that in the case of a fake boundary the normal vector points toward the interior of the set, but the opposite corner is still inside. This is due to the fact that a fake boundary is inside the set it should bound.

Remark. For a given interval $[t] \in [0, 2\pi]$ the tangent to the oriented boundary evaluated in $[\mathbf{g}]([t])$ belongs to $[\mathbf{J_g}([t])]$, and rotating $[\mathbf{J_g}([t])]$ by $-\frac{\pi}{2}$ gives a box containing the normal to the boundary. Note that the sign of the rotation depends on the orientation chosen for the unit circle (here counterclockwise). Figure 14 displays the oriented unit circle with a tangent and the associated normal vector.

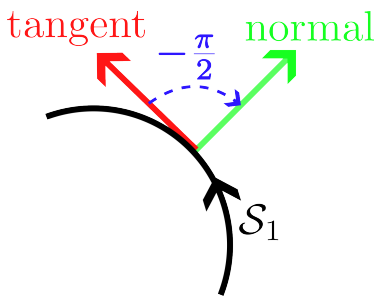


Figure 14: A tangent and the associated normal to the unit circle.

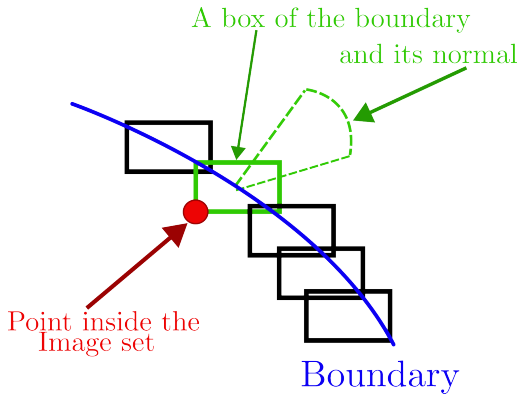


Figure 15: Determination of the inner areas.

Applying this to every box of the boundary, except the ones that were previously proven to belong to the self intersections, gives a set of points that are inside. The connected subsets containing at least one of these points can be marked as inside, and the other can be marked as outside. Figure 16 shows an example of output for this step.

4.1.4 Step 4: Suppressing the fake boundaries

As mentioned earlier, the normal to the boundary is supposed to point outwards. Figure 17 illustrates the fact that in the case of fake boundaries this normal points towards an area that was classified as inside in the last step. This can be seen as the fact that the normal of a fake boundary aims towards a fake exterior.

Suppressing the boxes with a normal pointing towards an inside area allows us to remove the fake boundaries to obtain a less pessimistic approximation of $\partial\mathbf{f}(\mathcal{D})$ as visible on Figure 18. Note that as the self-intersections in the contour have been detected in Step 2, it is possible to propagate the information of a box belonging to the fake boundary from neighbor to neighbor until an intersection is reached.

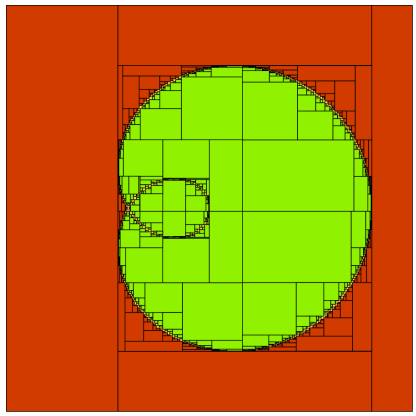


Figure 16: Inner (green) and outer (green + yellow) approximation of the image set with fake boundaries.

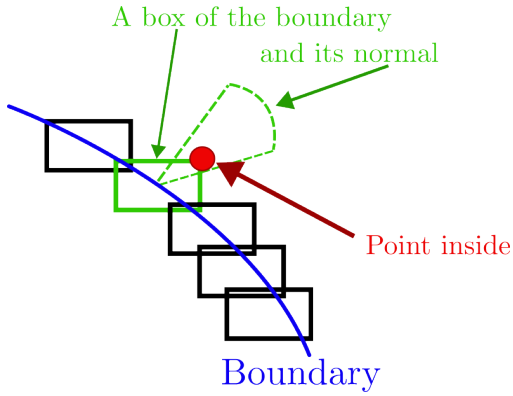


Figure 17: Case of a fake boundary.

