Automatica 187 (2026) 112897

Contents lists available at ScienceDirect

Automatica

journal homepage: www.elsevier.com/locate/automatica

Fault-tolerant control of nonlinear systems: An inductive synthesis approach☆

Daniele Masti [a], Davide Grande [b,*], Andrea Peruffo [c], Filippo Fabiani [d]

[a] Gran Sasso Science Institute, Viale F. Crispi 7, 67100 L'Aquila, Italy
[b] University College London, Gower St, WC1E 6BT London, UK
[c] TNO-ESI, High Tech Campus 25, 5656 AE Eindhoven, The Netherlands
[d] IMT School for Advanced Studies Lucca, Piazza San Francesco 19, 55100 Lucca, Italy

ARTICLE INFO

Article history:
Received 14 March 2025
Received in revised form 14 October 2025
Accepted 6 January 2026

Keywords:
Fault-tolerant
Computer-aided design tools
Guidance
Navigation and control of vehicles
Algorithms and software
Robust control of nonlinear systems
Control of constrained systems

ABSTRACT

Actuator faults greatly affect the performance and stability of control systems, an issue that is even more critical for systems required to operate autonomously under adverse environmental conditions, such as unmanned vehicles. To this end, passive fault-tolerant control (PFTC) systems can be employed, namely fixed-gain control laws that guarantee stability both in the nominal case and in the event of faults. In this paper, we propose a counterexample guided inductive synthesis (CEGIS)-based approach to design reliable PFTC policies for nonlinear control systems. Our approach takes into account actuator saturation, tackles both partial and total actuator faults, and employs a synthesis technique guaranteed to converge within finite-time. Extensive numerical simulations illustrate how the proposed method can be applied to realistic operational scenarios involving the control of velocity and heading of autonomous underwater vehicles (AUVs). Our PFTC technique exhibits comparatively short synthesis time (i.e. minutes) and requires low computational cost. These features render the presented method particularly suitable for embedded applications with limited availability of onboard energy and power resources.

1. Introduction

Unmanned vehicles are usually employed over long periods of time in unstructured or adverse environments to accomplish several tasks, *e.g.*, data collection, environmental monitoring and patrolling. During the mission, these robots may be subject to actuator or sensor faults. In words, a fault amounts to an undesired change in the dynamics of a signal of a sensor or an actuator, which does not compromise the entire functionality of the overall system (Ducard, 2009). A fault can be worked around so that the system can still perform its originally intended task, even with a certain degree of performance degradation (Blanke et al., 2006). Owing to the persistent exposure of the actuators to the surrounding environment, actuator faults represent one of the key issues in operations in domains such as underwater robotics (Liu et al., 2023), which will be the main application focus of this study. Faults at actuators in underwater vehicle operations have been associated to collisions with other vessels or with detrita (Queste et al., 2012), biofouling agents (Anderlini et al., 2021), or attacks by marine creatures (Grande, 2024, Ch. 2). An extensive report surveying 205 missions totalling over 5000 days of AUV deployments, concluded that 11 missions were aborted due to faults at actuators (one fault at any one time) (Brito et al., 2014), rendering actuator fault-tolerance a problem of academic interest with practical and timely field robotics implications. With the high design costs and the associated high risk of operating autonomous vehicles in challenging environments, extensive studies focused on reducing the actuator failure rate.

Dealing with actuator faults is traditionally addressed via two nearly complementary approaches. On the one hand, fault detection and isolation (FDI) algorithms (Caiti et al., 2015; Fabiani et al., 2016) can be designed to detect the presence of non-nominal operating conditions, isolating the faulty component and estimating the severity of the fault occurrence. As a consequence, the system's control policy is either rescheduled, or a completely new control architecture, tailored for the specific faulty behaviour, is applied. These approaches fall within the active fault-tolerant control (AFTC) techniques. On the other hand, PFTC methods consider a set of possible faulty dynamics and focus on the

☆ The material in this paper was not presented at any conference. This paper was recommended for publication in revised form by Associate Editor Angelo Alessandri under the direction of Editor Thomas Parisini (Control System Applications).

* Corresponding author.
E-mail addresses: daniele.masti@gssi.it (D. Masti), davide.grande@ucl.ac.uk, grande.rdev@gmail.com (D. Grande), andrea.peruffo.05@gmail.com (A. Peruffo), filippo.fabiani@imtlucca.it (F. Fabiani).

https://doi.org/10.1016/j.automatica.2026.112897

design of a fixed-gain control law, guaranteed to preserve closed-loop stability over the whole range of faulty behaviours. PFTC is frequently compared to solving a robust stabilisation problem, thereby requiring the simultaneous stabilisation of a plethora of dynamics catering for the nominal plant, and for the set of operating modes associated to the faults.

Related work: Leveraging the tight interaction between an FDI block with a control law scheduler or gain adaptive engine, AFTC methods offer the possibility to optimise a system's closed-loop performance in diverse operating conditions. However, AFTC techniques are typically computationally expensive, require accurate model descriptions, and need time to estimate the fault location and its severity, introducing a critical delay that might lead to instability (Verhaegen et al., 2010). PFTC architectures, instead, eliminate the need for monitoring sensors and are computationally inexpensive to implement, as they rely on fixed-gain controllers. When either extensive actuator sensoring, control design or online deployment possibilities are limited due to power, costs, or complexity constraints, PFTC represents the most effective control option. Although PFTCs result in more conservative control performance even under nominal operating conditions, in safety-critical applications, or when human-in-the-loop intervention is unfeasible, *e.g.,* AUVs operating under the polar ice caps (Webster et al., 2015), reliability and safety need to be prioritised (Kaminer et al., 1991), making PFTCs a premiere choice for our purposes. Available PFTC methods exploit both linear and nonlinear control theory. Linear methods conventionally stem from $\mathcal{H}_2$ and $\mathcal{H}_\infty$-control synthesis (Blanke et al., 2006; Kaminer et al., 1991); analytical nonlinear techniques instead rely on Lyapunov theory, for instance by designing a control law for the fault-free model, and adding extra terms catering for changes and/or faults in the actuator dynamics (Benosman & Lum, 2009). While coping with control affine nonlinear dynamics, the latter stream of methods can only deal with partial loss of actuator effectiveness and cannot take into account actuator saturation. Alternatively, the PFTC-ANLC method was proposed for nonlinear systems affected by partial and total actuator faults while accounting for actuator saturation (Grande, Fenucci, et al., 2023; Grande et al., 2024), but the synthesis of the control law does not have a formal theoretical guarantee of convergence.

Computer-aided design methods have recently been employed in the design of control systems. One of the most prominent examples refers to the CEGIS technique, which consists of a loop between two components, a learner and a verifier. Specifically, a *learner* is in charge of designing a candidate control solution, which is then handed over to a *verifier* that checks whether the candidate solution is valid over the whole state domain. CEGIS-based schemes have recently demonstrated their potential in the design of control Lyapunov functions (CLFs), or in extending the capabilities of traditional robust control approaches (Abate et al., 2022; Berger & Sankaranarayanan, 2022; Chang et al., 2019; Dai et al., 2020; Grande, Peruffo, et al., 2023; Masti et al., 2023; Solar-Lezama et al., 2006). In this regard, to overcome the intrinsic computational complexity resulting in the verifier's task, recent works focused on satisfiability modulo theory (SMT)-solvers as verification engines. For instance, ANLC (Grande, Peruffo, et al., 2023) and Fossil (Edwards et al., 2024) both employ SMT to verify the closed-loop stability of continuous-time systems, thus offering an alternative approach to standard techniques based on mixed-integer verification (Dai et al., 2020; Wu et al., 2024) or on Lipschitz-based optimisation (He et al., 2024; Masti et al., 2023). Despite the attractiveness of the automatic synthesis of correct-by-design CLFs, CEGIS-based methods are usually not formally guaranteed to converge. Besides introducing heuristics to increase the successful synthesis rate (Grande, Peruffo, et al., 2023), little work focused on theoretical guarantees of algorithmic termination. As we were finalising this manuscript, we became aware of a recent contribution (Hsieh et al., 2025) that presents an approach related to ours. However, the work in Hsieh et al. (2025) involves a significantly different technical analysis, leveraging Lipschitz continuity to learn Lyapunov functions for *unknown* dynamical systems, while relying on SMT for verification purposes.

Contribution: In this paper we propose a novel technique for PFTC, especially intended for AUVs that may be subject to multiple actuator faults. The devised method generates a unique, static, state-feedback robust control law to cope with both the nominal and multiple fault modes. This paper focuses on scenarios involving faults at a single actuator *at any one time*, owing to studies reporting that missions are often aborted due to a uniquely distinct root cause (*e.g.,* one buoyancy pump failure or one rudder stuck), and not to multiple actuator faults occurring at the same time (Brito et al., 2014). However, the method is not restricted to such limitation, as detailed later in this paper.

To this aim, we extend the CEGIS-based approach in Masti et al. (2023) (*i*) by considering *nonlinear* control systems; (*ii*) by tailoring the method to *fault-tolerant* policies; (*iii*) by supporting the method with theoretical results ensuring the finite-step convergence of the algorithm; and (*iv*) by including control inputs with actuator *saturations*. In contrast to Grande, Fenucci, et al. (2023), Grande et al. (2024), our method is based on a specialised verification engine designed to improve scalability, making this technique applicable to complex cyber–physical systems.

As a result, this paper illustrates a new PFTC method named *IS-sat*. IS-sat is the only PFTC method that, at the same time, (*i*) is capable of dealing with nonlinear systems; (*ii*) is designed to handle both partial and total faults at actuators; (*iii*) is able to cope with actuator saturation on every actuator; (*iv*) features a synthesis approach guaranteed to converge in finite-time; (*v*) can be easily implemented on unsophisticated hardware (i.e. requiring less than 1 MB of onboard RAM). As a further result of the presented analyses, we demonstrated how IS-sat can be synthesised in a few seconds/minutes using unsophisticated hardware for realistic nonlinear systems with multiple faults at actuators, ultimately showing how the method is promptly applicable to complex cyber–physical systems.

The rest of the paper is organised as follows: in Section 2 we formally introduce the PFTC problem under analysis; in Section 3 we detail our method while in Section 4 we show its experimental effectiveness over two nonlinear system benchmarks. The code developed is released open-source at: https://github.com/AndreaPuffo/IS-sat.

Notation

$\mathbf{1}_n$ is an n-dimensional vector of 1. $\mathbb{1}(\varphi)$ denotes the indicator of function of φ, i.e., $\mathbb{1}(\varphi) = 1$ when the condition φ is true, 0 otherwise. The operator $\text{conv}\{\cdot\}$ denotes the convex hull of its arguments. The operator $\text{sat}_{\mathscr{U}}(u)$ denotes the componentwise saturation function, i.e., for $u \in \mathbb{R}^p$,

$$\text{sat}_{\mathscr{U}}(u) = [\ldots,\ \text{sign}(u_i)\cdot\min\{u_{Mi}, |u_i|\},\ \ldots], \tag{1}$$

with sign function $\text{sign}(u_i)$ and i-th saturation threshold u_{Mi}.

2. Problem formulation and preliminaries

2.1. System description

Let us consider the following control-affine nonlinear model:

$$x(t+1) = f(x(t)) + g(x(t), \phi(t))u(t), \tag{2}$$

where $t \in \mathbb{N}_+$ denotes the discrete-time index, $x(t) \in \mathscr{D} \subseteq \mathbb{R}^n$ is the vector of state variables, $u(t) \in \mathscr{U} \subset \mathbb{R}^p$ is the vector

of control inputs, the mappings $f : \mathbb{R}^n \to \mathbb{R}^n$ and $g : \mathbb{R}^n \times [0,1]^p \to \mathbb{R}^{n\times p}$ are of class $\mathcal{C}^2$ with respect to (w.r.t.) $x(t)$, $\phi(t)$, and $\phi(t) \in [0,1]^p$ denotes the possible operation modes of the system. While $\phi_i = 1$, for all $i = 1,\dots,p$, represents the nominal operation, $\phi_i \in [0,1]$ models the loss of efficiency of actuator i, *e.g.,* $\phi_i = 0.6$ indicates that actuator i can only work at 60% of the nominal value. Finally, $\phi_i = 0$ denotes the total fault of actuator i. In this work, we assume that there can be just one entry of vector ϕ that assumes a value in $[0,1]$ at any one time, while the other entries are set to 1. More formally, we impose that $\phi(t) \in \Phi := \{\phi \in [0,1]^p \mid \sum_{i=1}^p \mathbb{1}(\phi_i = 1) \geq p-1\}$.

As mentioned, this assumption is commonly adopted in domain-specific literature, as faults are rare events. Yet, as we will discuss, it does not compromise the generality of the proposed method although it provides some computational simplification during the synthesis process. As such, multiple faults occurring during the same operation might require a complete stop of operations as the vehicles are not designed to cope with such unlikely scenarios. In the problem formulation presented in this section, we will consider how to design a unique control law for multiple faults, with at most one faults occurring *at every one time*, while extensions to *multiple contemporary* faults are discussed in Section 3.

