Robust Invariant Sets Computation for Discrete-Time Perturbed Nonlinear Systems

Bai Xue, Member, IEEE, and Naijun Zhan
1. State Key Lab. of Computer Science, Institute of Software, CAS, Beijing,China
2. Univ. of Chinese Academy of Sciences, Beijing, China
{xuebai,znj}@ios.ac.cn

***Abstract*—In this paper we study the maximal robust invariant set estimation problem for discrete-time perturbed nonlinear systems within the optimal control framework. The maximal robust invariant set of interest is a set of all states such that every possible trajectory starting from it never violates a specified state constraint, regardless of actual disturbances. The maximal robust invariant set is shown to be the zero level set of the unique bounded solution to a Bellman type equation, which is a functional equation being widely used in discrete-time optimal control. Consequently, the maximal robust invariant set estimation problem is reduced to a problem of solving a Bellman type equation. This is the main contribution of this work. The uniqueness of bounded solutions enables us to solve the derived Bellman type equation using numerical methods such as the value iteration and policy iteration, which provide an approximation of the maximal robust invariant set.Finally, two examples demonstrate the performance of our Bellman equation based method.**

***Index Terms*—Discrete-time Perturbed Nonlinear Systems; Maximal Robust Invariant Sets; Bellman Equations**

I. Introduction

The computation of robust invariant sets is central to the validation of systems such as nonlinear dynamical systems and hybrid-state extensions thereof [9], [11]. A robust invariant set of interest in this paper refers to a set of states such that every possible trajectory initialized in it never violates a specified safe state constraint irrespective of the actual disturbance. It has many other names in the literature, e.g., reachability tubes [20] and invariance kernels [2]. Due to its widespread applications in systems and control for stability analysis and control design [5], robust invariant sets generation has been the subject of extensive research over past several decades, e.g., [2], [4], [7], [10], [13], [18], [22].

One popular line for studying robust invariant sets generation problem is by exploiting the link to optimal control problems. When the system is continuous-time, Hamilton-Jacobi equations, which are widely used in optimal control theory [3], are explored for performing reachability analysis, e.g., [2], [14], [16]. From a theoretical point of view, one advantage of this method is that the exact reach sets of interest could be characterized by level sets of viscosity solutions to Hamilton-Jacobi equations. From a computational point of view, there exist well-developed numeric methods [1], [15], [23] for solving Hamilton-Jacobi equations with appropriate number of state variables, rendering possible the gain of exact reach sets. Recently, [24] proposed a Hamilton-Jacobi type equation and characterized the maximal robust invariant set as the zero level set of the unique Lipschitz continuous viscosity solution to this Hamilton-Jacobi type equation for state-constrained continuous-time perturbed systems.

Despite significant progress towards the computation of robust invariant sets for continuous-time systems within the optimal control framework, works on the computation of robust invariant sets for its counterpart, i.e. discrete-time systems, are relatively rare in this regard, especially for discrete-time nonlinear systems. Nonlinear discrete-time systems are widespread in many practical systems such as biological systems and economic systems, where the underlying models are in discrete-time [12]. Moreover, a growing number of digital devices are being used for information processing and control purposes in a variety of complex systems applications, including industrial processes, power networks and communication networks. For these applications, it is reasonable to model the system using a discrete-time nonlinear state space model [17]. Thus, the study of robust invariant sets for discrete-time nonlinear systems is conducive to improving these applications, as mentioned before.

In this paper we study the maximal robust invariant set estimation for discrete-time perturbed nonlinear systems in the framework of optimal control. We firstly define a bounded value function with a positive-valued parameter falling between zero and one such that its zero sub-level set equals the maximal robust invariant set. Then the value function is reduced to a bounded solution to a Bellman type equation. The Bellman equation, named after Richard Bellman, is a functional equation being widely used in discrete-time optimal control [3]. When the value of the positive-valued parameter is strictly less than one, the bounded solution to the Bellman type equation is unique. Both the value iteration and the policy iteration are capable of solving the equation, thereby enabling the gain of an estimation of the maximal robust invariant set. The estimation is just an approximation of the maximal robust invariant set. It is neither an outer-approximation nor an inner-approximation as in [19]. That is, trajectories starting from it may violate the specified state constraint and thus the invariance property of interest in this paper may not be guaranteed generally. Finally, two examples demonstrate the performance of our Bellman equation based method.

The works mostly close to the present one are [26] and [25]. The maximal robust non-termination set is equivalent to the maximal robust invariant set in this paper when the computer program in [26] is transformed into a discrete-time nonlinear system. The main difference between this work and [26] is that the derived Bellman type equation in this work has a unique bounded solution, thereby facilitating the use of existing numerical methods such as the value iteration and policy iteration to estimate the maximal robust invariant set. However, the equation in [26] does not feature this uniqueness property. A resulting consequence is that the value iteration and policy iteration for solving the equation in [26] cannot guarantee to terminate generally and thus we cannot obtain an estimation of the maximal robust invariant set. The work [25] presents a Bellman type equation for computing the maximal robust region of attraction. The maximal robust region of attraction is a set of all states such that every trajectory initialized in it will approach an equilibrium while never violating a specified state constraint, regardless of the actual perturbation. It differs from the maximal robust invariant set considered in this paper in that the maximal robust invariant set is not required to contain equilibria and trajectories starting from it are not required to approach an equilibrium. Recently, a Bellman equation based method was proposed for computing finite time horizon maximal invariant sets in [8]. Unlike [8], the present work focuses on generating the maximal robust invariant set over the infinite time horizon rather than finite time horizons.

