Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

REGULAR PAPER

An Algorithm For Computing Robust Forward Invariant Sets of Two Dimensional Nonlinear Systems

Shayok Mukhopadhyay*[1] | Fumin Zhang[2]

[1]Department of Electrical Engineering, American University of Sharjah, P.O. Box 26666, Sharjah, UAE

[2]Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, 30332, USA

Correspondence
*Shayok Mukhopadhyay, Email: smukhopadhyay@aus.edu

Summary

Robustness of nonlinear systems can be analyzed by computing robust forward invariant sets (RFISs). Knowledge of the smallest RFIS of a system, can help analyze system performance under perturbations. A novel algorithm is developed to compute an approximation of the smallest RFIS for two-dimensional nonlinear systems subjected to a bounded additive disturbance. The problem of computing an RFIS is formulated as a path planning problem, and the algorithm developed plans a path which iteratively converges to the boundary of an RFIS. Rigorous mathematical analysis shows that the proposed algorithm terminates in a finite number of iterations, and that the output of the proposed algorithm is an RFIS. Simulations are presented to illustrate the proposed algorithm, and to support the mathematical results. This work may aid future development, for use with higher dimensional systems.

KEYWORDS:
Nonlinear system, invariant set, computation, algorithm, path-planning

1 | INTRODUCTION

System outputs are often affected by disturbances. It is not necessarily a trivial task to establish a performance bound on the performance of a nonlinear system, e.g. for adaptive systems [20]. System performance in the presence of additive disturbances can be obtained using input-to-state-stability (ISS) [19] where the input represents a given disturbance. This may lead to a conservative estimate of system performance, because the shape of a ball corresponding to a particular norm may not always be identical to the shape of a set within which the system states stay. This can be understood as follows. The unit circles for few different norms are shown in Fig. 1. Also from [15] it can be seen that the for a system of the form $\dot{x} = f(t, x, u)$ such a system is called ISS if there is a class $\mathcal{KL}$ function β, and a class $\mathcal{K}$ function γ such that $\| x(t) \| \leq \beta \left(\left\| x(t_0) \right\|, t - t_0 \right) + \gamma \left(\sup_{t_0 \leq \tau \leq t} \| u(\tau) \| \right)$. For a nonlinear system affected by additive disturbances, as is the subject for this work, the input u can be replaced by the disturbance, then it is quickly apparent that any system state bounds based on the above mentioned ISS criteria involves an upper-bound dependent on the norm of the disturbance. This in addition to the shapes of unit circles shown in Fig. 1 is sufficient to convince one that there may be cases where an invariant set for a system, which may be used to evaluate system performance, may lie within sets shaped similar to the sets shown in Fig. 1. And so using the definition of ISS, and not explicitly finding the shape of such an invariant set may lead to significant room for conservativeness in the estimate of system performance bound.

A forward invariant set of a given dynamical system, is a set such that all trajectories of the given system with initial conditions $x(t_0)$ within this set remain in the set at all future times $t \geq t_0$. A robust forward invariant set (RFIS), is a forward invariant set for a given dynamical system in the presence of a disturbance of known bounded magnitude. If the smallest RFIS can be found

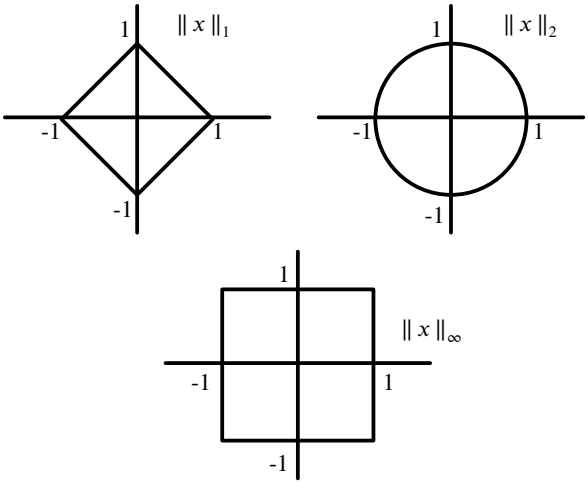

FIGURE 1 Figure showing unit circles for different norms.

around an equilibrium point of a given system, then this RFIS provides the least conservative system performance estimate around that equilibrium point in the presence of disturbances. This is because system performance bounds can be formulated as bounds on system states, and the smallest RFIS may be smaller than a ball corresponding to a particular norm used in the definition of ISS.

Work on computing RFISs exists for linear systems [16], [26]. Authors in [16] develop procedures to compute disturbance invariant (d-invariant) sets for discrete time linear time-invariant systems, where the disturbance inputs belong to a compact set. The work [26] deals with robust control invariant sets for discrete time linear systems with bounded additive disturbances, and shows that existence of a robust control invariant set can be verified by solving a convex optimization problem. Literature exists for computing polyhedral invariant sets for discrete-time linear systems [6], [7]. Computing robust control invariant sets for piecewise affine systems has been addressed by [1]. The authors in [4] use Newton's method and the secant method to find zeros for set-valued maps, and use this technique in [3] to find invariant sets for dynamical systems. Recent results exist [27], [28] for computing polyhedral invariant sets, however these are developed for discrete time linear autonomous systems. Finding the smallest RFIS, has similarities with estimating the shape and size of the domain of attraction of a given system. For a review of techniques used for domain of attraction estimation, readers are directed to [23, 21]. It is worth mentioning that use of sum-of-squares polynomials, along with optimization approaches has been widely used for computing invariant sets in the literature.

Work related to ours exists in [12], in which the problem of computing extremal trajectories and associated controllers is considered for a class of continuous-time systems with given initial conditions. Also, [11] deals with the largest estimate of the robust domain of attraction (LERDA), via the use of Lyapunov functions, and sum-of-squares techniques. Further, [34] deals with developing a numerical sampling-based algorithm for computing largest tolerable disturbance sets such that certain state constraints are satisfied. Our work is very closely related to [2], where the foundations of viability theory are provided. Viability is a special case of invariance, and the concept of an invariance envelope [2] is similar to the smallest RFIS that our work aims to compute. Further [29] shows that the viability kernel can be approximated by a sequence of discrete viability kernels, for a certain type of differential inclusions. This approach is similar to our work, where the algorithm proposed iteratively produces estimates of an RFIS, thus producing a sequence of sets which ideally converge to the required smallest RFIS. Estimation of reachable sets is another area of work related to the subject of this paper. Reachable set estimation deals with the estimation of a set containing all possible reachable states of a system given a set of initial states. Results related to reachable set estimation exist for linear systems [37], unmodeled systems via data driven approaches [10], and also for nonlinear systems of the form $\dot{x} = f(x)$ unaffected by disturbances [35]. Earlier mentioned sum-of-squares approaches also find applications in computing barrier certificates for obstacle avoidance [5], and for computing "funnels" [17] used for guaranteeing system performance within given bounds in the presence of disturbances.

A motivation for this work is related to model predictive control (MPC) [25], which may require the use of specialized or set-based [31] techniques to develop a controller meeting desired requirements [9]. As observed in [30], [32], knowledge of

the domain of attraction of a system, or the ability to estimate the shape and size of polyhedral invariant sets, is beneficial for MPC design. There is also work related to systems with non-ISS unmodeled dynamics [14], which could benefit from set-based performance evaluation that does not rely on ISS. More specifically, our work is inspired by the curve-tracking problem [36], which is important to path-following control of autonomous vehicles. Curve tracking controllers have been used for applications such as obstacle avoidance using wheeled robots, and marine sampling [22]. These controllers are robust to real world disturbances, which is justified in [19], [18] by using tools like ISS. The authors in [19] analytically find hexagonal RFISs for the curve tracking problem.

The novelty of our work is that the problem of finding the boundary of an approximation of the smallest RFIS is formulated as a path planning problem. Path planning algorithms, such as the A^* [13] algorithm, are well known to reduce computational effort, so we use the A^* algorithm as a part of the proposed RFIS computation algorithm. Also, the proposed RFIS computation algorithm is able to find an RFIS for general two-dimensional (2D) nonlinear systems with bounded additive disturbances.

Although our work deals with two dimensional systems, it is worth mentioning that creating an algorithm for finding the shape of the smallest RFIS is not a trivial task even for a two dimensional system affected by disturbances. The most significant contribution of this work is the ability to generate an estimate of the shape and size of the least conservative RFIS for 2D nonlinear systems with bounded additive disturbances. This is helpful for generating the tightest performance bounds for such nonlinear systems, which is an open problem with significant practical and theoretical value. We use path planning algorithms, and do not rely on Lyapunov based approaches or sum-of-squares relaxations based approaches, because as documented in [8], such approaches can provide conservative results. Generalizing this work to n-dimensional systems, and a detailed performance comparison with other approaches is outside the scope of this work and is left for future efforts. Preliminary results, and problem formulations exist in our previous efforts [23], [21]. In comparison to our earlier work, the specific contributions of this paper are as follows. In [23] only a preliminary formulation allowing the use of path planning algorithms for computing RFISs is presented with preliminary versions of some proofs. In [21] the above mentioned formulation is cleaned up, and it is shown that the proposed algorithm appears to produce correct shapes. In the current work, the notation used is further cleaned up. Also in [23, 21] a specific disturbance function appears to be used in the mathematical formulation, and in the result. In this paper, the formulation is altered so that no specific form of a disturbance function is required, but only discrete values/measurements of the disturbance function are required. This is also shown in the results where the RFIS computation algorithm produces results in the presence of different bounded functions used as the additive disturbance functions affecting the system dynamics. Further, for the first time, conditions on the system dynamics, and on the disturbance function and grid discretization are explicitly provided for achieving an output from the proposed algorithm that is an RFIS.

This paper is organized as follows. The required mathematical notations and definitions are presented in Section 2. The problem formulation is presented in Section 3. Details related to the proposed algorithm are presented in Section 4. Mathematical results are presented in Section 5. Simulation results are presented in Section 6. Followed by a concluding summary and discussion of future possibilities in Section 7.

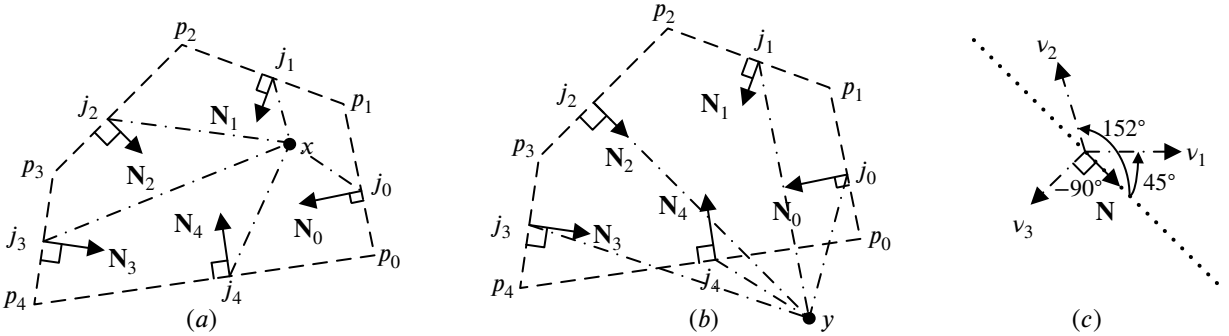

FIGURE 2 (a) A positively oriented simple closed curve is shown formed by segments joining vertices $p_0, \cdots, p_4$. Normal vectors $\mathbf{N}_0, \cdots, \mathbf{N}_4$ are shown at intermediate points $j_0, \cdots, j_4$ along these segments. Point x is lying in the interior region of a set whose boundary is a positively oriented simple closed curve, $\langle x - j_i, \mathbf{N}_i \rangle \geq 0$ for all $i \in \{0, 1, 2, 3, 4\}$. (b) Point y is lying in the exterior of a set whose boundary is a positively oriented simple closed curve, $\langle y - j_4, \mathbf{N}_4 \rangle < 0$. (c) Convention followed for measuring the angle a vector makes with the normal vector at a given point.

2 | MATHEMATICAL PRELIMINARIES

Consider the system $\dot{x}(t) = f(x(t)) + \delta(t)$. Here $f : \mathbb{R}^2 \to \mathbb{R}^2$. Also, $\delta : [0, +\infty) \to \mathcal{U}$, and $\mathcal{U} \subset \mathbb{R}^2$ is a compact set. Here $x(t)$ is a solution to $\dot{x}(t) = f(x(t)) + \delta(t)$. Further, $f(\cdot)$ represents system dynamics, and $\delta(\cdot)$ is a bounded disturbance. For computational simplicity, the set $\mathcal{U}$ is defined as $[-\delta_0, \delta_0] \times [-\delta_0, \delta_0]$, for some known value $\delta_0 \in [0, +\infty)$. This box structure for $\mathcal{U}$ is not essential, and can be relaxed. The definition of an RFIS is given as below.