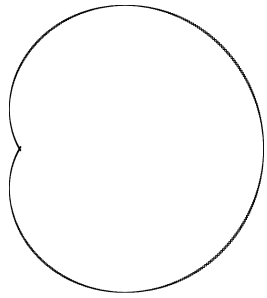


Figure 18: Fake boundaries removed.

4.2 Inner and outer approximation of the image set

Once the fake boundaries have been removed, Step 3 can be applied again with the remaining boxes to get the inner and outer approximation of the image set without fake boundaries.

To do so if we denote by C_r the set of remaining intervals of the cover C after the boundary simplification, we first proceed to an initial paving of $S = \bigcup_{[t]\in C_r} [\mathbf{g}([t])]$ by using the SIVIA algorithm a second time. Then the boxes that are not part of the boundary are sorted into connected subsets and are finally classified as inside or outside the image set with the criteria illustrated Figure 15.

Finally, we are able to characterize the inner and outer approximation of the image of the unit disk. Figure 19 shows the result obtained with the function $\mathbf{f}$ defined by Equation (16) and an initial cover where all the elements have a diameter of 0.01. The whole process, from the Box Chain decomposition to the final result took approximatly 0.6 seconds.

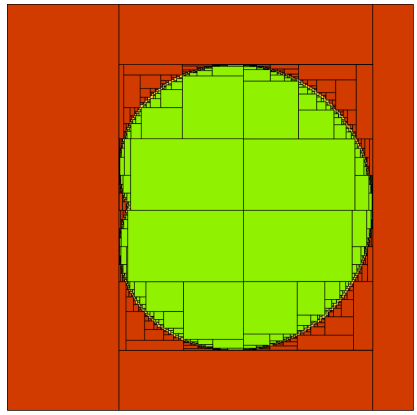


Figure 19: Inner (green) and outer (green + yellow) approximation of the image set without fake boundaries.

4.3 Additional example

The algorithm presented here also work in more complex cases. Algorithm 3 gives another example of a function $\mathbf{f} : \mathcal{D} \to \mathbb{R}^2$.

As earlier a function $\mathbf{g} : [0, 2\pi] \to \mathbb{R}^2$ can be defined by $\forall t \in [0, 2\pi], \mathbf{g}(t) = \mathbf{f}(\phi(t))$. Figure 20 shows the image of the unit circle $\mathcal{S}^1$ by this function $\mathbf{f}$. This result is obtained by considering a cover C of $[0, 2\pi]$ and computing $[\mathbf{g}([t])]$ for each interval $[t] \in C$. For this application each element of the cover has a diameter of 0.005.

With this function, Algorithm 1 gave 15 Box Chains as shown on Figure 21.

Algorithm 3 Pseudocode of the implementation of the **f** function.

Define $s(\tau, a, b, c) = a + (b - a) \cdot 0.5 \cdot (1 + \tanh(10 \cdot (\tau - c)))$

Input: $\mathbf{p} = (p_1, p_2)$ a point in the unit disk

$\rho = \sqrt{p_1^2 + p_2^2}$
$\alpha = arctan2(p_2, p_1)$
$\tau = 0.03 + \alpha \cdot \frac{1.01}{2\pi}$
$t_1 = 1 - \cos(2\pi\tau)$
$\mathbf{d} = \begin{pmatrix} 5 - 50 \cdot \cos(5 \cdot t_1) \\ 30 \cdot \sin(5 \cdot t_1) \end{pmatrix}$
$\theta = s(\tau, -3 \cdot \pi/2, -\pi/2, 0) + s(\tau, 0, \pi, 0.5) + s(\tau, 0, \pi, 1)$
$\mathbf{y} = \begin{pmatrix} 5 \cdot t_1 - 5 \cdot \sin(5 \cdot t_1) \\ 2 - 3 \cdot \cos(5 \cdot t_1) \end{pmatrix} + \frac{\rho}{\sqrt{d_1^2 + d_2^2}} \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \mathbf{d}$

Output: $\mathbf{y}$

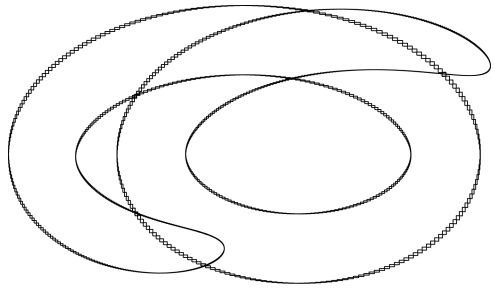


Figure 20: Boundary with fake boundaries.