The main contribution of this paper is summarized below:

1) The maximal robust invariant set is shown to be the zero level set of the *unique bounded* solution to a Bellman type equation. To our knowledge this is the first work on linking maximal robust invariant sets with Bellman type equations exhibiting unique bounded solutions.
2) Both the value iteration and policy iteration are capable of resolving the derived Bellman type equation for obtaining an estimation of the maximal robust invariant set. Due to the merits that some nice properties such as the rate of convergence and the maximum iteration number can be obtained a priori for the value iteration, we in this paper use the value iteration as the main tool to illustrate our Bellman equation based method. The convergence rate of the value iteration is linear and the same for all discrete-time nonlinear systems. This convergence property is generally not attainable for existing methods. Furthermore, the convergence rate is adjustable and can achieve a nearly superlinear one, enabling the value iteration to outperform the policy iteration when the convergence rate is close enough to the accuracy tolerance.

II. Preliminaries

In this section we describe discrete-time perturbed nonlinear systems, and maximal robust invariant sets of interest in this paper.

A. Problem Description

Before posing the problem studied, let us introduce some basic notions used in the rest of this paper: $\mathbb{N}$ stands for the set of nonnegative integers and $\mathbb{R}$ for the set of real numbers; Vectors are denoted by boldface letters.

In this paper we consider a discrete-time perturbed nonlinear system (abbr., DPNS), which could be modeled by iterative nonlinear maps of the following form:

$$\begin{aligned} &\boldsymbol{x}(l+1) = \boldsymbol{f}(\boldsymbol{x}(l), \boldsymbol{d}(l)), \forall l \in \mathbb{N}, \\ &\boldsymbol{x}(0) = \boldsymbol{x}_0, \end{aligned} \tag{1}$$

where $\boldsymbol{x}(\cdot) : \mathbb{N} \to \mathbb{R}^n$ are states, $\boldsymbol{d}(\cdot) : \mathbb{N} \to D$ are the disturbances with D being a compact set in $\mathbb{R}^m$, and $\boldsymbol{f}(\cdot,\cdot) : \mathbb{R}^n \times D \to \mathbb{R}^n$.

Remark 1. *It is worth remarking here that the continuity on the vector function $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{d})$ in* (1) *is not required. It is allowed to be discontinuous.*

In the following we define disturbance policies and their associated trajectories for DPNS.

Definition 1. *A disturbance policy π for* DPNS *is a sequence $(\boldsymbol{d}(l))_{l\in\mathbb{N}}$, where $\boldsymbol{d}(\cdot) : \mathbb{N} \to D$. Furthermore, we define Π as the set of all disturbance policies.*

Definition 2. *Given an initial state $\boldsymbol{x}_0 \in \mathbb{R}^n$ and a disturbance policy $\pi = (\boldsymbol{d}(l))_{l\in\mathbb{N}}$, the trajectory of* DPNS*, induced by $\boldsymbol{x}_0$ and π, is a sequence $(\boldsymbol{\phi}_{\boldsymbol{x}_0}^{\pi}(l))_{l\in\mathbb{N}}$ satisfying $\boldsymbol{\phi}_{\boldsymbol{x}_0}^{\pi}(l+1) = \boldsymbol{f}(\boldsymbol{\phi}_{\boldsymbol{x}_0}^{\pi}(l), \boldsymbol{d}(l))$ for $l \in \mathbb{N}$.*

Now, we define the maximal robust invariant set such that DPNS starting from it never leaves the state constraint set $X = \{\boldsymbol{x} \in \mathbb{R}^n \mid \bigwedge_{i=1}^{n_0} h_i(\boldsymbol{x}) \leq 0\}$, where $h_i(\cdot) : \mathbb{R}^n \to \mathbb{R}$ for $i = 1, \dots, n_0$.

Remark 2. *Similar to Remark 1, the continuity on functions $h_i(\boldsymbol{x})$ defining the set X is not required, $i = 1, \dots, n_0$. They are allowed to be discontinuous as well.*

Definition 3 (Maximal Robust Invariant Set)**.** *The maximal robust invariant set $\mathcal{R}_0$ is the set of all states in the state constraint set X such that every possible trajectory of* DPNS *starting from it never leaves X, i.e.*

$$\mathcal{R}_0 = \{\boldsymbol{x}_0 \in X \mid \boldsymbol{\phi}_{\boldsymbol{x}_0}^{\pi}(l) \in X, \forall l \in \mathbb{N}, \forall \pi \in \Pi\}.$$

Consequently, a robust invariant set is a subset of the maximal robust invariant set $\mathcal{R}_0$.

It is worth remarking here that a robust invariant set in Definition 3 is not an invariant set in the classical sense. Unlike invariant sets in the classical sense in which trajectories get trapped, it is a set of states such that every possible trajectory of DPNS starting from it does not leave the state constraint set X. Thus, trajectories starting from a robust invariant set in Definition 3 may leave it but cannot leave X.

The focus of this paper is on the computation of the maximal robust invariant set $\mathcal{R}_0$. In the rest of this paper we assume that the interior of the maximal robust invariant set $\mathcal{R}_0$ exists for DPNS.

Assumption 1. *$\mathcal{R}_0^{\circ} \neq \emptyset$, where $\mathcal{R}_0^{\circ}$ is the interior of the maximal robust invariant set $\mathcal{R}_0$.*

III. Computing Maximal Robust Invariant Sets

In this section we present our approach of synthesizing the maximal robust invariant set for DPNS.