Definition 1 (Robust forward invariant set)**.** A set $\mathcal{X} \subset \mathbb{R}^n$ is an RFIS, if for all initial conditions $x(t_0) \in \mathcal{X}$, and all $\delta(t) \in \mathcal{U}$ the solution $x(t) \in \mathcal{X}$ for all $t > t_0$.

A connected curve that does not cross over itself is called a simple closed curve. A simple closed curve is defined to be positively oriented if the interior region enclosed by such a curve lies to the left as the curve is traced anticlockwise. Given a point $p_0 \in \mathbb{R}^2$ on a positively oriented simple closed curve, suppose one arrives at point p_1 by tracing the curve anticlockwise. The tangent vector at p_0 is given by the vector $\lim_{\| p_1 - p_0 \| \to 0}(p_1 - p_0)$. The normal vector at p_0 is obtained by rotating the tangent vector anticlockwise through $\pi/2$ radians. Such a normal vector is said to point towards the 'left', or towards the interior region of the given positively oriented simple closed curve, at point p_0. Figure 2 provides an illustration of the above terms related to a simple closed curve. Given two points a and b belonging to $\mathbb{R}^2$, let seg $(a; b)$ denote the line segment joining points a, and b. A set $S \in \mathbb{R}^n$ is a *star-shaped set* if there is a $\bar{x} \in S$ such that for all $x \in S$, seg $(\bar{x}; x)$ lies in S.

Let $P = \{p_1, \cdots, p_n\}$ and $Q = \{q_1, \cdots, q_n\}$ be two sequences of equal length n consisting of points p_i and $q_j \in \mathbb{R}^2$ respectively. Let the distance between sequences P and Q be $d_2(P, Q) = \sum_{l=1}^{n} \| p_l - q_l \|_2$.

3 | PROBLEM FORMULATION

In this section the problem of finding the smallest RFIS is formulated, as an optimization problem.

3.1 | Problem statement

Using the notation and definitions in Section 2 the problem of interest to this work is stated as follows. Consider a system of the form

$$\dot{x}(t) = f(x(t)) + \delta(t) \tag{1}$$

where $f(x)$, $\delta(t)$ are Lipschitz locally in $x \in \mathcal{D}$, globally in $t \in \mathbb{R}$ respectively. The function $\delta(t)$ is not known, but $\delta(t) \in [-\delta_0, \delta_0] \times [-\delta_0, \delta_0]$, and δ_0 is known. Starting from an initial guess $B \subseteq \mathcal{D}$, find the smallest RFIS (see assumption 1) for the system in (1). The set $\mathcal{D}$ is the region of interest. Functions $f(\cdot)$, $\delta(\cdot)$ are assumed Lipschitz, because this is required for uniqueness of solutions, which is needed for the existence of an invariant set [8, 33, 2].

Assumption 1. We assume that the smallest RFIS for (1) with disturbance $\delta(\cdot)$ valued in $[-\delta_0, \delta_0] \times [-\delta_0, \delta_0]$ exists as a star-shaped RFIS contained in B. The smallest RFIS has the smallest area among all RFIS in B, and has a boundary that is a positively oriented simple closed curve.

For any system of the form in (1), if there is a domain of interest $\mathcal{D} \in \mathbb{R}^2$ which contains invariant sets, then it is reasonable to assume that one such invariant set will have smallest area. The boundary of such a set can be approximated by a piecewise linear closed curve (see Figure 3(a)). Details related to the choice of constructing star-shaped RFISs, and whether B is needed to be invariant or not, are provided further in the paper. Note that conditions for existence of RFISs, or smallest RFISs are beyond the scope of this paper, and not of interest to the material at hand. The focus of this paper is producing a computational algorithm for finding RFISs, and providing appropriate output to the user if an RFIS can, or cannot be found by the proposed algorithm in an initially guessed set B.

3.2 | Motivation for using star-shaped sets and path-planning algorithms

The literature [24] has results with the 'contingent cone' (or tangent cone) $C_{\mathcal{S}}(x)$ which uses distance between sets to find convex invariant sets for systems of the form $\dot{x}(t) = f(x(t))$, with the assumption that solutions are unique. This result holds for non-convex sets [8], and is further extended to systems of the form $\dot{x}(t) = f(x(t), w(t))$ in [2]. In both cases a set $\mathcal{S}$ is positively

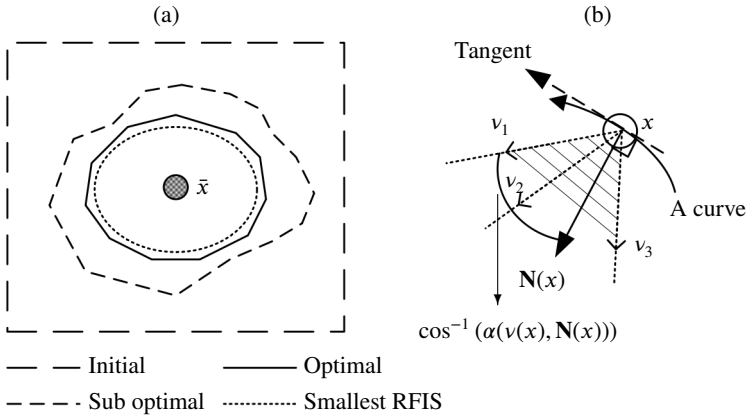

FIGURE 3 Illustrations: (a) Types of solutions to the problem of finding an RFIS. (b) An illustration of the quantity $\alpha(\nu(x), \mathbf{N}(x))$ at a point $x \in \mathbb{R}^2$. The magnitude of the angle between vector ν_1 and the normal vector $\mathbf{N}$ is bigger than the magnitudes of the angles between vectors ν_2 and $\mathbf{N}$, ν_3 and $\mathbf{N}$. All these angles are acute, so by properties of the cosine function, the value of the cosine of the angle between vector ν_1 and the normal vector $\mathbf{N}$ is the smallest compared to the other two vectors.

invariant if $f(\cdot) \in \mathcal{C}_{\mathcal{S}}(x)$ for all $x \in \mathcal{S}$. This can be replaced by $f(\cdot) \in \mathcal{C}_{\mathcal{S}}(x)$ for all $x \in \partial\mathcal{S}$ [8], where $\partial\mathcal{S}$ denotes the boundary of $\mathcal{S}$. As mentioned in [8], the geometric interpretation of the above results is the following.

Definition 2 (**Boundary condition**)**. If** for $x \in \partial\mathcal{S}$ the derivative $\dot{x}(t)$ points inside $\mathcal{S}$ or is tangential to $\mathcal{S}$, **then** the solution $x(t)$ stays in $\mathcal{S}$.

The above stated boundary condition can be intuitively understood as follows. Consider particles suspended in a flow field. If at every point on the boundary of a closed set, any flow is directed only inward into the set, or at worst directed along the boundary of the set, then no particles can ever leave such a set. Because the moment a particle reaches the set boundary, a particle is always acted upon by a flow which pushes it inside the set, or pushes it along the boundary of the set, but never pushes it outside the set, simply because the flow never takes such an outward flowing direction. Thus motivated, our work focuses on finding an approximation of the smallest RFIS which we define as follows.

Definition 3 (Approximation of the smallest RFIS)**.** An approximation of the smallest RFIS for the system in (1), is a star-shaped set in $\boldsymbol{B} \subset \mathbb{R}^2$ whose boundary is a simple closed curve such that the above **boundary condition** is satisfied, with disturbance $\delta(\cdot) \in [-\delta_0, \delta_0] \times [-\delta_0, \delta_0]$, and such that this set has the smallest area.

Consider a star-shaped set with a polygonal boundary such that the boundary doesn't intersect with itself, i.e. the boundary is a simple closed curve. Now consider triangles formed by joining line-segments from a point $\bar{x}$ in the interior of such a set, with two consecutive vertices of the polygonal boundary. The sum of the areas of all such triangles, is the area of the polygon. Thus minimizing the area of the polygon requires minimizing the area of each triangle, which can be achieved by reducing the lengths of the sides of the triangles. Next, it is shown that existing path planning algorithms can efficiently solve the problem of finding a star-shaped polygonal set satisfying the **boundary condition**, and minimize its area.

3.3 | Discretization of the search space

Let us begin with an initial guess for the smallest RFIS called $\boldsymbol{B}$. A naive, brute force approach to find the smallest RFIS would involve picking any n random points $a_1, a_2, \cdots, a_n$, then joining consecutive points $a_1, a_2; a_2, a_3; a_3, a_4; \cdots; a_{n-1}, a_n$, by line segments to form a simple closed curve, and then verifying if definition 3 holds for a set with the above simple closed curve as boundary. This process needs to be repeated an infinitely many number of times, until a result is achieved. Such a process may never end. Then, one can resort to Lyapunov based methods, which yield conservative results. So to get less conservative results, but also to reduce the number of points within $\boldsymbol{B}$ which need to be examined, we discretize the space in which an approximation of the smallest RFIS is to be found, as follows. Consider set $\boldsymbol{B}$, and a point $\bar{x}$ located in its interior. A radial grid is superimposed on $\boldsymbol{B}$ with grid lines emanating outward from $\bar{x}$. This particular grid choice helps find star-shaped sets. To create this grid, select $N + 1$ distinct points $\{b_0, b_1, \cdots, b_N\}$ from the boundary of $\boldsymbol{B}$. Then connect $\bar{x}$ to each point $b_0, b_1, \cdots, b_N$. Let the index

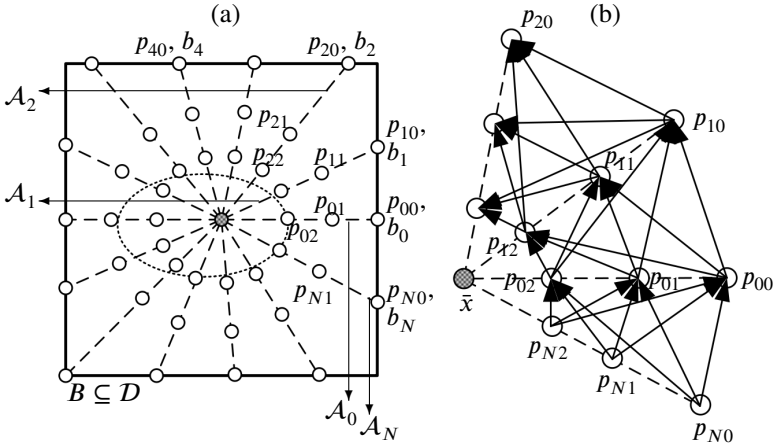

FIGURE 4 (a) Discretizing the search space to find the smallest RFIS (dotted ellipse) contained within a initial guess set $\boldsymbol{B}$. The shaded circle is a point $\bar{x}$ located within $\boldsymbol{B}$. (b) Embedding a directed graph structure - only partial graph structure shown.

$i \in \{0, 1, \cdots, N\}$. For simplicity, choose b_i so that the angle θ_i between each segment seg $\left(b_i; \bar{x}\right)$ and the x-axis is $\frac{2\pi i}{N+1}$. The points b_0, b_1, b_2, and b_N are indicated on the boundary of the square set in Figure 4(a). The set $\boldsymbol{B}$ is convex by choice, and $\bar{x}$ is in the interior of $\boldsymbol{B}$. So, $\boldsymbol{B}$ contains all line segments joining $\bar{x}$ to each point b_i by construction, i.e. it is star-shaped. Next, discretize each segment seg $\left(b_i; \bar{x}\right)$ into $n+1$ grid points p_{ij} where $i \in \{0, 1, \cdots, N\}$, $j \in \{0, 1, \cdots, n\}$, and $p_{ij} = b_i + \frac{j(\bar{x}-b_i)}{n}$. Let set $\mathcal{A}_i$ be a set of grid points $\{p_{ij} \in \text{seg}\left(b_i; \bar{x}\right)\}$. The sets $\mathcal{A}_0$ to $\mathcal{A}_N$ are illustrated by dashed line segments in Figure 4. The grid points p_{ij} belonging to each set $\mathcal{A}_i$ are marked by circles. Further, embedding the graph structure described in the following sub-section, allows converting the problem of computing an approximation of the smallest RFIS into $N+1$ separate unconstrained optimization problems.

