Acta Cybernetica **24** (2020) 431–448.

Characterizing Sliding Surfaces of Cyber-Physical Systems

Luc Jaulin[a] and Fabrice Le Bars[b]

Abstract

When implementing a non-continuous controller for a cyber-physical system, it may happen that the evolution function of the closed-loop system is not anymore piecewise continuous along the trajectory, mainly due to *if* statements inside the control algorithm. As a consequence, an unwanted chattering effect may occur. This behavior is often difficult to observe even in simulation. We propose here a set-membership method based on interval analysis to detect different types of discontinuities. One of them is the *sliding surface* where the state trajectory jumps indefinitely between two distinct behaviors. As an application, we consider the validation of a sailboat controller. We show that our approach is able to detect and explain some unwanted sliding effects that may be observed in rare and specific situations on our actual sailboat robots.

Keywords: sliding surface, interval analysis, sailboats

1 Introduction

Validating properties of cyber-physical systems [16, 29] is a difficult problem for which set membership techniques provide original and efficient solutions [25, 26].

Different types of set-membership approaches exist for the validation. Some require the integration of nonlinear differential equations [19, 28, 30]. Others are based on positive invariance approaches [1, 18]. For the numerical resolution some methods grid the state space [7, 27] which makes them computationally expensive. Lyapunov-based methods [24], level-set methods [20], or barrier functions [4] are attractive since they do not perform any integration through time. Now, these methods generally require a parametric expression for candidate Lyapunov-like functions [23].

This paper considers the validation of the controller of a sailboat robot which is an illustrative example of what is a cyber-physical system. Due to the control strategy used, the robot is an *hybrid system* [22] since it includes a physical system

[a]Lab-STICC, ENSTA Bretagne, Brest, France, E-mail: `lucjaulin@gmail.com`
[b]Lab-STICC, ENSTA Bretagne, Brest, France, E-mail: `fabrice.lebars@ensta-bretagne.fr`

DOI: 10.14232/actacyb.24.3.2020.9

(the sailboat) and an algorithm (the controller inside the computer of the robot). More precisely, it is a *controlled switching system* [10] due to some discrete state variables in the controller. The controller is an algorithm containing *if* statements and the validation requires approaches coming from invariance approaches [3], static analysis [12] and abstract interpretation [6].

To detect the discontinuities and Zeno effects, we propose in this paper to generate a set of equalities in the state space where undesirable switching phenomena could occur. The corresponding zone (called later *sliding surface*) may be stable and the system can be trapped inside without any possibility to escape. Characterizing these sliding zones will be done by using interval techniques [17, 21]. This characterization can be used for the validation of the controller or to correct it by eliminating the unwanted sliding surfaces.

The paper is organized as follows. Section 2 introduces the easy-boat model which is a simple sailboat with a controller. This model will be used to illustrate our approach. Section 3 provides the formalism and gives a list of three problems we want to solve. Section 4 shows how our approach can be used to validate the controller but also to detect and explain some unwanted sliding effects that occur on actual sailboat controllers. Section 5 concludes the paper and provides some perspectives.

2 Easy boat model

The easy-boat model is described by

$$\dot{d} = \sin u \tag{1}$$

under the constraint

$$\cos(\psi - u) + \cos\frac{\pi}{5} > 0. \tag{2}$$

It is a simple version of a sailboat following a line [13], where ψ is the angle of the wind, d is the algebraic distance to the line and u is the heading of the boat. This is illustrated by Figure 1, where s is the curvilinear abscissa.

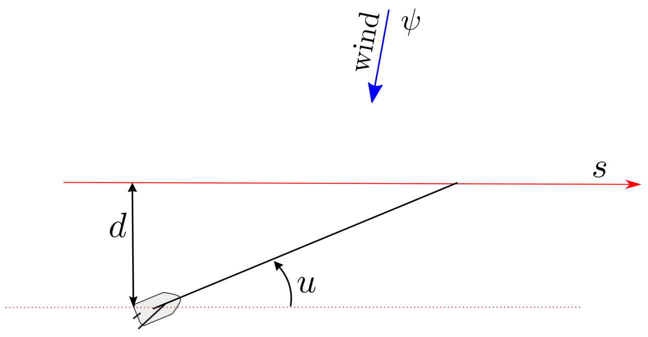

Figure 1: Easy-boat following the red line

We want that, after some transient period, the distance d becomes small ($|d| \leq 2$ for instance). The controller we propose is the following, where $q \in \{-1, 1\}$.

```
**Controller** in: $(d, \psi, q)$ ;   out:$u$
if $d^2 - 1 > 0$ then $q :=$ sign $(d)$
if $\cos(\psi + \operatorname{atan} d) + \cos\frac{\pi}{4} \leq 0$ or $\left(d^2 - 1 \leq 0 \text{ and } \cos\psi + \cos\frac{\pi}{4} \leq 0\right)$
        then $u := \pi + \psi - q\frac{\pi}{4}$.
        else $u := -\operatorname{atan} d$.
```

Figure 2 provides some simulations with $q = 1$ at time $t = 0$. We took different initial conditions to avoid the superposition of the curves, taking into account the fact that the behavior of the system does not depend on these initial values for d. When q switches between -1 to 1, the trajectories are not differentiable.

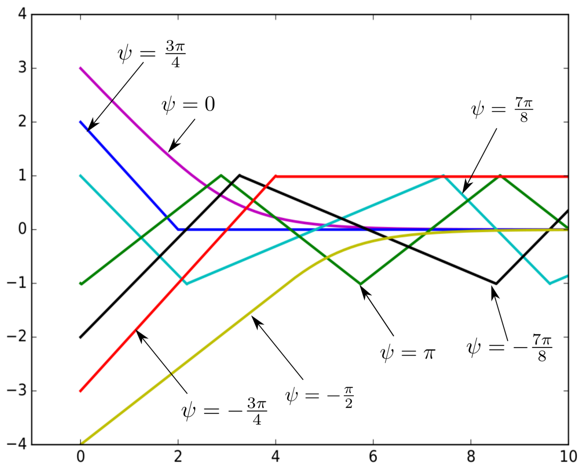

Figure 2: Simulation of the easy-boat model (t, d) with respect to different wind angles ψ

Remark. For a link to the sailboat, it is more interpretable to draw d with respect to the curvilinear abscissa $s = \int^t \cos u$ as in Figure 3. The boat has to follow the horizontal line, (s, d) corresponds to the position of the boat and u is the heading. The arrows represent different directions for the winds. As we can see on the figure, the boat never goes upwind: there always exists an angle between the heading and the wind greater than $\zeta = \frac{\pi}{5}$ where ζ is the angle defining the no-go zone. For the simulation, we added the state variable s which satisfies $\dot{s} = \cos u$.

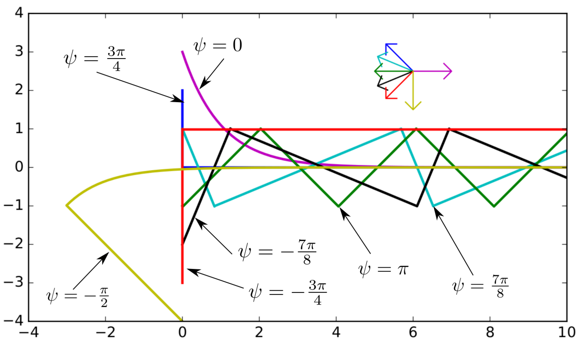

Figure 3: Simulation of the easy-boat in the (s, d) plane with different ψ

3 Formalism

This section provides an abstraction of our sailboat robot in order to give useful definitions, theorems and proofs. The corresponding formalism will be applied in the next section on the sailboat validation problems.