In our approach we reduce the maximal robust invariant set $\mathcal{R}_0$ as the zero (sub)level set of the bounded solution $v(\boldsymbol{x}) : \mathbb{R}^n \to \mathbb{R}$ to a Bellman type equation of the following form:

$$\min\Big\{\inf_{\boldsymbol{d}\in D}(v(\boldsymbol{x})-\alpha v(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d}))),\\ v(\boldsymbol{x})-\max_{j\in\{1,\dots,n_0\}} h'_j(\boldsymbol{x})\Big\}=0, \tag{2}$$

where $h'_j(\boldsymbol{x})=\frac{h_j(\boldsymbol{x})}{1+h_j^2(\boldsymbol{x})}$ and $\alpha\in(0,1]$ is a user-specified constant. When $\alpha\in(0,1)$, the bounded solution to (2) is unique. The proofs of the aforementioned claims will be presented afterwards. The uniqueness of bounded solutions facilitates the use of both the value iteration and the policy iteration in reinforcement learning to solve (2). A question arises naturally: *how can the solution to the equation* (2) *be obtained?*

In order to solve the above mentioned question, we first present a value function constructed based on a scalar value $\alpha\in(0,1]$ and the family of functions $(h_j(\boldsymbol{x}))_{j=1}^{n_0}$ defining the state constraint set X. Then, based on its underlying property of satisfying the dynamic programming principle as shown in Lemma 1, this value function finally boils down to the unique bounded solution to (2) when $\alpha\in(0,1)$, which is formally formulated in Theorem 1.

The value function $V:\mathbb{R}^n\to\mathbb{R}$ is defined by:

$$V(\boldsymbol{x}):=\sup_{\pi\in\Pi}\sup_{l\in\mathbb{N}}\max_{j\in\{1,\dots,n_0\}}\big\{\alpha^l h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi}(l))\big\}, \tag{3}$$

where $\alpha\in(0,1]$ is a user-specified constant. Obviously,

$$-1<h'_j(\boldsymbol{x})<1$$

over $\boldsymbol{x}\in\mathbb{R}^n$ for $j=1,\dots,n_0$. Thus,

$$-1\le V(\boldsymbol{x})\le 1,\forall \boldsymbol{x}\in\mathbb{R}^n.$$

Unlike [26], we introduce a positive-valued parameter α and new bounded functions $(h'_j(\boldsymbol{x}))_{j=1}^{n_0}$ into the construction of the value function (3). This enables us to reduce $V(\boldsymbol{x})$ to the unique bounded solution to (2) when $\alpha\in(0,1)$. When $\alpha=1$, $V(\boldsymbol{x})$ in (3) is a bounded solution to (2). As discussed in [26], the uniqueness of bounded solutions to (2) with $\alpha=1$ cannot be guaranteed and thus we cannot obtain the maximal robust invariant set via solving (2) generally.

The following proposition shows the relationship between the value function V and the maximal robust invariant set $\mathcal{R}_0$, The zero level set of $V(\boldsymbol{x})$ is equal to the maximal robust invariant set $\mathcal{R}_0$ when $\alpha\in(0,1)$, and the zero sub-level set of $V(\boldsymbol{x})$ is equal to $\mathcal{R}_0$ when $\alpha=1$.

Proposition 1. $\mathcal{R}_0=\{\boldsymbol{x}\in\mathbb{R}^n\mid V(\boldsymbol{x})\le 0\}$, *where $\mathcal{R}_0$ is the maximal robust invariant set in Definition 3. Furthermore, when $\alpha\in(0,1)$, $V(\boldsymbol{x})\ge 0$ for $\boldsymbol{x}\in\mathbb{R}^n$ and thus $\mathcal{R}_0=\{\boldsymbol{x}\in\mathbb{R}^n\mid V(\boldsymbol{x})=0\}$.*

Proof. Let $\boldsymbol{y}\in\mathcal{R}_0$. According to Definition 3, we have

$$h_j(\boldsymbol{\phi}_{\boldsymbol{y}}^{\pi}(i))\le 0,\forall i\in\mathbb{N},\forall\pi\in\Pi,\forall j\in\{1,\dots,n_0\} \tag{4}$$

holds, implying that

$$h'_j(\boldsymbol{\phi}_{\boldsymbol{y}}^{\pi}(i))\le 0,\forall i\in\mathbb{N},\forall\pi\in\Pi,\forall j\in\{1,\dots,n_0\}$$

and thus $V(\boldsymbol{y})\le 0$. Therefore, $\boldsymbol{y}\in\{\boldsymbol{x}\mid V(\boldsymbol{x})\le 0\}$.

On the other hand, if $\boldsymbol{y}\in\{\boldsymbol{x}\in\mathbb{R}^n\mid V(\boldsymbol{x})\le 0\}$, then $V(\boldsymbol{y})\le 0$, implying that (4) holds and consequently $\boldsymbol{y}\in\mathcal{R}_0$. Therefore, $\mathcal{R}_0=\{\boldsymbol{x}\in\mathbb{R}^n\mid V(\boldsymbol{x})\le 0\}$.

As to the case of $\alpha\in(0,1)$, it is obvious that $V(\boldsymbol{x})\ge 0$ for $\boldsymbol{x}\in\mathbb{R}^n$ since

$$\lim_{l\to\infty}\alpha^l\max_{j\in\{1,\dots,n_0\}}\{h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi}(l))\}=0,\forall\boldsymbol{x}\in\mathbb{R}^n,\forall\pi\in\Pi.$$

Therefore, $\mathcal{R}_0=\{\boldsymbol{x}\in\mathbb{R}^n\mid V(\boldsymbol{x})=0\}$ if $\alpha\in(0,1)$. □

Proposition 1 tells us that the maximal robust invariant set $\mathcal{R}_0$ can be obtained if the value function $V(\boldsymbol{x})$ in (3) is computed. However, it is too challenging to tackle the value function $V(\boldsymbol{x})$ in (3) directly, since it involves the manipulation of trajectories for DPNS over the infinite time horizon. Therefore, we reduce it to the unique bounded solution to the Bellman type equation (2) when $\alpha\in(0,1)$ and thus the problem of computing the maximal robust invariant set is reduced to solving the equation (2). The link between the value function (3) and the equation (2) is established via the following dynamic programming principle.