3.4 | Embedding a directed graph structure

To produce positively oriented simple closed curves around $\bar{x} \in \boldsymbol{B}$, the following graph structure is proposed. Excluding $\bar{x}$, connect every grid point in set $\mathcal{A}_i$ to every grid point in set $\mathcal{A}_{i+1}$. Do not connect any grid point in a set $\mathcal{A}_i$ to any other grid point in $\mathcal{A}_i$, or to any grid point in set $\mathcal{A}_{i-1}$, i.e. paths cannot go radially towards $\bar{x}$ or clockwise around $\bar{x}$. To produce closed paths, the index i wraps around i.e. $i+1=0$ if $i=N$, and $i-1=N$ if $i=0$. This connects grid points (excluding $\bar{x}$) in set $\mathcal{A}_N$ to grid points in set $\mathcal{A}_0$. An illustration of this graph structure, which is named $\boldsymbol{G'}$, is seen in Figure 4(b). Paths exist from grid points $p_{00}, p_{01}, p_{02} \in \mathcal{A}_0$ directed to grid points $p_{10}, p_{11}, p_{12} \in \mathcal{A}_1$. Similarly, directed paths exist from point $p_{N2} \in \mathcal{A}_N$, to points $p_{00}, p_{01}, p_{02} \in \mathcal{A}_0$.

By construction, if a set of paths is chosen from graph $\boldsymbol{G'}$ to form a directed simple closed curve, then the set formed by the region enclosed by such a curve contains $\bar{x}$. This is due to the fact that by-construction such a set formed is a star-shaped set with a simple closed curve boundary. The structure of graph $\boldsymbol{G'}$ is chosen to help produce star-shaped sets, which are essential to find a set of least area, as said in Section 3.2.

3.5 | Formulation of an optimization problem

The task of finding a smallest RFIS whose boundary is a simple closed curve involves picking points from the graph $\boldsymbol{G'}$ so that a polygonal set is formed. So for each edge of the polygon two points must be picked. Suppose for a particular edge, two points $p_{ij} \in \mathcal{A}_i$, and $p_{(i-1)k} \in \mathcal{A}_{i-1}$ are picked where $i \in \{0, 1, \cdots, N\}$, $j \in \{0, 1, \cdots, n\}$, then let

$$\mathbf{N}(p_{(i-1)k}, p_{ij}) = \Gamma_{\pi/2}(p_{ij} - p_{(i-1)k}), \tag{2}$$

The symbol $\Gamma_{\pi/2}$ represents the rotation matrix in $\mathbb{R}^2$, i.e.

$$\Gamma_{\pi/2} = \begin{bmatrix} \cos(\pi/2) & -\sin(\pi/2) \\ \sin(\pi/2) & \cos(\pi/2) \end{bmatrix} = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}. \tag{3}$$

The definition of vector $\mathbf{N}(p_{(i-1)k}, p_{ij})$ is understood as follows. The goal is to construct a positively oriented simple closed curve, which forms the boundary of the smallest RFIS (a polygon). A segment seg $\left(p_{(i-1)k}; p_{ij}\right)$ joining points p_{ij}, $p_{(i-1)k}$ is an edge of a polygon. This segment seg $\left(p_{(i-1)k}; p_{ij}\right)$ is to be tested to find if it qualifies as a part of the boundary of the required smallest RFIS, i.e. if the **boundary condition** is satisfied. By definition, a positively oriented simple closed curve has its interior on the left, and the exterior on the right when the curve is traced anticlockwise. Also by definition of such a curve, the positive direction of the normal vector points inwards (i.e. to the left). This is why the definition in (2) uses a rotation matrix which turns the candidate segment seg $\left(p_{(i-1)k}; p_{ij}\right)$ anticlockwise by $\pi/2$ radians to get an estimated direction of a "normal" vector $\mathbf{N}(p_{(i-1)k}, p_{ij})$. Further the task of testing if seg $\left(p_{(i-1)k}; p_{ij}\right)$ satisfies the **boundary condition**, is done as follows. Let,

$$\mathbf{F}(x) = \{ f(x) + \delta(\cdot) | x \in \mathbb{R}^2, \delta(\cdot) \in \{d_0, \cdots, d_s\} \times \{d_0, \cdots, d_s\} \}, \tag{4}$$

here $f(\cdot), \delta(\cdot)$ are from (1), and $d_{\bar{n}} = -\delta_0 + \dfrac{2\bar{n}\delta_0}{s}$, $\bar{n} = \{0, 1, \cdots, s\}$, where $s \in \mathbb{N}$. The set $\mathbf{F}(x)$ contains all vectors which satisfy (1), at a point $x \in \mathbb{R}^2$, and evaluated over a set of possible discrete disturbances valued in $\mathcal{U}$, i.e. $[-\delta_0, \delta_0] \times [-\delta_0, \delta_0]$. Note that to compute an RFIS, the analytical form of $\delta(t)$ is not needed. Only, the maximum, minimum values that each component of $\delta(t)$ can assume, are needed to sample the disturbance values in the set $\mathcal{U}$. Further let two points $p_{ij} \in \mathcal{A}_i$, and $p_{(i-1)k} \in \mathcal{A}_{i-1}$ be given, and consider the following.

$$v^\star(p_{ij}) = \begin{cases} \underset{v(p_{ij}) \in \mathbf{F}(p_{ij}),}{\arg\min} \dfrac{\langle \mathbf{N}(p_{(i-1)k}, p_{ij}), v(p_{ij}) \rangle}{\left\| \mathbf{N}(p_{(i-1)k}, p_{ij}) \right\| \left\| v(p_{ij}) \right\|}, \text{if } \| \mathbf{N} \| \| v \| \neq 0 \\ \\ \mathbf{N}(p_{(i-1)k}, p_{ij}), \text{ if } \| \mathbf{N} \| \| v \| = 0 \end{cases} \tag{5}$$

The vector $v^\star(p_{ij})$ represents a vector for which the term $\frac{\langle \mathbf{N}(p_{(i-1)k}, p_{ij}), v(p_{ij}) \rangle}{\| \mathbf{N}(p_{(i-1)k}, p_{ij}) \| \| v(p_{ij}) \|}$ has minimum value. This term is the cosine of the angle between the vector $\mathbf{N}(p_{(i-1)k}, p_{ij})$, and all vectors $v(p_{ij}) \in \mathbf{F}(p_{ij})$ satisfying (1) and affected by all possible disturbance values that $\delta(\cdot)$ can take. The term $\frac{\langle \mathbf{N}(p_{(i-1)k}, p_{ij}), v(p_{ij}) \rangle}{\| \mathbf{N}(p_{(i-1)k}, p_{ij}) \| \| v(p_{ij}) \|}$ can take values in $[-1, 1]$. Any vector $v(p_{ij})$ whose angle with respect to $\mathbf{N}(p_{(i-1)k}, p_{ij}) \in [-90, 90]$ degrees (i.e. cosine of such an angle belongs to $[0, 1]$), is a vector which points towards the 'left' of seg $\left(p_{(i-1)k}; p_{ij}\right)$, or points along seg $\left(p_{(i-1)k}; p_{ij}\right)$. If such a seg $\left(p_{(i-1)k}; p_{ij}\right)$ is selected as a candidate edge of a polygonal set, and subsequent such choices are made to produce a polygon with a positively oriented simple closed curve boundary, then the edges of this polygonal set would satisfy the **boundary condition**. Thus enforcing the boundary condition for every seg $\left(p_{(i-1),k}; p_{ij}\right)$ requires finding vectors for every seg $\left(p_{(i-1),k}; p_{ij}\right)$, for which the term $\frac{\langle \mathbf{N}(p_{(i-1)k}, p_{ij}), v(p_{ij}) \rangle}{\| \mathbf{N}(p_{(i-1)k}, p_{ij}) \| \| v(p_{ij}) \|}$ has minimum value. And by the definition of the cosine function, if this minimum value is positive, this implies that a given seg $\left(p_{(i-1)k}; p_{ij}\right)$ satisfies the **boundary condition**. Thus we define, the cosine of the angle between $v^\star(p_{ij})$, and

$$\alpha\left(v^\star(p_{ij}), \mathbf{N}(p_{(i-1)k}, p_{ij})\right) = \frac{\langle \mathbf{N}(p_{(i-1)k}, p_{ij}), v^\star(p_{ij}) \rangle}{\left\| \mathbf{N}(p_{(i-1)k}, p_{ij}) \right\| \left\| v^\star(p_{ij}) \right\|}, \tag{6}$$

and $\mathbf{N}(p_{(i-1)k}, p_{ij})$ as per (6). Also, from the above discussion, enforcing the **boundary condition** requires $\alpha(\cdot, \cdot) \geq 0$ for every seg $\left(p_{(i-1)k}; p_{ij}\right)$. The quantity $\alpha\left((v(\cdot), \mathbf{N}(\cdot))\right)$ is illustrated in Fig. 3 (b). It is worth noting that in Fig. 3 (b) the quantity $\alpha\left((v(\cdot), \mathbf{N}(\cdot))\right)$ is in terms of the variable $x \in \mathbb{R}^2$ (a continuous spatial variable). However for computation purposes we have to discretize the search space, as described in section 3.3. So here, and from now on the quantity $\alpha\left((v(\cdot), \mathbf{N}(\cdot))\right)$ is written in terms of a point p_{ij} in the discretized space. Also the normal vector cannot be defined with relation to a smooth curve as shown in Fig. 3 (b) because we are trying to reconstruct/find piece by piece, the curve which will form the boundary of the required smallest RFIS. So a normal vector perpendicular to a segment of such a curve being sought is represented by $\mathbf{N}(p_{(i-1)k}, p_{ij})$, here $p_{(i-1)k}, p_{ij}$ are endpoints of the segment under consideration.

Now the problem of computing an approximation of the smallest RFIS can be formulated as the following optimization problems. For each $i \in \{0, 1, \cdots, N\}$, given a point $p_{(i-1)k} \in \mathcal{A}_{i-1}$, $k \in \{0, 1, \cdots, n\}$ find a point $p_{ij}^\star \in \mathcal{A}_i$ minimizing the following

$$\sum_{\substack{i=0 \\ j \in \{0, 1, \cdots, n\}}}^{i=N} c(p_{(i-1)k}, p_{ij}), \tag{7}$$

where

$$c(p_{(i-1)k}, p_{ij}) = \eta(\alpha(v^\star(p_{ij}), \mathbf{N}(p_{(i-1)k}, p_{ij}))) + \left\| p_{ij} - \bar{x} \right\| \tag{8}$$

and

$$\eta(x) = \begin{cases} x, \text{ if } x \in [0,1] \\ \tilde{g}, \text{ if } x \in [-1,0), \quad \text{where } \tilde{g} > 0. \end{cases} \tag{9}$$

Equations (7)-(9) can be understood as follows. The goal is to use path planning algorithms to solve (7) and get points which can be joined by line segments to form a polygonal set which is also the smallest RFIS. Suppose it takes a certain cost $c(p_{ij}, p_{(i-1)k})$ to traverse along seg $\left(p_{(i-1)k}; p_{ij}\right)$ which is an edge of a polygon, (7) represents the total cost to traverse the boundary of the polygon. The first term on the R.H.S. of (8) is called the *boundary criteria checker* term, because it represents a component of the cost which ensures that every seg $\left(p_{(i-1)k}; p_{ij}\right)$ satisfies the **boundary condition**. So whenever an argument $\alpha(\cdot,\cdot) \geq 0$ is passed to the function $\eta(\cdot)$, the output is the argument itself, however if $\alpha(\cdot,\cdot) < 0$, then the function $\eta(\cdot)$ outputs a certain positive value $\tilde{g}$. Lemma 1 shows that if $\tilde{g}$ is a sufficiently large positive number, then the **boundary condition** is satisfied for seg $\left(p_{(i-1)k}; p_{ij}\right)$. The second term in (8) penalizes points p_{ij} which are far from the point $\bar{x} \in B$. If each point $p_{ij} \in \mathcal{A}_i$ is picked as close as possible to $\bar{x}$, then this reduces the areas of the triangles with vertices $p_{ij}, \bar{x}, p_{(i-1)k}$, thus reducing the area of the overall polygonal set as discussed in Section 3.2. Because each term in (8) is positive, minimizing the overall cost in (7) minimizes $c(\cdot,\cdot)$ in (8) for each $i \in \{0, 1, \cdots, N\}$, and thus satisfies the requirements of definition 3. A path planning approach, discussed below, can solve the above optimization problems.

4 | LEVERAGING PATH PLANNING ALGORITHMS

The following sections first provide some background information about the A^* path planning algorithm, and then provide information on how it is used in this paper.