Definition. Given $\mathbb{Q}^-$, $\mathbb{Q}^+$ two disjoint closed subsets of $\mathbb{R}^n$, two smooth functions $\mathbf{f}_a, \mathbf{f}_b : \mathbb{R}^n \times \{-1, 1\} \to \mathbb{R}^n$, we define the dynamical system

$$\mathcal{S}(\mathbb{A}) : \left\{ \begin{array}{ccccl} \dot{\mathbf{x}} & = & \mathbf{f}(\mathbf{x}, q) & = & \left\{ \begin{array}{ll} \mathbf{f}_a(\mathbf{x}, q) & \text{if } \mathbf{x} \in \mathbb{A} \\ \mathbf{f}_b(\mathbf{x}, q) & \text{if } \mathbf{x} \in \mathbb{B} = \overline{\mathbb{A}} \end{array} \right. \\ q & = & -1 & & \text{as soon as } \mathbf{x} \in \mathbb{Q}^- \\ & = & +1 & & \text{as soon as } \mathbf{x} \in \mathbb{Q}^+ \end{array} \right. \quad (3)$$

We assume that

- $\mathbf{f}_a$, $\mathbf{f}_b$ are continuous and differentiable,
- $\mathbb{A}$ is a closed subset of $\mathbb{R}^n$ that can be defined by inequalities linked by Boolean operators.

This definition is illustrated by the automaton of Figure 4 taking the conventions used for hybrid systems [2, 9]. The red arrows show transitions which may not be stable and which may generate the sliding phenomenons that are studied in this paper.

This definition trivially extends to situations where we have more than two guard sets $\mathbb{Q}^-, \mathbb{Q}^+$ and more than two fields $\mathbf{f}_a, \mathbf{f}_b$. An hybrid system which can be translated into the form (3) is said to be *expandable*.

Remark. In this paper, to avoid atypical situations, the closed sets are assumed to be topologically stable, i.e., they have the same boundary as their interior. For instance, a disk of $\mathbb{R}^2$ is topologically stable, but not the circle since its interior is empty. We will also assume that the closed sets can be defined as a finite composition (with unions and intersections) of sets of the form $\mathbb{X} = \{\mathbf{x} \in \mathbb{R}^n \,|\, c(\mathbf{x}) \leq 0\}$ where c is a smooth function.

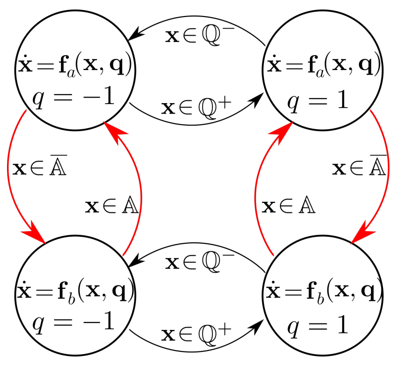
$\dot{\mathbf{x}}=\mathbf{f}_a(\mathbf{x},\mathbf{q})$
$q=-1$
$\dot{\mathbf{x}}=\mathbf{f}_a(\mathbf{x},\mathbf{q})$
$q=1$
$\dot{\mathbf{x}}=\mathbf{f}_b(\mathbf{x},\mathbf{q})$
$q=-1$
$\dot{\mathbf{x}}=\mathbf{f}_b(\mathbf{x},\mathbf{q})$
$q=1$
$\mathbf{x}\in\mathbb{Q}^-$
$\mathbf{x}\in\mathbb{Q}^+$
$\mathbf{x}\in\overline{\mathbb{A}}$
$\mathbf{x}\in\mathbb{A}$
$\mathbf{x}\in\mathbb{A}$
$\mathbf{x}\in\overline{\mathbb{A}}$
$\mathbf{x}\in\mathbb{Q}^-$
$\mathbf{x}\in\mathbb{Q}^+$

Figure 4: Automaton representing our Cyber Physical System

Since $\mathbb{A}$ is closed, the set $\mathbb{B}$ is open and the boundaries $\partial\mathbb{A}, \partial\mathbb{B}$ of $\mathbb{A}, \mathbb{B}$ satisfy

$$\partial\mathbb{A} = \partial\mathbb{B} = \mathbb{A} \cap \text{Clo}\,(\mathbb{B}) \tag{4}$$

where $\text{Clo}\,(\mathbb{B})$ denotes the smallest closed set which encloses $\mathbb{B}$. This common boundary can be defined by an equality. Moreover the pair $(\mathbf{x}, q)$ always satisfies the constraint

$$\begin{array}{ccc} \mathbf{x} \in \mathbb{Q}^+ & \Rightarrow & q = 1 \\ \mathbf{x} \in \mathbb{Q}^- & \Rightarrow & q = -1 \end{array} \tag{5}$$

This formula can be denoted equivalently by $\mathbf{x} \in \overline{\mathbb{Q}^{-q}}$, with the notation $\mathbb{Q}^{-1} = \mathbb{Q}^-$ and $\mathbb{Q}^1 = \mathbb{Q}^+$. The corresponding behavior is represented on Figure 5, where the blue arrows correspond to $\mathbf{f}\,(\mathbf{x}, -1)$ and the pink arrows to $\mathbf{f}\,(\mathbf{x}, 1)$.

In this paper, we consider three problems:

- the *constraint satisfaction problem* which checks that a given variable of the algorithm defining $\mathbf{f}$ is inside a feasible domain.
- the positive invariance for a set defined by inequalities
- the characterization of the sliding surface.

3.1 Constraint satisfaction

We want to show the state of the the cyber-physical system never reaches a forbidden domain. This can often be expressed as showing that we never have

$$h\,(\mathbf{x}, q) \leq 0, \tag{6}$$

with

$$\left\{ \begin{array}{cccc} h\,(\mathbf{x}, q) & = & h_a\,(\mathbf{x}, q) & \text{if } \mathbf{x} \in \mathbb{A} \\ & = & h_b\,(\mathbf{x}, q) & \text{if } \mathbf{x} \in \mathbb{B} \end{array} \right. \tag{7}$$

where h_a, h_b are continuous.

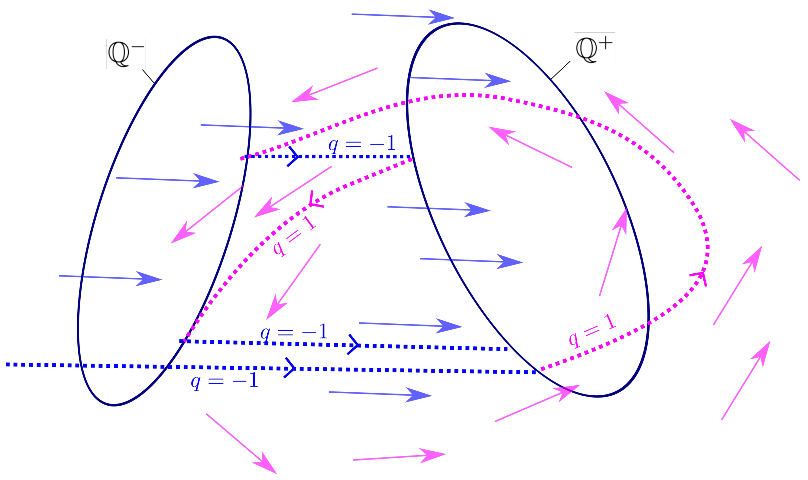

Figure 5: When the trajectory reaches $\mathbb{Q}^-$ (resp. $\mathbb{Q}^+$),the variable q switches to -1 (resp. $+1$)

Proposition 1. If the set

$$\begin{array}{rcl}\mathbb{H} & = & \cup_{q\in\{-1,1\}}\left(\{\mathbf{x}|h_a(\mathbf{x},q)\leq 0)\}\cap\mathbb{A}\cap\overline{\mathbb{Q}^{-q}}\right)\\ & & \cup\left(\{\mathbf{x}|h_b(\mathbf{x},q)\leq 0)\}\cap\mathbb{B}\cap\overline{\mathbb{Q}^{-q}}\right)\end{array} \tag{8}$$

is empty then we cannot have $h(\mathbf{x},q)\leq 0$.