Lemma 1. *For $\boldsymbol{x}\in\mathbb{R}^n$ and $l\in\mathbb{N}$, we have:*

$$V(\boldsymbol{x})=\sup_{\pi\in\Pi}\max\big\{\alpha^l V(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi}(l)),\\ \sup_{i\in[0,l)\cap\mathbb{N}}\max_{j\in\{1,\dots,n_0\}}\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi}(i))\big\}. \tag{5}$$

Proof. Let

$$W(l,\boldsymbol{x}):=\sup_{\pi\in\Pi}\max\big\{\alpha^l V(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi}(l)),\\ \sup_{i\in[0,l)\cap\mathbb{N}}\max_{j\in\{1,\dots,n_0\}}\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi}(i))\big\}.$$

We will prove that for $\epsilon>0$, $|W(l,\boldsymbol{x})-V(\boldsymbol{x})|<\epsilon$.

According to the definition of $V(\boldsymbol{x})$, i.e., (3), for any ϵ_1, there exists an input policy $\pi'=(\boldsymbol{d}'(i))_{i\in\mathbb{N}}\in\Pi$ such that

$$V(\boldsymbol{x})\le\sup_{i\in\mathbb{N}}\max_{j\in\{1,\dots,n_0\}}\{\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi'}(i))\}+\epsilon_1.$$

We then introduce two input policies $\pi_1=(\boldsymbol{d}_1(i))_{i\in\mathbb{N}}\in\Pi$ and $\pi_2=(\boldsymbol{d}_2(i))_{i\in\mathbb{N}}\in\Pi$ with $\boldsymbol{d}_1(j)=\boldsymbol{d}'(j)$ for $j=0,\dots,l-1$ and $\boldsymbol{d}_2(j)=\boldsymbol{d}'(j+l)$ for $j\in\mathbb{N}$ respectively. Thus we obtain that

$$\begin{aligned}W(l,\boldsymbol{x})&\ge\max\big\{\alpha^l V(\boldsymbol{y}),\sup_{i\in[0,l)\cap\mathbb{N}}\max_{j\in\{1,\dots,n_0\}}\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi_1}(i))\big\}\\ &\ge\max\Big\{\sup_{i\in[l,+\infty)\cap\mathbb{N}}\max_{j\in\{1,\dots,n_0\}}\{\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{y}}^{\pi_2}(i-l))\},\\ &\qquad\sup_{i\in[0,l)\cap\mathbb{N}}\max_{j\in\{1,\dots,n_0\}}\{\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi_1}(i))\}\Big\}\\ &=\max\Big\{\sup_{i\in[l,+\infty)\cap\mathbb{N}}\max_{j\in\{1,\dots,n_0\}}\{\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi'}(i))\},\\ &\qquad\sup_{i\in[0,l)\cap\mathbb{N}}\max_{j\in\{1,\dots,n_0\}}\{\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi'}(i))\}\Big\}\\ &=\sup_{i\in\mathbb{N}}\max_{j\in\{1,\dots,n_0\}}\{\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi'}(i))\}\\ &\ge V(\boldsymbol{x})-\epsilon_1,\end{aligned}$$

where $\boldsymbol{y}=\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi_1}(l)$. Therefore,

$$V(\boldsymbol{x}) \leq W(l,\boldsymbol{x})+\epsilon_1. \tag{6}$$

On the other hand, by the definition of $W(l,\boldsymbol{x})$, for every $\epsilon_1>0$, there exists $\pi_1=(\boldsymbol{d}_1(i))_{i\in\mathbb{N}}\in\Pi$ such that

$$W(l,\boldsymbol{x}) \leq \max\big\{\alpha^l V(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi_1}(l)), \sup_{i\in[0,l)\cap\mathbb{N}} \max_{j\in\{1,\dots,n_0\}}\{\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi_1}(i))\}\big\}+\epsilon_1.$$

Also, by the definition of $V(\boldsymbol{x})$, i.e., (3), for every $\epsilon_1>0$, there exists $\pi_2=(\boldsymbol{d}_2(i))_{i\in\mathbb{N}}\in\Pi$ such that

$$V(\boldsymbol{y}) \leq \sup_{i\in\mathbb{N}} \max_{j\in\{1,\dots,n_0\}}\{\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{y}}^{\pi_2}(i))\}+\epsilon_1,$$

where $\boldsymbol{y}=\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi_1}(l)$. We define $\pi=(\boldsymbol{d}(i))_{i\in\mathbb{N}}$ such that $\boldsymbol{d}(i)=\boldsymbol{d}_1(i)$ for $i=0,\dots,l-1$ and $\boldsymbol{d}(i+l)=\boldsymbol{d}_2(i)$ for $i\in\mathbb{N}$. Obviously, $\pi\in\Pi$. Then, it follows

$$\begin{aligned}W(l,\boldsymbol{x}) \leq& \max\Big\{\sup_{i\in\mathbb{N}\cap[l,\infty)} \max_{j\in\{1,\dots,n_0\}}\{\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{y}}^{\pi_2}(i-l))\},\\ &\sup_{i\in[0,l)\cap\mathbb{N}} \max_{j\in\{1,\dots,n_0\}}\{\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi_1}(i))\}\Big\}+2\epsilon_1\\ \leq& \sup_{i\in[0,+\infty)\cap\mathbb{N}} \max_{j\in\{1,\dots,n_0\}}\{\alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi}(i))\}+2\epsilon_1\\ \leq& V(\boldsymbol{x})+2\epsilon_1.\end{aligned}$$

