Robustly Complete Synthesis of Memoryless Controllers for Nonlinear Systems With Reach-and-Stay Specifications

Yinan Li, *Member, IEEE*, and Jun Liu, *Senior Member, IEEE*

***Abstract*—This article proposes a finitely terminating algorithm to solve reach-and-stay control problems for nonlinear systems. The algorithm is guaranteed to return a control strategy if the specification is robustly realizable. Such a feature is desirable as the commonly used abstraction-based methods are sound but not complete for systems that are not incrementally stable. Fundamental to the proposed method is a fixed-point characterization of the winning set of the system with respect to a given specification, i.e., the initial states that can be controlled to satisfy the specification. The use of an adaptive partitioning scheme not only guarantees the approximation precision of the winning set but also reduces computational time. The effectiveness and efficiency are illustrated by several benchmarking examples.**

***Index Terms*—Formal verification and synthesis, interval analysis, nonlinear control design, reachability analysis, robustness.**

I. Introduction

Reach-and-stay control synthesis for nonlinear systems is concerned with finding control strategies that can steer the state of the system to a target set and maintain it in the target set afterward. Such a problem exists in a variety of control applications, such as voltage regulation of electrical power converters [1], attitude control and flight path following in flight control systems [2], and regulation of room temperatures inside a building [3].

Seeking provably correct reach-and-stay control strategies at the presence of constraints and nonlinearity is challenging. Many nonlinear control methods, e.g., feedback linearization and Lyapunov-based control, are incapable to deal with constraints in system states or inputs. Integrating constraints and bounded perturbations in the stage of controller design was studied in [4] for linear systems and extended to nonlinear systems [5] where the reach-and-stay requirement is relaxed to reach a robustly controllable super set containing the target set when the target set is not controlled invariant. The set-theoretic approach used in [4] and [5] is similar to earlier theoretical framework proposed in [6] and [7] for solving invariance and reachability control problems with linear or nonlinear dynamics and constraints. We note that the results in [4], [5], and [7] do not offer completeness guarantees, whereas the convergence in [6] relies on exact computation of predecessor sets. Model predictive control was developed later as a standard framework for constrained control, but the feasibility of finding a controller is not guaranteed [8].

More recently, abstraction-based methods [9], which leverage model checking algorithms [10] for control synthesis by constructing discrete abstractions of the original system dynamics, has gained popularity for solving various control problems. To be sound and complete in control synthesis, abstractions that are (approximately) equivalent to the original systems is usually needed, which is shown feasible for incrementally stable systems [11], [12]. Without such a stability assumption, we can still construct over-approximations [13]–[16], but it does not always guarantee a feasible control strategy, even if one exists, because spurious transitions are introduced and control synthesis is separated from abstraction. Using sufficiently small granularities, approximately complete control synthesis can be achieved without stability assumptions [17] but it is at the cost of intractable computation.

In this article, we propose a sound and robustly complete control synthesis method for discrete-time nonlinear systems to verify the existence of a reach-and-stay control strategy and construct the strategy if it exists. A control synthesis method is called robustly complete if it returns a control strategy whenever the given specification is realizable for the same system under bounded disturbances. This is an extension of our previous work for switched systems [18]. At the core is a fixed-point algorithm that determines the winning set over the continuous state space with respect to the given specification. Only assuming that the target set is compact, we prove that such an algorithm is sound and complete and a memoryless control strategy is sufficient for nonlinear systems while the one used in [4] fails to be complete under the same condition.

One of our main contributions is that the proposed method is robustly complete, as opposed to most of the abstraction-based methods for nonlinear systems without stability assumptions (e.g., [3], [13], [14], [16], [19]) that are not complete. Another benefit of our method is that it practically gains computational efficiency by partitioning only the region in the state space where necessary. This is achieved by an automatic subdivision framework based on interval computation [20]. In contrast with other applications of interval analysis, such as [21] and [22], we deal with reach-and-stay control problems without the assumption that the target set is controlled invariant. Compared with the works with abstraction refinement mechanisms [3], [19], [23], in which parameters need to be chosen empirically or synthesis does not always terminate in finite time, we device a scheme for adaptive tuning of discretization precision under a given threshold related to system robustness level.

Notation: Let $\mathbb{Z}$, $\mathbb{Z}_{\geq 0}$, $\mathbb{R}$, $\mathbb{R}^n$ be the set of all integers, non-negative integers, reals, and n-dimensional real vectors, respectively, $|\cdot|$ is the infinity norm in $\mathbb{R}^n$ and $\mathbf{1}^n$ indicates the n-dimensional vector with all elements equal to 1; given two sets $A, B \subseteq \mathbb{R}^n$, $B \setminus A := \{x \in B \,|\, x \notin A\}$, $A \ominus B := \{c \in \mathbb{R}^n \,|\, c + b \in A, \forall b \in B\}$, and $\mathcal{B}_r := \{y \in \mathbb{R}^n \,|\, |y| \leq r\}$.

Manuscript received June 17, 2019; revised June 17, 2019 and January 28, 2020; accepted April 8, 2020. Date of publication April 14, 2020; date of current version February 26, 2021. This work was supported in part by the Natural Sciences and Engineering Council of Canada, in part by the Canada Research Chairs program, and in part by Ontario Early Researcher Award. Recommended by Associate Editor L. Palopoli. *(Corresponding author: Yinan Li.)*

The authors are with the Department of Applied Mathematics, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: yinan.li@uwaterloo.ca; j.liu@uwaterloo.ca).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAC.2020.2987711

II. Control Synthesis by Fixed-Point Iterations

A. Reach-and-Stay Control Synthesis Problem

Consider nonlinear control system in the form of

$$x_{t+1} = f(x_t, u_t) + d_t \tag{1}$$

where $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$, $t \in \mathbb{Z}_{\geq 0}$, $x_t \in \mathcal{X} \subseteq \mathbb{R}^n$ is the state, $u_t \in \mathcal{U} \subseteq \mathbb{R}^m$ is the control input, $d_t \in \mathcal{D} \subseteq \mathbb{R}^n$ is a bounded disturbance. The set $\mathcal{X}$ and $\mathcal{U}$ are assumed to be compact, and the bounded set $\mathcal{D} := \{d \in \mathbb{R}^n \mid |d| \leq \delta,\ \delta \geq 0\}$. When $\delta = 0$, system (1) reduces to the nominal one

$$x_{t+1} = f(x_t, u_t). \tag{2}$$

A sequence of control inputs $\mathbf{u} = \{u_i\}_{i=0}^{\infty}$, where $u_i \in \mathcal{U}$, is called a control signal. Similarly, we denote by $\mathbf{d} = \{d_i\}_{i=0}^{\infty}$ a disturbance. A solution of system (1) is an infinite sequence of states $\mathbf{x} = \{x_i\}_{i=0}^{\infty}$ generated by an initial condition $x_0 \in \mathcal{X}$, a control signal $\mathbf{u}$ and a disturbance $\mathbf{d}$ such that (1) is satisfied for all $t \in \mathbb{Z}_{\geq 0}$.

Definition 1: Let Ω be a subset of the state space $\mathcal{X}$ of a system (1). A reach-and-stay property of a solution $\mathbf{x} = \{x_i\}_{i=0}^{\infty}$ of the system (1) with respect to Ω, denoted by $\varphi(\Omega)$, requires that $x_k \in \Omega$ for all $k \geq j$, $k, j \in \mathbb{Z}_{\geq 0}$.

The purpose of this article is to design a control strategy, if there exists one, such that the resulting solution satisfies a given reach-and-stay objective $\varphi(\Omega)$. The set Ω will be omitted when the target area is clear from the context or we discuss a reach-and-stay objective in general. The form of control strategies is given in the following definition.

Definition 2: A *(memoryless) control strategy* of system (1) is a function $\kappa : \mathcal{X} \to 2^{\mathcal{U}}$. A control signal $\mathbf{u} = \{u_k\}_{k=0}^{\infty}$ is said to conform to a control strategy κ, if $u_k \in \kappa(x_k), \forall k \geq 0$, where $\{x_k\}_{k=0}^{\infty}$ is the resulting solution of system (1).