Proof. The proof is by contradiction. More precisely, we take $(\mathbf{x},q)$ such that $h(\mathbf{x},q)\leq 0$ and we show that $\mathbf{x}\in\mathbb{H}$. Since $\mathbb{B}=\overline{\mathbb{A}}$, we should consider two cases $\mathbf{x}\in\mathbb{A}$ and $\mathbf{x}\in\mathbb{B}$.

Case 1: $\mathbf{x}\in\mathbb{A}$. From Equation (7), $h(\mathbf{x},q)=h_a(\mathbf{x},q)$ and thus

$$\mathbf{x}\in\{\mathbf{x}|h_a(\mathbf{x},q)\leq 0)\}\cap\mathbb{A}.$$

Case 2: $\mathbf{x}\in\mathbb{B}$. From Equation (7), $h(\mathbf{x},q)=h_b(\mathbf{x},q)$ and thus

$$\mathbf{x}\in\{\mathbf{x}|h_b(\mathbf{x},q)\leq 0)\}\cap\mathbb{B}.$$

Since from Equation (5), we always have $\mathbf{x}\in\overline{\mathbb{Q}^{-q}}$, in both cases, $\mathbf{x}\in\mathbb{H}$. This is inconsistent with the fact that $\mathbb{H}=\emptyset$.■

3.2 Capture set

Consider a function $V:\mathbb{R}^n\to\mathbb{R}$. The set $\mathbb{C}=\{\mathbf{x}|V(\mathbf{x})\leq 0\}$ is called a *capture set* (or a *positive invariant set*) if all trajectories $\mathbf{x}(t)$ that enter inside $\mathbb{C}$ stay inside forever. To check that $\mathbb{C}$ is a capture set, we recall the notion of *Lie derivative* of V with respect to the field $\mathbf{f}:\mathbb{R}^n\to\mathbb{R}^n$ as

$$\mathcal{L}_{\mathbf{f}}^V(\mathbf{x})=\frac{dV}{d\mathbf{x}}(\mathbf{x})\cdot\mathbf{f}(\mathbf{x}). \tag{9}$$

We also define the Lie set as

$$\mathbb{L}_{\mathbf{f}}^{V}=\left\{\mathbf{x}|\mathcal{L}_{\mathbf{f}}^{V}(\mathbf{x})\leq 0\right\}. \tag{10}$$

In our context, the field depends on $i\in\{a,b\}$ and q. We will write

$$\begin{array}{rcl}\mathcal{L}_{i}^{V}(\mathbf{x},q) & = & \mathcal{L}_{\mathbf{f}_i(\cdot,q)}^{V}(\mathbf{x})\\ \mathbb{L}_{i}^{V}(q) & = & \mathbb{L}_{\mathbf{f}_i(\cdot,q)}^{V}\end{array} \tag{11}$$

Proposition 2. Define the set

$$\mathbb{V}=\bigcup_{q\in\{-1,1\}}\left(\overline{\mathbb{L}_{a}^{V}(q)}\cap\mathbb{A}\cap\overline{\mathbb{Q}^{-q}}\right)\cup\left(\overline{\mathbb{L}_{b}^{V}(q)}\cap\mathbb{B}\cap\overline{\mathbb{Q}^{-q}}\right). \tag{12}$$

If $\mathbb{V}\cap\overline{\mathbb{C}}=\emptyset$ then $\mathbb{C}$ is a capture set.

Proof. The proof is by contradiction. Assume that $\mathbb{C}$ is not a capture set. There exists a trajectory leaving $\mathbb{V}$ at a point $\mathbf{x}$. Assume first that $\mathbf{x}\in\mathbb{A}$. Then, $\mathcal{L}_{a}^{V}(\mathbf{x},q)\geq 0$ or equivalently, $\mathbf{x}\in\overline{\mathbb{L}_{a}^{V}(q)}$. Taking into account that from (5), $\mathbf{x}\in\overline{\mathbb{Q}^{-q}}$, we get that $\mathbf{x}\in\overline{\mathbb{L}_{a}^{V}(q)}\cap\mathbb{A}\cap\overline{\mathbb{Q}^{-q}}$. If now we assume that $\mathbf{x}\in\mathbb{B}$, we get $\mathbf{x}\in\overline{\mathbb{L}_{b}^{V}(q)}\cap\mathbb{B}\cap\overline{\mathbb{Q}^{-q}}$.■

3.3 Sliding surface

The *sliding surface* $\mathbb{S}(\mathbb{A})$ [8] for $\mathcal{S}(\mathbb{A})$ (see Equation (3)) is defined as the largest subset of the boundary $\partial\mathbb{A}$ between $\mathbb{A}$ and $\mathbb{B}=\overline{\mathbb{A}}$ such that the system can stay inside for a non degenerated interval of time.

If $\mathbb{A}$ is defined by the inequality $c(\mathbf{x})\leq 0$, then $\mathbb{B}$ is defined by $c(\mathbf{x})>0$ and the boundary by $c(\mathbf{x})=0$. The sliding surface is

$$\begin{array}{rcl}\mathbb{S}(\mathbb{A}) & = & \partial\mathbb{A}\cap\left\{\mathbf{x}\,|\,\exists q,\mathbf{x}\in\overline{\mathbb{Q}^{-q}},\mathcal{L}_{a}^{c}(\mathbf{x},q)\geq 0\wedge\mathcal{L}_{b}^{c}(\mathbf{x},q)\leq 0\right\}\\ & = & \partial\mathbb{A}\cap\bigcup_{q\in\{-1,1\}}\overline{\mathbb{Q}^{-q}}\cap\overline{\mathbb{L}_{a}^{c}(q)}\cap\mathbb{L}_{b}^{c}(q).\end{array} \tag{13}$$

Figure 6 illustrates the principle of this proposition in the case where $\mathbb{A}$ is described by one inequality $c(\mathbf{x})\leq 0$ and with no discrete variable q. In this case

$$\mathbb{S}(\mathbb{A})=\partial\mathbb{A}\cap\{\mathbf{x}\,|\,\mathcal{L}_{a}^{c}(\mathbf{x})\geq 0\wedge\mathcal{L}_{b}^{c}(\mathbf{x})\leq 0\}. \tag{14}$$