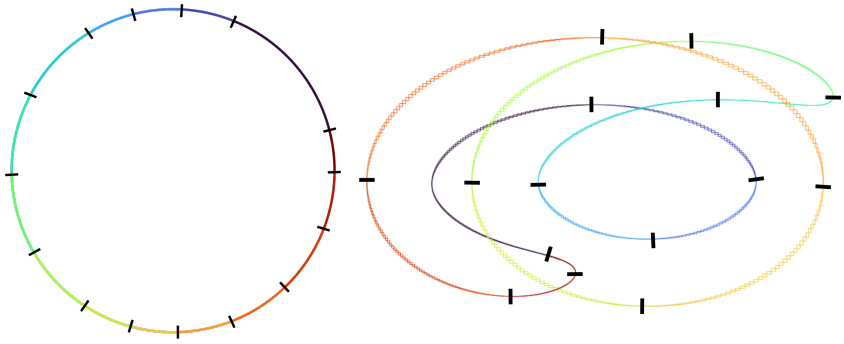


Figure 21: Fifteen Box Chains in $\mathcal{S}^1$ (left) and their image by **g** (right).

Thanks to this Box Chain decomposition, we were able to detect the self-intersections in the boundary as shown in red on Figure 22.

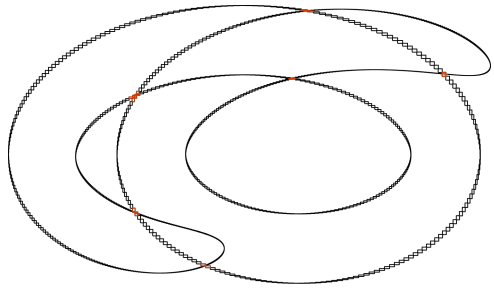
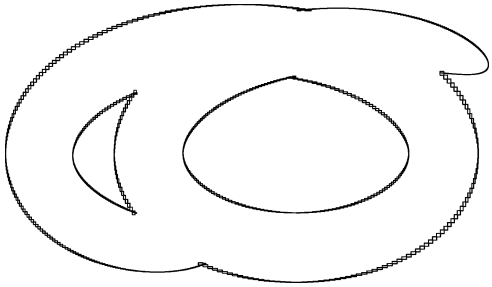


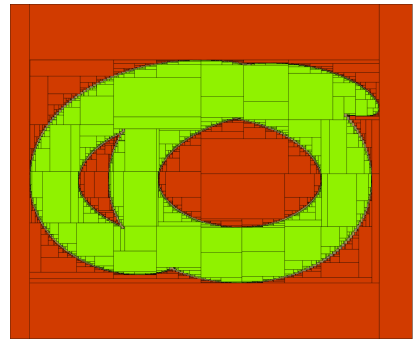
Figure 22: Self-intersections in the boundary (in red).

Finally, the fake boundaries are detected and removed before computing the inner and outer approximation of the image set without fake boundaries as displayed on Figures 23a and 23b.

For this application the whole process took 14 seconds. This increase in the computation time is due to two factors. First the higher resolution of the cover leads to an increase in the computation time as the Box Chain decomposition algorithm has a complexity in $\mathcal{O}(n^2)$ (see Section 3.3.2). In addition, the area to pave in this example is larger while the same accuracy was used in the SIVIA algorithm. Indeed in the first example the area to cover is a 5x5 square, whereas in this example the area is a 16x12 rectangle. As the SIVIA paving is called twice, this second factor influences even more the computation time.



23(a) Fake boundaries removed



23(b) Inner (green) and outer (green + yellow) approximation of the image set

Figure 23: Fake boundaries are detected and removed before computing the inner and outer approximation of the image set without fake boundaries

5 Conclusion

This paper presented the problematic of fake boundaries in the computation of the image of a set by a function. A method to remove them in the case of a function from $\mathcal{D}$ to $\mathbb{R}^2$ was presented. The method was finally applied with two examples where it was indeed able to remove the fake boundaries.

The notion of Box Chains was introduced with the definition of the neighborhood and the Box Chain relations. This Box Chain relation relies on an injectivity criterion and an example for the case of a function from $\mathbb{R}$ to $\mathbb{R}^2$ was proposed. These Box Chains can be used to make the detection of self-intersections in the boundary easier. In addition, a Box Chain decomposition algorithm in $\mathcal{O}(n^2)$ was presented.