4.1 | Background information about the A^* algorithm

Figure 5 shows a graph with a source node (shaded in gray), a goal node (shaded solid black), and the current node at which path planning has reached i.e. node 'n'. The numbers mentioned on the links in Fig. 5 are the path costs, solid paths represent paths already taken to reach node 'n', and the dashed paths represents paths not taken yet. The task for the A^* path planning algorithm is to find a least cost path from the source node to the goal node. For this, A^* uses costs calculated as follows. The cost to get to node 'n' from the source is $2 + 3 = 5$, which is represented as $g(n)$, and the cost to go from node 'n' to the goal node is represented as $h(n)$. This second cost function $h(n)$ is a heuristic cost, i.e. based on experience. When these costs are added, it gives total navigation cost $f(n) = g(n) + h(n)$ at a node 'n'. It is realistic to assume that actual values of $g(n), h(n)$ are not available accurately. So, the A^* algorithm [13] uses an evaluation function of the form $\hat{f}(\cdot) = \hat{g}(\cdot) + \hat{h}(\cdot)$. Here $\hat{g}(\cdot)$ is the estimated cost to go from a source node to a given node 'n', and $\hat{h}(\cdot)$ is the estimated cost-to-go to a goal node from a given node 'n'. The functions $\hat{h}(\cdot)$ and $h(\cdot)$ are known as estimated, and actual heuristic costs. The function $\hat{h}(\cdot)$ is chosen based on experience. As long as $\hat{h}(\cdot) \leq h(\cdot)$, and the actual cost $g(\cdot)$ is positive for traveling between any two consecutive nodes in a given graph, then the A^* algorithm terminates and produces an optimal path [13] i.e. minimizes $\hat{f}(\cdot)$. These details can be found in [13, Theorem 1]. As an example of a heuristic cost, we can see that the cost to reach the goal node from any node in Fig. 5, is

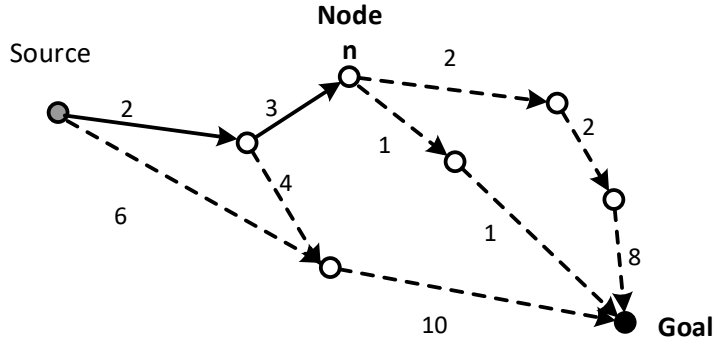

FIGURE 5 An illustration for A^* based path planning.

at least 1. So a choice for an estimated heuristic cost function could be $\hat{h}(n) = 1$, this way the requirement $\hat{h}(n) \leq h(n)$ is met. For further detailed information on A^* readers are referered to [13].

4.2 | The proposed algorithm for finding an approximation of the smallest RFIS

Path planning algorithms need a graph, a starting point (source node), endpoint (goal node), and an evaluation function to estimate the cost of a path. Readers are directed to [13] for details on path planning. Finding the smallest RFIS requires the set of source nodes and goal nodes to be the same, so that the starting and ending point of the closed path coincide. The A^* algorithm [13] does not start planning a path if the source node belongs to the goal set. To separate the source nodes from the goal set, a path must be planned in at least two parts. The first part originates at a source node belonging to the set $\mathcal{A}_0$ and terminates in the set $\mathcal{A}_{N'}$, where $N' = \lfloor N/2 \rfloor$. The second part originates at the node where the first part of the path terminated in a set $\mathcal{A}_{N'}$, and the goal set is the set $\mathcal{A}_0$.

Given a graph G' as mentioned in Section 3.4, the following cases are considered for computing the evaluation function $\hat{f}(\cdot)$ required by A^* to find an optimal path, which is hypothesized to be the boundary of the smallest RFIS. For (10)-(11), $i \in \{1, \cdots, N\}$, and $j, k, l \in \{0, 1, \cdots, n-1\}$. When traveling from a point (node) $p_{(i-1)k} \in \mathcal{A}_{i-1}$ to a point (node) $p_{ij} \in \mathcal{A}_i$,

$$\hat{f}(p_{(i-1)k}, p_{ij}) = \hat{g}(p_{(i-1)k}, p_{ij}) + \hat{h}(p_{(i-1)k}, p_{ij}), \tag{10}$$

where $\hat{h}(p_{(i-1)k}, p_{ij}) = 0$, and

$$\hat{g}(p_{(i-1)k}, p_{ij}) = c(p_{(i-1)k}, p_{ij}) + \hat{g}(p_{(i-2)l}, p_{(i-1)k}), \tag{11}$$

with the only exception that the initial cost $\hat{g}(p_{Nl}, p_{0k}) = 0$ for all $l, k \in \{0, 1, \ldots, n-1\}$.

Now, Algorithm 1 presents the pseudo code proposed to find an approximation of the smallest RFIS, the operation of Algorithm 1 is described in the following subsection.

4.3 | Operation of the RFIS computation algorithm

The quantity r in Algorithm 1 is the iteration count with the initial value 1, $r_{\max}$ represents the maximum number of iterations. The values of n, N control the discretization of the search space, the value of s controls the discretization of the disturbance set. The sequence P_0 is initialized to $\{b_0, b_1, \cdots, b_N\}$ where each point $b_i \in \mathcal{A}_i$, $i \in \{0, 1, \cdots, N\}$, lies on the boundary of the initial guess $\boldsymbol{B}$. Algorithm 1 generates sequence P_r which iteratively approaches the desired RFIS boundary. The variable σ measures if P_r has converged to a particular sequence of points, σ is initialized to a value greater than ϵ, where ϵ is a small non-zero positive constant chosen by the user.

Let $\tilde{p}_{r,0}$ represent a source node $p_{0j} \in \mathcal{A}_0$. During every iteration r, the first segment of the resulting path is planned starting from the point $\tilde{p}_{r,0}$ to the goal set $\mathcal{A}_{N'}$, where $N' = \lfloor N/2 \rfloor$. The points (nodes) picked by A^* from the sets $\mathcal{A}_0, \cdots, \mathcal{A}_{N'}$ are stored into the sequence P_r in the order $\{\tilde{p}_{r,0}, \tilde{p}_{r,1}, \tilde{p}_{r,2}, \cdots, \tilde{p}_{r,N'}\}$ The above process is shown graphically in Fig. 6. The second segment of the resulting path is planned starting from the point $\tilde{p}_{r,N'} \in \mathcal{A}_{N'}$ to the goal set $\mathcal{A}_0$. The sequence P_r is updated with the points picked by A^* from the sets $\mathcal{A}_{N'+1}, \cdots, \mathcal{A}_N$ as $P_r = \{\tilde{p}_{r,0}, \tilde{p}_{r,1}, \cdots, \tilde{p}_{r,N'}, \tilde{p}_{r,(N'+1)}, \cdots, \tilde{p}_{r,(N-1)}, \tilde{p}_{r,N}\}$, where $\tilde{p}_{r,i} \in \mathcal{A}_i$, and $i \in \{0, 1, \cdots, N\}$. Also let the sequence $\Psi_0 = \{0, \cdots, 0\}$ be initialized with N zeros as shown in Algorithm 1. Sequence Ψ_r is updated in each iteration r with *boundary criteria checker* terms as defined in (8). By definition and from the discussion following (8), given two points $\tilde{p}_{r,(i-1)}, \tilde{p}_{r,i}$ the *boundary criteria checker* term for seg $\left(\tilde{p}_{r,(i-1)}; \tilde{p}_{r,i}\right)$ is $\eta\left(\alpha(v^\star(\tilde{p}_{r,i}), \mathbf{N}(\tilde{p}_{r,(i-1)}, \tilde{p}_{r,i}))\right)$.

The algorithm continues until the given maximum number of iteration $r_{\max}$ are performed, or if $\sigma < \epsilon$, i.e. the sequence P_r has converged to some sequence of points chosen by the A^* algorithm. If the maximum number of iterations are reached and $\sigma > \epsilon$, this means Algorithm 1 has not converged to a particular sequence of points, thus the algorithm exists by returning empty sequences P_r, Ψ_r. On the other hand, if in a certain iteration $r < r_{\max}$, we have $\sigma < \epsilon$, then the Algorithm 1 checks if $\tilde{g} \in \Psi_r$. If true, it means that for the least cost path that Algorithm 1 could find in the given initial set $\boldsymbol{B}$, there is at least one seg $\left(\tilde{p}_{r,(i-1)}; \tilde{p}_{r,i}\right)$ that does not satisfy the **boundary condition**, and thus the result output by the algorithm is not an RFIS. If either of the above events occur, the user can change the initial guess, or change the location of $\bar{x}$, or change the spatial and disturbance set discretization by changing values of n, N, s, or increase $r_{\max}$ and run Algorithm 1 again. Further if $\sigma < \epsilon$, and $\tilde{g} \notin \Psi_r$ then no seg $\left(\tilde{p}_{r,(i-1)}; \tilde{p}_{r,i}\right)$ violates the **boundary condition**, and thus it is hypothesized that the result output by Algorithm 1 is a candidate for an approximation of the smallest RFIS. The following section proves the above hypothesis.

Algorithm 1 Compute an approximation of a robust forward invariant set

Data: Sets $\mathcal{A}_0$ to $\mathcal{A}_N$, graph G', $\hat{f}(\cdot)$, $p_{0j} \in \mathcal{A}_0$, positive numbers $\epsilon, \tilde{g}, \in \mathbb{R}$, and $n, N, s, r_{\max} \in \mathbb{N}$.
Result: P_r, Ψ_r

1: Let $r = 1, N' = \lfloor N/2 \rfloor, \sigma = 2\epsilon, P_0 = \{b_0, b_1, \cdots, b_N\}$;
2: Let $\tilde{p}_{r,0} = p_{0j} \in \mathcal{A}_0, \Psi_0 = \{0, 0, \cdot\ (N \text{ elements}) \cdot, 0\}$;
3: **while** $(\sigma > \epsilon)$ **and** $(r \leq r_{\max})$ **do**
4: Use A^* to find an optimal path from $\tilde{p}_{r,0} \in \mathcal{A}_0$ to the goal set $\mathcal{A}_{N'}$;
5: Let $\tilde{p}_{r,N'}$ represent the point picked by A^* in line 4 from the set $\mathcal{A}_{N'}$;
6: Store the points obtained above as the sequence $P_r = \{\tilde{p}_{r,0}, \in \mathcal{A}_0, \tilde{p}_{r,1} \in \mathcal{A}_1, \cdots, \tilde{p}_{r,N'} \in \mathcal{A}_{N'}\}$
7: Store *boundary criteria checker* terms in sequence $\Psi_r = \left\{\eta\left(\alpha(\nu^\star(\tilde{p}_{r,1}), \mathbf{N}(\tilde{p}_{r,0}, \tilde{p}_{r,1}))\right), \cdots, \eta\left(\alpha(\nu^\star(\tilde{p}_{r,N'}), \mathbf{N}(\tilde{p}_{r,(N'-1)}, \tilde{p}_{r,N'}))\right)\right\}$;
8: Use A^* to find an optimal path from $\tilde{p}_{r,N'} \in \mathcal{A}_{N'}$ to the goal set $\mathcal{A}_0$;
9: Let $\tilde{p}_{r,N}$ represent the point picked by A^* in line 8 from the set $\mathcal{A}_N$;
10: Let $\tilde{p}_{r+1,0}$ represent the point picked by A^* in line 8 from the set $\mathcal{A}_0$;
11: Update sequence P_r as $P_r = \{\tilde{p}_{r,0} \in \mathcal{A}_0, \cdots, \tilde{p}_{r,N'} \in \mathcal{A}_{N'}, \tilde{p}_{r,N'+1} \in \mathcal{A}_{N'+1}, \cdots, \tilde{p}_{r,N} \in \mathcal{A}_N\}$;
12: Update sequence Ψ_r as $\Psi_r = \left\{\cdots, \eta\left(\alpha(\nu^\star(\tilde{p}_{r,(N'+1)}), \mathbf{N}(\tilde{p}_{r,N'}, \tilde{p}_{r,(N'+1)}))\right),\ \cdots, \eta\left(\alpha(\nu^\star(\tilde{p}_{r,N}), \mathbf{N}(\tilde{p}_{r,(N-1)}, \tilde{p}_{r,N}))\right)\right\}$;
13: Compute $\sigma_r = d_2(P_r, P_{r-1})$;
14: Let $\sigma = \sigma_r, \tilde{p}_{r,0} = \tilde{p}_{r+1,0} \in \mathcal{A}_0$;
15: **if** $(\sigma > \epsilon)$ **and** $(r < r_{\max})$ **then**
16: $r = r + 1$;
17: **else if** $(\sigma > \epsilon)$ **and** $(r = r_{\max})$ **then**
18: **display**("No approximation of smallest RFIS found in $r_{\max}$ iterations.");
19: $P_r = \{\}, \Psi_r = \{\}$;
20: Go to line 31;
21: **else if** $(\sigma < \epsilon)$ **then**
22: **if** $\tilde{g} \notin \Psi_r$ **then**
23: **display**("Approximation of RFIS found.");
24: Go to line 31;
25: **else**
26: **display**("Polygonal set found in iteration r, but boundary condition violated.");
27: Go to line 31;
28: **end if**
29: **end if**
30: **end while**
31: **return** P_r, Ψ_r;

4.3.1 | Tie breaker rule

For any $i \in \{0, 1, \cdots, N\}$ it is possible that when the A^* algorithm is trying to find an optimal path from a point $p_{(i-1)k} \in \mathcal{A}_{i-1}$ to the set $\mathcal{A}_i$, there are multiple $p_{ij} \in \mathcal{A}_i$ providing a path of equal cost. In such situations a point p_{ij} with the largest index $j \in \{0, 1, \cdots, n\}$ such that the **boundary condition** is satisfied, is selected as a solution to (7). This guarantees a unique solution to (7) for all $i \in \{0, 1, \cdots, N\}$. Also based on the discretization of the search space, choosing a point p_{ij} with the largest value of j gives a point closer to $\bar{x}$, which reduces the area of an RFIS.