The boundary $\partial\mathbb{A}$ of $\mathbb{A}$ is composed of four parts :

$$\begin{array}{rcl}\partial\mathbb{A}\cap\overline{\mathbb{L}_{a}^{c}(q)}\cap\overline{\mathbb{L}_{b}^{c}(q)} & \rightarrow & \text{magenta}\\ \partial\mathbb{A}\cap\overline{\mathbb{L}_{a}^{c}(q)}\cap\mathbb{L}_{b}^{c}(q) & \rightarrow & \text{red}\\ \partial\mathbb{A}\cap\mathbb{L}_{a}^{c}(q)\cap\overline{\mathbb{L}_{b}^{c}(q)} & \rightarrow & \text{yellow}\\ \partial\mathbb{A}\cap\mathbb{L}_{a}^{c}(q)\cap\overline{\mathbb{L}_{b}^{c}(q)} & \rightarrow & \text{black}\end{array}$$

One trajectory (dotted line) $\mathbf{x}(t)$ is also represented. Before the yellow arc, $c(\mathbf{x})$ is positive and decreases. When it crosses the yellow arc, $c(\mathbf{x})=0$ for some isolated

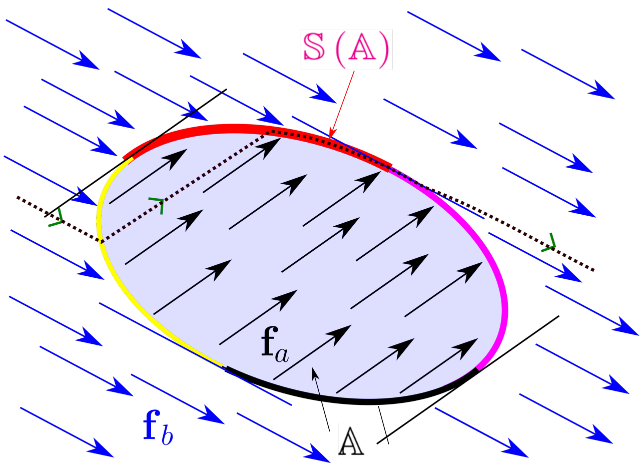

Figure 6: Sliding set $\mathbb{S}(\mathbb{A})$ (red) for $\mathbb{A} = \{\mathbf{x}|c(\mathbf{x}) \leq 0\}$

time point t_1. Then $\mathbf{x}(t)$ remains inside $\mathbb{A}$ until it reaches the red arc. It slides in the red arc for some non-degenerated time interval. When $\mathbf{x}(t)$ reaches the magenta arc, it leaves $\mathbb{A}$.

Proposition 3. Consider two closed sets $\mathbb{A}_1$ and $\mathbb{A}_2$. As illustrated by Figure 7, we have

$$\begin{array}{rlcl} (i) & \mathbb{S}(\mathbb{A}_1 \cap \mathbb{A}_2) & = & (\mathbb{S}(\mathbb{A}_1) \cap \mathbb{A}_2) \cup (\mathbb{S}(\mathbb{A}_2) \cap \mathbb{A}_1) \\ (ii) & \mathbb{S}(\mathbb{A}_1 \cup \mathbb{A}_2) & = & \left(\mathbb{S}(\mathbb{A}_1) \cap \mathrm{clo}\overline{\mathbb{A}_2}\right) \cup \left(\mathbb{S}(\mathbb{A}_2) \cap \mathrm{clo}\overline{\mathbb{A}_1}\right) \end{array} \quad (15)$$

Proof. Let us first prove (i). If $\mathbf{x} \in \mathbb{S}(\mathbb{A}_1 \cap \mathbb{A}_2)$, then $\mathbf{x}$ belongs to the boundary $\partial(\mathbb{A}_1 \cap \mathbb{A}_2)$ of $\mathbb{A}_1 \cap \mathbb{A}_2$. Now, since $\mathbb{A}_1, \mathbb{A}_2$ are both closed, we have $\partial(\mathbb{A}_1 \cap \mathbb{A}_2) = (\partial\mathbb{A}_1 \cap \mathbb{A}_2) \cup (\partial\mathbb{A}_2 \cap \mathbb{A}_1)$. Thus, we have to consider two cases: (a) $\mathbf{x} \in \partial\mathbb{A}_1 \cap \mathbb{A}_2$ and the system slides on $\partial\mathbb{A}_1$ (i.e., $\mathbf{x} \in \mathbb{S}(\mathbb{A}_1)$) or (b) $\mathbf{x} \in \partial\mathbb{A}_2 \cap \mathbb{A}_1$ and the system slides on $\partial\mathbb{A}_2$ (i.e., $\mathbf{x} \in \mathbb{S}(\mathbb{A}_2)$). Considering the two cases, we get

$$\begin{array}{rcl} \mathbb{S}(\mathbb{A}_1 \cap \mathbb{A}_2) & = & (\partial\mathbb{A}_1 \cap \mathbb{A}_2 \cap \mathbb{S}(\mathbb{A}_1)) \cup (\partial\mathbb{A}_2 \cap \mathbb{A}_1 \cap \mathbb{S}(\mathbb{A}_2)) \\ & = & (\mathbb{A}_2 \cap \mathbb{S}(\mathbb{A}_1)) \cup (\mathbb{A}_1 \cap \mathbb{S}(\mathbb{A}_2)). \end{array} \quad (16)$$

Let us now prove (ii). If $\mathbf{x} \in \mathbb{S}(\mathbb{A}_1 \cup \mathbb{A}_2)$, then $\mathbf{x}$ belongs to the boundary $\partial(\mathbb{A}_1 \cup \mathbb{A}_2)$ of $\mathbb{A}_1 \cup \mathbb{A}_2$. Now, $\partial(\mathbb{A}_1 \cup \mathbb{A}_2) = (\partial\mathbb{A}_1 \cap \mathrm{clo}\mathbb{B}_2) \cup (\partial\mathbb{A}_2 \cap \mathrm{clo}\mathbb{B}_1)$. Again, we have to consider two cases: (a) $\mathbf{x} \in (\partial\mathbb{A}_1 \cap \mathrm{clo}\mathbb{B}_2)$ and then $\mathbf{x} \in \mathbb{S}(\mathbb{A}_1) \cap \mathrm{clo}\mathbb{B}_2$ and (b) $\mathbf{x} \in (\partial\mathbb{A}_2 \cap \mathrm{clo}\mathbb{B}_1)$ then $\mathbf{x} \in \mathbb{S}(\mathbb{A}_2) \cap \mathrm{clo}\mathbb{B}_1$.■

Proposition 3 can be used to compute the sliding surface of a set $\mathbb{A}$ as soon as $\mathbb{A}$ can be defined by inequalities connected by Boolean operators such as *and, or, not.* The proposition is illustrated by Figure 8 in the case where $\mathbb{A} = \mathbb{A}_1 \cup (\mathbb{A}_2 \cap \mathbb{A}_3)$ and $\mathbb{A}_i = \{\mathbf{x}|c_i(\mathbf{x}) \leq 0\}$. The trajectory (green) slides twice, first on $\partial\mathbb{A}_1$, then it slides on $\partial\mathbb{A}_2$. The sliding surfaces are painted red.