Without loss of generality, we also assume the origin being an equilibrium point of (2), i.e., for a given $\bar{\phi} \in \Phi$, there exists some control law $\bar{u} \in \mathscr{U}$ such that $f(0) + g(0,\bar{\phi})\bar{u} = 0$.

For computational purposes, we finally constraint both the state variables and control inputs within polytopes $\mathscr{D} := \{x \in \mathbb{R}^n \mid Lx \leq \mathbf{1}_\ell\}$ and $\mathscr{U} := \{u \in \mathbb{R}^p \mid |u| \leq \bar{u}\}$, respectively, for some $L \in \mathbb{R}^{\ell\times n}$, $\operatorname{rank}(L) = \ell$, and $\bar{u} \in \mathbb{R}^p_+$. Note that $\mathscr{U}$ actually coincides with a saturation on the control input, element which we take into account in the presented derivation, in line with realistic control engineering applications.

Our goal is hence to design a control law for model (2) under all possible faults happening one at a time, i.e., under all possible values ϕ can take in Φ, with the aim of guaranteeing the maximal region of attraction, whilst abiding by the constraints imposed on both state variables and control inputs. As a byproduct of the control method we propose, a CLF is synthesised to certify the closed-loop stability.

2.2. Reformulation as uncertain system

Typically, a controller for system (2) is designed by means of robust control techniques based on, *e.g.,* system linearisation or Lyapunov arguments. The latter offers analytical solutions for a partial actuator fault, but conventionally cannot cope with the case of full loss of efficiency, namely $\phi_i = 0$ (Benosman & Lum, 2009). Inspired by the uncertain system literature, we reformulate the stabilisation of (2) as the uncertain model:

$$x(t+1) = A(t)x(t) + B(t)u(t), \tag{3}$$

where matrices $A(t) \in \mathbb{R}^{n\times n}$ and $B(t) \in \mathbb{R}^{n\times p}$ belong to:

$$\Omega := \left\{ (A,B) \mid A = \left.\frac{df}{dx}\right|_{x=\bar{x}}, \; B = \left.\frac{dg}{dx}\right|_{\substack{x=\bar{x}\\ \phi=\bar{\phi}}}, \text{ for } \bar{x} \in \mathscr{D},\ \bar{\phi} \in \Phi \right\}, \tag{4}$$

which collects all the Jacobian matrices $A(\cdot)$ of the autonomous (i.e. time-invariant) dynamics in (2), as well as those related to the affine control term $B(\cdot)$, which consider all possible variations associated to an actuator fault through $\phi \in \Phi$. We will then assume that both $A(t)$ and $B(t)$ are bounded for all $x \in \mathscr{D}$ and $t \in \mathbb{R}_+$, which is a natural condition for standard nonlinear models describing physical system, *e.g.,* the motion of AUVs (Fossen, 2011) considered as case study in Section 4. Note that, in view of the results in Kothare et al. (1996), Liu (1968), guaranteeing the closed-loop stability of (3) with $(A(t),B(t)) \in \Omega$, implies the local closed-loop stability of the original system (2), since Ω includes all possible behaviours that (2) may exhibit. In other words, designing a controller for the system in (3) yields a valid fault-tolerant policy for (2).

To conclude, note that Ω can assume any (possibly nonconvex) shape, which in view of Rudin et al. (1964, Theorem 4.14 and Theorem 4.22) it is however connected and compact. Moreover, recall that if Ω was a polytope with s vertices (as in, *e.g.,* (Rubin et al., 2020)), it could be written as $\Omega = \operatorname{conv}\{(A_i,B_i)\}_{i=1}^s$, also guaranteeing that some $\alpha \in \mathbb{R}^s_+$ exists so that $A(t) = \sum_{i=1}^s \alpha_i A_i$ and $B(t) = \sum_{i=1}^s \alpha_i B_i$, for all $t \geq 0$. This will be key for our subsequent developments.

2.3. Tackling input saturation

From our assumption on $\mathscr{U}$, we aim to find a PFTC law for (2) of the form:

$$u(t) = \operatorname{sat}_{\mathscr{U}}(Kx(t)), \tag{5}$$

where $K \in \mathbb{R}^{p\times n}$ is a linear gain to be designed, and the saturation function, namely $\operatorname{sat}_{\mathscr{U}}$, restricts the control actions to the set $\mathscr{U}$ (with each saturation limit set independently).

Various solutions were proposed to compute such a function. Among them, Hu and Lin (2001), Rubin et al. (2020) rely on linear difference inclusions (LDIs) . First, let us introduce an ellipsoid defined as $\mathcal{E}(Q) := \{x \in \mathbb{R}^n \mid x^\top Q^{-1} x \leq 1\}$, for some $Q \succ 0$ that has to be suitably designed. The saturation function can hence be reformulated using matrices $\{E_j\}_{j=1}^{2^p}$, $E_j \in \mathbb{R}^{p\times p}$, representing the collection of diagonal matrices with $(0,1)$-entries. Thus, the control law in (5) can be rewritten as:

$$\operatorname{sat}_{\mathscr{U}}(Kx) = \sum_{j=1}^{2^p} \sigma_j (E_j K x + E_j^- H x), \; \sum_{j=1}^{2^p} \sigma_j = 1, \; \sigma_j \geq 0, \tag{6}$$

where $E_j^- := I - E_j$, and $|Hx| < \bar{u}$ holds within the ellipsoid $\mathcal{E}(Q)$. This reformulation entails an extended control design problem, in which we strive to find a (possibly maximal) invariant ellipsoid[1] $\mathcal{E}(Q)$, along with matrices K, H.

To this end, we introduce the three matrices $Q \succ 0$, $Y = KQ$, $Z = HQ$, and we impose the following linear matrix inequality (LMI), where "$\star$" denotes the corresponding transposed block-element to make the left-hand side (LHS) symmetric:

$$\begin{bmatrix} \tau Q & \star & \star \\ 0 & (1-\tau)I & \star \\ A_i Q + B_i E_j Y + B_i E_j^- Z & 0 & Q \end{bmatrix} \succcurlyeq 0,$$

$$\text{for all } i = 1,\dots,s,\ j = 1,\dots,2^p, \tag{7a}$$

where $\tau \in [0,1]$ is an hyperparameter whose meaning is discussed in detail in Rubin et al. (2020, §4.1) and where the pair of matrices (A_i, B_i) represents the i-th vertex of Ω when Ω is defined as a polytope. This consideration will be key in the definition of the learner's task.

We then further add the LMI conditions to satisfy state and input constraints, as follows:

[1] Restricting our analysis to ellipsoidal domains of attraction only can be seen as limiting but represent a quite effective trade off between expressive power and computational affordability. See also Remark 3.1.

$$\begin{bmatrix} 1 & l_i Q \\ Q l_i^\top & Q \end{bmatrix} \succcurlyeq 0, \text{ for all } i = 1, \dots, \ell \tag{7b}$$

where l_i is the i-th row of matrix L, and

$$\begin{bmatrix} \bar{u}_i^2 & z_i \\ z_i^\top & Q \end{bmatrix} \succcurlyeq 0, \text{ for all } i = 1, \dots, p, \tag{7c}$$

where z_i is the i-th row of matrix Z. Finding a set of matrices satisfying (7) amounts to solving a semidefinite program (SDP), which is a convex optimisation problem. In our specific PFTC application, we will then incorporate (7) into an optimisation problem to find the control law with the maximal invariant ellipsoid, thus guaranteeing attraction for the largest possible region contained in $\mathscr{D}$. This can be accomplished by exploiting the following lemma:

Lemma 2.1 (*Rubin et al., 2020*)**.** *The largest invariant ellipsoid $\mathcal{E}(Q)$ w.r.t. system (2) and any saturated control $u(t) = \text{sat}_{\mathscr{U}}(Kx(t))$, can be computed as $K = YQ^{-1}$ by solving the SDP:*

$$\begin{aligned} \max_{Q,Y,Z} \quad & \text{trace}(Q) \\ \text{s.t.} \quad & \text{(7a), (7b), (7c).} \end{aligned} \tag{8}$$

□

Solving (8) returns three matrices: Q represents the invariant ellipsoid $\mathcal{E}(Q)$, Y is used to find the controller $K = YQ^{-1}$, and similarly $H = ZQ^{-1}$.

Despite its convexity, finding a solution to (8) is hard due to the combinatorial number of constraints (7a). The LDI formulation introduces 2^p LMIs, nested with 2^v constraints coming from the uncertainty set (v denotes the number of uncertain parameters of the matrices $A(t)$ and $B(t)$ belonging to Ω), producing a total of 2^{p+v} constraints. The SDP in (8) is hence computationally challenging even for low dimensional models. Moreover, to enhance generalisability of the present work, we do *not* restrict the study to Ω being necessarily a polytope. Nevertheless, Ω can be over-approximated, without loss of generality, by the convex hull composed of the 2^v uncertainty constraints.

3. CEGIS-based synthesis of PFTC policies

3.1. Counterexample-guided iterative procedure

Recently, the CEGIS learning paradigm has been adapted to the design of control functions for systems affected by actuator faults, *e.g.,* (Grande, Fenucci, et al., 2023; Grande et al., 2024). As illustrated in Fig. 1, CEGIS relies on the adversarial interaction between a *learner* and a *verifier*, aiming at designing a function $h : \mathbb{R}^n \to \mathbb{R}$ from a hypothesis space $\mathcal{H}$, by exploiting a dataset of (counter-)examples $\mathcal{S}_k$ (at the k-th algorithmic step). The procedure follows two iterative steps:

(1) The learner takes the dataset $\mathcal{S}_k$ as input and either synthesises a function $h_k \in \mathcal{H}$ according to a given criterion, or establishes that such a function cannot be synthesised;
(2) The verifier checks if $h_k \in \Theta$, with Θ representing a verification criterion, namely $\Theta = \{h : \mathbb{R}^n \to \mathbb{R} \mid r(h(z)) \leq 0, \text{ for all } z \in \mathcal{Z}\}$, for some $r : \mathbb{R} \to \mathbb{R}$ and set $\mathcal{Z} \subseteq \mathbb{R}^n$ over which the variable z takes values. The verifier can therefore take two different conclusions:
   (a) It finds a *counterexample* $\bar{z}_{k+1} \in \mathcal{Z}$ such that $r(h_k(\bar{z}_{k+1})) > 0$. In such a case $\mathcal{S}_{k+1} \leftarrow \mathcal{S}_k \cup \{\bar{z}_{k+1}\}$, and a new iteration step follows;
   (b) It certifies that no counterexample exists, yielding the successful conclusion of the procedure.

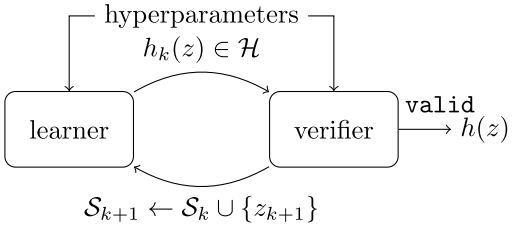


Fig. 1. At every iteration $k \in \mathbb{N}_+$, the learner proposes a candidate function $h_k(z)$, while the verifier checks its validity through $r(h(z)) \leq 0$.

To design PFTC policies, we propose to solve the SDP in (8) by using a set of strategically placed constraints which are added iteratively based on a CEGIS approach. In the following sections, we illustrate in detail how the learner and verifier can be designed to solve a PFTC problem.

3.2. Learner's task

Let us consider a set $\mathcal{S} = \{(\hat{A}_i, \hat{B}_i)\}_{i=1}^k$ consisting of pair matrices $(\hat{A}, \hat{B}) \in \Omega$. At the algorithmic step $k \in \mathbb{N}_+$, the learner aims at solving the following reduced instance of (8):

$$\max_{Q,Y,Z} \quad \text{trace}(Q) \tag{9a}$$

$$\text{s.t.} \quad \begin{bmatrix} \tau Q & \star & \star \\ 0 & (1-\tau)I & \star \\ \hat{A}_i Q + \hat{B}_i E_j Y + \hat{B}_i E_j^- Z & 0 & Q \end{bmatrix} \succcurlyeq \varepsilon I, \quad \text{for all } i = 1, \dots, V_k,\ j = 1, \dots, 2^p, \tag{9b}$$

$$\text{(7b),(7c)}$$

$$Q \preccurlyeq \eta I,\ Y \in \mathcal{Y},\ Z \in \mathcal{Z}, \tag{9c}$$

which considers only the V_k vertices of the convex hull of $\mathcal{S}$. In contrast to (8), formulation (9) considers two additional hyperparameters $\eta \geq \varepsilon > 0$ that can be chosen arbitrarily. As in Masti et al. (2023), η and ε are meant to upper (respectively, lower) bound the largest (smallest) eigenvalue of matrix Q. Finally, we assume $\mathcal{Y}$ and $\mathcal{Z}$ to be nonempty, convex and compact subset of $\mathbb{R}^{m \times n}$.