Therefore,

$$V(\boldsymbol{x}) \geq W(l,\boldsymbol{x})-2\epsilon_1. \tag{7}$$

Combining (6) and (7), we finally have $|V(\boldsymbol{x})-W(l,\boldsymbol{x})|\leq\epsilon=2\epsilon_1$. Since ϵ_1 is arbitrary, $V(\boldsymbol{x})=W(l,\boldsymbol{x})$ holds for $\boldsymbol{x}\in\mathbb{R}^n$ and $l\in\mathbb{N}$. This completes the proof. ☐

Based on Lemma 1, we derive (2), to which $V(\boldsymbol{x})$ is a unique bounded solution when $\alpha\in(0,1)$.

Theorem 1. *If $\alpha\in(0,1]$, the value function $V(\boldsymbol{x}):\mathbb{R}^n\to\mathbb{R}$ in (3) is a bounded solution to the Bellman type equation (2). Moreover, $V(\boldsymbol{x})$ is the unique bounded solution to (2) when $\alpha\in(0,1)$.*

Proof. When $l=1$, (5) is reduced to

$$V(\boldsymbol{x})=\sup_{\pi\in\Pi}\max\big\{\alpha V(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi}(1)), \sup_{i\in[0,1)\cap\mathbb{N}} \max_{j\in\{1,\dots,n_0\}} \alpha^i h'_j(\boldsymbol{\phi}_{\boldsymbol{x}}^{\pi}(i))\big\},$$

which is further equivalent to

$$V(\boldsymbol{x})=\sup_{\boldsymbol{d}\in D}\max\big\{\alpha V(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d})), \max_{j\in\{1,\dots,n_0\}} h'_j(\boldsymbol{x})\big\}.$$

Thus, (2) is the special case of (5) when $l=1$. In the rest we just prove the statement that $V(\boldsymbol{x})$ is the unique bounded solution to (2) when $\alpha\in(0,1)$.

Assume that $U(\boldsymbol{x})$ is a bounded solution to (2) as well, and there exists $\boldsymbol{y}\in\mathbb{R}^n$ such that $U(\boldsymbol{y})\neq V(\boldsymbol{y})$. Without loss of generality, we assume that $U(\boldsymbol{y})<V(\boldsymbol{y})$, i.e., there exists $\delta>0$ such that $V(\boldsymbol{y})-U(\boldsymbol{y})=\delta$.

Since $U(\boldsymbol{y})-\max_{j\in\{1,\dots,n_0\}} h'_j(\boldsymbol{y})\geq 0$,

$$V(\boldsymbol{y})-\max_{j\in\{1,\dots,n_0\}} h'_j(\boldsymbol{y})>0$$

holds. Consequently, we have that $V(\boldsymbol{y})=\alpha\sup_{\boldsymbol{d}\in D}V(\boldsymbol{f}(\boldsymbol{y},\boldsymbol{d}))$. Also, due to the fact that $U(\boldsymbol{y})\geq\alpha\sup_{\boldsymbol{d}\in D}U(\boldsymbol{f}(\boldsymbol{y},\boldsymbol{d}))$,

$$\alpha\sup_{\boldsymbol{d}\in D}V(\boldsymbol{f}(\boldsymbol{y},\boldsymbol{d}))-\alpha\sup_{\boldsymbol{d}\in D}U(\boldsymbol{f}(\boldsymbol{y},\boldsymbol{d}))\geq\delta$$

holds, implying that

$$\alpha\sup_{\boldsymbol{d}\in D}(V(\boldsymbol{f}(\boldsymbol{y},\boldsymbol{d}))-U(\boldsymbol{f}(\boldsymbol{y},\boldsymbol{d})))\geq\delta.$$

Therefore, for $1<\beta<\frac{1}{\alpha}$, there exists $\boldsymbol{d}\in D$ such that

$$\alpha(V(\boldsymbol{f}(\boldsymbol{y},\boldsymbol{d}))-U(\boldsymbol{f}(\boldsymbol{y},\boldsymbol{d})))\geq\beta\alpha\delta.$$

Let $\boldsymbol{d}_1$ satisfy

$$V(\boldsymbol{f}(\boldsymbol{y},\boldsymbol{d}_1))-U(\boldsymbol{f}(\boldsymbol{y},\boldsymbol{d}_1))\geq\beta\delta,$$

and $\boldsymbol{y}_1=\boldsymbol{f}(\boldsymbol{y},\boldsymbol{d}_1)$. It is obvious that

$$V(\boldsymbol{y}_1)-U(\boldsymbol{y}_1)\geq\beta\delta.$$

Repeating the above procedure, we can construct a sequence $(\boldsymbol{y}_j)_{j=1}^{\infty}$ satisfying $V(\boldsymbol{y}_j)-U(\boldsymbol{y}_j)\geq\beta^j\delta$. Thus,

$$V(\boldsymbol{y}_j)\geq U(\boldsymbol{y}_j)+\beta^j\delta, \forall j\in\mathbb{N}.$$

Since U is bounded over $\mathbb{R}^n$ and $\lim_{j\to\infty}\beta^j\delta=\infty$, we have that $V(\boldsymbol{y}_j)$ approach infinity when j tends to infinity, contradicting that V is bounded over $\mathbb{R}^n$. Thus, $V(\boldsymbol{y})=U(\boldsymbol{y})$. Based on the above deduction technique, a similar contradiction can be obtained for the case when $U(\boldsymbol{y})>V(\boldsymbol{y})$.