If there exists an initial condition $x_0 \in \mathcal{X}$ and a memoryless control strategy κ such that, for any disturbance, the resulting solution of system (1) under a control signal that conforms to κ satisfies the objective φ, we say φ is realizable for (1), and the control strategy κ realizes φ for (1). The set of all realizable initial conditions is the winning set of (1) with respect to φ, written as $\text{Win}^{\delta}(\varphi)$. For the nominal system (2), the notation is simplified to $\text{Win}(\varphi)$. If $\text{Win}(\varphi) \neq \varnothing$ or $\text{Win}^{\delta}(\varphi) \neq \varnothing$, then φ is realizable for system (2) or (1).

Problem 1 (Reach-and-Stay Control Synthesis): Consider a reach-and-stay objective φ for system (1) given as follows:

i) determine if φ is realizable;
ii) synthesize a control strategy such that the closed-loop system satisfies φ if possible;

We assume neither that a target set Ω is controlled invariant (i.e., every state $x \in \Omega$ can be controlled inside Ω for all time) nor the existence of a control strategy, which is to be determined by solving Problem 1 (i).

B. Fixed-Point Characterization of Realizability

The realizability of a reach-and-stay objective $\varphi(\Omega)$ with $\Omega \subseteq \mathcal{X}$ for system (1) can be determined through a fixed-point algorithm. The winning set of $\varphi(\Omega)$ is characterized by the fixed point, which is a subset of $\mathcal{X}$ returned by the algorithm. Considering that the system state space is always bounded in real applications, it is fair to assume the compactness of the state space $\mathcal{X}$ and the target set Ω.

Definition 3: Given a set $Y \subseteq \mathcal{X}$, the predecessor of Y with respect to system (1) is a set of states defined by

$$\text{Pre}^{\delta}(Y) := \{x \in \mathcal{X} \mid \exists u \in \mathcal{U},\ \text{s.t. } f(x,u) + d \in Y, \forall d \in \mathcal{D}\}.$$

The predecessor is simplified to $\text{Pre}(Y)$ for $\delta = 0$. The set of valid control values that lead to one-step transition to Y for an $x \in \text{Pre}^{\delta}(Y)$ is

$$U_Y(x) := \{u \in \mathcal{U} \mid f(x,u) + d \in Y,\ \forall d \in \mathcal{D}\}.$$

For $X, Y \subseteq \mathcal{X}$, the predecessor Y that resides in a set X is the set $X \cap \text{Pre}^{\delta}(Y)$. To simplify the notation, we let

$$\text{Pre}^{\delta}(Y|X) = X \cap \text{Pre}^{\delta}(Y).$$

The following properties are straightforward without further assumptions on system (1).

Proposition 1: Let $A, B \subseteq \mathcal{X}$ and $\delta \geq 0$. Then

i) $\text{Pre}^{\delta}(A) \subseteq \text{Pre}^{\delta}(B)$ if $A \subseteq B$.
ii) $\text{Pre}^{\delta}(X) = \text{Pre}(X \ominus \mathcal{B}_{\delta})$ and $\text{Pre}^{\delta_2}(A) \subseteq \text{Pre}^{\delta_1}(A)$ if $0 \leq \delta_1 \leq \delta_2$.

If continuity is imposed to the dynamics, then $\text{Pre}^{\delta}(\cdot)$ has some additional property.

Assumption 1: The function $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ in system (1) is continuous with respect to both arguments, and the state space $\mathcal{X}$ and the input space $\mathcal{U}$ are compact.

Proposition 2 ([24]): Under Assumption 1, if $A \subseteq \mathcal{X}$ is closed (compact), then $\text{Pre}^{\delta}(A)$ is closed (compact).

It is straightforward to see that Proposition 2 still holds if $\mathcal{U}$ is a finite set.

Assumption 2: The function $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ in system (1) is continuous with respect to the first argument. The state space $\mathcal{X}$ is compact and input space $\mathcal{U}$ is finite.

We now present the following algorithm (3) for reach-and-stay control synthesis, which consists of two nested fixed-point iterations:

$$Y_{\infty} \Leftarrow \begin{cases} Y_0 = \emptyset, X_0^{\infty} = \emptyset \\ \left.\begin{array}{l} X_{i+1}^0 = Y_i \cup \Omega \\ X_{i+1}^{j+1} = \text{Pre}^{\delta}(X_{i+1}^j | X_{i+1}^j) \end{array}\right\} \Rightarrow X_{i+1}^{\infty} \\ \kappa(x) \leftarrow U_{X_{i+1}^{\infty}}(x), \forall x \in \Omega \cap \left(X_{i+1}^{\infty} \setminus X_i^{\infty}\right) \\ Y_{i+1} = \text{Pre}^{\delta}(X_{i+1}^{\infty}) \\ \kappa(y) \leftarrow U_{X_{i+1}^{\infty}}(y), \forall y \in Y_{i+1} \setminus (Y_i \cup \Omega). \end{cases} \tag{3}$$

In the following proposition, we show that algorithm (3) characterizes the winning set $\text{Win}(\varphi(\Omega))$. This implies that the realizability $\varphi(\Omega)$ for system (1) can be determined by checking the emptyness of Y_{∞}.

Proposition 3: Let $\Omega \subseteq \mathcal{X}$ be compact. Suppose that Assumption 1 or 2 holds. Let $Y_{\infty} = \bigcup_{i=0}^{\infty} Y_i$ be a fixed point of (3), where $\{Y_i\}_{i=0}^{\infty}$ is a sequence of subsets of $\mathcal{X}$ generated from (3). Then

i) $\text{Win}^{\delta}(\varphi(\Omega)) = Y_{\infty}$.
ii) κ is a memoryless control strategy that realizes $\varphi(\Omega)$.

Proof: We only consider $\Omega \neq \emptyset$. Otherwise the results trivially hold. We first show $Y_{\infty} \subseteq \text{Win}^{\delta}(\varphi(\Omega))$ by induction. Trivially $Y_0 = \emptyset \subseteq \text{Win}^{\delta}(\varphi(\Omega))$ and X_1^{∞} is compact. The induction step aims to show that, for all $i \geq 1$, $Y_{i+1} \subseteq \text{Win}^{\delta}(\varphi(\Omega))$ if $Y_i \subseteq \text{Win}^{\delta}(\varphi(\Omega))$. Assume that X_i^{∞} is compact. Then, $Y_i = \text{Pre}^{\delta}(X_i^{\infty})$ and thus $X_{i+1}^0 = \Omega \cup Y_i$ is compact. The sequence $\{X_{i+1}^j\}_{j=0}^{\infty}$ is compact and decreasing by induction, using Proposition 1 (i) and Proposition 2 since $X_{i+1}^0 = \Omega \cup Y_i$ is compact. It is also easy to show that $\{Y_i\}_{i=0}^{\infty}$ is increasing by induction. Furthermore, the compact limit set $X_{i+1}^{\infty} = \lim_{j\to\infty} X_{i+1}^j = \bigcap_{j=0}^{\infty} X_{i+1}^j$ (with respect to Painlevé-Kuratowski convergence [25]) is the maximal controlled invariant set inside $\Omega \cup Y_i$ [6, Prop. 4]. If $Y_i \subseteq \text{Win}^{\delta}(\varphi(\Omega))$, then $X_{i+1}^{\infty} \subseteq \text{Win}^{\delta}(\varphi(\Omega))$ because X_{i+1}^{∞} is a controlled invariant set inside $\Omega \cup Y_i$, which gives $Y_{i+1} = \text{Pre}^{\delta}(X_{i+1}^{\infty}) \subseteq \text{Win}^{\delta}(\varphi(\Omega))$ by Definition 3. Hence, $\bigcup_{i=0}^{\infty} Y_i \subseteq \text{Win}^{\delta}(\varphi(\Omega))$.

Applying the control inputs generated by κ, for all $i \geq 0$ and for all $x \in \Omega \cap (X_{i+1}^{\infty} \setminus X_i^{\infty})$ and $x \in Y_{i+1} \setminus (Y_i \cup \Omega)$, the state x will be controlled inside $\Omega \cup Y_i$ and Y_i in one step, respectively. That means any state $x \in Y_{i+1}$ will be controlled into Y_i until it enters $X_1^{\infty} \subseteq \Omega$, which is controlled invariant. Hence, we have also shown (ii) that κ realizes $\varphi(\Omega)$.