An optimal solution to (9) consists of a triplet (Q^*, Y^*, Z^*), which allows us to determine, at each k-th iteration of the algorithm, a candidate CLF $V(x) = x^\top P_k x$ and a linear controller through the following quantities:

$$P_k = (Q^*)^{-1},\ K_k = Y^*(Q^*)^{-1},\ H_k = Z^*(Q^*)^{-1}, \tag{10}$$

which are successively handed over to the verification task.

Remark 3.1 (*On the use of Quadratic CLFs*)**.** As discussed in Blanchini (1995), the search for a common quadratic Lyapunov function can be seen as an excessively restrictive design choice. Yet, in practical terms, this choice is key for our approach to run within reasonable time on unsophisticated hardware (i.e. within a few minutes). While more general techniques exist (Dai et al., 2021; Grande, Fenucci, et al., 2023), the associated calculations might take days to run on the same unsophisticated hardware.

3.3. Verifier's task

The verifier ensures that the triplet (Q_k^*, Y_k^*, Z_k^*), provided by the learner as an optimal solution to (9) at the k-th iteration of the iterative procedure, is actually valid over Ω and, hence, for all $x \in \mathscr{D}$ and $\phi \in \Phi$ according to the discussion in Section 2.2.

We first notice that the two conditions (7b)–(7c) concern solely the state and input domains and thus are satisfied by design. To check the validity of (9b) uniformly over Ω—rather just on the considered samples—one can verify if the optimal value of the following optimisation problem is positive:

$$\lambda^* = \min_{(A,B)\in\Omega, j=1,\dots,2^p} \lambda_{\min}(\Xi_k(A,B,j)), \tag{11}$$

where $\lambda_{\min}$ indicates the minimum eigenvalue of matrix Ξ_k, which corresponds to the block matrix in (9b), computed using (Q_k^*, Y_k^*, Z_k^*), as a function of matrices A and B and the index j. In the following, we drop the dependence on the j index momentarily to alleviate notation clutter.

Therefore, at iteration k of the CEGIS loop, the verifier computes Ξ_k using the triplet (Q_k^*, Y_k^*, Z_k^*) returned by the learner, and solves (11). If $\lambda^* > 0$, no counterexample exists and the CEGIS procedure terminates with output $(P_k, K_k, H_k) = ((Q^*)^{-1}, Y^*(Q^*)^{-1}, Z^*(Q^*)^{-1})$. Conversely, if the optimisation finds $\lambda^* \leq 0$, the verifier then adds the minimiser pair $(A^*, B^*) =: (\hat{A}_{k+1}, \hat{B}_{k+1})$ to the training set $\mathcal{S}_k$ such that the learner can compute a new candidate triplet $P_{k+1}, Y_{k+1}, Z_{k+1}$, and the process repeats.

Solving (11) is a nonconvex optimisation problem that shall be solved *globally* to prove that no counterexample exists. To achieve global optimality, an idea is to exploit the fact that the minimum eigenvalue $\lambda_{\min}(\cdot)$ of a symmetric matrix is a Lipschitz continuous map of the entries of the matrix itself:

Lemma 3.2 (*Horn & Johnson, 1994*)**.** *Consider two symmetric matrices K and L of the same dimension. Then, it holds that $|\lambda_{\min}(K) - \lambda_{\min}(K+L)| \leq \|L\|_{\text{op}}$, where $\|\cdot\|_{\text{op}}$ is the operator norm of a matrix, induced by the ℓ_2-norm.* □

Yet, problem (11) still involves optimising over Ω, which is an unknown, potentially nonconvex, set. Even though one may know the domain of the state variables $\mathscr{D}$, inferring useful information for Ω from $\mathscr{D}$ itself is not straightforward. If we however further assume that $A(t)$, $B(t)$ are Lipschitz continuous with respect to x, and recalling (4), one can exploit the fact that the composition of two Lipschitz functions is itself Lipschitz, i.e.:

$$\begin{aligned} &\| A|_x - A|_y \| \leq \kappa_A \|x-y\|, \ \text{ for all } x, y \in \mathscr{D}, \\ &\| B|_{x,\phi} - B|_{y,\psi} \| \leq \kappa_B(\|x-y\| + \|\phi-\psi\|), \\ &\qquad\qquad \text{for all } x, y \in \mathscr{D}, \ \phi, \psi \in \Phi, \end{aligned} \tag{12}$$

for a matrix norm of interest and constants $\kappa_A, \kappa_B \geq 0$.[2]

Since bounded linear operators preserve Lipschitz continuity, we are then able to prove the following result:

Theorem 3.3. *Let $P = (Q^*)^{-1}$ and $K = Y^*(Q^*)^{-1}$, $H = Z^*(Q^*)^{-1}$, with (Q^*, Y^*, Z^*) a solution of (9), with a given $\eta \geq \varepsilon > 0$. Then, there exists some constant $\ell = \ell(\varepsilon) \geq 0$ such that:*

$$\begin{aligned} &| \lambda_{\min}(\Xi(A,B)) - \lambda_{\min}(\Xi(A+\Delta A, B+\Delta B)) | \\ &\leq \ell(\| \Delta A \|_{\text{op}} + \| \Delta B \|_{\text{op}}) \end{aligned} \tag{13}$$

where ΔA, ΔB denote any perturbation of A, B with $(A,B) \in \Omega$. □

Proof. To prove this statement, we make use of similar arguments as in Masti et al. (2023, Theorem 1). In fact, from Lemma 3.2 we get:

$$\begin{aligned} &| \lambda_{\min}(\Xi(A,B)) - \lambda_{\min}(\Xi(A+\Delta A, B+\Delta B)) | \\ &\qquad \leq \left\| \begin{matrix} 0 & 0 & \Delta_{AB} \\ 0 & 0 & 0 \\ \Delta_{AB}^\top & 0 & 0 \end{matrix} \right\|_{\text{op}}, \end{aligned}$$

where $\Delta_{AB} := \Delta A Q + \Delta B(E_j Y + E_j^- Z)$, and $\|\cdot\|_{\text{op}}$ is the matrix norm induced by the ℓ_2 one. Manipulating the right-hand side (RHS), we obtain:

$$\begin{aligned} \left\| \begin{matrix} 0 & 0 & \Delta_{AB} \\ 0 & 0 & 0 \\ \Delta_{AB}^\top & 0 & 0 \end{matrix} \right\|_{\text{op}} &:= \sup_{\|x\|=1} \left\| \begin{bmatrix} 0 & 0 & \Delta_{AB} \\ 0 & 0 & 0 \\ \Delta_{AB}^\top & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \right\|_2 \\ &= \sup_{\|x\|=1} \left\| \begin{bmatrix} \Delta_{AB} x_3 \\ 0 \\ \Delta_{AB}^\top x_1 \end{bmatrix} \right\|_2 \end{aligned}$$

$$= \sup_{\|x\|=1} \sqrt{\| \Delta_{AB} \|_{\text{op}} \| x_1 \| + \| \Delta_{AB} \|_{\text{op}} \| x_3 \|} = \| \Delta_{AB} \|_{\text{op}},$$

$$\begin{aligned} \| \Delta_{AB} \|_{\text{op}} &\leq \|\Delta A\|_{\text{op}} \|Q\|_{\text{op}} + \|\Delta B\|_{\text{op}} \|E_j Y + E_j^- Z\|_{\text{op}} \\ &\leq \|\Delta A\|_{\text{op}} \eta + \|\Delta B\|_{\text{op}} (\|Y\|_{\text{op}} + \|Z\|_{\text{op}}). \end{aligned}$$

This follows from the fact that $\| E_j \|_{\text{op}} = 1, \| E_j^- \|_{\text{op}} = 1 \ \forall j$. Moreover, since $\mathcal{Y}$ and $\mathcal{Z}$ are compact, both Y and Z are norm bounded, i.e. it holds that $\|Y\|_{\text{op}} \leq \eta_Y$ and $\|Z\|_{\text{op}} \leq \eta_Z$, for some $\eta_Y, \eta_Z > 0$, and therefore:

$$\begin{aligned} &| \lambda_{\min}(\Xi(A,B)) - \lambda_{\min}(\Xi(A+\Delta A, B+\Delta B)) | \\ &\qquad \leq \|\Delta A\|_{\text{op}} \eta + \|\Delta B\|_{\text{op}} (\eta_Y + \eta_Z). \end{aligned}$$

In turn, if η is chosen such that $\|Y\|_{\text{op}} \leq \frac{\eta}{2}$ and $\|Z\|_{\text{op}} \leq \frac{\eta}{2}$, it holds that:

$$\begin{aligned} &| \lambda_{\min}(\Xi(A,B)) - \lambda_{\min}(\Xi(A+\Delta A, B+\Delta B)) | \\ &\qquad \leq \eta \cdot (\|\Delta A\|_{\text{op}} + \|\Delta B\|_{\text{op}}), \end{aligned}$$

which concludes this proof. □

Remark 3.4. By exploiting (12) we can further say that, if $\Delta A = A|_x - A|_y$ and $\Delta B = B|_{x,\phi} - B|_{y,\psi}$, then it holds that:

$$\begin{aligned} &| \lambda_{\min}(\Xi(A,B)) - \lambda_{\min}(\Xi(A+\Delta A, B+\Delta B)) | \\ &\qquad \leq \eta\kappa_A \|x-y\| + \eta\kappa_B(\|x-y\| + \|\phi-\psi\|). \end{aligned}$$ □

As a consequence of Theorem 3.3, we can recast (11) as:

$$\begin{aligned} \lambda^* = \min_{x\in\mathscr{D}, \phi\in\Phi} \ & \lambda_{\min}(\Xi_k(A,B)), \\ \text{s.t.} \ & (4), \end{aligned} \tag{14}$$

and, exploiting Lemma 3.2, solve (14) to global optimality using a global Lipschitz solver.

Remark 3.5 (*Computational Considerations*)**.** To operatively solve Problem (14), there is no need to provide the values of the above-mentioned Lipschitz constants κ_A and κ_B, as many solvers embed proceed to estimate such a quantity automatically (see, *e.g.,* Jones et al., 1993). We also note that, to further speed up the computation of the eigenvalue of high dimensional matrices, one can resort on techniques such as (Kangal & Mengi, 2020; Manucci et al., 2024).

Note that the peculiar nature of Φ simplifies the resolution of (14), allowing us to split the verification problem in p subproblems, with one loss of efficiency on ϕ at any one time. Such problem formulation helps tackling the nonconvex nature of Φ, which further complicates the verifier's task, and supports the computation of the solution via Lipschitz solvers, which are known to suffer from the curse of dimensionality.

[2] Later discussed in Remark 3.5.

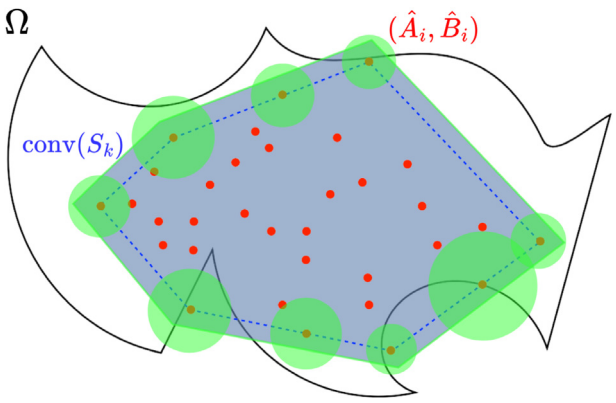


Fig. 2. The region identified by the union of the green "balls" centred on each vertex pair of matrices $\{(\hat{A}_i, \hat{B}_i)\}_{i=1}^{V_k}$ (red dots, among all the other samples) and the associated set of matrices with eigenvalues $(\epsilon/\lambda^2_{\max}(P))$-distant from those belonging to cl(conv($\mathcal{S}_k$)) (green line and its interior, with conv($\mathcal{S}_k$) represented by the blue dashed line), denotes the actual portion of volume where it is guaranteed that no counterexample can be found. As the iterations of Algorithm 1 progress, the resulting area will cover the whole of Ω, regardless of its shape.