Thus, this concludes that the value function $V(\boldsymbol{x}):\mathbb{R}^n\to\mathbb{R}$ in (3) is the unique bounded solution to (2). ☐

Remark 3. *In this paper we take the function $h'_j(x)$ of the particular form $\frac{h_j(\boldsymbol{x})}{1+h_j^2(\boldsymbol{x})}$, $j=1,\dots,n_0$. Actually, it can take various forms such as $\frac{h_j(\boldsymbol{x})}{1+|h_j(\boldsymbol{x})|}$ which can still make Proposition 1, Lemma 1 and Theorem 1 hold, but it should satisfy the following properties:*

1) $\{\boldsymbol{x}\mid\bigwedge_{j=1}^{n_0}h'_j(\boldsymbol{x})\leq 0\}=\{\boldsymbol{x}\mid\bigwedge_{j=1}^{n_0}h_j(\boldsymbol{x})\leq 0\}=X$;
2) *$h'_j(\boldsymbol{x})$ is bounded over $\mathbb{R}^n$, $j=1,\dots,n_0$.*

The difference with various $h'_j(\boldsymbol{x})$'s in computing the maximal robust invariant set will be investigated as the future work.

From Theorem 1, we conclude that the maximal robust invariant set $\mathcal{R}_0$ can be obtained by solving (2). A technique for solving (2) with $\alpha\in(0,1)$ is the value iteration in the framework of reinforcement learning.

Theorem 2. *Suppose the sequence of functions $(V_i(\boldsymbol{x}))_{i\in\mathbb{N}}$ with $V_i(\cdot):\mathbb{R}^n\to\mathbb{R}$ is generated by the value iteration starting from some bounded function $V_0:\mathbb{R}^n\to\mathbb{R}$ according to*

$$V_{i+1}(\boldsymbol{x})=\max\big\{\sup_{\boldsymbol{d}\in D}\alpha V_i(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d})), \max_{j\in\{1,\dots,n_0\}} h'_j(\boldsymbol{x})\big\} \tag{8}$$

for $\boldsymbol{x}\in\mathbb{R}^n$ and $i\in\mathbb{N}$, then $V_i(\boldsymbol{x})$ uniformly approximates $V(\boldsymbol{x})$ over $\mathbb{R}^n$ if $\alpha\in(0,1)$ as i tends to infinity, where $V(\boldsymbol{x})$ is the unique bounded solution to (2).

Proof. According to (8), we have

$$\begin{aligned}
&V_{i+1}(\boldsymbol{x})-V_i(\boldsymbol{x})\\
&=\max\Big\{\alpha\sup_{\boldsymbol{d}_i\in D}V_i\big(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d}_i)\big),\max_{j\in\{1,\dots,n_0\}}h'_j(\boldsymbol{x})\Big\}-\\
&\quad\max\Big\{\alpha\sup_{\boldsymbol{d}_i\in D}V_{i-1}\big(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d}_i)\big),\max_{j\in\{1,\dots,n_0\}}h'_j(\boldsymbol{x})\Big\}\\
&\leq\max\Big\{\alpha\sup_{\boldsymbol{d}_i\in D}\Big(V_i\big(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d}_i)\big)-V_{i-1}\big(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d}_i)\big)\Big),0\Big\}\\
&\leq\max\Big\{\alpha^i\sup_{\boldsymbol{d}_1\in D}\cdots\sup_{\boldsymbol{d}_i\in D}\Big(V_1\big(\boldsymbol{g}(\boldsymbol{x},\boldsymbol{d}_1,\cdots,\boldsymbol{d}_i)\big)-\\
&\qquad\qquad V_0\big(\boldsymbol{g}(\boldsymbol{x},\boldsymbol{d}_1,\cdots,\boldsymbol{d}_i)\big)\Big),0\Big\}
\end{aligned}\tag{9}$$

and

$$\begin{aligned}
&V_{i+1}(\boldsymbol{x})-V_i(\boldsymbol{x})\\
&=-\min\Big\{\alpha\inf_{\boldsymbol{d}_i\in D}-V_i\big(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d}_i)\big),-\max_{j\in\{1,\dots,n_0\}}h'_j(\boldsymbol{x})\Big\}\\
&+\min\Big\{\alpha\inf_{\boldsymbol{d}_i\in D}-V_{i-1}\big(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d}_i)\big),-\max_{j\in\{1,\dots,n_0\}}h'_j(\boldsymbol{x})\Big\}\\
&\geq\min\Big\{\alpha\inf_{\boldsymbol{d}_i\in D}\Big(V_i\big(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d}_i)\big)-V_{i-1}\big(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d}_i)\big)\Big),0\Big\}\\
&\geq\min\Big\{\alpha^i\inf_{\boldsymbol{d}_1\in D}\cdots\inf_{\boldsymbol{d}_i\in D}\Big(V_1\big(\boldsymbol{g}(\boldsymbol{x},\boldsymbol{d}_1,\cdots,\boldsymbol{d}_i)\big)-\\
&\qquad\qquad V_0\big(\boldsymbol{g}(\boldsymbol{x},\boldsymbol{d}_1,\cdots,\boldsymbol{d}_i)\big)\Big),0\Big\},
\end{aligned}\tag{10}$$

where

$$\boldsymbol{g}(\boldsymbol{x},\boldsymbol{d}_1,\cdots,\boldsymbol{d}_i)=\underbrace{\boldsymbol{f}\Big(\boldsymbol{f}\Big(\cdots\boldsymbol{f}\big(\boldsymbol{f}}_{i}(\boldsymbol{x},\boldsymbol{d}_i),\boldsymbol{d}_{i-1}\big),\cdots,\boldsymbol{d}_2\Big),\boldsymbol{d}_1\Big).$$