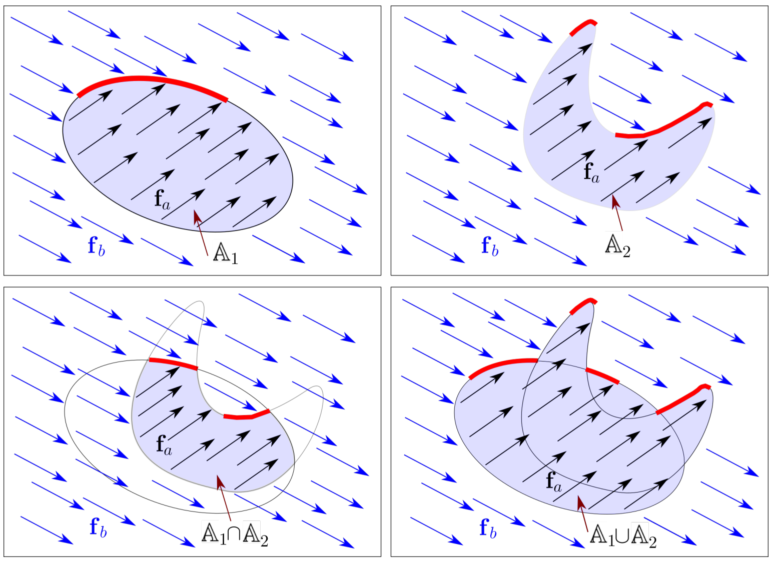
$\mathbf{f}_a$
$\mathbf{f}_b$
$\mathbb{A}_1$
$\mathbb{A}_2$
$\mathbb{A}_1 \cap \mathbb{A}_2$
$\mathbb{A}_1 \cup \mathbb{A}_2$

Figure 7: Illustration of Proposition 3, the sliding surfaces are painted red

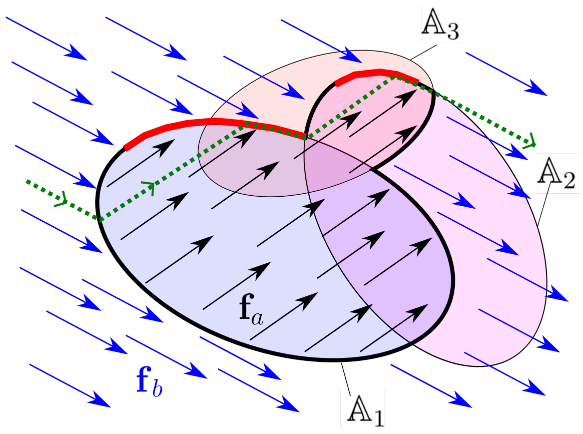

Figure 8: Sliding surfaces for $\mathbb{A} = \mathbb{A}_1 \cup (\mathbb{A}_2 \cap \mathbb{A}_3)$

4 Application to our easy-boat model

Taking into account the dynamic in (1), the controller given in Section 2, and setting $\mathbf{x} = (d, \psi)$, we obtain the following evolution function for the closed loop easy-boat model:

```
**Function** $\mathbf{f}(\mathbf{x}, q)$
If $\cos(x_2 + \operatorname{atan} x_1) + \cos\frac{\pi}{4} \leq 0 \vee \left(x_1^2 - 1 \leq 0 \wedge \cos x_2 + \cos\frac{\pi}{4} \leq 0\right)$
    then return $\begin{pmatrix} \sin\left(\pi + x_2 - q\frac{\pi}{4}\right) \\ 0 \end{pmatrix}$
    else return $\begin{pmatrix} \sin(-\operatorname{atan} x_1) \\ 0 \end{pmatrix}$
```

Therefore, our easy-boat model can be described by the expandable form (3) by taking the following correspondences:

$$\begin{array}{l} \mathbf{x} = (d, \psi) \\ \mathbf{f}_a(\mathbf{x}, q) = \begin{pmatrix} \sin\left(\pi + x_2 - q\frac{\pi}{4}\right) \\ 0 \end{pmatrix} \\ \mathbf{f}_b(\mathbf{x}) = \begin{pmatrix} \sin(-\operatorname{atan} x_1) \\ 0 \end{pmatrix} \\ \mathbb{A}_1 = \left\{\mathbf{x} \mid \cos(x_2 + \operatorname{atan} x_1) + \cos\frac{\pi}{4} \leq 0\right\} \\ \mathbb{A}_2 = \left\{\mathbf{x} \mid x_1^2 - 1 \leq 0\right\} \\ \mathbb{A}_3 = \left\{\mathbf{x} \mid \cos x_2 + \cos\frac{\pi}{4} \leq 0\right\} \\ \mathbb{A} = \mathbb{A}_1 \cup (\mathbb{A}_2 \cap \mathbb{A}_3) \\ \mathbb{Q}^- = \{\mathbf{x} \mid x_1 + 1 \leq 0\} \\ \mathbb{Q}^+ = \{\mathbf{x} \mid 1 - x_1 \leq 0\} \end{array} \tag{17}$$

We can now illustrate the resolution of the three problems treated at Section 3.

4.1 Constraint satisfaction

Using Proposition 1, we want to prove that the easyboat never goes upwind (see Equation (2)), i.e., we never have

$$\cos(x_2 - u) + \cos\frac{\pi}{5} \leq 0 \tag{18}$$

where u is given by the controller (see Section 2)

$$\left\{ \begin{array}{lll} u = & \pi + x_2 - q\frac{\pi}{4} & \text{if } \mathbf{x} \in \mathbb{A} \\ = & -\operatorname{atan} x_1 & \text{otherwise} \end{array} \right. \tag{19}$$

Thus, the no-go zone constraint can be expressed as

$$h(\mathbf{x}) = \cos(x_2 - u) + \cos\frac{\pi}{5} \leq 0 \tag{20}$$

with

$$\left\{\begin{array}{ccccll} h(\mathbf{x}) & = & h_a(\mathbf{x}) & = & \cos\left(-\pi - q\frac{\pi}{4}\right) + \cos\frac{\pi}{5} & \text{if } \mathbf{x}\in\mathbb{A} \\ & & & = & \cos\frac{3\pi}{4} + \cos\frac{\pi}{5} & \\ & = & h_b(\mathbf{x}) & = & \cos(x_2 + \operatorname{atan} x_1) + \cos\frac{\pi}{5} & \text{otherwise} \end{array}\right. \tag{21}$$

As required by (8), we compute the set

$$\mathbb{H} = \left(\mathbb{H}_a(1)\cap\overline{\mathbb{Q}^-}\cap\mathbb{A}\right)\cup\left(\mathbb{H}_a(-1)\cap\overline{\mathbb{Q}^+}\cap\mathbb{A}\right)\cup(\mathbb{H}_b\cap\mathbb{B}) \tag{22}$$

where

$$\begin{array}{ccc} \mathbb{H}_a(q) & = & \{\mathbf{x}|h_a(\mathbf{x},q)\le 0\} \\ \mathbb{H}_b & = & \{\mathbf{x}|h_b(\mathbf{x})\le 0\} \end{array} \tag{23}$$

Using the interval based solver PyIbex[1] we easily show that this set has no solution. From Proposition 1, we conclude that the forbidden constraint $\cos(x_2-u)+\cos\frac{\pi}{5}\le 0$ is never reached.