Algorithm 1 CEGIS-based PFTC learning method

Initialization: Set $\eta \geq \epsilon > 0$, $\mathcal{S}_1 = (\hat{A}_1, \hat{B}_1) \in \Omega$
Iteration ($k \in \mathbb{N}_+$)**:**
Ⓛ Identify vert(conv($\mathcal{S}_k$))
Ⓛ Solve (9), set (P_k, K_k, H_k) as in (10)
  If (9) `infeasible` **then** `exit`
Ⓥ Solve (14), $\forall j$, using Lipschitz global optimisation
  If $\lambda^* \leq 0$ **:** $\mathcal{S}_{k+1} \leftarrow \mathcal{S}_k \cup \{(A^*, B^*)\}$, `repeat`
  If $\lambda^* > 0$ **:** $V = x^\top P_k x$, $K = K_k$, $H = H_k$, `exit`

Remark 3.6 (*Multiple Contemporary Faults*)**.** As previously discussed, limiting failures to one actuator at a time is a common restriction in fault-tolerance studies due to practical domain considerations, *e.g.,* in AUV missions. However, our method is not fundamentally limited to this assumption. Such a limitation only applies when constructing the set Φ and is considered in this presentation as a means to speed up the solution of (14). To consider multiple contemporary actuator faults, one only needs to modify the shape of Φ.

3.4. CEGIS-based procedure and analysis

Algorithm 1 reports the main steps of the proposed CEGIS-based iterative scheme for the design of PFTC policies based on Lyapunov arguments. Specifically, the Ⓛ bullets refer to tasks performed by the learner, while the Ⓥ bullet to the one performed by the verifier. An intuitive depiction of the proposed procedure is shown in Fig. 2, illustrating the geometrical intuition behind how a generic-shaped Ω is gradually covered by the polytope conv($\mathcal{S}_k$) and the "safe balls" spawning from every point in its frontier, thus proving the overall stability of the whole closed loop. This follows from the fact that if the Jacobian of a nonlinear system can be shown to lie within a set Ω, the original nonlinear system can be conservatively approximated by an uncertain Linear Time-Varying (LTV) system and the stability of such a system is then implied by the stability of the latter (Kothare et al., 1996; Liu, 1968). In this sense, the purpose of the verifier, and specifically Problem (14), is thus to span all possible linearisation points to implicitly construct the set Ω and, by extension, to prove global stability within the specified domain.

Algorithm 1 may terminate with two possible outcomes, denoted by the `exit` commands. First, if the verifier finds no counterexample, Algorithm 1 has converged and a control function K is returned. Alternatively, if the verifier returns a counterexample, the latter is added to the set $\mathcal{S}_{k+1}$ and the next learner iteration is started by solving a new instance of (9). Our procedure is susceptible to the quality of the counterexample produced by the verifier, in turn associated to the choice of the hyperparameters of our method. Indeed, it may occur that an instance of (9) may fail to generate a controller (declaring infeasibility when solving (9)) due to the specific choice of said hyperparameters. In such a case, the user may want to restart the algorithm with a new choice of hyperparameters. In case (9) remains feasible, Algorithm 1 enjoys finite-time convergence. This implies that either the procedure declares infeasibility, or it returns a PFTC policy within a finite-number of steps, as established hereby.

Theorem 3.7. *Let $\eta \geq \varepsilon > 0$ and $(\hat{A}, \hat{B}) \in \Omega$. Then, in a finite number of steps Algorithm 1 either declares infeasibility or return a triplet (Q, Y, Z) so that $V(x) = x^\top Q^{-1} x$ is a CLF for (3) with saturated input $u(t) = \mathrm{sat}_{\mathscr{U}}(Kx(t))$.*

Proof. The proof follows the same rationale behind the one of Masti et al. (2023, Theorem 2). For this reason, we only sketch it by referring to the diagram in Fig. 2, highlighting key points and where we employ the results developed in this paper.

By considering the generic M-th iteration of Algorithm 1, the learner solves (9) based on the vertices of the convex-hull of samples (i.e., counterexamples) $S^M := \{(A_i, B_i)\}_{i=1}^M$ available up to that iteration (red dots in Fig. 2). Standard robust control arguments allow us to conclude that the proposed solution, consisting of a candidate Lyapunov function and linear controller in (10), is also a valid one for all the pair matrices belonging to conv(S^M) (grey area in Fig. 2). As such, the controller $K_M = Y^*(Q^*)^{-1}$ stabilises all the dynamics in conv(S^M) while meeting the saturation constraints, and hence the verifier can only find counterexamples strictly outside conv(S^M). Moreover, by pre and postmultiplying each expression in (9b) by diag$((Q^*)^{-1}, (Q^*)^{-1}, (Q^*)^{-1})$, making the substitution in (10) and relying on the other constraints in (9), one can find a lower bound ε/η^2 for the resulting matrix expression $\Xi_M(A_i, B_i, j)$, which imposes an extra limitation on the verifier's task: if a counterexample is found, then it should be (at least) (ε/η^2)-far (in terms of eigenvalues explored through (11)) from the convex hull of the examples currently provided conv$(\{(\hat{A}_i, \hat{B}_i)\}_{i=1}^k)$ (green line delimiting the grey area in Fig. 2). Note that this is particularly true in view of the Lipschitz continuity property established in Theorem 3.3, which allows us to claim that any new $(M+1)$-th counterexample $\bar{z}_{M+1} = (\bar{A}_{M+1}, \bar{B}_{M+1})$ shall satisfy:

$$\|\bar{A}_{M+1} - A_s\| + \|\bar{B}_{M+1} - B_s\| > \frac{\varepsilon^2}{\eta^2}, \ \forall (A_s, B_s) \in S^M.$$

In view of this erosion property, it holds that $S^M \subset S^{M+1} \subseteq \Omega$. In turn, by means of standard contradiction arguments, one can show that the verifier cannot generate counterexamples indefinitely, and hence there will exist a sufficiently large k such that $S^{M+k} \supseteq \Omega$, since the set Ω is bounded and closed (although it can be disconnected, nonconvex of any shape), as well as because 2^p is finite. By referring to Fig. 2 once again, there will exist an iteration index k so that the grey area has covered the whole of Ω, and hence no counterexample can be generated. □

Remark 3.8. From Remark 3.4 and (12), it also follows that for any tuple $[\bar{x}_{M+1}, \bar{\phi}_{M+1}]$ such that $z_{M+1} = (A|_{\bar{x}_{M+1}} = A|_{x_{M+1}}, B_{M+1} = B|_{\bar{x}_{M+1}, \bar{\phi}_{M+1}})$, it will hold that:

$$\kappa_A \|\bar{x}_{M+1} - y\| + \kappa_B (\|\bar{x}_{M+1} - y\|$$

$$+ \|\bar{\phi}_{M+1} - \psi\|) > \frac{\varepsilon}{\eta} \quad \forall [y, \psi] \in D^M,$$

where $D^M := \{[y, \psi] \mid y \in \mathscr{D}, \psi \in \Phi, (A|_y, B|_{y,\psi}) \in S^M\}$. Moreover, due to f, g being C^2 and S^M being a closed and connected set, we know that D^M is closed and connected. From these considerations it follows that $D^{M+1} \supset D^M$ and therefore there exists a sufficiently large k such that $D^k \supseteq \mathscr{D} \times \Phi$. □

Remark 3.9 (*Selection of η and ε*)**.** Besides affecting the learner convex optimisation problem (9), we note that η and ε assume a key role also in characterising the performance of both Algorithm 1 and the verification task. In fact, from Theorem 3.7, it is evident that by setting η large enough, the convergence of Algorithm 1 may require fewer steps, as this combination allows to "erode" larger portions of the set Ω at each iterations (see also Fig. 2). In addition, larger η can also favour the performance of the technique employed to solve (14). Indeed, convergence of algorithms for Lipschitz-continuous global optimisation problems, as the one in (14), is well-studied in the literature. As a general rule of thumb inferred, the smaller the Lipschitz constant, the better the guarantee on the performance of the underlying algorithm (Malherbe & Vayatis, 2017). However, following the considerations above may undesirably complicate the learner's optimisation problem, as the feasible set of (9) would shrink, also possibly making it excessively conservative w.r.t. to (7). This naturally induces a tradeoff in the choice of the pair (η, ε), which shall hence be set according to the problem data at hand.

Having now completed the theoretical derivations, we next discuss how the described method can be employed in realistic control case scenarios involving nonlinear dynamical systems subjected to a range of faults at actuators.

4. Case study: PFTC for a hover-capable AUV

In this section, we test our IS-sat controller on two realistic benchmarks, and compare our bespoke method against an $\mathcal{H}_\infty$ control, representing a common passive fault-tolerant alternative.

The code relative to the IS-sat control is written in Python 3.11, employing `cvxpy` as the optimisation library solving (9), while the verification task uses the `SHGo` algorithm (Endres et al., 2018) from Scipy library to solve the problem as formulated in (11). The experiments are run on a laptop computer featuring an Intel Core i7-14650HX CPU and 14 cores, with the dedicated GPU disabled.

4.1. A hover-capable AUV

Let us consider a hover-capable AUV, which represents an AUV operating in condition of neutral buoyancy, capable of maintaining prescribed position and attitude in presence of environmental disturbances. Hover-capable AUVs are conventionally employed in surveillance of underwater maritime structures and in sonar-searches and mapping, where they are tasked with following a path at constant depth and constant forward speed. In order to follow a predefined path, the innermost control loop is designed to maintain the forward (surge) speed at a constant set value, to regulate the angular speed to zero, and to track desired time-varying heading setpoints (Fenucci et al., 2024). In such precise guidance and control applications, where AUVs operate nearby the seabed carrying sensitive and expensive instruments, faults at thruster can result in collisions with ground, or, in the worst case scenarios, in damages to the structures under surveillance or in the loss of the vehicle.

To this aim, let us consider the two-state AUV model from Grande et al. (2024, Case study A), representing a simplified version of an Autosub Hover AUV. The vehicle mounts three fixed

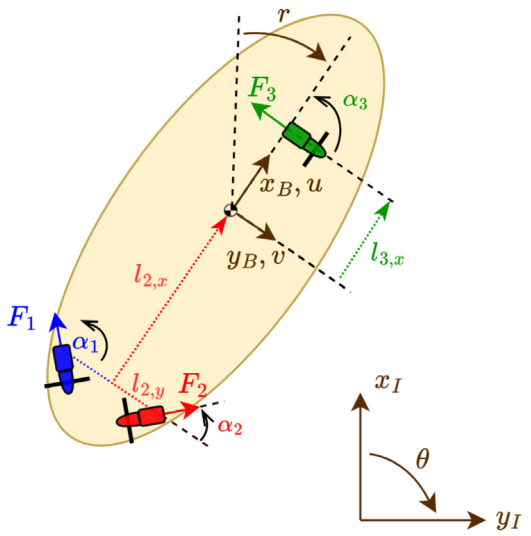


Fig. 3. Hover-capable AUV with three (fixed) thrusters moving in the horizontal plane.

(i.e. non-rotating) thrusters, generating force along the thruster axis in both the positive and negative directions, as illustrated in Fig. 3. Such a vehicle is designed to operate near infrastructures at low speeds and constant depth, while retaining the ability to control pitch and yaw angle (Fenucci et al., 2024). The equations of motion are derived from the robot-like vectorial model for marine crafts (Fossen, 2011), while restricting our analysis to the surge and yaw dynamics, resulting in[3]:

$$\begin{cases} \dot{x}_1 = \dfrac{-X_u x_1 - X_{uu} x_1^2 + \phi_1 F_{1,x} + \phi_2 F_{2,x} + \phi_3 F_{3,x}}{m} \\ \dot{x}_2 = \dfrac{-N_r x_2 - N_{rr} x_2^2 + (-F_{1,x} l_{1,y} + F_{1,y} l_{1,x})\phi_1}{J_z} \\ \qquad + \dfrac{(-F_{2,x} l_{2,y} + F_{2,y} l_{2,x})\phi_2}{J_z} \\ \qquad + \dfrac{(-F_{3,x} l_{3,y} + F_{3,y} l_{3,x})\phi_3}{J_z} \end{cases}, \tag{15}$$

where $x := [x_1, x_2]^\top$ denotes the surge speed and angular rate about the vehicle vertical axis, and $u := [F_1, F_2, F_3]^\top$ denotes the control input encompassing the forces generate by the three thrusters, namely the aft port (F_1), aft starboard (F_2) and bow thruster (F_3). Moreover, the model includes $u \in [-\bar{u}, \bar{u}]$, where $\bar{u}$ denotes the saturation value of the thrusters and $F_{i,x} = F_i \sin(\alpha_i)$ and $F_{i,y} = F_i \cos(\alpha_i)$ represent the projections of the thruster force F_i along the x_b and y_b body-axes, with the convention reported in Fig. 3. Finally, m and J_z denote the mass and inertia of the vehicle (including added inertial terms); X_u, X_{uu} indicate the linear and quadratic surge drag coefficients; N_r, N_{rr} represent the linear and quadratic yaw drag coefficients; and ϕ_i characterises the efficiency associated to the i-th thruster.