Moreover, since V_0 and $\max_{j\in\{1,\dots,n_0\}}h'_j$ are bounded over $\mathbb{R}^n$, therefore, V_1 is bounded as well. Thus, according to (9), (10) and $\alpha\in(0,1)$, we have that $V_i(\boldsymbol{x})$ uniformly approximates a function $V'(\boldsymbol{x})$ over $\mathbb{R}^n$ as i tends to infinity. In the rest we just need to prove that $V'(\boldsymbol{x})=V(\boldsymbol{x})$ over $\boldsymbol{x}\in\mathbb{R}^n$. This conclusion can be assured by replacing $V_{i+1}(\boldsymbol{x})-V_i(\boldsymbol{x}$ in (9) and (10) with $V_{i+1}(\boldsymbol{x})-V(\boldsymbol{x})$, resulting in that $V_{i+1}(\boldsymbol{x})$ uniformly approximates $V(\boldsymbol{x})$ over $\mathbb{R}^n$ as i tends to infinity, where $V(\boldsymbol{x})=\max\{\alpha\sup_{\boldsymbol{d}\in D}V(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d})),\max_{j\in\{1,\dots,n_0\}}h'_j(\boldsymbol{x})\}$. □

Remark 4. *From Theorem 2, we observe that the initial function $V_0(\boldsymbol{x})$ for the value iteration* (8) *can be an arbitrary bounded function from $\mathbb{R}^n$ to $\mathbb{R}$. Let $|V_0(\boldsymbol{x})|\leq M$ for $\boldsymbol{x}\in\mathbb{R}^n$ and $\alpha\in(0,1)$, where $M\geq 0$. From* (9) *and* (10) *we can obtain that*

$$\sup_{\boldsymbol{x}\in\mathbb{R}^n}|V_{i+1}(\boldsymbol{x})-V_i(\boldsymbol{x})|\leq\alpha^i\max\{2M,1+M\}.\tag{11}$$

Consequently, $V_i(\boldsymbol{x})$ converges to $V(\boldsymbol{x})$ over $\mathbb{R}^n$ with the rate of convergence α. This also implies that the smaller the rate of convergence α is, the faster the convergence is. This will be reflected in the experimental section as well.

In addition, (11) *indicates that a smaller M results in a faster convergence to the unique bounded solution to* (2). *Thus, $M=0$, i.e., $V_0(\boldsymbol{x})\equiv 0$ for $\boldsymbol{x}\in\mathbb{R}^n$, is the best choice. This claim holds when $V(\boldsymbol{x})$ in* (3) *is unknown. Due to the fact that $V(\boldsymbol{x})=\max\{\alpha\sup_{\boldsymbol{d}\in D}V(\boldsymbol{f}(\boldsymbol{x},\boldsymbol{d})),\max_{j\in\{1,\dots,n_0\}}h'_j(\boldsymbol{x})\}$, we have that $V_0(\boldsymbol{x})=V(\boldsymbol{x})$ for $\boldsymbol{x}\in\mathbb{R}^n$ is the best choice if $V(\boldsymbol{x})$ is known.*

The value iteration for addressing (2) with $\alpha\in(0,1)$ is described in Alg. 1.

Algorithm 1 The value iteration for solving (2)

```
1) Set $V_0(\boldsymbol{x}) := 0$ over $\boldsymbol{x}\in\mathbb{R}^n$ and $k := 0$, and decide on
   a grid $\Lambda=\{\boldsymbol{x}_1,\dots,\boldsymbol{x}_N\}$ on $X$ for the state variable $\boldsymbol{x}$,
   and a grid $\Delta=\{\boldsymbol{d}_1,\dots,\boldsymbol{d}_M\}$ in $D$ for the disturbance
   variable $\boldsymbol{d}$.
2) Choose a value in $(0,1)$ for $\alpha$;
3) Choose an accuracy tolerance $\epsilon > 0$;
4) For each $\boldsymbol{x}_i\in\Lambda$, $i=1,\dots,N$, compute
       $\boldsymbol{x}'_{i,j}=\boldsymbol{f}(\boldsymbol{x}_i,\boldsymbol{d}_j), j=1,\dots,M,$
   then compute an interpolated value function at each
   $\boldsymbol{x}'_{i,j}$ : $\tilde{V}_k(\boldsymbol{x}'_{i,j})$ and compute
       $V_{k+1}(\boldsymbol{x}_i)=\max\{\alpha\max_{j\in\{1,\dots,M\}}\tilde{V}_k(\boldsymbol{x}'_{i,j}),$
                 $\max_{j\in\{1,\dots,n_0\}}h'_j(\boldsymbol{x}_i)\}.$
5) If $\max_{\boldsymbol{x}\in\Lambda}|V_{k+1}(\boldsymbol{x})-V_k(\boldsymbol{x})| < \epsilon$, go to step 6);
   otherwise, $k := k+1$ and go back to 4);
6) Obtain the final solution $V(\boldsymbol{x})$ as $V(\boldsymbol{x})\approx V_{k+1}(\boldsymbol{x})$.
```

Remark 5. *From* (11) *we can obtain that given the accuracy tolerance ϵ, the maximum iteration number in Alg. 1 is predictable a priori, i.e., the maximum iteration number is less than or equal to* $\max\{\lceil\log_\alpha\epsilon\rceil,1\}$*, where $\lceil\log_\alpha\epsilon\rceil$ is the smallest integer being larger than or equal to $\log_\alpha\epsilon$.*

The termination of the value iteration algorithm described in Alg. 1 is guaranteed by Theorem 2. In this way the value of $V(\boldsymbol{x})$ can be calculated on the grid points of the set X.