4.2 Capture set

To show that the easyboat stays inside a corridor of radius 2, we take $V(\mathbf{x}) = x_1^2-4$. We have

$$\begin{array}{ccccc} \mathcal{L}_a^V(\mathbf{x},q) & = & \frac{dV}{d\mathbf{x}}(\mathbf{x})\cdot\mathbf{f}_a(\mathbf{x},q) & = & 2x_1\cdot\sin\left(\frac{q\pi}{4}-x_2\right) \\ \mathcal{L}_b^V(\mathbf{x}) & = & \frac{dV}{d\mathbf{x}}(\mathbf{x})\cdot\mathbf{f}_b(\mathbf{x},q) & = & \frac{-2x_1^2}{\sqrt{x_1^2+1}} \end{array} \tag{24}$$

We compute the set

$$\mathbb{V} = \left(\overline{\mathbb{L}_a^V(1)}\cap\overline{\mathbb{Q}^-}\cap\mathbb{A}\right)\cup\left(\overline{\mathbb{L}_a^V(-1)}\cap\overline{\mathbb{Q}^+}\cap\mathbb{A}\right)\cup\left(\overline{\mathbb{L}_b^V}\cap\mathbb{B}\right) \tag{25}$$

where

$$\begin{array}{ccc} \mathbb{L}_a^V(q) & = & \left\{\mathbf{x}|\mathcal{L}_a^V(\mathbf{x},q)\le 0\right\} \\ \mathbb{L}_b^V & = & \left\{\mathbf{x}|\mathcal{L}_b^V(\mathbf{x})\le 0\right\} \end{array} \tag{26}$$

Since we need to compute with sets defined by non-linear inequalities that are connected with intersection, union, complementary operators, we decided to use separators [15] instead of contractors [5] (which do not allow the use of complementary operators).

We prove that set $\mathbb{V}\cap\overline{\mathbb{C}}$ is empty using PyIbex. From Proposition 2, we conclude that $\mathbb{C}$ is a capture set.

[1] `http://benensta.github.io/pyIbex/`

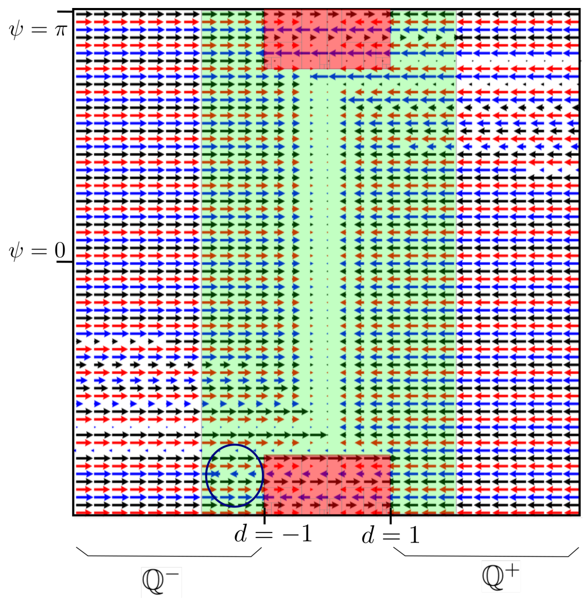

Figure 9: Fields $\mathbf{f}_a(\mathbf{x}, q)$, $\mathbf{f}_b(\mathbf{x})$, the set $\mathbb{C}$ (green) and the set $\mathbb{V}$ (red)

Figure 9 gives a superposition of the fields for $\mathbf{f}_a(\mathbf{x}, 1)$ (blue), $\mathbf{f}_a(\mathbf{x}, -1)$ (black) and $\mathbf{f}_b(\mathbf{x})$ (red). Is also represented the capture set $\mathbb{C}$ (green) and the set $\mathbb{V}$ (red) which may not respect the constraint as soon as it is inside $\mathbb{C}$. Since the wind is constant, the arrows are horizontal. Since we have $\mathbf{x} \in \mathbb{Q}^- \Rightarrow q = -1$, the blue arrow going left in the blue circle cannot be reached by a trajectory. From the figure, we can see that outside $\mathbb{C}$,all fields are oriented toward the line $d = 0$ which is consistent with the results obtained in [14].

4.3 Sliding surface

Assume that for all i, $\mathbb{A}_i$ is defined by the inequality $c_i(\mathbf{x}) \leq 0$, $\mathbb{B}$ by $c_i(\mathbf{x}) > 0$ and the boundary $\partial\mathbb{A}_i$ by $c_i(\mathbf{x}) = 0$. From (13), the sliding surface for $\mathbb{A}_i$ is

$$\begin{array}{rcl} \mathbb{S}(\mathbb{A}_i) & = & \partial\mathbb{A}_i \cap \bigcup_{q\in\{-1,1\}} \overline{\mathbb{Q}^{-q}} \cap \overline{\mathbb{L}_a^i(q)} \cap \mathbb{L}_b^i \\ & = & \partial\mathbb{A}_i \cap \mathbb{L}_b^i \cap \left(\overline{\mathbb{L}_a^i(1)} \cap \overline{\mathbb{Q}^-} \cup \overline{\mathbb{L}_a^i(-1)} \cap \overline{\mathbb{Q}^+}\right) \end{array} \qquad (27)$$

where

$$\begin{array}{rcl} \mathbb{L}_a^i(q) & = & \{\mathbf{x} | \mathcal{L}_a^{c_i}(\mathbf{x}, q) \leq 0\} \\ \mathbb{L}_b^i & = & \{\mathbf{x} | \mathcal{L}_b^{c_i}(\mathbf{x}) \leq 0\} \end{array} \qquad (28)$$

Now, we have

$$\begin{array}{rcccc}
\mathcal{L}_a^{c_1}(\mathbf{x},q) & = & \frac{dc_1}{d\mathbf{x}}(\mathbf{x})\cdot\mathbf{f}_a(\mathbf{x},q) & = & \frac{-\sin(\frac{q\pi}{4}-x_2)\cdot\sin(\mathrm{atan}(x_1)+x_2)}{x_1^2+1} \\
\mathcal{L}_b^{c_1}(\mathbf{x}) & = & \frac{dc_1}{d\mathbf{x}}(\mathbf{x})\cdot\mathbf{f}_b(\mathbf{x},q) & = & \frac{\sin(\mathrm{atan} x_1+x_2)\cdot x_1}{\sqrt{x_1^2+1}^3} \\
\mathcal{L}_a^{c_2}(\mathbf{x},q) & = & \frac{dc_2}{d\mathbf{x}}(\mathbf{x})\cdot\mathbf{f}_a(\mathbf{x}) & = & 2\sin(\frac{q\pi}{4}-x_2)\cdot x_1 \\
\mathcal{L}_b^{c_2}(\mathbf{x}) & = & \frac{dc_2}{d\mathbf{x}}(\mathbf{x})\cdot\mathbf{f}_b(\mathbf{x},q) & = & \frac{-2x_1^2}{\sqrt{x_1^2+1}} \\
\mathcal{L}_a^{c_3}(\mathbf{x},q) & = & \frac{dc_3}{d\mathbf{x}}(\mathbf{x})\cdot\mathbf{f}_a(\mathbf{x},q) & = & 0 \\
\mathcal{L}_b^{c_3}(\mathbf{x}) & = & \frac{dc_3}{d\mathbf{x}}(\mathbf{x})\cdot\mathbf{f}_b(\mathbf{x},q) & = & 0
\end{array} \tag{29}$$

$$\begin{array}{rcc}
\mathbb{S}(\mathbb{A}_1) & = & \partial\mathbb{A}_1\cap\mathbb{L}_b^1\cap\left(\overline{\mathbb{L}_a^1(1)}\cap\overline{\mathbb{Q}^-}\cup\overline{\mathbb{L}_a^1(-1)}\cap\overline{\mathbb{Q}^+}\right) \\
\mathbb{S}(\mathbb{A}_2) & = & \partial\mathbb{A}_2\cap\mathbb{L}_b^2\cap\left(\overline{\mathbb{L}_a^2(1)}\cap\overline{\mathbb{Q}^-}\cup\overline{\mathbb{L}_a^2(-1)}\cap\overline{\mathbb{Q}^+}\right) \\
\mathbb{S}(\mathbb{A}_3) & = & \partial\mathbb{A}_3
\end{array} \tag{30}$$