To see $\text{Win}^{\delta}(\varphi(\Omega)) \subseteq Y_{\infty}$, we aim to show that $x \notin \text{Win}^{\delta}(\varphi(\Omega))$ for all $x \notin Y_{\infty}$. Let $x \notin Y_{\infty}$ be arbitrary. Since Y_{∞} is a fixed point of (3), i.e., $Y_{\infty} = \text{Pre}^{\delta}(V)$, where V is the maximal controlled invariant set inside $\Omega \cup Y_{\infty}$. Then, $x \notin \text{Pre}^{\delta}(V)$, which means that for all $\{u_i\}_{i=0}^{\infty}$ there exists k and $\{d_i\}_{i=0}^{k}$ such that the resulting sequence of (1) satisfies $x_k \notin (\Omega \cup Y_{\infty})$. Since $x_k \notin Y_{\infty}$, we can show in the same manner that for all $\{u_i\}_{i=k}^{\infty}$ there exists $k' \geq k$ and $\{d_i\}_{i=k}^{k'}$ such that the k'th state $x_{k'}$ of the resulting solution satisfies $x_{k'} \notin (\Omega \cup Y_{\infty})$. In this way, for all $\{u_i\}_{i=0}^{\infty}$, we can find an infinite sequence $\{d_i\}_{i=0}^{\infty}$ for any $x \notin Y_{\infty}$ so that the resulting solution of (1) goes outside of Ω infinitely often. Hence, $x \notin \text{Win}^{\delta}(\varphi(\Omega))$, which shows $\text{Win}^{\delta}(\varphi(\Omega)) \subseteq Y_{\infty}$. The proof is now complete. ■

Proposition 3 is a generalized result for the reach-and-stay problem. A algorithm for solving the reach-and-stay problem was first proposed in [4] [as shown in (4)], which relies on the assumption that the target set is compact and convex

$$\left.\begin{array}{l} X_0 = \Omega \\ X_{i+1} = \text{Pre}^{\delta}(X_i | X_i) \end{array}\right\} \Rightarrow X_{\infty}$$
$$\kappa(x) \leftarrow U_{X_{\infty}}(x), \forall x \in X_{\infty}$$
$$\left.\begin{array}{l} Z_0 = X_{\infty} \\ Z_{i+1} = \text{Pre}^{\delta}(Z_i) \\ \kappa(z) \leftarrow U_{Z_{i+1}}(z), \forall z \in Z_{i+1} \setminus Z_i \end{array}\right\} \Rightarrow Z_{\infty}. \tag{4}$$

As opposed to (3), algorithm (4) is composed of two sequential fixed-point iterations, which fails to yield the real winning set. This can be illustrated in the following example.

Example 1: Consider a target set $\Omega = [-0.3, 0.3] \cup [0.8, 1.1]$ and the dynamics $x_{t+1} = -x_t(x_t^2 - 2.05x_t + 0.05) + u_t + d_t$, where $x_t \in [-0.65, 1.1]$, $u_t \in \{0, 10\}$ and $d_t \in [-5, 5] \times 10^{-4}$ for $t \in \mathbb{Z}_{\geq 0}$. We obtain the real winning set $\text{Win}^{\delta}(\varphi(\Omega)) = Y_{\infty} = [-0.6311, 1.1]$ by using algorithm (3) while algorithm (4) only gives a subset $Z_{\infty} = [-0.6311, -0.6082) \cup (-0.6021, 0.9914) \cup (1.0135, 1.1]$. Fig. 1 illustrates the difference between these two different algorithms.

Remark 1: In the literature, reach-avoid-stay objectives are also considered (e.g., [3]), which additionally require the system state to avoid unsafe regions. By the definition of predecessor and algorithm (3), the winning set, as well as every intermediate set, is bounded in the state space $\mathcal{X}$, which guarantees that any controlled trajectory using the synthesized control strategy κ lives inside $\mathcal{X}$. Hence, (3) can also be applied to solve reach-avoid-stay control problems by restricting the state space $\mathcal{X}$ to safe regions only.

C. Robustly Complete Control Synthesis

One problem with algorithm (3), however, is that it might not terminate in a finite number of iterations.

Example 2: Consider the system (in polar coordinates): $r_{t+1} = r_t^2$, $\theta_{t+1} = \mod(\theta_t + \theta_0/2\pi)$, $\theta_0 \in [0, 2\pi)$. For this system, there is an unstable limit cycle given by $O = \{(r, \theta) \in \mathbb{R} \times [0, 2\pi) \mid r = 1\}$. Let the target set Ω be a subset of O that contains the origin. The winning set $\text{Win}(\varphi)$ is the interior of O, which is open. Since the set returned by (3) after a finite number of iterations is always closed, the algorithm cannot terminate in finite time.

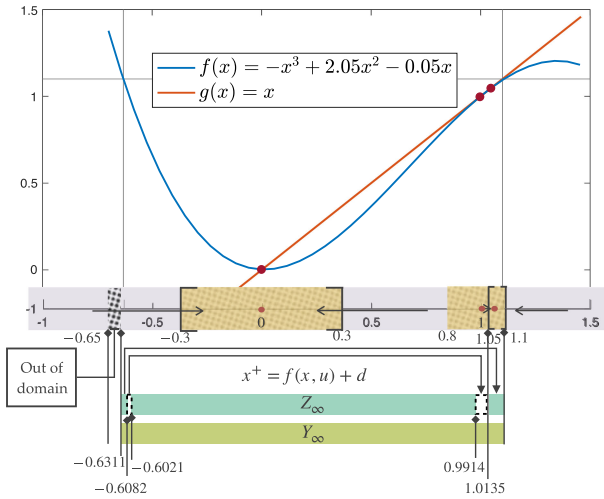


Fig. 1. Let $u_t = 0$ for all t. The fixed points 0 and 1.05 are stable while 1 is unstable, which leads to $X_{\infty} = [-0.3, 0.3] \cup [1.0370, 1.1]$ if (4) is used. Computing the reachable set of X_{∞} only gives a subset Z_{∞} of the real winning set.

Another difficulty is the computation of predecessors under nonlinear mappings. Only for some special cases, e.g., predecessors of polyhedral sets with respect to linear dynamics, which can be characterized by linear inequalities, the exact computation is possible. For general nonlinear dynamics, a practical way to overcome this difficulty is to use approximations. Inner-approximations are often used for control synthesis, since valid control values cannot be found for all states in an outer-approximation. However, such a numerical compromise is at the expense of the loss of completeness, because the emptiness of the approximated winning set does not reflect the realizability of a given reach-and-stay property.

Alternatively, we study a relaxed problem based on the robust realizability of a specification introduced in the following.

Definition 4: A reach-and-stay objective φ is said to be δ-robustly realizable for system (2) if it is realizable for (1). If $\delta > 0$, then φ is called robustly realizable for (2).

Problem 2 (Robustly Complete Reach-and-Stay Control Synthesis): Consider a reach-and-stay property φ for system (2). Answer one of the two following questions.

i) Construct a control strategy if φ is robustly realizable for system (2).
ii) Verify that φ is not realizable for system (1) with $\delta > 0$.

III. Reach-and-Stay Control Synthesis is Robustly Complete

This section presents our main results: there is a computer algorithm based on approximations of predecessors that will return reach-and-stay control strategies for nonlinear systems whenever the specification is robustly realizable. Such a conclusion is made because the inner-approximated winning set can be lower bounded by the winning set of the same system with additional perturbations, provided some condition to the precision of predecessor approximations is satisfied.