Next, we detail the design of the control laws. The state domain is defined as $\mathcal{D} = [-2, 2]^2$, which extensively covers the usual range of linear and angular velocities of the hoover-capable AUVs. In this case study, the control saturation value is set to $\bar{u} = 38.0$ N for all three actuators, value selected within the conventional saturation range of common underwater thrusters, such as the BlueRobotics T200 thrusters. It is worth noting that the positive and negative saturation limits are not restricted to be equal, and such limits can be arbitrarily customised for each

[3] For a detailed derivation of the model, we refer the interested reader to Grande (2024, Ch. 4, 6.5.2).

actuator. As this case study is based on the Autosub Hover AUV, which features three equal thrusters (as the majority of AUVs with non-axial single thrusters), the saturation limits are set to the same value. To synthesise the IS-sat control law, we consider a discrete-time version of model (15), which is discretised via the explicit Euler method with a time step of 0.01 s. By setting $\eta = 50$, $\varepsilon = 10^{-4}$, $\tau = 0.999$,[4] we are able to synthesise a sound controller within 7 iterations of Algorithm 1 and with an overall run time of < 3 s, resulting in the following feedback gain:

$$K_{\text{IS-sat}} = 10^3 \begin{bmatrix} -43.987 & 0.308 \\ -30.985 & 7.948 \\ -1.187 & 37.481 \end{bmatrix}, \tag{16}$$

which is in turn applied in closed-loop as:

$$u(t) = 38.0 \cdot \text{sat}_{\mathscr{U}}(K_{\text{IS-sat}} e(t)). \tag{17}$$

Recall that the IS-sat method represents an optimised version of a classical polytopic formulation for uncertain systems. For comparison, the standard polytopic formulation from Rubin et al. (2020), under the same modelling uncertainties, yields an optimisation problem with 2^{13} constraints due to the 10 time-varying coefficients of the A, B matrices and the 3 control signals, which is beyond the computational capacity of a common office laptop.[5]

Performance comparison: The proposed approach is tested against two $\mathcal{H}_\infty$ control laws, which represent a conventional framework to solve PFTC problems in AUV applications (see, for instance, Grande et al., 2024; Kaminer et al., 1991). As in Grande et al. (2024), we formulate the $\mathcal{H}_\infty$ optimisation problem as the simultaneous stabilisation of *four* operational modes, namely the faultless mode and the three modes characterised by one (distinct) thruster failing. Each mode is obtained by linearising equation (15) around $\bar{x} = [0.5, 0]$ and by selecting $\phi_1 = \phi_2 = \phi_3 = 1$ in the faultless mode, and $\phi_i = 0$ for the mode corresponding to the i-th thruster at fault. To devise controllers with different performance characteristics, two optimisation problems are formulated. A target response time of 10 s (a realistic value for the application being considered) is set as a design constraint (hard goal) for both the optimisation problems, while the maximum tolerated steady-state error is set as objective (soft goal) and varied between the two controllers. The first control law, defined 'aggressive' and denoted as $\mathcal{H}_\infty^a$, is designed to ensure a comparatively low maximum steady-state error, set at 5%, assuming that energy and power consumption are not the key design driver. Such 'aggressive' choice entails that no restriction is imposed on the actuator saturations. The second 'conservative' control law, denoted as $\mathcal{H}_\infty^c$, is designed relaxing the maximum steady-state error at 40% and by imposing a saturation on the actuator as soft optimisation constraint. The synthesised controllers result in:

$$\mathcal{H}_\infty^a = 10^4 \begin{bmatrix} -0.2400 & -0.2865 \\ -0.2553 & 0.3037 \\ 0.0013 & 2.2510 \end{bmatrix}, \tag{18}$$

$$\mathcal{H}_\infty^c = \begin{bmatrix} -232.1081 & -277.2772 \\ -183.4074 & 219.4158 \\ 0.0298 & 776.4082 \end{bmatrix}. \tag{19}$$

Next, we evaluate the performance of the three control laws, namely (17), (18), (19), by tracking the reference point $\bar{x} = [0.5, 0]$ over a simulation horizon of $T = 30$ s. The simulated scenario is divided into three phases to represent different operational modes. In the first section of the simulation ($0 \leq t \leq 10$ s), the system operates without faults. In the second phase ($10 < t \leq 20$ s), thruster F_3 operates at 10% of its nominal efficiency, while the other thrusters operate nominally. Finally, in the last third of the scenario ($20 < t \leq 30$ s), F_2 operates at 10% of its nominal efficiency, while the other thrusters operate nominally. Fig. 4 shows the simulation results in terms of the magnitude of the control effort and tracking error (it is recalled that even if the $\mathcal{H}_\infty$ controllers do not explicitly exhibit the saturation limits within the laws (18) and (19), the physical limits of the actuators are accounted for within u in model (15)). From these results, it can be noted that our controller exhibits similar closed-loop performance to the 'aggressive' $\mathcal{H}_\infty^a$ control design in terms of tracking error.

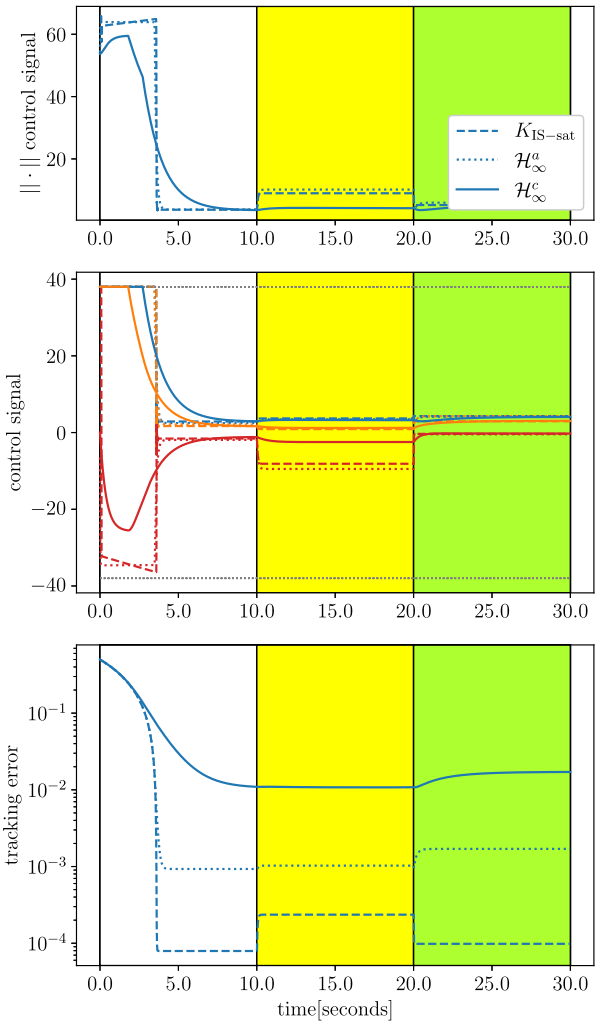


Fig. 4. Comparison of our proposed controller $K_{\text{IS-sat}}$ with the two $\mathcal{H}_\infty$ controllers for tracking a constant reference $\bar{x} = [0.5, 0]$. The top plot shows the norm of the control input over time. The middle plot shows the control action applied to each input channel, with grey lines indicating saturation limits. The bottom plot shows the norm of the tracking error (on both components). The three vertical coloured regions indicate the three fault modes.

Next, we further investigate the performance of our controller and of the $\mathcal{H}_\infty^a$ controller while tracking sinusoidal references on both x_1 and x_2. We report the result in Fig. 5, illustrating a simulation with a time horizon $T = 120$ s. In this case, IS-sat also demonstrates tracking performance at least as good as $\mathcal{H}_\infty^a$.

Remarkably, our controller exhibits significantly better theoretical robustness properties, which can be observed by comparing the largest ellipsoidal domains of attraction for the three

[4] A systematic hyperparameter analysis is provided in Section 4.2.

[5] We provide further computational details and comparisons in the more extensive case study reported in Section 4.2.

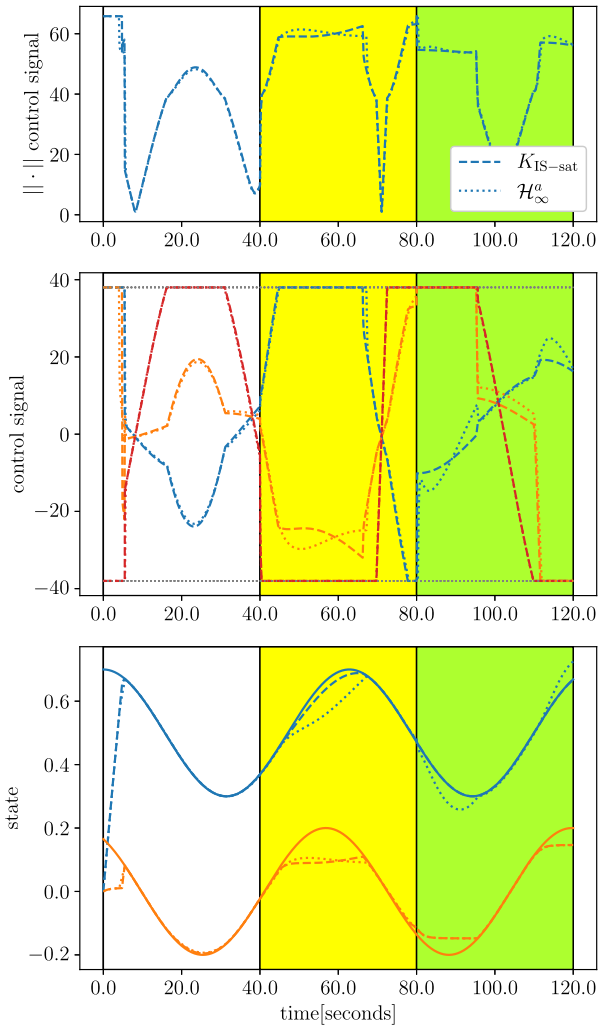


Fig. 5. Comparison of the our controller $K_{\text{IS-sat}}$ with the aggressive $\mathcal{H}_\infty^a$ controller from Grande et al. (2024) for tracking a sinusoidal reference signal. In the third plot, the solid line represents the reference trajectory, with x_1 in blue and x_2 in orange.

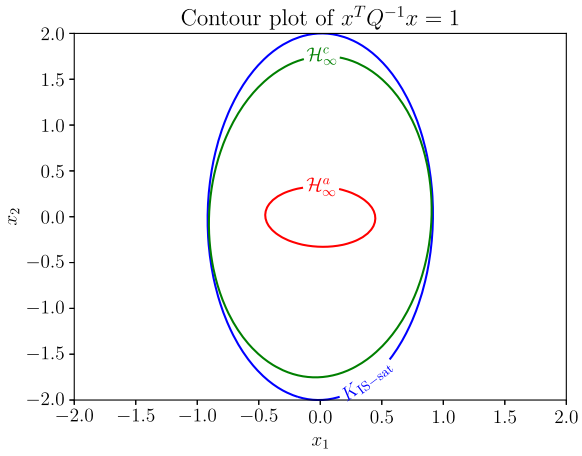


Fig. 6. Domains of attraction (computed according to Rubin et al. (2020)), associated to the IS − sat and the two $\mathcal{H}_\infty$ controllers from Grande et al. (2024). The proposed controller achieves the largest domain of attraction. This is particularly noteworthy given its comparable performance to the aggressive $\mathcal{H}_\infty^a$ controller, which exhibits significantly smaller domains of attraction.

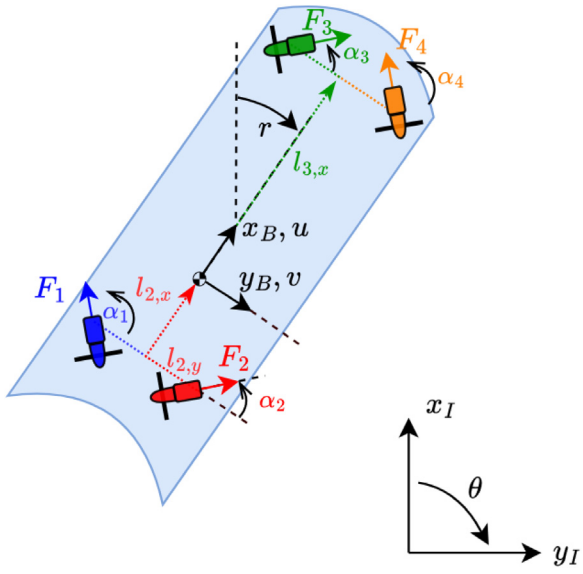


Fig. 7. Hover-capable AUV with four (fixed) thrusters moving in the horizontal plane.

feedback laws. To compute the domains of attraction, we solve Problem (9) with the additional constraint $KQ = Y$, still using the CEGIS loop as suggested in Rubin et al. (2020). The resulting domains of attraction are shown in Fig. 6. As expected, our approach achieves the largest domain of attraction among the three control laws. Notably, our approach also compares well with the neural network-based approach proposed in Grande et al. (2024), which has a domain of attraction equal to a sphere of radius 1 (not depicted in Fig. 6, but reported in Grande et al. (2024, §9)). It is important to note that the domains of attractions reported in Fig. 6 are in actuality very conservative estimates of the real domains of attraction. Nevertheless, especially when considered in conjunction with the results in Fig. 4, they demonstrate that our technique is not significantly affected by the technical constraints in Eq. (9), which are introduced to enhance the convergence of the control synthesis.