4.3.2 | Zero cost $\hat{h}(\cdot)$

Setting $\hat{h}(\cdot) = 0$, may appear to make using A^* algorithm unnecessary, and may seem to make using Djikstra's algorithm more appropriate. Also, even if setting $\hat{h}(\cdot) = 0$ may result in Djikstra's algorithm being used, yet using the notation as mentioned in section 4.1, allows us to mold the formulation along the lines of [13], and further allows usage of results from [13] to prove convergence of the proposed algorithm. This is a motivating factor for setting up notation related to A^* and using it in this work. Also, incorporating any experience to choose an appropriate function $\hat{h}(\cdot)$ has potential to reduce computation time significantly [13]. For example, suppose after a few runs of the proposed algorithm it is seen that the algorithm is not converging to a convex

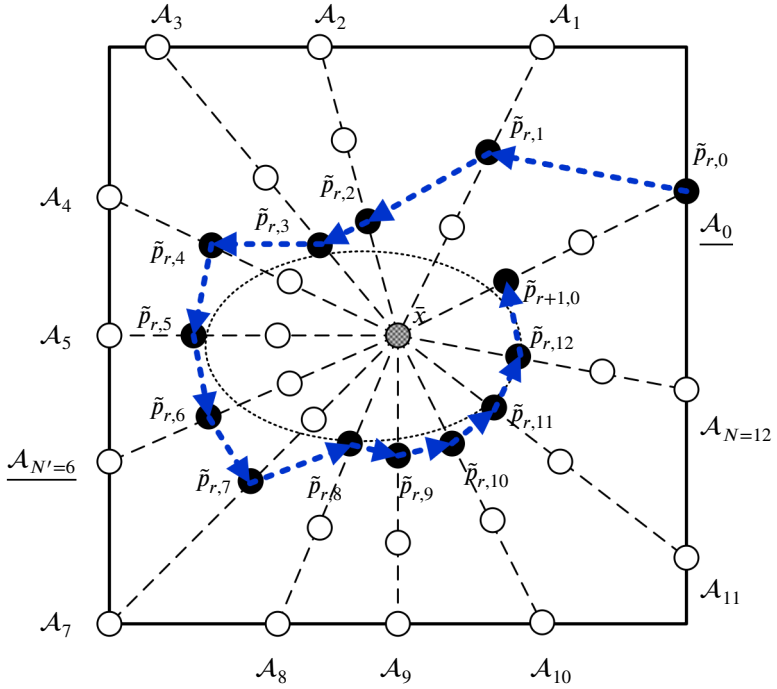

FIGURE 6 An illustration of planning the required path in two halves, first from $\mathcal{A}_0$ to $\mathcal{A}_{N'}$, and then from $\mathcal{A}_{N'}$ back to $\mathcal{A}_0$. Note that the set $\mathcal{A}_0$ is shown to not lie on a horizontal line segment because any set can be chosen to be the set $\mathcal{A}_0$ as desired.

shaped set, but to a star shaped set. Then, at least from the results, an idea of the general size of the set can be approximately formed. Based on this the $\hat{h}(\cdot)$ cost can be updated at each node to allow for possibly faster convergence of the algorithm proposed in this work.

5 | MATHEMATICAL JUSTIFICATION

The following result establishes lower bounds that $\tilde{g}$ in (9) must satisfy so that given a point $p_{(i-1),k} \in \mathcal{A}_{i-1}$, the next point $p_{ij} \in \mathcal{A}_i$ is picked such that $\alpha(v^\star(p_{ij}), \mathbf{N}(p_{(i-1,k)}, p_{ij})) \in [0,1]$, and as mentioned briefly after (6), having $\alpha(v^\star(p_{ij}), \mathbf{N}(p_{(i-1,k)}, p_{ij})) \in [0,1]$ is required to enforce the **boundary condition** along seg $\left(p_{(i-1)k}; p_{ij}\right)$. Thus Lemma 1 establishes conditions on $\tilde{g}$ which are required for enforcing the **boundary condition**. Intuitively, this result can be understood as follows. The cost functions described in section 3.5 are setup such that if any segment is chosen by the proposed algorithm to be a part of the boundary of the computed RFIS, then the least navigation cost occurs along segments (paths) located along the true boundary of the RFIS (in other words along points/paths meeting the **boundary condition**). So to prevent the proposed algorithm from picking a point, which if chosen, would violate the boundary condition, such points are simply assigned an extremely high navigation cost, so that the path planning algorithm does not pick them. In fact, it turns out that the exact value of the cost to be assigned to such points, so they are not chosen, is not hard to find. Such a cost assignment is done in our formulation by assigning a value for the constant $\tilde{g}$, details for which are available in Lemma 1 below.

Lemma 1. Let $i \in \{0, 1, \cdots, N\}$, and $j, k \in \{0, 1, \cdots, n\}$, and $\bar{x}$ be any point in the interior of the initial set $B \subset \mathbb{R}^2$. Given a point $p_{(i-1)k} \in \mathcal{A}_{i-1}$, suppose there exists at least one j such that $\alpha(v^\star(p_{ij}), \mathbf{N}(p_{(i-1)k}, p_{ij})) \in [0,1]$, where $p_{ij} \in \mathcal{A}_i$. Let p_{ij^*} be a node chosen by A^* . **If** the constant $\tilde{g} > 1 + \max_{i,j}\left(\left\| p_{ij} - \bar{x} \right\|_2\right)$, **then** $\alpha(v^\star(p_{ij^*}), \mathbf{N}(p_{(i-1)k}, p_{ij^*})) \in [0,1]$.

Proof. Given a point $p_{(i-1)k}$, $\hat{f}(p_{(i-1)k}, p_{ij})$ is the cost of traveling to a point p_{ij} as per the evaluation function in (10). In every step of execution of the A^* algorithm this term $\hat{f}(p_{(i-1)k}, p_{ij})$ is evaluated [13]. Evaluation of $\hat{f}(p_{(i-1)k}, p_{ij})$ requires evaluation of $\hat{g}(p_{(i-1)k}, p_{ij})$ in (11), $c(p_{(i-1)k}, p_{ij})$ in (8), $\eta\left(\alpha(v^\star(p_{ij}), \mathbf{N}(p_{(i-1)k}, p_{ij}))\right)$ in (9), and $\alpha(v^\star(p_{ij}), \mathbf{N}(p_{(i-1)k}, p_{ij}))$ in (6). The quantity $\alpha(\cdot)$ is the cosine of the angle between vectors $v^\star(p_{ij})$, and $\mathbf{N}(p_{(i-1)k}, p_{ij})$ and so $\alpha(\cdot) \in [-1, 1]$. If $\alpha(\cdot) \in [-1, 0)$, then by definition in (9) we have $\eta(\alpha(\cdot)) = \tilde{g}$, and if $\alpha(\cdot) \in [0, 1]$, then by definition in (9) we have $\eta(\alpha(\cdot)) \in [0, 1]$. Now let $p_{ij'} \in \mathcal{A}_i$ be a point such that $\alpha(v^\star(p_{ij'}), \mathbf{N}(p_{(i-1)k}, p_{ij'})) \in [-1, 0)$, and let $p_{ij^*} \in \mathcal{A}_i$ be a point such that $\alpha(v^\star(p_{ij^*}), \mathbf{N}(p_{(i-1)k}, p_{ij^*})) \in [0, 1]$. Then

by the above discussion, and by the definitions of $\hat{f}(\cdot,\cdot)$, and $c(\cdot,\cdot)$ we have $\hat{f}(p_{(i-1)k}, p_{ij'}) = \tilde{g} + \left\| p_{ij'} - \bar{x} \right\| + \hat{g}(p_{(i-2)l}, p_{(i-1)k})$. And $1 + \left\| p_{ij^*} - \bar{x} \right\| + \hat{g}(p_{(i-2)l}, p_{(i-1)k}) \geq \hat{f}(p_{(i-1)k}, p_{ij^*})$. So it is trivial to see that if $\tilde{g} > 1 + \max_{i,j} \left(\left\| p_{ij} - \bar{x} \right\|_2 \right)$, then the cost $\hat{f}(p_{(i-1)k}, p_{ij'})$ to go from $p_{(i-1)k} \in \mathcal{A}_{i-1}$, to $p_{ij'} \in \mathcal{A}_i$ is greater than the cost $\hat{f}(p_{(i-1)k}, p_{ij^*})$ to go from $p_{(i-1)k} \in \mathcal{A}_{i-1}$, to $p_{ij^*} \in \mathcal{A}_i$. But the A^* algorithm always picks the optimal path, and by the assumptions of this lemma, a point p_{ij^*} is available such that $\alpha(\nu^\star(p_{ij^*}), \mathbf{N}(p_{(i-1)k}, p_{ij^*})) \in [0,1]$. Thus, A^* will pick point p_{ij^*} so that $\alpha(\nu^\star(p_{ij^*}), \mathbf{N}(p_{(i-1)k}, p_{ij^*})) \in [0,1]$. This completes the proof. □

The next result shows that Algorithm 1 terminates in a finite number of iterations, and as the discretization of the search space becomes finer, the output of Algorithm 1 tends to an RFIS. The idea on which the below result is based is as follows. Suppose a set is picked in which we wish to seek an RFIS. If there is an RFIS in this set, then the proposed algorithm can converge to it, only if the discretization of the search space is such that all points belonging to the boundary of this RFIS belong to the sets $\mathcal{A}_i$ in graph G'. So assuming that the discretization is so, the algorithm proposed in this work searches for the RFIS. Because as mentioned before Lemma 1, the formulation is such that the least navigation cost occurs along the boundary of the RFIS. So the proposed path planning algorithm at some instant (i.e. on some segment set $\mathcal{A}_i$) will thus pick a point from among the points in graph G' that belongs to the boundary of the RFIS. Following this, to stay on the least cost path, every point further picked for navigation will also be on this least cost path, i.e. the boundary of the required RFIS. If this keeps happening, and because the path being planned is a closed path, so at some instant, the proposed algorithm will simply be picking points it has already picked. That is, every point picked on every segment set A_i in a current iteration will be the same as the point picked in the last iteration on every set A_i. If a sequence of these points is maintained every iteration, and the points stored in a sequence in a current iteration are compared with the points stored in the previous iteration. Then at some instant, the sequences will be identical. And when this happens, the proposed algorithm will terminate having found the RFIS formed by segments joining these consecutive points stored in such a sequence. However, for this to happen, the system under consideration must satisfy some assumptions, i.e. the system must be generally well behaved in every neighborhood under consideration so that the system behavior at one end of a particular discrete segment is not radically different from behavior at the other end. These conditions are simply the Lipschitz conditions as mentioned below.

Theorem 1. Let $i \in \{0, 1, \cdots, N\}$, and $j, k \in \{0, 1, \cdots, n\}$, and $B \subseteq \mathcal{D} \subset \mathbb{R}^2$ be a convex set. Suppose a graph $G' \in B$ with points $p_{ij} \in \mathcal{A}_i$ is given, $\bar{x}$ is a point in the interior of B, $\tilde{g} > 1 + \max_{i,j} \left\| p_{ij} - \bar{x} \right\|$, and for the system in (1), $f(\cdot)$ is L-Lipschitz locally in $x \in \mathcal{D} \subset \mathbb{R}^2$ with $L > 0$, $\delta(\cdot)$ is γ-Lipschitz globally in $t \in \mathbb{R}$ with $\gamma > 0$. **If** there exists at least one RFIS whose boundary is a simple closed curve, which is formed by joining consecutive segments $\text{seg}\left(p_{(i-1)k}; p_{ij}\right)$ where points $p_{ij} \in \mathcal{A}_i$, and $p_{(i-1)k} \in \mathcal{A}_{i-1}$, **then** Algorithm 1 converges to a sequence P_r in a finite number of iterations. Further, let $\mathscr{R}$ be the set whose boundary is formed by joining the segments $\text{seg}\left(\tilde{p}_{r,0}; \tilde{p}_{r,1}\right)$, $\text{seg}\left(\tilde{p}_{r,1}; \tilde{p}_{r,2}\right)$, $\cdots$, $\text{seg}\left(\tilde{p}_{r,N-1}; \tilde{p}_{r,N}\right)$, $\text{seg}\left(\tilde{p}_{r,N}; \tilde{p}_{r,0}\right)$, where $\tilde{p}_{r,i} \in P_r$. **If** n, N are arbitrarily large, and γ is arbitrarily small, **then** $\mathscr{R}$ is an RFIS. Furthermore, $\mathscr{R}$ has the smallest area among all possible star-shaped RFISs that can be formed by joining consecutive segments $\text{seg}\left(p_{(i-1)k}; p_{ij}\right)$.