Thus

$$\begin{array}{rcc}
\mathbb{S}(\mathbb{A}_1\cup(\mathbb{A}_2\cap\mathbb{A}_3)) & = & \left(\mathbb{S}(\mathbb{A}_1)\cap\mathrm{clo}\left(\overline{\mathbb{A}_2\cap\mathbb{A}_3}\right)\right)\cup\left(\mathbb{S}(\mathbb{A}_2\cap\mathbb{A}_3)\cap\mathrm{clo}\overline{\mathbb{A}_1}\right) \\
\mathbb{S}(\mathbb{A}_2\cap\mathbb{A}_3) & = & (\mathbb{S}(\mathbb{A}_2)\cap\mathbb{A}_3)\cup(\mathbb{S}(\mathbb{A}_3)\cap\mathbb{A}_2)
\end{array} \tag{31}$$

The abstract syntax tree associated to the expression of the sliding surface $\mathbb{S}$ is depicted on Figure 10. It can be generated automatically using the rules provided by Proposition 3. The complexity of the tree illustrates the advantage of using separator algebra for the characterization of the solution set.

We obtain Figure 11 where two horizontal segments appear. They correspond to a wind angle corresponding to $\pm\frac{3\pi}{4}$ as expected.

To have a deeper understanding, let us draw the trajectories associated to the simulations of Figure 2 (see also Figure 12). The red set, obtained with PyIbex, corresponds to $\mathbb{A} = \mathbb{A}_1\cup(\mathbb{A}_2\cap\mathbb{A}_3)$. We can see that most of trajectories cross the singularities at one time t. But the red stays on the sliding surface for time period that maybe long. Thus make the sailboat loosing a lot of time due to many unneeded maneuvers. The controller alternates indefinitely between two strategies: $\bar{\theta} := \varphi$ and $\bar{\theta} := \pi+\psi-q\zeta$. Recall that this hesitation can be seen on simulations but also sometimes for short periods during real experiments with our actual sailboat.

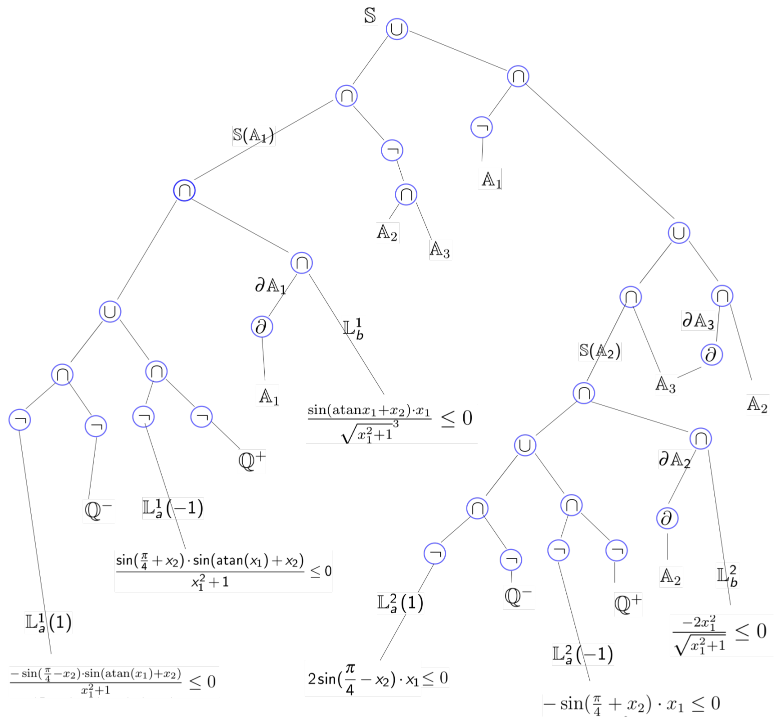

Figure 10: Abstract syntax tree associated to the expression of $\mathbb{S}$

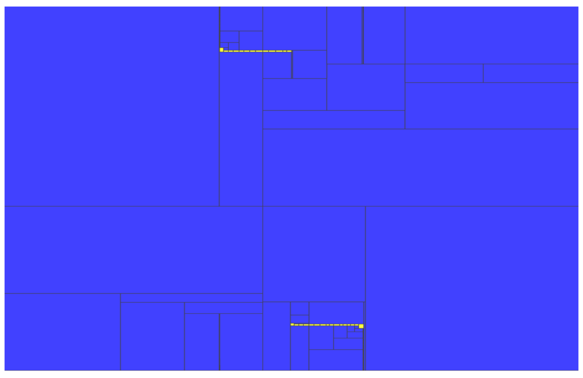

Figure 11: Sliding surface (yellow)

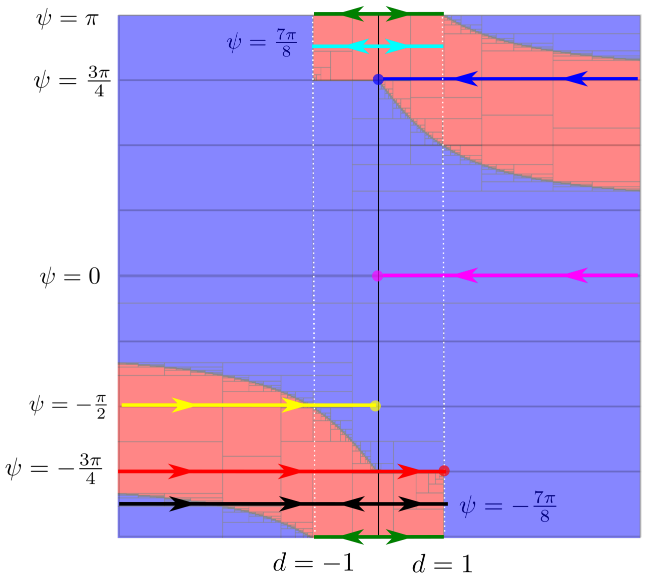

Figure 12: Several trajectories in the state space

5 Conclusion

In this paper, we have presented a new approach based on contractor/separator programming to compute the sliding surfaces of a cyber-physical system. If the state of the system is on this surface, it may hesitate indefinitely between two different strategies. As a result, the system may be trapped on this surface and the designed mission may fail. It is thus important to detect and compute the sliding surface in order to eliminate them by changing the controller.

Further researches we would like to address in the future are the following.

- Generalize the method to situations where we have more than two continuous evolution functions $f_i, i \in \{a, b, \ldots\}$ and where q may take more than two values.
- Take into account quantifiers to consider different kinds of uncertainties [11].
- Build a tool able to cast automatically a physical system with a controller described by an algorithm with if-statements into the expandable form (3). This could be done, for instance, by obtaining a *disjonctive normal form* (BNF) of the controller. Or equivalently to replace all *if-then-else* in the controller by a single *switch-case* statement.
- Find a new controller for our sailboat, as efficient as the existing one, but without any sliding surface.