A. Approximated Synthesis for Reach-and-Stay Control

There are some computational redundancies in (3): the sequence $\{Y_i\}_{i=0}^{\infty}$ is increasing and so is $\{X_i^j\}_{i=0}^{\infty}$ for all j. Hence, it is only necessary to compute the incremental parts between two adjacent sets in

the sequences. Also, considering that predecessors cannot be precisely computed, we present the following approximated control synthesis algorithm (5) based on an approximation $\widehat{\text{Pre}}$ of the predecessor operator Pre:

$$\widehat{Y}_\infty \Leftarrow \begin{cases} \widehat{Y}_0 = \widehat{X}_0^\infty = \emptyset, V_0 = \mathcal{X} \setminus \Omega \\ W_i^0 = \Omega \setminus \widehat{Y}_i \\ \widehat{X}_{i+1}^j = \widehat{Y}_i \cup W_i^j \\ W_i^{j+1} = \widehat{\text{Pre}}(\widehat{X}_{i+1}^j | W_i^j) \end{cases} \Rightarrow \begin{matrix} W_i^\infty \\ \widehat{X}_{i+1}^\infty \end{matrix} \\ \begin{cases} \kappa(x) \leftarrow U_{X_{i+1}^\infty}(x), \forall x \in W_i^\infty \\ Z_i = \widehat{\text{Pre}}(\widehat{X}_{i+1}^\infty | V_i) \\ \kappa(x) \leftarrow U_{X_{i+1}^\infty}(x), \forall x \in Z_i \\ V_{i+1} = V_i \setminus Z_i \\ \widehat{Y}_{i+1} = X_{i+1}^\infty \cup Z_i. \end{cases} \tag{5}$$

Theorem 1: Let Y_∞ and Y_∞^r ($r > 0$) be the outputs of (3) with operator Pre and Pre^r, respectively. Suppose that $\widehat{\text{Pre}}(X)$ satisfies Propositions 1, 2, and $\text{Pre}^r(X) \subseteq \widehat{\text{Pre}}(X) \subseteq \text{Pre}(X)$ for any $X \subseteq \mathcal{X}$. Then

$$Y_\infty^r \subseteq \widehat{Y}_\infty \subseteq Y_\infty.$$

Proof: Let $\{\widetilde{X}_i^\infty\}$ ($\{\widetilde{X}_i^{r\infty}\}$) and $\{\widetilde{Y}_i\}$ ($\{\widetilde{Y}_i^r\}$) be the sequences of sets generated by algorithm (5) with $\widehat{\text{Pre}} = \text{Pre}$ ($\widehat{\text{Pre}} = \text{Pre}^r$). We prove the theorem in the following structure.

i) Show that (5) is equivalent to (3) when set computation is accurate, i.e., $\widetilde{X}_i^\infty = X_i^\infty$ ($\widetilde{X}_i^{r\infty} = X_i^{r\infty}$) and $\widetilde{Y}_i = Y_i$ ($\widetilde{Y}_i^r = Y_i^r$) for all i.
ii) Show $\widetilde{Y}_\infty^r \subseteq \widehat{Y}_\infty \subseteq \widetilde{Y}_\infty$ under the given condition.

First of all, we show that $Y_i \subseteq \text{Pre}(Y_i)$ for all i. Since X_i^∞ is a controlled invariant set, $X_i^\infty \subseteq \text{Pre}(X_i^\infty)$. By the definition of Y_i in (3) and monotonicity of Pre, $Y_i = \text{Pre}(X_i^\infty) \subseteq \text{Pre}(\text{Pre}(X_i^\infty)) = \text{Pre}(Y_i)$. We now prove (i) by induction. The base case clearly holds since $\widetilde{Y}_0 = Y_0 = \widetilde{X}_0^\infty = X_0^\infty = \emptyset$. Suppose that $\widetilde{X}_i^\infty = X_i^\infty$ and $\widetilde{Y}_i = Y_i$ for some $i > 0$. Then $\widetilde{X}_{i+1}^0 = \widetilde{Y}_i \cup (\Omega \setminus \widetilde{Y}_i) = \widetilde{Y}_i \cup \Omega = X_{i+1}^0$, and

$$\begin{aligned} \widetilde{X}_{i+1}^{j+1} &= \widetilde{Y}_i \cup W_i^{j+1} = \widetilde{Y}_i \cup (\text{Pre}(\widetilde{X}_{i+1}^j) \cap W_i^j) \\ &= (\widetilde{Y}_i \cup \text{Pre}(\widetilde{X}_{i+1}^j)) \cap (\widetilde{Y}_i \cup W_i^j) \\ &= (\widetilde{Y}_i \cup \text{Pre}(\widetilde{X}_{i+1}^j)) \cap \widetilde{X}_{i+1}^j. \end{aligned}$$

Also, $\text{Pre}(\widetilde{X}_{i+1}^j) = \text{Pre}(\widetilde{Y}_i \cup W_i^j) \supseteq \text{Pre}(\widetilde{Y}_i) \supseteq \widetilde{Y}_i$, which implies that $\widetilde{X}_{i+1}^{j+1} = \text{Pre}(\widetilde{X}_{i+1}^j) \cap \widetilde{X}_{i+1}^j$. This is the same as the iteration step in (3), and thus $\widetilde{X}_{i+1}^\infty = X_{i+1}^\infty$. Now consider the sequence $\{V_i\}_{i=0}^\infty$. We have $V_0 = \mathcal{X} \setminus \Omega$ and $V_{i+1} = V_i \setminus (\text{Pre}(\widetilde{X}_i^\infty) \cap V_i) = V_i \setminus \text{Pre}(\widetilde{X}_i^\infty)$. Unfolding V_i until V_0 and using that $\text{Pre}(\widetilde{X}_i^\infty) \subseteq \text{Pre}(\widetilde{X}_{i+1}^\infty)$, we can derive $V_i = \mathcal{X} \setminus (\Omega \cup \text{Pre}(\widetilde{X}_i^\infty)) = \mathcal{X} \setminus (\Omega \cup Y_i) = \mathcal{X} \setminus \widetilde{X}_{i+1}^0$. Then

$$\begin{aligned} \text{Pre}(\widetilde{X}_{i+1}^\infty) &= \text{Pre}(\widetilde{X}_{i+1}^\infty) \cap (\widetilde{X}_{i+1}^0 \cup V_i) \\ &= \left[\text{Pre}(\widetilde{X}_{i+1}^\infty) \cap \widetilde{X}_{i+1}^0\right] \cup \left[\text{Pre}(\widetilde{X}_{i+1}^\infty) \cap V_i\right] \\ &= \widetilde{X}_{i+1}^\infty \cup \text{Pre}(\widetilde{X}_{i+1}^\infty | V_i) = \widetilde{Y}_{i+1}. \end{aligned} \tag{6}$$

The equality $\text{Pre}(\widetilde{X}_{i+1}^\infty) \cap \widetilde{X}_{i+1}^0 = \widetilde{X}_{i+1}^\infty$ can be derived by contradiction. If there exists $A \subseteq \widetilde{X}_{i+1}^0 \setminus \widetilde{X}_{i+1}^\infty$ such that $A \subseteq \text{Pre}(\widetilde{X}_{i+1}^\infty)$ then $\widetilde{X}_{i+1}^\infty \cup A \subseteq \text{Pre}(\widetilde{X}_{i+1}^\infty \cup A)$, which indicates $A \cup \widetilde{X}_{i+1}^\infty$ is a larger controlled invariant set inside $\widetilde{X}_{i+1}^0$, but $\widetilde{X}_{i+1}^\infty$ is the maximal one. Therefore, $Y_{i+1} = \widetilde{Y}_{i+1}$. The abovementioned argument also applies to prove $\widetilde{X}_i^{r\infty} = X_i^{r\infty}$ and $\widetilde{Y}_i^r = Y_i^r$.

To prove (ii), we aim to show $X_i^{r\infty} \subseteq \widehat{X}_i^\infty \subseteq X_i^\infty$ and $Y_i^r \subseteq \widehat{Y}_i \subseteq Y_i$ for all i. Clearly $X_1^{r0} = \widehat{X}_1^0 = X_1^0 = \Omega$ and $\text{Pre}^r(X_1^{r0} | W_1^0) \subseteq \widehat{\text{Pre}}(\widehat{X}_1^0 | W_1^0) \subseteq \text{Pre}(X_1^0 | W_1^0)$ since $\text{Pre}^r(X) \subseteq \widehat{\text{Pre}}(X) \subseteq \text{Pre}(X)$ and Proposition 1 (ii). This means $X_1^{r1} \subseteq \widehat{X}_1^1 \subseteq X_1^1$. By induction, we can easily achieve $X_1^{rj} \subseteq \widehat{X}_1^j \subseteq X_1^j$ for any j. Thus, $X_1^{r\infty} \subseteq \widehat{X}_1^\infty \subseteq X_1^\infty$. As shown in (6), $\widehat{Y}_i = \widehat{\text{Pre}}(\widehat{X}_i^\infty)$. Then, $Y_1^r = \text{Pre}^r(X_1^{r\infty}) \subseteq \text{Pre}^r(\widehat{X}_1^\infty) \subseteq \widehat{Y}_1 \subseteq \text{Pre}(\widehat{X}_i^\infty) \subseteq \text{Pre}(X_i^\infty) = Y_1$. Therefore, (ii) can also be shown using induction. ■

B. Robustly Complete Control Synthesis via Interval Arithmetic

To implement the operator $\widehat{\text{Pre}}$ in (5), we use interval arithmetic. This is because any compact set can be approximated by intervals with convergence guarantee under mild assumptions and interval operations are simple. An interval vector (box) in $\mathbb{R}^n$ is denoted by $[x]$, where $[x] := [x_1] \times \cdots \times [x_n] \subseteq \mathbb{R}^n$ and $[x_i] = [\underline{x}_i, \overline{x}_i] \subseteq \mathbb{R}$ for $i = 1, \ldots, n$ with $\underline{x}_i$ as the infimum of $[x_i]$ and $\overline{x}_i$ the supremum. Let the set of all boxes in $\mathbb{R}^n$ be $\mathbb{IR}^n$. The width of the interval $[x]$ is defined as $\text{wid}([x]) := \max_{1 \le i \le n} \{\overline{x_i} - \underline{x_i}\}$.