4.2. A 5-dimensional AUV model

Let us now analyse the performance of a higher-dimensional, higher-fidelity version of the AUV model employed in Section 4.1. Besides the previous two state variables, the model now accounts for the sway dynamics (x_2) and for the yaw angle (x_4). Introducing the sway dynamics allows to account for the Coriolis forces, which introduce cross-coupling terms between the degrees of freedom, in turn increasing the nonlinearity of the model. Finally, a fifth state (without direct physical meaning) is introduced as the integral action of the yaw angle, to ensure zero-error tracking for constant heading setpoints (Zanon & Bemporad, 2021, §IV-A).

For this study, we employ an AUV with four thrusters in a X-shape configuration (Baldini et al., 2018). Each of the four non-azimuthing (i.e. not rotating) thrusters, two located aft and two stern of the centre of gravity, can generate positive and negative thrust force. Such a vehicle is illustrated in Fig. 7, with the overall model dynamics reported in (20) (derived from Grande, 2024, Ch. 7.3).

It should be noted that the nonlinear part of the AUV dynamics (20) depends on three state variables, namely x_1, x_2, and x_3. In the corresponding uncertain representation, these variables, together with the four control inputs, give rise to a total of 21

$$
\begin{cases}
\dot{x}_1 = & \dfrac{-X_u x_1 - X_{uu} x_1^2 + m x_2 x_3 + \phi_1 F_{1,x} + \phi_2 F_{2,x} + \phi_3 F_{3,x} + \phi_4 F_{4,x}}{m} \\
\dot{x}_2 = & \dfrac{-Y_v x_2 - Y_{vv} x_2^2 - m x_1 x_3 + \phi_1 F_{1,y} + \phi_2 F_{2,y} + \phi_3 F_{3,y} + \phi_4 F_{4,y}}{m} \\
\dot{x}_3 = & \dfrac{-N_r x_3 - N_{rr} x_3^2 + (-F_{1,x} l_{1,y} + F_{1,y} l_{1,x})\phi_1 + (-F_{2,x} l_{2,y} + F_{2,y} l_{2,x})\phi_2 + (-F_{3,x} l_{3,y} + F_{3,y} l_{3,x})\phi_3 + (-F_{4,x} l_{4,y} + F_{4,y} l_{4,x})\phi_4}{J_z} \\
\dot{x}_4 = & x_3 \\
\dot{x}_5 = & x_4
\end{cases}
\tag{20}
$$

Box I.

Table 1
Comparison of computational resources required by the nonlinear MPC law(s) and IS-sat law(s).

Method	Synthesis		Deployment	
	time [s]	RAM [MB]	time [s]	RAM [MB]
IS-sat (selected tuning)	71	220	< 1	< 1
IS-sat mean(std)	76(8)	215(22)	< 1	< 1
Nonlinear MPC (selected tuning)	< 1	< 1	60.7	149
Nonlinear MPC mean(std)	< 1	< 1	94(61)	155(35)

time-varying uncertain elements, with $A \in \mathbb{R}^{3\times3}$, $B \in \mathbb{R}^{3\times4}$.The approach enumerating all vertices of a hypercube-shaped Ω accounting for 4 possible faults encompasses $2^{21} \times 2^4$ constraints, which is clearly beyond the capabilities of commonly accessible contemporary hardware.

To synthesise the $K_{\text{IS-sat}}$ controller, we start by selecting a state domain as $\mathcal{D} = [-2, 2]^3$ for the first three components of the state vector, whilst we do not restrict the remaining two variables (namely the integral variables): this domain choice well exceeds the usual range of dynamics encountered by slow moving vehicles, which dynamics are usually bounded under 1 m/s in the linear speeds and under 0.18 rad/s (i.e. ≈10 deg/s) in yaw rate. The saturation threshold is once again set to $\bar{u} = 38.0$ N for all actuators to resemble realistic commercial thruster values such as the T200 by BlueRobotics.

For this case study, we repeat the tuning exercise by defining over 40 combinations of η and ε values spanning over several orders of magnitude (from 10^{-4} to 10^2), observing consistent results among the synthesis runs. We discuss such results in detail in the following paragraph and summarise them in Table 1. It was noted that the synthesised control laws lead to qualitatively similar performance; hence, for this comparison, we illustrate the control law obtained via the same hyperparameter values of the previous test, namely $\eta = 50$, $\varepsilon = 10^{-4}$, $\tau = 0.999$. This choice of parameters achieved convergence in 8 CEGIS loop iterations, taking 71.4 s of synthesis time.

$$
u(t) = 38.0\ \text{sat}_{\mathscr{U}}\left(\begin{bmatrix} -50.93 & -48.31 & 592.27 & 114.93 & 5.49 \\ 45.28 & -45.56 & 540.61 & 107.57 & 5.14 \\ 46.77 & -47.90 & -540.79 & -104.30 & -4.98 \\ -41.89 & -44.06 & -492.74 & -96.09 & -4.59 \end{bmatrix} e(t)\right). \tag{21}
$$

Let us now compare our control gain $K_{\text{IS-sat}}$ against a fully fledged nonlinear MPC scheme built using (20) in no-fault condition, which represents the *de facto* standard approach to control dynamical systems accounting for state and input constraints. To provide a fair comparison, we explored several dozens MPC laws, varying gain and prediction horizon over different orders of magnitude. Specifically, we explored time horizons varying from a few seconds to a few minutes (10 s to 150 s), and gain spanning over 10^{-4} to 10^2. These ranges were initially identified as likely to produce stable closed-loop dynamics and good

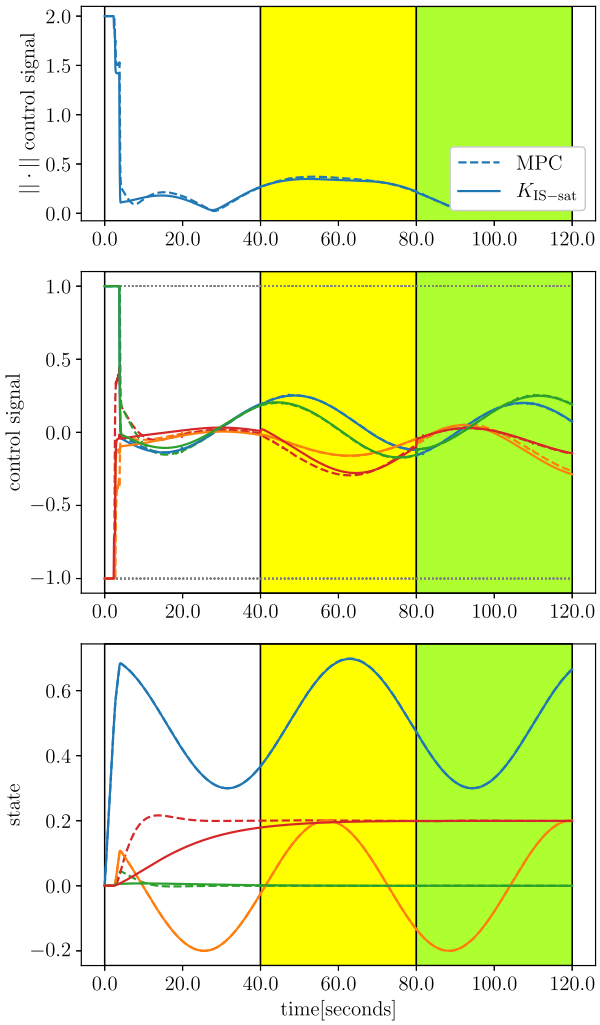


Fig. 8. Comparison of IS-sat with a nonlinear MPC, with model dynamics initialised to $x = 0$. Control signal plots have been normalised in the $[-1, 1]$ range for ease of readability. State plot legend: x_1 in blue, x_2 in orange, x_3 in green, x_4 in red; solid line associated to $K_{\text{IS-sat}}$, dashed line to MPC. It can be appreciated how IS-sat and MPC have comparative tracking performance for x_1 and x_2 (the solid and dashed lines superimpose), while MPC tracks x_4 6x faster.

tracking performance. For the comparison with IS-sat, we specifically selected the tuning which best tracks the reference (while naturally discarding the unstable laws), resulting in an LQR-like cost function (Duan & Yu, 2013, Ch. 8.4) with $Q = 1$ and $R = 3.162 \cdot 10^{-2}$, with a prediction horizon of 56 steps.

As a first test, the AUV is set to track two sinusoidal references on x_1, x_2, while tracking a constant yaw reference of $\bar{x}_4 = 0.2$. As in the case study discussed in Section 4.1, we consider three sequential phases consisting of a no-fault phase, followed by a phase where the second actuator (F_2) functions at 10% efficiency to terminate with a phase where the third actuator (F_3) stands at 10% efficiency. Next, we consider two different initial conditions and perform two tests: (a) the simulation starts at the origin, illustrated in Fig. 8; (b) the simulation starts at the corner of our domain, namely $x(0) = [2, 2, 2, 2, 2]^T$, reported in Fig. 9. As it can be appreciated in the state tracking plot within Fig. 8, both $K_{\text{IS-sat}}$ and MPC are capable of satisfactory performance in tracking the sine signals on x_1, x_2, whilst the regulation of x_4 is much faster for the MPC controller with respect to our proposed approach (10 s vs 60 s). However, when the initial condition is at the boundary of the domain, the MPC approach behaves rather poorly compared to IS-sat, as illustrated in the tracking error graph reported within Fig. 9.

Finally, we compare the computational burden of the two methods. As expected, it is noted that the main cost of running a nonlinear MPC is almost entirely related to the online optimisation, while the main cost of IS-sat is due to the initial (offline) synthesis. Simulations involving over 40 MPC tuning required $\approx$ 94 s, while the runtime required by the IS-sat static controller was negligible (i.e. $<$ 1 s). It must be noted that the 42% of the synthesised MPC laws took longer to run than the operational scenario of 120 s, namely being slower than real-time and hence bearing limited use in actual applications. In Table 1 we include a comparison between the different tuned control laws, illustrating both the synthesis and the deployment times and required RAM resources.

As almost the entirety of the IS-sat computational burden is experienced during the synthesis, which is only required once, the controller is particularly suited for applications requiring inexpensive online deployments. The IS-sat was on average synthesised in 76 s on an standard office computer without GPU, proving to be a feasible method even for nonlinear systems affected by a set of possible faults. Moreover, in the specific case under analysis, IS-sat required $<$ 1 MB of RAM when deployed in closed-loop, while the MPC[6] requires $\approx$ 155 MB of onboard RAM, with a consequent increased computation and energy demand, which is at premium in real underwater embedded applications. This case study further confirmed that IS-sat can be deployed in embedded applications running on unsophisticated hardware and limited computational budget.

4.3. Validation of the control in the OpenMAUVe simulator

In this section, we focus on further enhancing the trustworthiness of the proposed control method. More specifically, we test and discuss the performance of the control law designed in Section 4.2 in a higher fidelity simulation environment.

Hover-capable AUVs can be tasked with autonomous inspection of monopiles, representing the submerged structures of the offshore wind turbines lying on the seabed (Pedersen et al., 2024). Routine inspection of monopiles is of paramount importance to monitor the structure degradation due to corrosion and biofouling, which can lead to catastrophic outcomes if advancing undetected (Sindi et al., 2024). AUVs represent the ideal candidates to effectively scale operations as a growing number of wind farms are being installed. A monopile inspection is conventionally performed by an AUV carrying a front-looking camera, moving sideways around the monopile, keeping the camera (i.e. the vehicle fore end) pointed towards the monopile.