References

[1] Asarin, E., Dang, T., and Girard, A. Hybridization methods for the analysis of non-linear systems. *Acta Informatica*, 7(43):451–476, 2007. DOI: 10.1007/s00236-006-0035-7.

[2] Asarin, E., Dang, T., and Maler, O. The d/dt tool for verification of hybrid systems. In *In International Conference on Computer Aided Verification*, pages 365–370. Springer, 2002. DOI: 10.1007/3-540-45657-0_30.

[3] Blanchini, F. and Miani, S. *Set-Theoretic Methods in Control.* Springer Science & Business Media, October 2007. DOI: 10.1007/978-0-8176-4606-6.

[4] Bouissou, O., Chapoutot, A., Djaballah, A., and Kieffer, M. Computation of parametric barrier functions for dynamical systems using interval analysis. In *2014 IEEE 53rd Annual Conference on Decision and Control (CDC)*, pages 753–758, December 2014. DOI: 10.1109/CDC.2014.7039472.

[5] Chabert, G. and Jaulin, L. Contractor Programming. *Artificial Intelligence*, 173:1079–1100, 2009. DOI: 10.1016/j.artint.2009.03.002.

[6] Cousot, P. and Cousot, R. Abstract interpretation: A unified lattice model for static analysis of programs by construction or approximation of fixpoints. In *Conference Record of the Fourth ACM Symposium on Principles of Programming Languages*, pages 238–252, Los Angeles, California, 1977. DOI: 10.1145/512950.512973.

[7] Delanoue, N., Jaulin, L., and Cottenceau, B. Attraction domain of a nonlinear system using interval analysis. In *Twelfth International Conference on Principles and Practice of Constraint Programming (IntCP 2006)*, France, Nantes, 2006.

[8] Drakunov, S. and Utkin, V. Sliding mode control in dynamic systems. *International Journal of Control*, 55(4):1029–1037, 1992. DOI: 10.1080/00207179208934270.

[9] Frehse, G. Phaver: Algorithmic verification of hybrid systems. *International Journal on Software Tools for Technology Transfer*, 10(3):23–48, 2008. DOI: 10.1007/978-3-540-31954-2_17.

[10] Fribourg, L. and Soulat, R. *Control of Switching Systems by Invariance Analysis: Application to Power Electronics.* Wiley-ISTE, 2013. DOI: 10.1002/9781118791486.

[11] Goldsztejn, A. and Chabert, G. On the approximation of linear AE-solution sets. In *12th International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics, Duisburg, Germany, (SCAN 2006)*, 2006. DOI: 10.1109/SCAN.2006.33.

[12] Goubault, E. and Putot, S. Static analysis of numerical algorithms. In *In Proceedings of SAS 06, LNCS 4134*, pages 18–34. Springer-Verlag, 2006. DOI: `10.1007/11823230_3`.

[13] Jaulin, L. and Bars, F. Le. A simple controller for line following of sailboats. In *5th International Robotic Sailing Conference*, pages 107–119, Cardiff, Wales, England, 2012. Springer. DOI: `10.1007/978-3-642-33084-1_11`.

[14] Jaulin, L. and Bars, F. Le. An Interval Approach for Stability Analysis; Application to Sailboat Robotics. *IEEE Transaction on Robotics*, 27(5), 2012. DOI: `10.1109/TRO.2012.2217794`.

[15] Jaulin, L. and Desrochers, B. Introduction to the algebra of separators with application to path planning. *Engineering Applications of Artificial Intelligence*, 33:141–147, 2014. DOI: `10.1016/j.engappai.2014.04.010`.

[16] Konecny, M., Taha, W., Duracz, J., Duracz, A., and Ames, A. Enclosing the behavior of a hybrid system up to and beyond a zeno point. In *Cyber-Physical Systems, Networks, and Applications (CPSNA)*, 2013. DOI: `10.1109/CPSNA.2013.6614258`.

[17] Kreinovich, V., Lakeyev, A.V., Rohn, J., and Kahl, P.T. Computational complexity and feasibility of data processing and interval computations. *Reliable Computing*, 4(4):405–409, 1997.

[18] Mézo, T. Le, Jaulin, L., and Zerr, B. An interval approach to compute invariant sets. *IEEE Transaction on Automatic Control*, 62:4236–4243, 2017. DOI: `10.1109/TAC.2017.2685241`.

[19] Mitchell, I. Comparing forward and backward reachability as tools for safety analysis. In Bemporad, A., Bicchi, A., and Buttazzo, G., editors, *Hybrid Systems: Computation and Control*, pages 428–443. Springer-Verlag, 2007. DOI: `10.1007/978-3-540-71493-4_34`.

[20] Mitchell, I., Bayen, A., and Tomlin, C. Validating a Hamilton-Jacobi Approximation to Hybrid System Reachable Sets. In Benedetto, M. and Sangiovanni-Vincentelli, A., editors, *Hybrid Systems: Computation and Control*, number 2034 in Lecture Notes in Computer Science, pages 418–432. Springer Berlin Heidelberg, 2001. DOI: `10.1007/3-540-45351-2_34`.

[21] Moore, R. E. *Methods and Applications of Interval Analysis.* SIAM, Philadelphia, PA, 1979. DOI: `10.1137/1.9781611970906`.

[22] Ramdani, N. and Nedialkov, N. Computing Reachable Sets for Uncertain Nonlinear Hybrid Systems using Interval Constraint Propagation Techniques. *Nonlinear Analysis: Hybrid Systems*, 5(2):149–162, 2011. DOI: `10.1016/j.nahs.2010.05.010`.

[23] Ratschan, S. Approximate quantified constraint solving by cylindrical box decomposition. *Reliable Computing*, 8(1):21–42, 2002.

[24] Ratschan, S. and She, Z. Providing a Basin of Attraction to a Target Region of Polynomial Systems by Computation of Lyapunov-like Functions . *SIAM J. Control and Optimization*, 48(7):4377–4394, 2010. DOI: `10.1109/ICCCYB.2006.305705`.

[25] Rauh, A. and Auer, E. Interval approaches to reliable control of dynamical systems. In *Computer-assisted proofs - tools, methods and applications*, 2009.

[26] Rohou, S., Jaulin, L., Mihaylova, M., Bars, F. Le, and Veres, S. Reliable non-linear state estimation involving time uncertainties. *Automatica*, pages 379–388, 2018. DOI: `10.1016/j.automatica.2018.03.074`.

[27] Saint-Pierre, P. Hybrid kernels and capture basins for impulse constrained systems. In Tomlin, C.J. and Greenstreet, M.R., editors, *in Hybrid Systems: Computation and Control*, volume 2289, pages 378–392. Springer-Verlag, 2002. DOI: `10.1007/3-540-45873-5_30`.

[28] Sandretto, J. Alexandre Dit and Chapoutot, A. Validated simulation of differential algebraic equations with Runge-Kutta methods. *Reliable Computing*, 22, 2016.

[29] Taha, W. and Duracz, A. Acumen: An open-source testbed for cyber-physical systems research. In *CYCLONE'15*, 2015. DOI: `10.1007/978-3-319-47063-4_11`.

[30] Wilczak, D. and Zgliczynski, P. Cr-Lohner algorithm. *Schedae Informaticae*, 20:9–46, 2011. DOI: `10.4467/20838476SI.11.001.0287`.