We have described in [26, Algorithm 1] a algorithm to obtain an inner approximation of $\text{Pre}(Y|X)$ with ε precision ($\varepsilon > 0$), denoted by $[\text{Pre}]^\varepsilon(Y|X)$ here, which is a union of a finite number of intervals. Central to this algorithm is the convergent inclusion function $[f] : \mathbb{IR}^n \to \mathbb{IR}^m$ of $f : \mathbb{R}^n \to \mathbb{R}^m$ such that $f([x]) \subseteq [f]([x])$ for all $[x] \in \mathbb{IR}^n$ and $\lim_{\text{wid}([x]) \to 0} \text{wid}([f]([x])) = 0$. The parameter ε controls the minimum width of intervals for approximating $\text{Pre}(Y|X)$. We now discuss the relation between ε and the error of set approximation in two scenarios.

1) Finite Control Values: In this case, system (1) can be treated as switched system, which has been discussed in [18] and [27]. We summarize the result with its assumption as follows.

Assumption 3: For system (1), there exists a constant $\rho > 0$ such that

$$|f(x,u) - f(y,u)| \le \rho |x - y|, \quad \forall x, y \in \mathcal{X}. \tag{7}$$

By Assumption 3, we can always construct the centered-form convergent inclusion function $[f]([x], u) = f(\bar{x}, u) + \rho([x] - \bar{x})\mathbf{1}^n$ based on (7) for all $[x] \subseteq \mathcal{X}$.

Under Assumption 3, [27, Lemma 1] can be used directly and presented as the following lemma.

Lemma 1: Let $Y, X \subseteq \mathcal{X}$ be compact. If system (1) satisfies Assumption 3 in an neighborhood of X, then

$$\text{Pre}(Y \ominus \mathcal{B}_{\rho\varepsilon} | X) \subseteq [\text{Pre}]^\varepsilon(Y|X) \subseteq \text{Pre}(Y|X).$$

2) Infinite Control Values: The compact set $\mathcal{U} \subseteq \mathbb{R}^m$ might contain an infinite number of elements in $\mathcal{U}$. A straightforward way is to uniformly sample points in within the control set, e.g., an under-sampled set of controls

$$[\mathcal{U}]_\eta := \eta \mathbb{Z}^m \cap \mathcal{U} \tag{8}$$

where $\mathbb{Z}^m$ denotes the m-dimensional integer grid, and $\eta\mathbb{Z}^m = \{\eta z \mid z \in \mathbb{Z}^m, \eta > 0\}$. We additionally assume that for all $x \in \mathcal{X}$ and $u, v \in \mathcal{U}$

$$|f(x,u) - f(x,v)| \le \rho |u - v|. \tag{9}$$

Similar to Lemma 1, we prove the following approximation error by using under-sampled control values.

Lemma 2: Consider (1) with under-sampled control values (8). Let $Y, X \subseteq \mathcal{X}$ be compact. If system (1) satisfies Assumption 3 and (9) in

a neighborhood of X, then

$$\mathrm{Pre}(Y \ominus \mathcal{B}_{\rho(\varepsilon+\eta)}|X) \subseteq [\mathrm{Pre}]^{\varepsilon}(Y|X) \subseteq \mathrm{Pre}(Y|X).$$

Proof: We define a new predecessor operator $\mathrm{Pre}_\eta(X) := \{x \in \mathcal{X} \mid \exists u \in [\mathcal{U}]_\eta, \text{ s.t. } f_u(x) + d \in X, \forall d \in \mathcal{D}\}$.

Let $Z = \mathrm{Pre}(Y|X)$, $Z_\eta = \mathrm{Pre}_\eta(Y|X)$ and $\widetilde{Y} = Y \ominus \mathcal{B}_{\rho\frac{\eta}{2}}$. We first claim that $\mathrm{Pre}(\widetilde{Y}|X) \subseteq Z_\eta \subseteq Z$. Trivially $Z_\eta \subseteq Z$ because $[\mathcal{U}]_\eta$ is a subset of $\mathcal{U}$. By Definition 3, for all $z \in \mathrm{Pre}(\widetilde{Y}|X)$, there exists a $u \in \mathcal{U}$ such that $f(z,u) + d \in \widetilde{Y}$ for all $d \in \mathcal{D}$. With (9), for all $u \in \mathcal{U}$, there exists a $v \in [\mathcal{U}]_\eta$ such that $f(z,v) \in f(z,u) \oplus \mathcal{B}_{\rho\frac{\eta}{2}}$. Then, $f(z,v) + d \in f(z,u) \oplus \mathcal{B}_{\rho\frac{\eta}{2}} + d = (f(z,u) + d) \oplus \mathcal{B}_{\rho\frac{\eta}{2}} \in \widetilde{Y} \oplus \mathcal{B}_{\rho\frac{\eta}{2}} = Y \ominus \mathcal{B}_{\rho\frac{\eta}{2}} \oplus \mathcal{B}_{\rho\frac{\eta}{2}} \in Y$ by [28, Th. 2.1 (ii)], which means that $z \in Z_\eta$. Hence, the claim holds.

By Lemma 1, $\mathrm{Pre}_\eta(\widetilde{Y} \ominus \mathcal{B}_{\rho\varepsilon}|X) \subseteq \underline{Z} \subseteq Z_\eta$. Applying the claim above, we have $\mathrm{Pre}(\widetilde{Y} \ominus \mathcal{B}_{\rho\varepsilon} \ominus \mathcal{B}_{\rho\frac{\eta}{2}}|X) \subseteq \mathrm{Pre}_\eta(\widetilde{Y} \ominus \mathcal{B}_{\rho\varepsilon}|X)$. Therefore, $\mathrm{Pre}(Y \ominus \mathcal{B}_{\rho(\varepsilon+\eta)}|X) \subseteq \underline{Z} \subseteq Z_\eta \subseteq \mathrm{Pre}(Y|X)$, which completes the proof. ■

Remark 2: The abovementioned result can also be established for system $x_{t+1} = f(x_t, u_t, w_t) + d_t$, where $w_t \in \mathcal{W} \subseteq \mathbb{R}^p$ and $\mathcal{W}$ is a bounded set of nonadditive disturbances. As indicated in [17, Lemma 1], to achieve higher approximation precision in computing $[f]([x], u, \mathcal{W})$, we can mince the set $\mathcal{W}$ into smaller subintervals and take the union of the images of all the subintervals under the inclusion functions. Suppose that $\mathcal{W}$ is uniformly partitioned with size μ, i.e., $[\mathcal{W}]_\mu$. Replacing the inclusion function for (1) by $\bigcup_{[w]\in[\mathcal{W}]_\mu}[f]([x], u, [w])$, we can show, without much effort, that $\mathrm{Pre}(Y \ominus \mathcal{B}_{\rho(\varepsilon+\mu)}|X) \subseteq \underline{Z} \subseteq \mathrm{Pre}(Y|X)$.