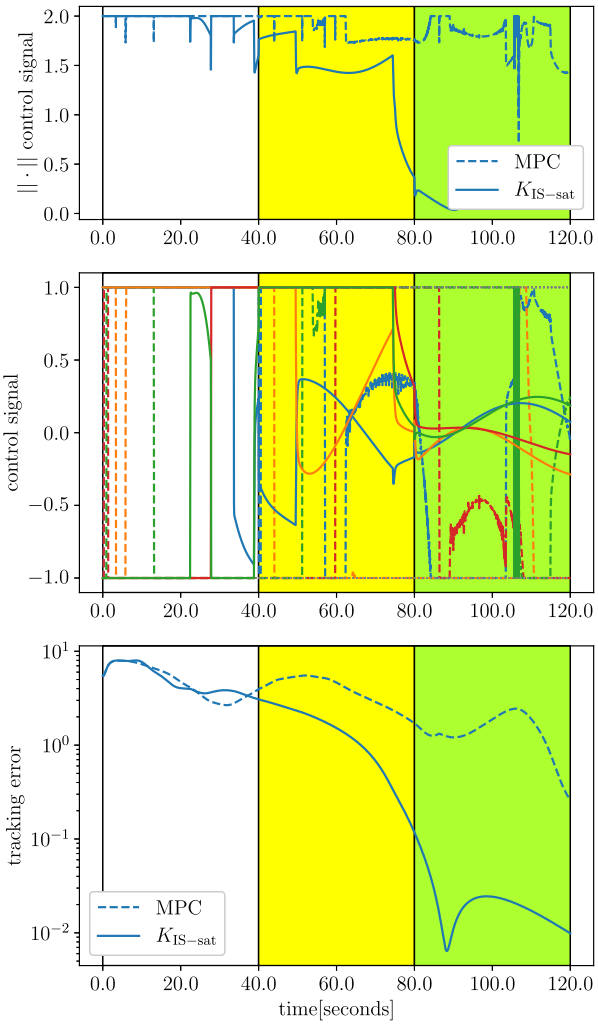


Fig. 9. Comparison of IS-sat with a nonlinear MPC, with model dynamics initialised to $x = 2$, and with the same reference signal of Fig. 8. Control signal plots have been normalised in the $[-1, 1]$ range for ease of readability. State plot legend: x_1 in blue, x_2 in orange, x_3 in green, x_4 in red; solid line associated to $K_{\text{IS-sat}}$, dashed line to MPC. It can be appreciated how, upon convergence, IS-sat tracks the reference values much more accurately.

This case study is performed within the OpenMAUVe simulator environment (Grande et al., 2025). OpenMAUVe allows to simulate the underwater dynamics of AUVs, accounting for both nonlinear cross-coupling and nonlinear effects associated to the hydrostatics and the hydrodynamics of fully submerged vehicles in motion. For the present study, a hover-capable neutrally buoyant vehicle is designed within the simulator, with viscous effects encompassing skin friction, form damping and added mass effects kept into account. Four thrusters are then added to the vehicle according to the actuator scheme previously depicted in Fig. 7.

Next, we focus on designing the overall onboard autonomous architecture as composed of two elements: (a) an overarching *guidance* law; (b) a *low-level* control law. The guidance law is in charge of defining a set of waypoints that, starting from an initial random location, allows the AUV to approach the monopile

[6] The MPC is implemented in JAX with Just-In-Time compilation and with a prediction horizon of 56 steps.

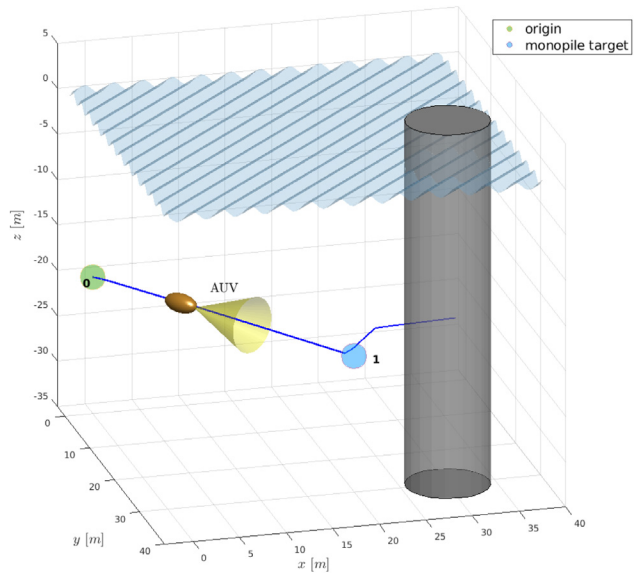


Fig. 10. AUV (in orange) moving from an initial random position (waypoint 0) to a location close to the submerged monopile (waypoint 1). The AUV hovers at a depth of 20 m, mounting a forward looking camera, with the camera viewing cone displayed in yellow.

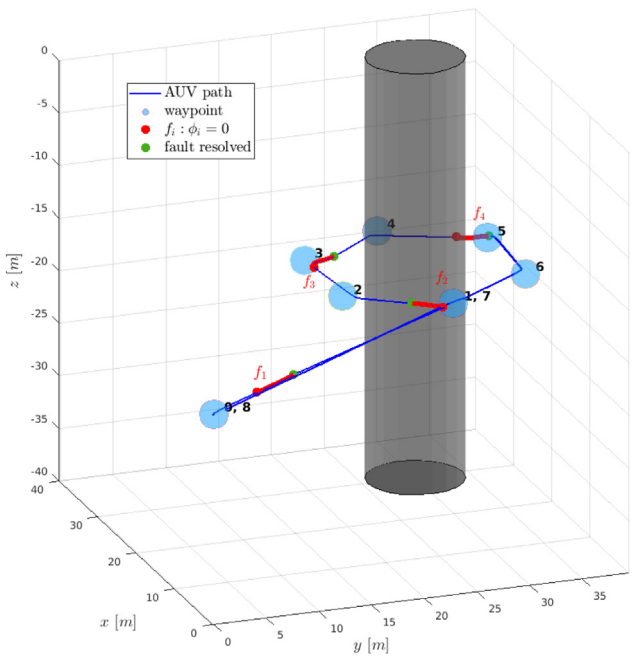


Fig. 11. AUV overall path (in blue) with the segments characterised by faults at actuators (in red).

and to perform the inspection routine by revolving around the structure. The guidance law produces the set of reference speeds and heading, which is handed over to the low-level control. In turn, the low-level control, which we focus on in this article, generates the forces to track the desired reference values. For this case study, we select a line of sight-based guidance law which allows the AUV to follow an hexagonal path around the monopile, resembling a realistic mode of operations (different guidance laws could be selected (Breivik & Fossen, 2008): for the scope of this work, we are interested in any guidance law which provides realistic time-varying references).

The first segment of the AUV path is defined such that the vehicle transitions from an initial random position (waypoint 0) to the closest point at a fixed distance of 5 m from the monopile (waypoint 1). Such an initial (approaching) phase is illustrated in Fig. 10. The overall AUV path is illustrated in Fig. 11: once the AUV terminates the inspection of the monopile (i.e. reaching waypoint 7), a command to return to the original location (waypoint 8) is sent. During the operation, the vehicle undergoes *four* fault modes, each one associated to the fault of one (unique) thruster. To verify the reliability of the proposed controller, we test the worst case scenarios, namely we test $f_i : \phi_i = 0$ for $i \in [1, 4]$, the mode corresponding to the i-th thruster complete rupture (0% of efficiency, while the other three thrusters work nominally). The faults are injected both during the approaching phase and during the inspection phase, with the faults lasting for 60 s each. The simulation automatically terminates after 2295 s, when the AUV reaches the terminal waypoint; overall, the scenario takes about 2 s to be simulated.

Finally, we analyse the tracking performance of the IS-sat control law in the aforementioned scenario. We specifically focus on x_1, x_2, x_4, as surge speed, sway speed and yaw angle are the driving variables steering the AUV during the path tracking (the references for x_3 and for x_5 are set to 0) (Fossen, 2011). The reference surge speed is illustrated in Fig. 12, the sway speed in Fig. 13 and the yaw angle tracking in Fig. 14. As can be seen, the guidance law provides a time-varying surge speed $x_1^\star = 0.12$ m/s during the approaching phase (with $x_2^\star = 0.0$ m/s), while it requires a reference $x_2^\star = -0.055$ m/s (with $x_1^\star = 0.0$ m/s) during

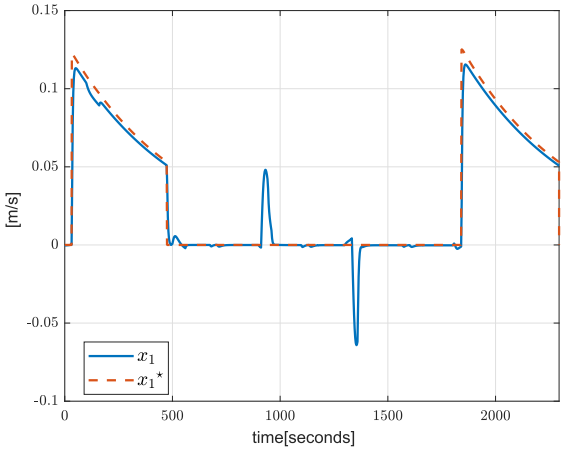


Fig. 12. AUV surge speed (x_1) reference tracking with the control law IS-sat during the path illustrated in Fig. 11.

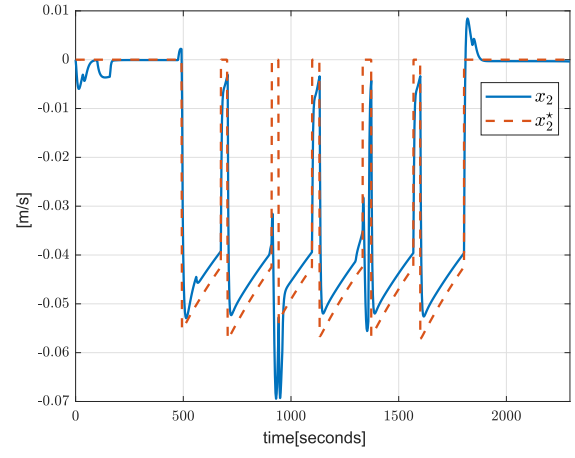


Fig. 13. AUV sway speed (x_2) reference tracking with the control law IS-sat during the path illustrated in Fig. 11.

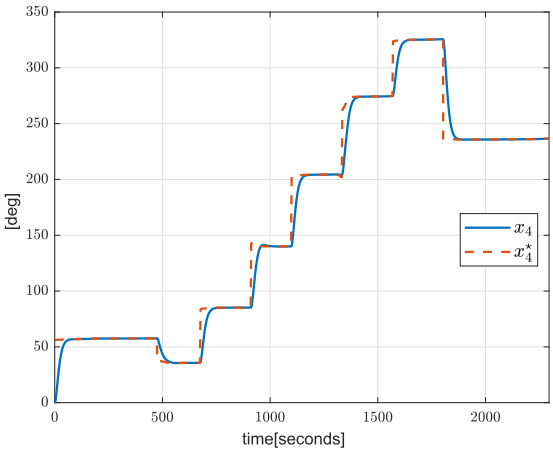


Fig. 14. AUV yaw angle (x_4) reference tracking with the control law IS-sat during the path illustrated in Fig. 11.

the inspecting phase (the negative sign denoting that the AUV is requested to move port). Both reference values are gradually decreased as the AUV approaches the next waypoint to slow down the vehicle, in turn allowing for a smoother transition to the next waypoint. As can be noticed in Fig. 14, the yaw angle reference is updated 8 times, corresponding to the requests to switch to the next waypoint. Despite the four different faults being injected, the vehicle is still able to accurately track the required setpoints. Although such an occurrence is highly unlikely to occur in practical operations, we purposely time the faults to occur when the guidance law is switching waypoint. The worst case tracking performance is in fact recorded in one such occasion: as can be seen in Fig. 13, the worst case performance occurs in the interval 920-945 s, when a relative error of 36% in the sway speed is recorded. Such high error is due to the fault injected just before waypoint 3 is reached, namely when the control law was already attempting to compensate for the change in actuator dynamics. Even in this case scenario, it can be seen that the control architecture is able to compensate for the fault, proceeding to reach the next target waypoint and eventually autonomously terminating the mission profile.

5. Conclusion

Capitalising on Lyapunov arguments and LMI reformulations, we presented an automated method to design PFTC laws tailored to nonlinear systems affected by both partial and total actuators faults, including actuator saturation and time-varying tracking capability. Our IS-sat tackles nonlinear systems via translation to linear parameter-varying models, whose uncertainty set is bounded for models accounting for actuator faults. By exploiting a counterexample-based approach, our technique is able to outperform conventional techniques both from the realm of robust control (i.e., $\mathcal{H}_\infty$) and from the nonlinear systems literature (i.e., nonlinear MPC). Notably, with respect to other counterexample-based techniques, our technique is guaranteed to converge within a finite number of iterations, either providing a control solution or declaring infeasibility. The proposed control approach provides encouraging results for experimentation with higher dimensional systems, which are usually problematic with automatic synthesis techniques. Moreover, since IS-sat consists of a static gain, it can be easily implemented in embedded applications where energy consumption is critical or where limited computational resources are allocated to the control subsystem. In the presented study, the nonlinear MPC requested ≈155x the RAM usage of IS-sat. We also concluded that computing IS-sat laws is within the computational capacity of commonly accessible contemporary hardware, having systematically completed the control synthesis for nonlinear systems affected to up to four faults within minutes. Our approach was specifically applied to the control of AUVs, but is relevant to every control application where extensive sensors or fault monitoring algorithms cannot be designed, implemented, or run in real-time due to energy, hardware or cost constraints. Future work will focus on studying the scalability to more complex, higher-dimensional models.