Proof. Algorithm 1 uses A^* internally to pick points $p_{ij} \in \mathcal{A}_i$ so that $\hat{f}(\cdot)$ in (10) is minimized. From (7), (10) and (11) it is seen that minimizing (10) minimizes (7). Also, it is given that there exists at least one RFIS whose boundary is a simple closed curve, which is formed by joining consecutive segments $\text{seg}\left(p_{(i-1)k}; p_{ij}\right)$ where points $p_{ij} \in \mathcal{A}_i$, and $p_{(i-1)k} \in \mathcal{A}_{i-1}$. This coupled with the discussion of the **boundary condition** in section 3.5 implies that for a point $p_{(i-1)k} \in \mathcal{A}_{i-1}$ which belongs to an RFIS, there exists at least one point $p_{ij} \in \mathcal{A}_i$ such that $\alpha(\nu^\star(p_{ij}), \mathbf{N}(p_{(i-1)k}, p_{ij})) \in [0,1]$. Now there exist only a finite number of nodes in graph G' which are to be searched by the A^* algorithm, and it is known that the A^* algorithm converges in finite iterations to its optimal path [13]. So this implies that in a certain iteration r of Algorithm 1, A^* picks a point $p_{(i-1)k} \in \mathcal{A}_{i-1}$ such that $p_{(i-1)k}$ is a vertex of the above mentioned RFIS. Now by assumption we have $\tilde{g} > 1 + \max_{i,j} \left\| p_{ij} - \bar{x} \right\|$ and as discussed above for a given point $p_{(i-1),k} \in \mathcal{A}_{i-1}$ which belongs to an RFIS, there exists at least one point $p_{ij} \in \mathcal{A}_i$ such that $\alpha(\nu^\star(p_{ij}), \mathbf{N}(p_{(i-1),k}, p_{ij})) \in [0,1]$. Now invoking Lemma 1 we know that A^* picks such a point $p_{ij} \in \mathcal{A}_i$, and repeating the above argument we get that every subsequent point $p_{(i+1)j} \in \mathcal{A}_{i+1}$ and so on, picked by A^* satisfies the **boundary condition** i.e. $\alpha(\cdot) \in [0,1]$. So by the definition of $\eta(\cdot)$ in (9) and the definition of sequence Ψ_r in Algorithm 1 we see that no subsequent entries in Ψ_r will contain $\tilde{g}$. This shows why we use line 22 in Algorithm 1 to verify that the result satisfies the **boundary condition**, required for an RFIS.

From the pseudocode of Algorithm 1, all points $p_{ij} \in \mathcal{A}_i$ picked by A^* are stored in a sequence P_r. Further, because A^* picks optimal points, and considering the tie-breaker rule mentioned in subsection 4.3.1 allows us to establish that for each

$i \in \{0, 1, \cdots, N\}$ there is a unique point $\tilde{p}_{r,i} \in P_r$ which minimizes (7). Let the simple closed curve formed by joining such points $\tilde{p}_{r,i}$ be $\partial\mathscr{R}$. Note that a curve formed by joining consecutive points $\tilde{p}_{r,i} \in P_r$ is a simple closed curve by virtue of the construction of the embedded graph structure described in Section 3.4. Now, using arguments exactly similar to the ones in the first half of the previous paragraph we know there exists some finite iteration number r in which $\tilde{p}_{r,i}$ will lie on $\partial\mathscr{R}$. And now again, because there is only one optimal simple closed curve $\partial\mathscr{R}$, and because A^* picks optimal points, and because $\tilde{p}_{r,i} \in \partial\mathscr{R}$, implies that every subsequent point chosen by A^* and stored in P_r will now belong to $\partial\mathscr{R}$. Therefore, it is trivial to see that after a certain number of iterations r, points stored in P_{r+1} will be identical to the points in P_r. At the end of this iteration $d_2(P_r, P_{r+1}) = 0$, and Algorithm 1 will terminate. This completes the first result required.

Now consider points $\tilde{p}_{r,i-1}, \tilde{p}_{r,i}$ from P_r, where seg $\left(\tilde{p}_{r,i-1}; \tilde{p}_{r,i}\right)$ is an edge forming the boundary $\partial\mathscr{R}$ of the set $\mathscr{R}$. By construction, and from Algorithm 1 we know that these points $\tilde{p}_{r,i-1}, \tilde{p}_{r,i}$ are some points $p_{(i-1)k} \in \mathcal{A}_{i-1}$, $p_{ij} \in \mathcal{A}_i$ respectively. So seg $\left(\tilde{p}_{r,i-1}; \tilde{p}_{r,i}\right)$ can be represented by a seg $\left(p_{(i-1)k}; p_{i,j}\right)$. Also by Lemma 1 we know that given a point $p_{(i-1)k} \in \mathcal{A}_{i-1}$, the A^* algorithm picks a point $p_{ij} \in \mathcal{A}_i$ such that $\alpha(\nu^\star(p_{ij}), \mathbf{N}(p_{(i-1),k}, p_{ij})) \in [0, 1]$. So from the definitions of $\alpha(\cdot,\cdot)$ in (6), $\nu^\star(\cdot)$ in (5) we know $0 \leq \langle \mathbf{N}(p_{(i-1)k}, p_{ij}), \nu(p_{ij})\rangle$ for all vectors $\nu(p_{ij}) \in \mathbf{F}(p_{ij})$. Now by definition, such a vector $\nu(p_{ij}) = f(p_{ij}) + \delta(\cdot)$. For use in Algorithm 1, the set $\mathbf{F}(\cdot)$ contains such vectors $\nu(\cdot)$ evaluated at discrete values that an unknown disturbance function $\delta(\cdot)$ can take. However, in reality, such disturbances would be parametrized by time as per (1). So consider $\nu(p_{ij}) = f(p_{ij}) + \delta(t_1)$, and $\nu(y) = f(y) + \delta(t_2)$, where t_1, t_2 are some time instants, and points $p_{ij}, y \in$ seg $\left(p_{(i-1),k}; p_{i,j}\right)$. Then from above we get $0 \leq \langle \mathbf{N}(p_{(i-1)k}, p_{ij}), \nu(p_{ij}) + \nu(y) - \nu(y)\rangle$. Re-arranging, and using properties of inner-products gives $\langle \mathbf{N}(p_{(i-1)k}, p_{ij}), \nu(y) - \nu(p_{ij})\rangle \leq \langle \mathbf{N}(p_{(i-1)k}, p_{ij}), \nu(y)\rangle$. Now using the Cauchy-Schwarz inequality, and then substituting $\nu(y), \nu(p_{ij})$ in the above, and simplifying using assumptions gives $\left|\langle \mathbf{N}(p_{(i-1)k}, p_{ij}), \nu(y) - \nu(p_{ij})\rangle\right| \leq |\mathbf{N}(\cdot)| \left(\left|f(y) - f(p_{ij})\right| + \left|\delta(t_2) - \delta(t_1)\right|\right) \leq \mathbf{N}(\cdot) \left(L \left|y - p_{ij}\right| + \gamma \left|t_2 - t_1\right|\right)$. Now because n, N are arbitrarily large, enabling a fine discretization of space, so it is possible to pick points $p_{(i-1)k}, p_{ij}$ such that the length of segments $\in$ seg $\left(p_{(i-1),k}; p_{i,j}\right)$ are arbitrarily small, so it is possible to have $\left|y - p_{ij}\right| < \dfrac{\epsilon}{2L}$, and similarly as γ is arbitrarily small, it is possible to have $\gamma \left|t_2 - t_1\right| < \frac{\epsilon}{2}$ for any given t_2, t_1. This provides $\left|\langle \mathbf{N}(p_{(i-1)k}, p_{ij}), \nu(y) - \nu(p_{ij})\rangle\right| < \epsilon$, i.e. from above $-\epsilon < \langle \mathbf{N}(p_{(i-1)k}, p_{ij}), \nu(y) - \nu(p_{ij})\rangle \leq \langle \mathbf{N}(p_{(i-1)k}, p_{ij}), \nu(y)\rangle$. So we see that as n, N become larger, and γ becomes smaller, then $\epsilon \to 0$, i.e. for any point $y \in$ seg $\left(p_{(i-1),k}; p_{i,j}\right)$, the **boundary condition** is satisfied. Also this implies that at vertices p_{ij} the vectors $\nu(\cdot) \in \mathbf{F}(\cdot)$ satisfy boundary conditions along two segments which have a common vertex at p_{ij}. This shows that any point on the boundary $\partial\mathscr{R}$ satisfies the **boundary condition**, for disturbance $\delta(\cdot)$ satisfying given assumptions, and thus the set $\mathscr{R}$ is an RFIS. Further this set is formed by minimizing (7), the second term of which minimizes the distance of each point $p_{ij} \in \mathcal{A}_i$ from an interior point $\bar{x}$. This results in decreasing the area of triangles $p_{(i-1)k}, p_{ij}, \bar{x}$. Because all points $p_{ij} \in \mathcal{A}_i$ are on the optimal path, no further decrease in area of individual triangles $p_{(i-1)k}, p_{ij}, \bar{x}$ is possible. So summing the areas of all such triangles which lie in the interior of $\mathscr{R}$ by construction, we have the RFIS $\mathscr{R}$ with least possible area. This completes the proof.

□

The above result shows that Algorithm 1 outputs a set which is an RFIS of minimum area. By construction the set is star-shaped, so this also satisfies the definition of an approximation of the smallest RFIS. Further, the initial guess B is not needed to be invariant, although it is required to contain an RFIS. Also the sequence Ψ_r acts as an indicator sequence, because it indicates to the user if the boundary condition is violated, by inserting $\tilde{g} \in \Psi_r$. This is useful if it is not known whether an initial guess B contains any RFISs. It is worth mentioning that despite the conditions on the Lipschitz constant γ for the disturbance function $\delta(\cdot)$ in the above result, simulation results in Section 6 show the versatility of Algorithm 1 developed in this paper, when used with different disturbance functions. Relaxing the assumptions for the above result, and extending it to higher dimensions is left for future efforts.

6 | SIMULATIONS

Consider the curve tracking problem [22] as given below.

$$\dot{\rho} = -\sin(\phi) \tag{12}$$

$$\dot{\phi} = (\rho - \rho_0)\cos(\phi) - \mu \sin(\phi) + \delta(t). \tag{13}$$

The values of certain constants used for testing this system in (12)-(13) with the proposed RFIS computation Algorithm 1 are $\delta_0 = 0.15$, $\rho_0 = 1$, $\mu = 6.42$ (this applies to all results mentioned here in this section). The following subsections provide an

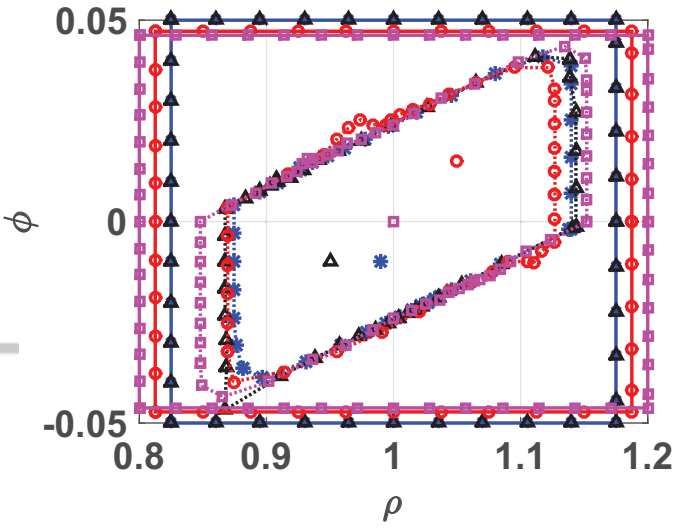

FIGURE 7 Test 1: Results of choosing different initial guesses $\boldsymbol{B}$ along with corresponding choices for $\bar{x}$.