Theorem 2: Consider system (2) under Assumption 2. Let $\Omega \subseteq \mathcal{X}$ be compact and Assumption 3 holds on $\mathcal{X}$. Suppose that $\varphi(\Omega)$ is δ-robustly realizable for system (2). Then, algorithm (5) with sets represented in intervals and $\widehat{\mathrm{Pre}} = [\mathrm{Pre}]^{\varepsilon}$ terminates in finite time and the following holds if $\rho\varepsilon \leq \delta$:

$$\mathrm{Win}^{\delta}(\varphi(\Omega)) \subseteq Y^{\varepsilon} \subseteq \mathrm{Win}(\varphi(\Omega)). \tag{10}$$

Proof: Relation (10) is an immediate result from Lemma 1 and Theorem 1, so we only show the finite termination here. Assume that Ω is an interval or a union of intervals. Under a given precision $\varepsilon > 0$, W_i can only be partitioned to finite number of intervals. Then, for the inner loop, there must exist a positive integer N such that $W_i^N = \emptyset$ if $W_i^j \neq W_i^{j+1}$ for all $j \in \mathbb{Z}_{\geq 0}$ because $\{W_i^j\}_{j=0}^{\infty}$ is decreasing. Thus, the inner loop terminates within each outer loop. Likewise, the outer loop is also terminating since $\{V_i\}_{i=0}^{\infty}$ is decreasing and $\mathcal{X}$ only consists of a finite number of intervals. ■

A similar result can also be established for systems under Assumption 1.

Theorem 3: Consider system (2) under Assumption 1 with a set of under-sampled control values (8). Let $\Omega \subseteq \mathcal{X}$ be compact and Assumption 3 and (9) hold. Suppose that $\varphi(\Omega)$ is δ-robustly realizable for system (2). Then, algorithm (5) with sets represented in intervals and $\widehat{\mathrm{Pre}} = [\mathrm{Pre}]^{\varepsilon}$ terminates in finite time and the output Y^{ε} satisfies (10) if $\rho(\varepsilon + \eta) \leq \delta$.

Remark 3: It is worth noting that the precision control parameter ε in the inner and outer loops of algorithm (5) can be set to different values, especially when the target area is volumetrically minuscule relative to the state space, e.g., in practical regulation problems. Furthermore, the precision control parameters are not necessarily fixed throughout the computation, but change with respect to the winning set obtained at each iteration.

Remark 4: The abovementioned theorems, however, cannot trivially lead to the convergence result, i.e., $\lim_{\varepsilon\to 0} Y^{\varepsilon} = \mathrm{Win}(\varphi)$. This is because $\lim_{\delta\to 0}\mathrm{Win}^{\delta}(\varphi) = \mathrm{Win}(\varphi)$ does not always hold under Assumption 3.

By Theorem 2 (Theorem 3) and a controller defined in [18, Prop. 3], algorithm (5) is guaranteed to generate a nonempty winning set along with a memoryless control strategy if the specification is robustly realizable. Even if algorithm (5) returns an empty set, we can still make some conclusion on the robust realizability property of the specification, which are spelled out in the following corollary.

Corollary 1: Suppose that the assumptions for system (2) in Theorem 2 (Theorem 3) hold. Then, control synthesis with respect to a reach-and-stay specification φ is robustly complete, i.e., there exists an algorithm that

i) generates a memoryless control strategy that realizes φ, if φ is robustly realizable for system (2);
ii) verifies that φ is not δ-realizable for system (2) for $\delta \geq \rho\varepsilon$ ($\delta \geq \rho(\varepsilon + \eta)$), if it returns no result.

Remark 5: The conditions in Theorems 2 and 3 serve as criteria for choosing the precision control parameter if the bound of disturbance δ and the Lipschitz constant ρ over the state space can be trivially determined. Using such a criterion in actual computation is usually too conservative due to the evaluation of the Lipschitz constant over the entire state space. A practical benefit of Theorems 2 and 3 is the guarantee that the winning set can be approximated more precisely by using a smaller precision parameter. Corollary 1 implies that if we start computation with a large ε and iteratively reduce it until the algorithm achieves a nonempty result, algorithm (5) can also estimate the bound of the disturbances that can be tolerated without breaking the realizability of the given specification.

C. Example: Automatic Parallel Parking

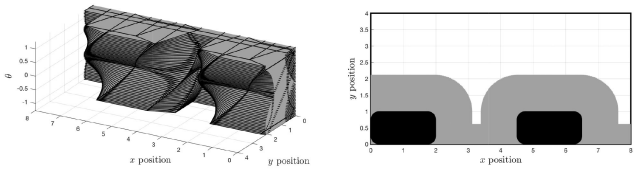
We now demonstrate the effectiveness of the proposed algorithm on automatic parallel parking of the unicycle model [29]: $\dot{x} = v\cos(\gamma + \theta)\cos(\gamma)^{-1}$, $\dot{y} = v\sin(\gamma + \theta)\cos(\gamma)^{-1}$, $\dot{\theta} = v\tan(\phi)$, where (x, y) is the planar position of center of the unicycle, θ is its orientation, the control variable v represents the velocity, ϕ is the steering angle command, and $\gamma = \arctan(a\tan(\phi)/b)$ with $a/b = 1/2$.

In our simulation, the state space is $\mathcal{X} = [0, 8] \times [0, 4] \times [-72^\circ, 72^\circ]$, sampling time is $\tau_s = 0.3$s, and the set of control values is $\mathcal{U} = \{\pm 0.9, \pm 0.6, \pm 0.3, 0\}$, which is sampled by uniform discretization of the space $[-1, 1] \times [-1, 1]$ with grid width $\eta = 0.3$. An exact discrete-time model of the unicycle can be obtained [30] and readily verified Lipschitz continuous over $\mathcal{X}$ for all control values in $\mathcal{U}$.

Suppose that the length and width of the unicycle be $L = 2$ and $H = 1$, respectively. We consider two problem settings: parking with a wide marginal space $\Delta = L = 2$ and a narrow marginal space $\Delta = 0.5$. The marginal space is the distance between the front and rear vehicles in addition to L. For both cases, the rear vehicle center is at (1,0.5), and thus the front vehicle center is at $(1 + 3L/2 + \Delta, 0.5)$. The target area is $\Omega = [1 + L, 1 + L + \Delta] \times [0.5, 0.6] \times [-3^\circ, 3^\circ]$.

The collision area (the center position and orientation of the unicycle that would cause collision with the parked vehicles and the curb) needs to be determined before control synthesis. We assume that vehicles and the curb are rectangles. Then, the collision area can be interpreted by inequalities of the form $g(x) \leq 0$, which is derived by checking if two polyhedra intersect. It is clear that the center of the unicycle has different admissible regions with different orientations. Hence, the collision area is not simply a hyper-rectangle in $\mathbb{R}^3$, as shown in Fig. 2(a). The free configuration space (the admissible position of the unicycle center in $\mathbb{R}^3$) determined by such a constraint can be handled by algorithm (5).

We perform control synthesis for both cases using ROCS [26]. By Corollary 1 (i), if parallel parking is robustly realizable with the given

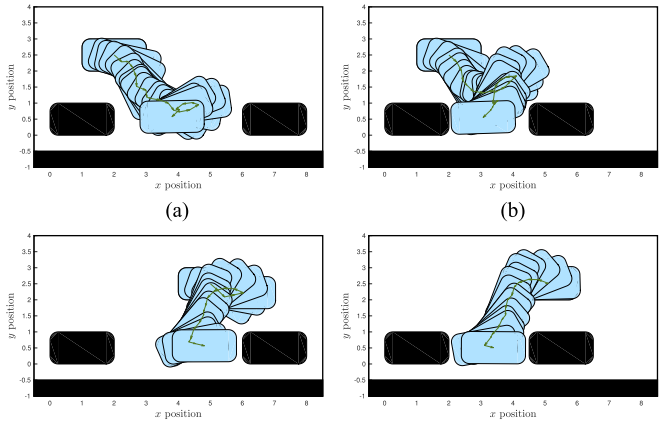


(a) (b)

Fig. 2. Collision area when $\Delta = 0.5$. In (b), the gray area is the xy plane projection of the 3-D collision area, and the two black rectangles represent the bodies of rear and front vehicle. (a) $x - y - \theta$ view. (b) $x - y$ view.