Acknowledgements

This work has been partially funded by the European Union – NextGenerationEU under the Italian Ministry of University and Research (MUR) through the National Innovation Ecosystem grant ECS00000041 - VITALITY (CUP: D13C21000430001). Daniele Masti is also part of INdAM/GNAMPA.

References

Abate, A., Edwards, A., & Giacobbe, M. (2022). Neural abstractions. *Advances in Neural Information Processing Systems*, *35*, 26432–26447.

Anderlini, E., Real-Arce, D. A., Morales, T., Barrera, C., Hernandez-Brito, J. J., Phillips, A. B., & Thomas, G. (2021). A marine growth detection system for underwater gliders. *IEEE Journal of Oceanic Engineering*, *46*(4), 1099–1113.

Baldini, A., Fasano, A., Felicetti, R., Freddi, A., Longhi, S., & Monteriù, A. (2018). A model-based active fault tolerant control scheme for a remotely operated vehicle. *IFAC-PapersOnLine*, *51*(24), 798–805.

Benosman, M., & Lum, K.-Y. (2009). Passive actuators' fault-tolerant control for affine nonlinear systems. *IEEE Transactions on Control Systems Technology*, *18*(1), 152–163.

Berger, G. O., & Sankaranarayanan, S. (2022). Learning fixed-complexity polyhedral Lyapunov functions from counterexamples. In *2022 IEEE 61st conference on decision and control* (pp. 3250–3255). IEEE.

Blanchini, F. (1995). Nonquadratic Lyapunov functions for robust control. *Automatica*, *31*(3), 451–461.

Blanke, M., Kinnaert, M., Lunze, J., Staroswiecki, M., & Schröder, J. (2006). *Diagnosis and fault-tolerant control*: *vol. 2*, Springer.

Breivik, M., & Fossen, T. I. (2008). Guidance laws for planar motion control. In *2008 47th IEEE conference on decision and control* (pp. 570–577). IEEE.

Brito, M., Smeed, D., & Griffiths, G. (2014). Underwater glider reliability and implications for survey design. *Journal of Atmospheric and Oceanic Technology*, *31*(12), 2858–2870.

Caiti, A., Di Corato, F., Fabiani, F., Fenucci, D., Grechi, S., & Pacini, F. (2015). Enhancing autonomy: Fault detection, identification and optimal reaction for over-actuated AUVs. In *OCEANS 2015 - genova* (pp. 1–6).

Chang, Y.-C., Roohi, N., & Gao, S. (2019). Neural Lyapunov control. *Advances in Neural Information Processing Systems*, *32*.

Dai, H., Landry, B., Pavone, M., & Tedrake, R. (2020). Counter-example guided synthesis of neural network Lyapunov functions for piecewise linear systems. In *2020 59th IEEE conference on decision and control* (pp. 1274–1281). IEEE.

Dai, H., Landry, B., Yang, L., Pavone, M., & Tedrake, R. (2021). Lyapunov-stable neural-network control. In *Robotics: science and systems*.

Duan, G.-R., & Yu, H.-H. (2013). *LMIs in control systems: Analysis, design and applications*. CRC Press.

Ducard, G. J. (2009). *Fault-tolerant flight control and guidance systems: Practical methods for small unmanned aerial vehicles*. Springer Science & Business Media.

Edwards, A., Peruffo, A., & Abate, A. (2024). Fossil 2.0: Formal certificate synthesis for the verification and control of dynamical models. In *Proceedings of the 27th ACM international conference on hybrid systems: computation and control* (pp. 1–10).

Endres, S. C., Sandrock, C., & Focke, W. W. (2018). A simplicial homology algorithm for Lipschitz optimisation. *Journal of Global Optimization*, *72*(2), 181–217.

Fabiani, F., Grechi, S., Della Tommasina, S., & Caiti, A. (2016). A NLPCA hybrid approach for AUV thrusters fault detection and isolation. In *2016 3rd conference on control and fault-tolerant systems* (pp. 111–116).

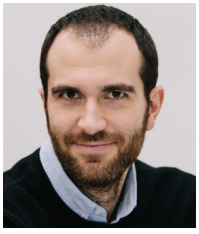
Fenucci, D., Fanelli, F., Consensi, A., Salavasidis, G., Pebody, M., & Phillips, A. B. (2024). A multi-platform guidance, navigation and control system for the autosub family of autonomous underwater vehicles. *Control Engineering Practice*, *146*, Article 105902.

Fossen, T. I. (2011). *Handbook of marine craft hydrodynamics and motion control*. John Wiley & Sons.

Grande, D. (2024). *Actuator faults in autonomous underwater vehicles: simulation and computer-aided control synthesis* (Ph.D. thesis), UCL (University College London).
Grande, D., Farì, S., Corsini, G., Grech La Rosa, A., Smith, T., Pawling, R., & Thomas, G. (2025). OpenMAUVe: an open-source modelica-based simulator for multibody underwater vehicle dynamics-demo. In *AQ2UASIM*.
Grande, D., Fenucci, D., Peruffo, A., Anderlini, E., Phillips, A. B., Thomas, G., & Salavasidis, G. (2023). Systematic synthesis of passive fault-tolerant augmented neural Lyapunov control laws for nonlinear systems. In *2023 62nd IEEE conference on decision and control* (pp. 5851–5856). IEEE.
Grande, D., Peruffo, A., Anderlini, E., Anderlini, E., & Salavasidis, G. (2023). Augmented neural Lyapunov control. *IEEE Access*.
Grande, D., Peruffo, A., Salavasidis, G., Anderlini, E., Fenucci, D., Phillips, A. B., Kosmatopoulos, E. B., & Thomas, G. (2024). Passive fault-tolerant augmented neural Lyapunov control: A method to synthesise control functions for marine vehicles affected by actuators faults. *Control Engineering Practice*, *148*, Article 105935.
He, K., Shi, S., van den Boom, T., & De Schutter, B. (2024). Approximate dynamic programming for constrained linear systems: A piecewise quadratic approximation approach. *Automatica*, *160*, Article 111456.
Horn, R. A., & Johnson, C. R. (1994). *Topics in matrix analysis*. Cambridge University Press.
Hsieh, C., Waga, M., & Suenaga, K. (2025). Certifying Lyapunov stability of black-box nonlinear systems via counterexample guided synthesis. In *28th ACM international conference on hybrid systems: computation and control*.
Hu, T., & Lin, Z. (2001). *Control systems with actuator saturation: analysis and design*. Springer Science & Business Media.
Jones, D. R., Perttunen, C. D., & Stuckman, B. E. (1993). Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, *79*(1), 157–181.
Kaminer, I., Pascoal, A. M., Silvestre, C. J., & Khargonekar, P. P. (1991). Control of an underwater vehicle using h/sub infinity/synthesis. In *[1991] proceedings of the 30th IEEE conference on decision and control* (pp. 2350–2355). IEEE.
Kangal, F., & Mengi, E. (2020). Nonsmooth algorithms for minimizing the largest eigenvalue with applications to inner numerical radius. *IMA Journal of Numerical Analysis*, *40*(4), 2342–2376.
Kothare, M. V., Balakrishnan, V., & Morari, M. (1996). Robust constrained model predictive control using linear matrix inequalities. *Automatica*, *32*(10), 1361–1379.
Liu, R.-W. (1968). Convergent systems. *IEEE Transactions on Automatic Control*, *13*(4), 384–391.
Liu, F., Ma, Z., Mu, B., Duan, C., Chen, R., Qin, Y., Pu, H., & Luo, J. (2023). Review on fault-tolerant control of unmanned underwater vehicles. *Ocean Engineering*, *285*, Article 115471.
Malherbe, C., & Vayatis, N. (2017). Global optimization of Lipschitz functions. In *ACM ICML*.
Manucci, M., Mengi, E., & Guglielmi, N. (2024). Uniform approximation of eigenproblems of a large-scale parameter-dependent hermitian matrix. arXiv preprint arXiv:2409.05791.
Masti, D., Fabiani, F., Gnecco, G., & Bemporad, A. (2023). Counter-example guided inductive synthesis of control Lyapunov functions for uncertain systems. *IEEE Control Systems Letters*, *7*, 2047–2052.
Pedersen, S., Liniger, J., Von Benzon, M., Sørensen, F. F., & Mai, C. (2024). Towards autonomous subsea maintenance on offshore structures with unmanned underwater vehicles. In *OCEANS 2024-Singapore* (pp. 1–7). IEEE.
Queste, B. Y., Heywood, K. J., Kaiser, J., Lee, G. A., Matthews, A., Schmidtko, S., Walker-Brown, C., & Woodward, S. W. (2012). Deployments in extreme conditions: Pushing the boundaries of seaglider capabilities. In *2012 IEEE/OES autonomous underwater vehicles* (pp. 1–7). IEEE.
Rubin, D., Mercader, P., Gutman, P.-O., Nguyen, H.-N., & Bemporad, A. (2020). Interpolation based predictive control by ellipsoidal invariant sets. *IFAC Journal of Systems and Control*, *12*, Article 100084.
Rudin, W., et al. (1964). *Principles of mathematical analysis*: *vol. 3*, McGraw-hill New York.
Sindi, A., Kim, H. J., Yang, Y. J., Thomas, G., & Paik, J. K. (2024). Advancing digital healthcare engineering for aging ships and offshore structures: an in-depth review and feasibility analysis. *Data-Centric Engineering*, *5*, Article e18.
Solar-Lezama, A., Tancau, L., Bodik, R., Seshia, S., & Saraswat, V. (2006). Combinatorial sketching for finite programs. In *Proceedings of the 12th international conference on architectural support for programming languages and operating systems* (pp. 404–415).
Verhaegen, M., Kanev, S., Hallouzi, R., Jones, C., Maciejowski, J., & Smail, H. (2010). Fault tolerant flight control - a survey. In *Fault tolerant flight control: a benchmark challenge* (pp. 47–89). Berlin, Heidelberg: Springer Berlin Heidelberg, ISBN: 978-3-642-11690-2.
Webster, S. E., Freitag, L. E., Lee, C. M., & Gobat, J. I. (2015). Towards real-time under-ice acoustic navigation at mesoscale ranges. In *2015 IEEE international conference on robotics and automation* (pp. 537–544). IEEE.
Wu, J., Clark, A., Kantaros, Y., & Vorobeychik, Y. (2024). Neural Lyapunov control for discrete-time systems. *Advances in Neural Information Processing Systems*, *36*.
Zanon, M., & Bemporad, A. (2021). Constrained controller and observer design by inverse optimality. *IEEE Transactions on Automatic Control*, *67*(10), 5432–5439.

Daniele Masti received his Ph.D. in Systems Science in 2021 at IMT School for Advanced Studies Lucca, Italy. From 2022 to March 2024, he was a researcher in Cyber Security at IMT School for Advanced Studies in Lucca, Italy. He currently serves as PostDoc at Gran Sasso Science Institute, Italy. His main research interest lies at the intersection between control theory and machine learning, with the overall aim of bridging the gap between the two disciplines.

Davide Grande received the B.Sc. (2015) and the M.Sc. (2015) degrees in Control engineering from Politecnico di Milano (IT), and the Ph.D. degree in Mechanical engineering (2024) from University College London (UK). From 2017 to 2018, he worked with INESC-TEC (PT) on the modelling and control of underwater vehicles, and, from 2018 to 2020, he was with Airbus Defence and Space (UK) supporting the design of spacecraft guidance and attitude control systems. Since 2023 he has been a Postdoctoral Research Fellow at University College London. His research focuses on hardware design of autonomous maritime vehicles, on fault-tolerant control systems and on formally-correct machine learning-based control methods.



Andrea Peruffo is a research fellow in formal verification at TNO-ESI. He was a postdoctoral researcher at the Delft Center for Systems and Control. He obtained the DPhil in Computer Science at the University of Oxford in 2021. He received a B.Sc. (2012) and an M.Sc. (2015, cum laude) in Automation Engineering, from the University of Padova (IT). He worked as a research engineer at Inria Rennes in 2016 and visited the Erato MMSD Group at NII, Tokyo, in 2020. His research interests include hybrid systems, formal verification and control theory.

Filippo Fabiani received the B.Sc. degree in bio-engineering, the M.Sc. degree in automatic control engineering, and the Ph. D. degree in automatic control from the University of Pisa, Pisa, Italy, in 2012, 2015, and 2019, respectively. He is currently an Assistant Professor with the IMT School for Advanced Studies Lucca, Lucca, Italy. In 2018–2019, he was Postdoctoral Research Fellow with the Delft Center for Systems and Control, TU Delft, Delft, The Netherlands, while in 2019–2022, he was a Postdoctoral Research Assistant with the Control Group, Department of Engineering Science, University of Oxford, Oxford, U.K. Since 2025, he is Associate Editor for the IEEE Control Systems Letters. His research interests include machine learning for control and game theory, with applications in smart grids and automated driving.