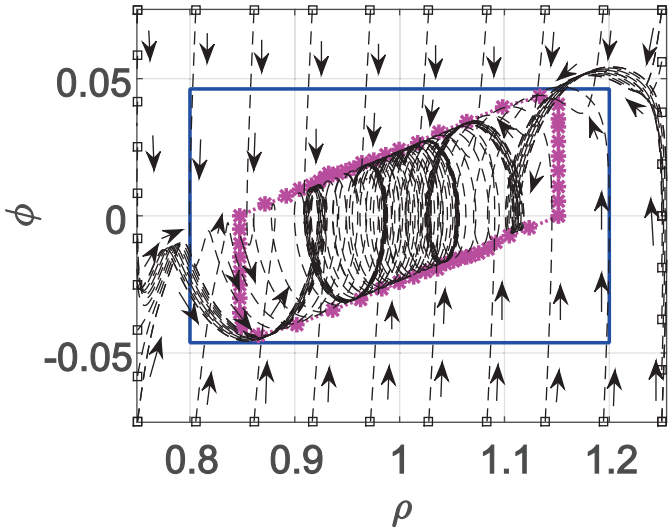

FIGURE 8 Test 2: Results using a disturbance function of the form $\delta(t) = \delta_0 \sin(t)$.

analysis of the results obtained on performing various tests with the proposed algorithm. Seven tests are considered as follows. Test 1 considers the effects of changing initial guesses i.e. set $\boldsymbol{B}$, and changing locations of the point $\bar{x}$, on the output of the proposed algorithm. Tests 2 to 4 consider the effects of different bounded disturbance functions $\delta(t)$ on the output of the proposed RFIS comutation algorithm, when used with the problem in (12)-(13). Tests 5 and 6 plot trajectories (solutions) of the problem in (12)-(13) for ten random intial conditions in the area of interest. The time required for calculating these is noted down. These results are provided as a means to compare the algorithm in this paper with one which would simply compute simulated system trajectories and hope to construct an estimate of the smallest RFIS from such simulation. This is done because a full fledged comparison of the proposed approach with other approaches existing in the literature is completely out of the scope of the current manuscript, but at least such a test shows the advantages of the algorithm proposed in this work when compared to a naive unguided search. Test 7 simply shows that the algorithm proposed produces correct and expected shapes for the RFISs, this is verified by testing the proposed algorithm with a simple system that has nested circular trajectories.

6.1 | Test 1

Figure 7 shows results output from Algorithm 1 for different (rectangular) initial sets $\boldsymbol{B}$, and corresponding different $\bar{x}$, when tested for finding RFISs for the system given in (12)-(13). The outer square sets with solid lines are the different initial guesses $\boldsymbol{B}$. The inner rhomboidal sets with dotted lines are the outputs. To read Fig. 7, readers must look for points having the same color. For Example, the solid purple rectangular set with square markers is an initial guess $\boldsymbol{B}$ for a particular test run. Corresponding to this, the single purple point (square marker) at (1,0) is $\bar{x}$ for this run, and the dotted purple rhombus shaped set with square markers is the output. Similarly, the solid black rectangular set with triangular markers is an initial guess $\boldsymbol{B}$ for the next test run. Corresponding to this, the single black point (triangular marker) at approximately (0.95,-0.01) is $\bar{x}$ for this run, and the dotted black rhombus shaped set with triangular markers is the output. This output set almost overlaps with a few other output sets in Fig. 7. This is not bad, because searching around slightly different points i.e. $\bar{x} = (1,0)$ or (0.95,-0.01), should not provide widely different results for an RFIS (if there is an unique smallest RFIS). A similar discussion as above applies for the sets and points shown in red and blue in Fig. 7, so the discussion has not been repeated. It is worth noting that the initial guess set marked with a solid blue line and blue asterisks as markers is the same as the initial guess set shown with solid black lines and triangular markers. Thus as seen from Fig. 7 changing the sizes of the initial guesses, or the discretization, or the location of $\bar{x}$ has relatively less effect on the output. Although $\bar{x}$ located centrally within $\boldsymbol{B}$ appears to provide smoother boundaries of sets. For the first try (purple set with square markers), the following values were used for Algorithm 1, $n = 51, N = 59, s = 630, r_{\max} = 10, \epsilon = 0.1, \tilde{g} = 4.5035 \times 10^{15}$. All runs of Algorithm 1 which produced Fig. 7 ended in 4 iterations, and took about 3 to 5.6 minutes to terminate and output results. Note that all results are obtained using MATLAB on a Microsoft Surface Pro 4 dual-core tablet

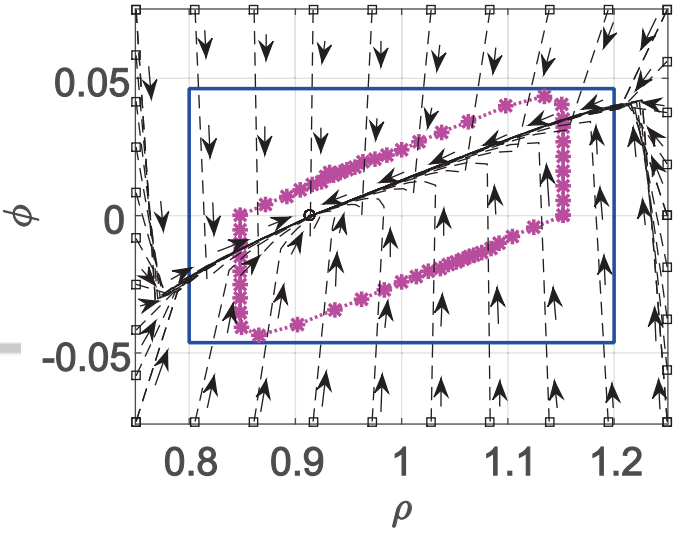

FIGURE 9 Test 3: Results using a disturbance function of the form $\delta(t) = \frac{2 \times 0.087}{\pi} \tan^{-1}(t)$.

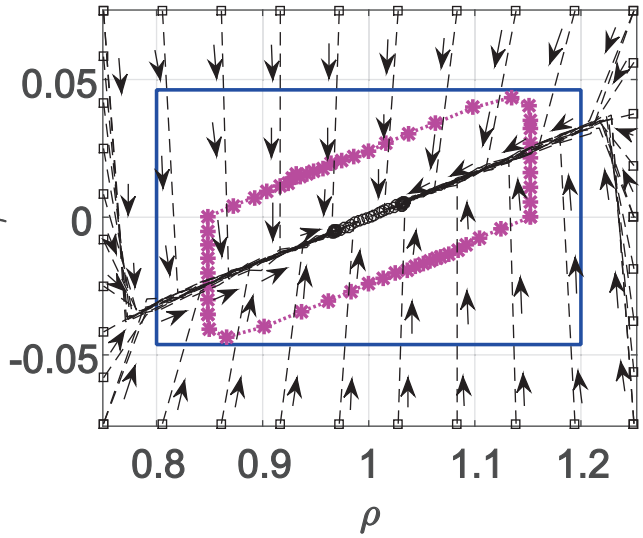

FIGURE 10 Test 4: Results using a disturbance function of the form $\delta(t) = \mathrm{U}(-\delta_0, \delta_0)$.

PC with 4GB RAM, 2.5GHz CPU. The times reported above include time for setting up storage variables, and post-termination plotting.

6.2 | Tests 2 to 4

Figures 8 to 10 show verification of the proposed RFIS computation algorithm when tested using the problem in equations (12)-(13) with different disturbance functions as indicated below. It is worth noting that the purple set shown in figures 8-10 is the same purple output set shown in Fig. 7. So the amount of computation required for this is similar to that mentioned above is around 3 to 5.6 minutes. Figure 8 shows test results when the proposed RFIS computation algorithm's result is tested using the problem in equations (12)-(13) along with the disturbance function $\delta(t) = \delta_0 \sin(t)$. Figure 9 shows test results when the proposed RFIS computation algorithm's result is tested using the problem in equations (12)-(13) along with the disturbance function $\delta(t) = \frac{2 \times 0.087}{\pi} \tan^{-1}(t)$. Figure 10 shows test results when the proposed RFIS computation algorithm's result is tested using problem in equations (12)-(13) along with the disturbance function $\delta(t) = \mathrm{U}(-\delta_0, \delta_0)$. The notation $\mathrm{U}(-\delta_0, \delta_0)$ is used to represent a uniformly distributed random value in $[-\delta_0, \delta_0]$. For each of these tests, and in each of the figures 8-10, the initial guess set $\boldsymbol{B}$ is shown as a solid blue rectangle, and the pink/purple starred set is the computed RFIS output by the proposed Algorithm 1. The arrows in figures 8-10 show the vector field directions, and solutions are shown in black dashed lines starting from arbitrarily chosen initial conditions shown by unfilled black square markers. From figures 8-10 we see that the set produced i.e the pink/purple starred set output by the proposed RFIS computation algorithm is invariant, because the solutions (dashed black lines) do not appear to leave the pink/purple starred set once they enter it. This is expected, and verifies that the proposed RFIS computation algorithm works because the algorithm is used only once with the problem in equations (12)-(13), it does not have any knowledge of the functional form of the disturbance that affects the system in (12)-(13). And when the computed results is tested with different different disturbance functions as mentioned above, and as shown in figures 8-10, the trajectories computed do not leave the pink/purple set with starred markers. This result shows that the proposed algorithm works and it is not dependent on the form of the additive disturbance used, as long as the magnitude of the disturbance is known.

6.3 | Test 5 and 6

While a detailed comparison of the performance of this proposed algorithm with other results from the literature is beyond the scope of the current work, the results presented in figures 11, and 12 provide an insight into the performance gained by using a path planning based approach for RFIS computation as opposed to using simple trajectory ensemble based RFIS computation. Computing the trajectories shown in Fig. 11 with ten random initial conditions took around 6.507 seconds, when the problem

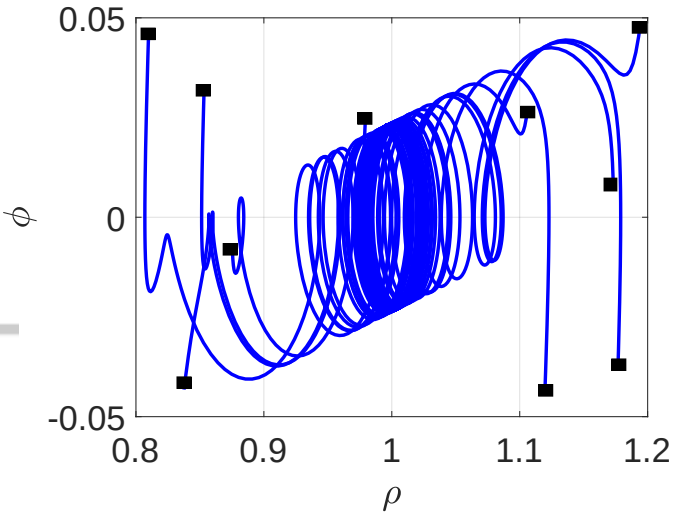

FIGURE 11 Test 5: A plot of ten system solution trajectories using a disturbance function of the form $\delta(t) = \delta_0 \sin(t)$. Black square markers show initial conditions.

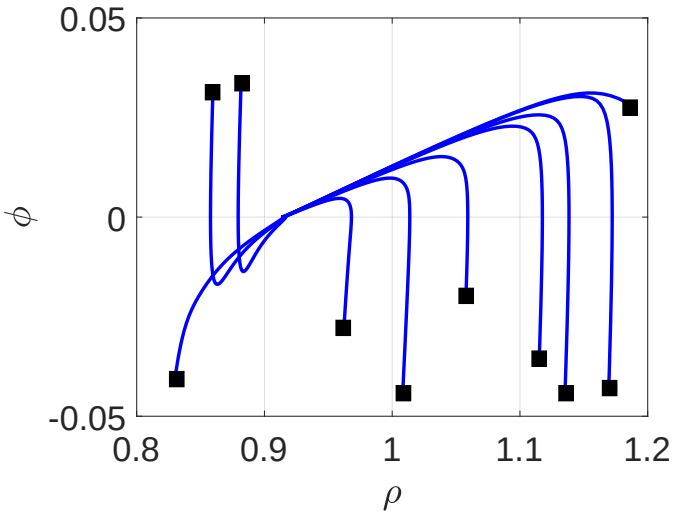

FIGURE 12 Test 6: A plot of ten system solution trajectories using a disturbance function of the form $\delta(t) = \frac{2 \times 0.087}{\pi} \tan^{-1}(t)$. Black square markers show initial conditions.