TABLE I
CONTROL SYNTHESIS WITH DIFFERENT PRECISIONS

ε	$\#\mathcal{P}_1$	t_1 (s)	$\#\mathcal{P}_2$	t_2 (s)
0.07	176786	102.93	–	–
0.06	176666	103.19	1797027	295.68
0.02	203166	127.44	1832589	327.50
0.01	274694	176.20	1920929	427.48



(a) (b) (c) (d)

Fig. 3. Controlled parking trajectories from an initial condition (x_0, y_0) with wide and narrow marginal parking spaces. (a) $\Delta = 2$, $(x_0, y_0) = (2, 2.5)$. (b) $\Delta = 0.5$, $(x_0, y_0) = (2, 2.5)$. (c) $\Delta = 2$, $(x_0, y_0) = (5, 2.5)$. (d) $\Delta = 0.5$, $(x_0, y_0) = (5, 2.5)$.

marginal space, we can always synthesize a control strategy using a sufficiently small precision without calculating the Lipschitz constant. To see if the specifications in these two parking scenarios are realizable, we use different precision control parameters. The corresponding control synthesis results regarding the number of partitions ($\#\mathcal{P}_{1,2}$) and the run time ($t_{1,2}$) are summarized in Table I.

For both scenarios, the unicycle can be successfully parked into the target spot from any point of the free configuration space. The controlled parking trajectories with the resulting memoryless control strategies are presented in Fig. 3, which all meet the parallel parking specification.

When the marginal parking space Δ is 0.5, we need a control synthesis precision no greater than 0.06 so that a memoryless control strategy can be generated. Additionally, for this specific example, using a smaller ε only increases the winning set by adding intervals close to the boundary of the free configuration space.

IV. EVALUATION OF TIME COMPLEXITY

To show how well the proposed method performs in terms of computational time, we compare the time complexities of abstraction-based methods and the proposed method. Although theoretical analysis shows the equivalency of both methods in the worst case, the proposed method outperforms abstraction-based methods in solving many of the control synthesis problems practically.

A. Complexity Analysis

Let ε and η ($\varepsilon, \eta > 0$) be the grid size of the state space $\mathcal{X}$ and input space $\mathcal{U}$, respectively. Assume that the cost in terms of run time for each computation of the predecessor is some constant $c > 0$, and $c_1, c_2 > 0$ are some constants related to the width of the state and input space.

For abstraction-based control synthesis based on a uniform partition of the state space, the number of discrete states and inputs are $N_S = \lceil \frac{c_1}{\varepsilon} \rceil^n$ and $N_U = \lceil \frac{c_2}{\eta} \rceil^m$, respectively. Then, the time complexity for computing abstractions is $\mathcal{O}(cN_S N_U)$. Under Assumption 3, the number of transitions N_T is $(\lceil \rho \rceil + 1)^n N_S N_U$. Using the classical co-Büchi algorithm, the time for solving the discrete control synthesis problem is $\mathcal{O}(N_S N_T)$, which yields the overall time complexity of abstraction-based control synthesis

$$\mathcal{O}(cN_S N_U + (\lceil \rho \rceil + 1)^n N_S^2 N_U). \tag{11}$$

We now analyze the time complexity of algorithm (5) via interval computation implemented based on a binary tree data structure. According to the bisection scheme for predecessor approximation, the greatest depth of the binary tree is $h_{\max} = \lceil n \log_2(\frac{c_1}{\varepsilon}) \rceil \approx \log_2 N_S$. Set membership tests are performed by searching the binary tree. Hence, in the worst case where the tree is of depth $h_{\max}$, computation of each predecessor, including membership test, takes approximately $(h_{\max} + c)N_U$ operational time. Let N_G be the number of the set of intervals that represents the target set Ω. Then, the number of intervals outside of Ω is $N_S - N_G$. In the worst case for algorithm (5), the set elements in the sequences $\{Y_i\}_{i=0}^{\infty}$ and $\{X_i^j\}_{j=0}^{\infty}$ differ by one interval. Then, the number of iterations N_I is

$$\sum_{i=0}^{N_G}(i^2 + i) + \sum_{i=0}^{N_S - N_G} i = \frac{N_G^3 + 3N_G^2 + 8N_G}{6} + \frac{(N_S - N_G)^2 + (N_S - N_G)}{2}.$$

If $N_G << N_S$, then $N_I \approx (N_S^2 + N_S)/2$. Hence, the time complexity of the algorithm (5) is of

$$\mathcal{O}\left(\frac{c}{2} N_U N_S^2 + \frac{1}{2} N_U N_S^2 \log_2 N_S\right). \tag{12}$$

By comparing (11) with (12), the time complexity of algorithm (5) is of $\mathcal{O}(N_U N_S^2 \log N_S)$ while the abstraction-based methods is quadratic in N_S. The overhead of algorithm (5) primarily comes from the set inclusion tests by searching the binary tree, i.e., the part induced by $h_{\max}$. When only a high precision is necessary to yield a control strategy, the overhead run time is relatively large, which makes (5) less efficient than abstraction-based methods.

The worst case, however, rarely exists in practical control problems. On the other hand, the use of a nonuniform partitioning scheme avoids partitioning the region in the state space without helping in control synthesis. This usually leads to fewer discrete states for a given precision. In this sense, the proposed method is less sensitive to the state and input discretization precisions than abstraction-based control synthesis methods. Such results from complexity analysis will be shown by the comparison tests in the following section.

From the relationship between system dimension and the time complexity as discussed earlier, the main limitation of the proposed method,

TABLE II
PERFORMANCE COMPARISON TESTS: TO = *TIME OUT* (> 86400 s) AND "–" = *CONTROL SYNTHESIS FAILS*

Examples	Parameters			ROCS			SCOTS				
	n	N_U	ε	N_S	#Iter	time(s)	N_S	N_T	#Iter	time(s)	
										Abst	Syn
DC-DC converter	2	2	0.005	22433	76(529)	0.53	40401	291068	84(671)	0.69	15.90
			0.001	162261	76(272)	3.48	1002001	7243320	77(431)	29.83	481.97
Motion planning	3	49	0.2	280291	381(1)	151.01	91035	3.73×10^7	–	82.80	–
			0.1	1850830	297(1)	1062.97	724271	2.95×10^8	313(2266)	2004.66	17568.2
Parallel parking $\Delta = 0.5$	3	49	0.02	1832589	133(1)	327.50	10075125	TO	TO	TO	TO
Parallel parking $\Delta = 2$			0.07	167155	123(8)	94.32	83025	3.277×10^7	–	73.14	–

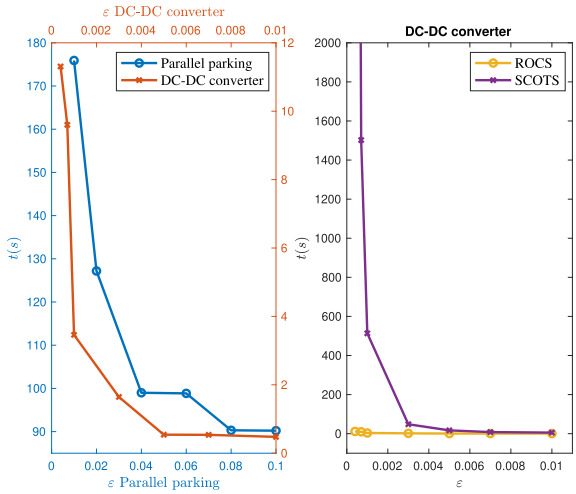


Fig. 4. Changes of run time under different precisions.

which also exists in abstraction-based methods, is that it still suffers the curse of dimensionality.

B. Experimental Tests on Performance

We now compare the performance of our proposed method with abstraction-based methods (implemented in SCOTS [31]) on solving different benchmarking examples. The results on a 3.6 GHz processor (Intel Core i3) are shown in Table II. The column of #Iter indicates the number of outer loops and the total number of inner loops (in the bracket) running (3). The time for abstraction-based control methods is split into the part for abstraction (indicated as *Abst*) and the one for synthesis (as *Syn*).

The state space in the dc–dc converter example (see [27] for the detailed model) is $\mathcal{X} = [0.649, 1.65] \times [0.9898, 1.19]$ and the target region Ω of the reach-and-stay specification $\varphi(\Omega)$ is $[1.1, 1.6] \times [1.08, 1.18]$. While the full setting of the motion planning example can be found in multiple works (e.g., [13], [16]), we consider in our experiments the reach-and-stay control objective instead of just reachability.