Finally a boundary simplification algorithm was displayed. It relies on Box Chains to facilitate the detection of self intersections in the contour. Once the fake boundaries have been removed, a better inner approximation of the image set can be obtained.

References

[1] Alamo, T., Bravo, J. M., and Camacho, E. F. Guaranteed state estimation by zonotopes. *Automatica*, 41(6):1035–1043, 2005. DOI: 10.1016/j.automatica.2004.12.008.

[2] Brateau, Q., Le Bars, F., and Jaulin, L. Considering adjacent sets for computing the visibility region. *Acta Cybernetica*, 2025. DOI: 10.14232/actacyb.314640.

[3] Costa Vianna, M., Goubault, E., Jaulin, L., and Putot, S. Estimating the coverage measure and the area explored by a line-sweep sensor on the plane. *International Journal of Approximate Reasoning*, 169:109162, 2024. DOI: 10.1016/j.ijar.2024.109162.

[4] Dabadie, C., Kaynama, S., and Tomlin, C. J. A practical reachability-based collision avoidance algorithm for sampled-data systems: Application to ground robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4161–4168. IEEE, 2014. DOI: 10.1109/IROS.2014.6943149.

[5] Franek, P. and Ratschan, S. Effective topological degree computation based on interval arithmetic. *Mathematics of Computation*, 84(293):1265–1290, 2015. DOI: 10.1090/S0025-5718-2014-02877-9.

[6] Guyonneau, R., Lagrange, S., Hardouin, L., and Lucidarme, P. Guaranteed interval analysis localization for mobile robots. *Journal on Advanced Robotics*, 28(16):1067–1077, 2014. DOI: 10.1080/01691864.2014.908742.

[7] Jaulin, L., Kieffer, M., Didrit, O., and Walter, E. *Applied Interval Analysis: With Examples in Parameter and State Estimation, Robust Control and Robotics*. Engineering Online Library. Springer-Verlag, London, 2001. DOI: 10.1007/978-1-4471-0249-6.

[8] Karnaugh, M. The map method for synthesis of combinational logic circuits. *Transactions of the American Institute of Electrical Engineers, Part I: Communication and Electronics*, 72(5):593–599, 1953. DOI: 10.1109/TCE.1953.6371932.

[9] Kieffer, M., Jaulin, L., Braems, I., and Walter, E. Guaranteed set computation with subpavings. *Scientific Computing, Validated Numerics, Interval Methods*, pages 167–178, 2000. DOI: 10.1007/978-1-4757-6484-0_14.

[10] Malone, N., Chiang, H.-T., Lesser, K., Oishi, M., and Tapia, L. Hybrid dynamic moving obstacle avoidance using a stochastic reachable set-based potential field. *IEEE Transactions on Robotics*, 33(5):1124–1138, 2017. DOI: 10.1109/TRO.2017.2705034.

[11] Merlet, J.-P. Interval analysis and robotics. In *Robotics Research*, Volume 66 of *Springer Tracts in Advanced Robotics*, pages 147–156, 2007. DOI: 10.1007/978-3-642-14743-2_13.

[12] Rego, B. S., Raffo, G. V., Scott, J. K., and Raimondo, D. M. Guaranteed methods based on constrained zonotopes for set-valued state estimation of nonlinear discrete-time systems. *Automatica*, 111:108614, 2020. DOI: 10.1016/j.automatica.2019.108614.

[13] Rohou, S., Desrochers, B., and Le Bars, F. The Codac Library. *Acta Cybernetica*, 26(4):871–887, 2024. DOI: 10.14232/actacyb.302772.

[14] Rohou, S., Jaulin, L., Mihaylova, L., Le Bars, F., and Veres, S. M. *Reliable Robot Localization: A Constraint-Programming Approach Over Dynamical Systems*. Wiley-ISTE, 1 edition, 2020. ISBN: 9781848219700.

[15] Stenger, F. Computing the topological degree of a mapping in $\mathbb{R}^n$. *Numerische Mathematik*, 25(1):23–38, 1975. DOI: 10.1007/BF01419526.

[16] Welte, A., Jaulin, L., Ceberio, M., and Kreinovich, V. Avoiding fake boundaries in set interval computing. *Journal of Uncertain Systems*, 11(2):137–148, 2017. URL: https://ensta-bretagne.hal.science/hal-01698416/file/jusVol11No2paper07.pdf.