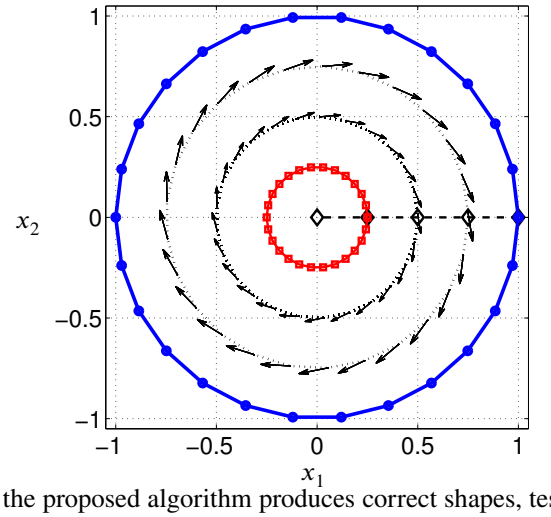

FIGURE 13 Test 7: Verifying that the proposed algorithm produces correct shapes, testing on a simple problem with circular trajectories.

in (12)-(13) is used with a disturbance of the form $\delta(t) = \delta_0 \sin(t)$. Similarly, computing the trajectories shown in Fig. 12 with ten random initial conditions took around 0.3406 seconds, when the problem in (12)-(13) is used with a disturbance of the form $\delta(t) = \frac{2\times0.087}{\pi} \tan^{-1}(t)$. From Fig. 11, at least an idea of the smallest RFISs shaped and size can be achieved, but even doing this with sufficient fidelity would require computing trajectories starting from several initial conditions. On the other hand while computing the trajectories as shown in Fig. 12 takes lesser time, but these trajectories do not provide any idea about the shape or size of the smallest RFIS. This shows that trajectory generation and subsequent RFIS computation is a time consuming process. This is because, several (ideally infinite) trajectories need to be considered. Then a set has to be found such that the union of all such trajectories, with all allowed functional forms of disturbances considered added to the system dynamics, stays within this set for all time. For testing only 20 initial conditions and resulting trajectories across two types of disturbances already consumes around 6.9 seconds. If infinitely many trajectories (or at least a large number say ten thousand trajectories are considered), a linear estimate for computation to generate trajectories just for two cases of disturbances is 57.5 minutes. This does not even include the time required for an algorithm which has to combine all of this information to produce an estimate for an RFIS.

This analysis quickly shows that the proposed approach, even if it takes a few minutes to complete computation, is many times efficient compared to a pure simulation based trajectory generation approach for computing RFISs.

6.4 | Test 7

The results in Fig. 13 show the result of using the proposed algorithm to compute an RFIS for a system with the following dynamics.

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} x(t) \tag{14}$$

As is obvious from the system dynamics, it is a simple linear system with complex eigenvalues, as a result of which the trajectories are circular. The equilibrium point is (0,0), and no disturbance is used for this system. Details about the discretization used and other details about settings used for the algorithm proposed, are available as reported in [21]. Thus it is obvious from the results seen in Fig. 13, that the proposed algorithm has the ability to produce sets of the correct shape. It is worth noting that the smallest RFIS for the problem in (14) as tested in the absence of disturbances is the equilibrium point itself, however due to discretization of the search space, the result achieved is the smallest set closest to the equilibrium point, that can be reached by the proposed algorithm.

7 | CONCLUSION AND FUTURE WORK

In summary, this paper presented an algorithm for finding the boundary of the smallest robust forward invariant sets (RFISs) for two dimensional systems subject to bounded additive disturbances. It has been shown that the proposed algorithm terminates in a finite number of iterations, and outputs an RFIS of minimal area, with precision only constrained by the discretization of the search space. We have demonstrated that path planning algorithms can be introduced as potential numerical algorithms to compute an RFIS. The contribution of this paper is that for the proposed algorithm, no specific form of an additive disturbance function is required to be known to compute the smallest RFIS of a system. Only discrete values/measurements of the disturbance function are required. This is also shown in the simulation results where the RFIS computation algorithm's results are verified in the presence of different bounded functions used as the additive disturbance functions affecting the system dynamics. Further, conditions on the system dynamics, and on the disturbance function and grid discretization are provided for letting the proposed algorithm converge to an RFIS.

7.1 | Future work

The benefit of using such path planning based algorithms is computational efficiency associated with sampling of the search space, required for computationally estimating RFISs. Future extensions to higher dimensions is underway [33]. Future efforts may consider rapidly exploring random tree (RRT) based approaches for estimating the boundary of an RFIS. This is envisioned keeping in mind that RRTs can operate in any dimensional space. As an example for a three dimensional problem which may possibly be solved without using higher dimensional approaches like RRTs, the space can be divided into two dimensional planes, with the path planning approach applied on each plane. Merging the results of each such planar search, is one possible approach for estimating the size of an RFIS in three dimensions.

ACKNOWLEDGMENTS

The authors acknowledge comments from Dr. Michael Malisoff, Roy P. Daniels Professor #3, Department of Mathematics, 303 Lockett Hall Louisiana State University, Baton Rouge, LA 70803-4918 USA.

Financial disclosure

The research work is supported by ONR grants N00014-19-1-2556 and N00014-19-1-2266; NSF grants OCE-1559475, CNS-1828678, and S&AS-1849228; NRL grants N00173-17-1-G001 and N00173-19-P-1412; and NOAA grant NA16NOS0120028.

Conflict of interest

None.

References

[1] Alamo, T., M. Fiacchini, A. Cepeda, D. Limon, J. Bravo, and E. Camacho, 2007: On the computation of robust control invariant sets for piecewise affine systems. *Assessment and Future Directions of Nonlinear Model Predictive Control*, Springer Berlin Heidelberg, volume 358, 131–139.

[2] Aubin, J. P., 2009: *Viability Theory*. Birkhäuser Boston.

[3] Baier, R., M. Dellnitz, M. H.-V. Molo, S. Sertl, and I. G. Kevrekidis, 2014: The computation of convex invariant sets via Newton's method. *Journal of Computational Dynamics*, **1**, no. 1, 39–69.

[4] Baier, R. and M.-V. Molo, 2012: Neton's method and secant method for set-valued mappings. *Large-Scale Scientific Computing*, Springer Berlin Heidelberg, volume 7116, 91–98.

[5] Barry, A. J., A. Majumdar, and R. Tedrake, 2012: Safety verification of reactive controllers for UAV flight in cluttered environments using barrier certificates. *2012 IEEE International Conference on Robotics and Automation*, 484–490.

[6] Bitsoris, G., 1988: On the positive invariance of polyhedral sets for discrete-time systems. *Systems & Control Letters*, **11**, no. 3, 243 – 248.

[7] Bitsoris, G., 1988: Positively invariant polyhedral sets of discrete-time linear systems. *International Journal of Control*, **47**, no. 6, 1713–1726.

[8] Blanchini, F., 1999: Set invariance in control. *Automatica*, **35**, no. 11, 1747 – 1767.

[9] Canale, M., L. Fagiano, and M. Signorile, 2013: Design of robust predictive control laws using set membership identified models. *Asian Journal of Control*, **15**, no. 6, 1714–1722.

[10] Chakrabarty, A., A. Raghunathan, S. Di Cairano, and C. Danielson, 2018: Data-driven estimation of backward reachable and invariant sets for unmodeled systems via active learning. *2018 IEEE Conference on Decision and Control (CDC)*, 372–377.

[11] Chesi, G., 2013: Rational Lyapunov functions for estimating and controlling the robust domain of attraction. *Automatica*, **49**, no. 4, 1051 – 1057.

[12] Chesi, G. and Y. S. Hung, 2008: Analysis and synthesis of nonlinear systems with uncertain initial conditions. *IEEE Transactions on Automatic Control*, **53**, no. 5, 1262–1267.

[13] Hart, P. E., N. J. Nilsson, and B. Raphael, 1968: A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, **4**, no. 2, 100–107.

[14] Hua, C., G. Liu, Y. Li, and X. Guan, 2019: Adaptive neural tracking control for interconnected switched systems with non-ISS unmodeled dynamics. *IEEE Transactions on Cybernetics*, **49**, no. 5, 1669–1679.

[15] Khalil, H. K., 2002: *Nonlinear Systems*. 3nd ed., Prentice Hall.

[16] Kolmanovsky, I. and E. G. Gilbert, 1998: Theory and computation of disturbance invariant sets for discrete-time linear systems. *Mathematical Problems in Engineering*, **4**, no. 4, 317–367.

[17] Majumdar, A. and R. Tedrake, 2017: Funnel libraries for real-time robust feedback motion planning. *The International Journal of Robotics Research*, **36**, no. 8, 947–982.

[18] Malisoff, M., F. Mazenc, and F. Zhang, 2011: Input-to-State Stability for Curve Tracking Control: A Constructive Approach. *Proceedings of the American Control Conference*, Baltimore, MD, 1984–1989.

[19] Malisoff, M., F. Mazenc, and F. Zhang, 2011: Stability and Robustness Analysis for Curve Tracking Control using Input-to-State Stability. *IEEE Transactions on Automatic Control*, **57**, no. 5, 1320–1326.

[20] Mohamed, M., X. Yan, Z. Mao, and B. Jiang, 2019: Adaptive sliding mode observer for nonlinear interconnected systems with time varying parameters. *Asian Journal of Control*, **21**, no. 1, 405–414.

[21] Mukhopadhyay, S., 2014: *Robust Forward Invariant Sets for Nonlinear Systems*. Ph.d. Dissertation, Georgia Institute of Technology.

[22] Mukhopadhyay, S., C. Wang, S. Bradshaw, V. Bazie, S. Maxon, L. Hicks, M. Patterson, and F. Zhang, 2012: Controller performance of marine robots in reminiscent oil surveys. *Proceedinggs of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1766–1771.

[23] Mukhopadhyay, S. and F. Zhang, 2014: A path planning approach to compute the smallest robust forward invariant sets. *Proceedings of the American Control Conference (ACC), 2014*, 1845–1850.

[24] Nagumo, M., 1942: Über die Lage der Integralkurven gewöhnlicher Differentialgleichungen. *Proceedings of the Physico-Mathematical Society of Japan. 3rd Series*, **24**, 551–559.

[25] Qiu, Z., S. Hu, and X. Liang, 2019: Model predictive control for constrained image-based visual servoing in uncalibrated environments. *Asian Journal of Control*, **21**, no. 2, 783–799.

[26] Raković, S. V. and M. Barić, 2010: Parameterized robust control invariant sets for linear systems: Theoretical advances and computational remarks. *IEEE Transactions on Automatic control*, **55**, no. 7, 1599–1614.

[27] Rubin, D., H. Nguyen, and P. Gutman, 2018: Computation of polyhedral positive invariant sets via linear matrix inequalities. *2018 European Control Conference (ECC)*, 2941–2946.

[28] Rubin, D. Y., H. Nguyen, and P. Gutman, 2018: Yet another algorithm for the computation of polyhedral positive invariant sets. *2018 IEEE Conference on Control Technology and Applications (CCTA)*, 698–703.

[29] Saint-Pierre, P., 1994: Approximation of the viability kernel. *Applied Mathematics & Optimization*, **29**, no. 2, 187–209.

[30] Shamaghdari, S. and M. Haeri, 2020: Model predictive control of nonlinear discrete time systems with guaranteed stability. *Asian Journal of Control*, **22**, no. 2, 657–666.

[31] Shi, D. and Z. Mao, 2019: Multi-step control set-based nonlinear model predictive control with persistent disturbances. *Asian Journal of Control*, **21**, no. 2, 868–878.

[32] Shokrollahi, A. and S. Shamaghdari, 2020: Offline Robust Model Predictive Control for Lipschitz Non-Linear Systems Using Polyhedral Invariant Sets. *Asian Journal of Control*, **22**, no. 1, 288–296.

[33] Varnell, P., S. Mukhopadhyay, and F. Zhang, 2016: Discretized boundary methods for computing smallest forward invariant sets. *2016 IEEE 55th Conference on Decision and Control (CDC)*, IEEE, 6518–6524.

[34] Varnell, P. and F. Zhang, 2018: Computing largest tolerable disturbance sets. *2018 Annual American Control Conference (ACC)*, 6114–6119.

[35] Wang, T., S. Lall, and M. West, 2013: Polynomial level-set method for polynomial system reachable set estimation. *IEEE Transactions on Automatic Control*, **58**, no. 10, 2508–2521.

[36] Zhang, F. and N. E. Leonard, 2007: Coordinated patterns of unit speed particles on a closed curve. *Systems & Control Letters*, **56**, no. 6, 397–407.

[37] Zuo, Z., Y. Fu, and Y. Wang, 2012: Brief paper: results on reachable set estimation for linear systems with both discrete and distributed delays. *IET Control Theory Applications*, **6**, no. 14, 2346–2350.