Our proposed method outperforms abstraction-based methods in those examples. In the motion planning example, using a grid size of 0.1 succeeds in synthesis while using 0.2 fails for abstraction-based methods because abstractions are more conservative for larger grid size. In contrast, our proposed method solves the problem in 151 s by using $\varepsilon = 0.2$. This is because the minimum width of the partitions can be less than 0.2 by the bisection criterion in [26, Algorithm 1]. As opposed to the motion planning case where obstacles are distributed evenly across the state space, the constraints for parallel parking are highly nonlinear and only posed to a corner of the state space, and varying the discretization precision of the state space will save computational time in a great deal. Such a difference in those two case settings explains why the gain in time efficiency by using our method is more profound in the parallel parking cases.

As seen in (11) and (12), both methods are equivalently sensitive to the size of the discretized systems. The experimental results show that the worst case as in (12) is rather pessimistic in practice and our proposed method is more scalable to the discretization precision than abstraction-based methods. Analyzing the example of dc–dc converter, we can observe in the right-hand side of Fig. 4 that the run time of the proposed method changes slowly while the one for abstraction-based method explodes as precision ε decreases. The left-hand side of Fig. 4 compares the run time of the proposed method for two cases of different sizes, which indicates the dimensionality problem of the proposed method.

V. CONCLUSION

Under mild assumptions, we derived conditions so that reach-and-stay control synthesis is sound and robustly complete for discrete-time nonlinear systems in the sense that control strategies can be found if the specification can be satisfied for the perturbed system. A fixed-point algorithm based on interval computation was proposed as a practical control synthesis method. This is an improvement over abstraction-based methods, which are often not complete for systems without incremental stability. By adaptively partitioning the state space with respect to both dynamics and the given specification, the winning set for the given reach-and-stay problem can be inner-approximated with sufficiently high precision while reducing computational burdens. The efficiency was substantiated by performance tests on several benchmarking examples.

REFERENCES

[1] L. Fribourg and R. Soulat, *Control of Switching Systems by Invariance Analysis: Application to Power Electronics*. Hoboken, NJ, USA: Wiley, 2013.
[2] T. Faulwasser, B. Kern, and R. Findeisen, "Model predictive path-following for constrained nonlinear systems," in *Proc. IEEE Conf. Decis. Control*, 2009, pp. 8642–8647.
[3] P. Nilsson, N. Ozay, and J. Liu, "Augmented finite transition systems as abstractions for control synthesis," *Discrete Event Dyn. Syst.*, vol. 27, no. 2, pp. 301–340, 2017.
[4] F. Blanchini, "Minimum-time control for uncertain discrete-time linear systems," in *Proc. IEEE Conf. Decis. Control*, 1992, pp. 2629–2634.
[5] G. Pin and T. Parisini, "On the robustness of nominal nonlinear minimum-time control and extension to non-robustly controllable target sets," *IEEE Trans. Autom. Control*, vol. 59, no. 4, pp. 863–875, Apr. 2014.
[6] D. P. Bertsekas, "Infinite-time reachability of state-space regions by using feedback control," *IEEE Trans. Autom. Control*, vol. AC-17, no. 5, pp. 604–613, Oct. 1972.
[7] D. Bertsekas and I. Rhodes, "On the minimax reachability of target sets and target tubes," *Automatica*, vol. 7, no. 2, pp. 233–247, 1971.
[8] L. Grüne and J. Pannek, *Nonlinear Model Predictive Control: Theory and Algorithms*. Berlin, Germany: Springer, 2011.

[9] P. Tabuada, *Verification and Control of Hybrid Systems: A Symbolic Approach*. Berlin, Germany: Springer, 2009.
[10] E. Clarke, O. Grumberg, and D. A. Peled, *Model Checking*. Cambridge, MA, USA: MIT Press, 1999.
[11] G. Pola, A. Girard, and P. Tabuada, "Approximately bisimilar symbolic models for nonlinear control systems," *Automatica*, vol. 44, no. 10, pp. 2508–2516, 2008.
[12] A. Girard, G. Pola, and P. Tabuada, "Approximately bisimilar symbolic models for incrementally stable switched systems," *IEEE Trans. Autom. Control*, vol. 55, no. 1, pp. 116–126, Jan. 2010.
[13] M. Zamani, G. Pola, M. Mazo, and P. Tabuada, "Symbolic models for nonlinear control systems without stability assumptions," *IEEE Trans. Autom. Control*, vol. 57, no. 7, pp. 1804–1809, Jul. 2012.
[14] J. Liu, N. Ozay, U. Topcu, and R. Murray, "Synthesis of reactive switching protocols from temporal logic specifications," *IEEE Trans. Autom. Control*, vol. 58, no. 7, pp. 1771–1785, Jul. 2013.
[15] J. Liu and N. Ozay, "Finite abstractions with robustness margins for temporal logic-based control synthesis," *Nonlinear Anal., Hybrid Syst.*, vol. 22, pp. 1–15, 2016.
[16] G. Reissig, A. Weber, and M. Rungger, "Feedback refinement relations for the synthesis of symbolic controllers," *IEEE Trans. Autom. Control*, vol. 62, no. 4, pp. 1781–1796, Apr. 2017.
[17] J. Liu, "Robust abstractions for control synthesis: Completeness via robustness for linear-time properties," in *Proc. Int. Conf. Hybrid Syst., Comput. Control*, 2017, pp. 101–110.
[18] Y. Li and J. Liu, "Robustly complete reach-and-stay control synthesis for switched systems via interval analysis," in *Proc. Annu. Amer. Control Conf.*, 2018, pp. 2350–2355.
[19] K. Hsu, R. Majumdar, K. Mallik, and A.-K. Schmuck, "Multi-layered abstraction-based controller synthesis for continuous-time systems," in *Proc. Int. Conf. Hybrid Syst., Comput. Control*, 2018, pp. 120–129.
[20] L. Jaulin, *Applied Interval Analysis: With Examples in Parameter and State Estimation, Robust Control and Robotics*. Berlin, Germany: Springer, 2001.
[21] P. Collins and A. Goldsztejn, "The reach-and-evolve algorithm for reachability analysis of nonlinear dynamical systems," *Theor. Comput. Sci.*, vol. 223, pp. 87–102, 2008.
[22] J. Wan, J. Vehí, N. Luo, and P. Herrero, "Control of constrained nonlinear uncertain discrete-time systems via robust controllable sets: A modal interval analysis approach," *ESAIM: Control, Optim. Calculus Variations*, vol. 15, no. 1, pp. 189–204, 2009.
[23] A. Girard, G. Gossler, and S. Mouelhi, "Safety controller synthesis for incrementally stable switched systems using multiscale symbolic models," *IEEE Trans. Autom. Control*, vol. 61, no. 6, pp. 1537–1549, Jun. 2016.
[24] S. Rakovic, E. Kerrigan, D. Mayne, and J. Lygeros, "Reachability analysis of discrete-time systems with disturbances," *IEEE Trans. Autom. Control*, vol. 51, no. 4, pp. 546–561, Apr. 2006.
[25] R. T. Rockafellar and R. J.-B. Wets, *Variational Analysis*. Berlin, Germany: Springer, 2009.
[26] Y. Li and J. Liu, "ROCS: A robustly complete control synthesis tool for nonlinear dynamical systems," in *Proc. Int. Conf. Hybrid Syst., Comput. Control*, 2018, pp. 130–135.
[27] Y. Li and J. Liu, "Invariance control synthesis for switched nonlinear systems: An interval analysis approach," *IEEE Trans. Autom. Control*, vol. 63, no. 7, pp. 2206–2211, Jul. 2018.
[28] I. Kolmanovsky and E. G. Gilbert, "Theory and computation of disturbance invariant sets for discrete-time linear systems," *Math. Problems Eng.*, vol. 4, no. 4, pp. 317–367, 1998.
[29] K. J. Astrom and R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton, NJ, USA: Princeton Univ. Press, 2008.
[30] Y. Li and J. Liu, "Robustly complete synthesis of memoryless controllers for nonlinear systems with reach-and-stay specifications," Feb. 2018. [Online]. Available: http://arxiv.org/abs/1802.09082v2
[31] M. Rungger and M. Zamani, "SCOTS: A tool for the synthesis of symbolic controllers," in *Proc. Int. Conf. Hybrid Syst., Comput. Control*, 2016, pp. 99–104.