When $\alpha=1$, we cannot guarantee the convergence of $(V_i)_{i\in\mathbb{N}}$ in (8). This is also reflected in Remark 4. Even if the sequence $(V_i)_{i\in\mathbb{N}}$ converges, the convergence to $V(\boldsymbol{x})$ is not guaranteed, where $V(\boldsymbol{x})$ is the value function in (3), since (2) may not have a unique bounded solution.

IV. Experiments

In this section we evaluate the performance of our Bellman equation based method equipped with Alg. 1 on two illustrative examples. Moreover, we compare the Bellman equation based method in this paper with that in [26] based on these two examples, and compare the Bellman equation based methods propsoed in this paper equipped with the value iteration and the policy iteration based on these two examples.

The parameters that control the performance of our method are presented in Table I. All computations were performed on an i7-7500U 2.70GHz CPU with 32GB RAM running Windows 10. For the value iteration in Alg. 1, uniform grids are adopted for state and disturbance spaces. The computational state spaces for Examples 1 and 2 are restricted to $[-1.1,1.1]\times[-1.1,1.1]$ and $[-1,1]\times[-1,1]$, respectively.

Ex.	α	ϵ	N	M	$T_{\mathtt{VI}}$
1	10^{-16}	10^{-20}	10^4	10	88.10
2	10^{-16}	10^{-20}	10^4	10	87.05

TABLE I
Parameters and performance of our implementations on the Examples 1 and 2. α: the parameter value in (3)*; ϵ: the stopping criterion in the value iteration method in Alg. 1; N, M: numbers of elements in Λ and Δ respectively in the value iteration method in Alg. 1; $T_{\mathtt{VI}}$: computation times (seconds) in solving* (2) *using the value iteration method in Alg. 1.*

Example 1. *In this example we consider a computer-based model of the following perturbed continuous-time system describing the motion of a whirling pendulum [6],*

$$\begin{cases} \dot{x}_1 = x_2, \\ \dot{x}_2 = -\frac{2}{d}x_2 + 0.81\sin(x_1)\cos(x_1) - \sin(x_1), \end{cases}$$

where $d \in [0.9, 1.1]$.

When performing computer simulations, the Euler's method is used. It utilizes the idea of local linearity or linear approximation. When the simulation time step is 0.4, the resulting discrete-time system is:

$$\begin{cases} x_1(l+1) = x_1(l) + 0.4x_2(l), \\ x_2(l+1) = x_2(l) + 0.4\Big(-\frac{2}{d(l)}x_2(l)+ \\ \qquad\qquad 0.81\sin(x_1(l))\cos(x_1(l)) - \sin(x_1(l))\Big), \end{cases}$$

where $D = [0.9, 1.1]$ and $l \in \mathbb{N}$.

We take the state constraint set $X = \{(x_1, x_2) \mid x_1^2 + x_2^2 \leq 1\}$. The computed maximal robust invariant set is illustrated in Fig. 1, which also showcases the computed $V(\boldsymbol{x})$. Four trajectories, where two trajectories respect the state constraint and two trajectories violate the state constraint, are illustrated in Fig. 1 as well. These trajectories are generated by extracting the disturbance $d(l)$ from D randomly for $l \in \mathbb{N}$.

Example 2. *We consider a* DPNS *with $\boldsymbol{f}(x,y) = 1_{X_1} \cdot \boldsymbol{f}_1(x,y,d) + 1_{X_2} \cdot \boldsymbol{f}_2(x,y,d)$, where $1_{X_i} : X_i \to \{0,1\}$ represents the indicator function of the set X_i, $i = 1,2$, i.e.,*

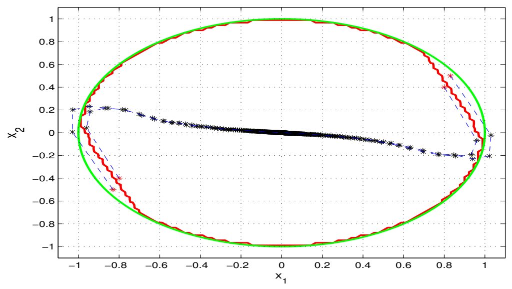
$$1_{X_i} := \begin{cases} 1, & \text{if } \boldsymbol{x} \in X_i, \\ 0, & \text{if } \boldsymbol{x} \notin X_i, \end{cases}$$

$$\boldsymbol{f}_1(x,y,d) = (x; (0.5+d)x - 0.1y),$$
$$\boldsymbol{f}_2(x,y,d) = (y; 0.2x - (0.1+d)y + y^2),$$

$X = \{(x,y) \mid x^2 + y^2 - 0.8 \leq 0\}$, $X_1 = \{(x,y) \mid 1 - (x-1)^2 - y^2 \leq 0\}$, $X_2 = \{(x,y) \mid -1 + (x-1)^2 + y^2 < 0\}$ and $D = \{d \mid d^2 - 0.01 \leq 0\}$.

Fig. 2 presents the maximal robust invariant set and the level sets of the solution to (2)*, which are computed via Alg. 1. Six trajectories, where three trajectories respect the state constraint and three trajectories violate the state constraint, are illustrated in Fig. 2 as well. Like Example 1, these trajectories are generated by extracting the disturbance $d(l)$ from D randomly for $l \in \mathbb{N}$.*

The plots in Fig. 1 and Fig. 2 confirm the statement in Proposition 1 that the solution to (2) with $\alpha \in (0,1)$ is non-negative. In the following we first compare the Bellman



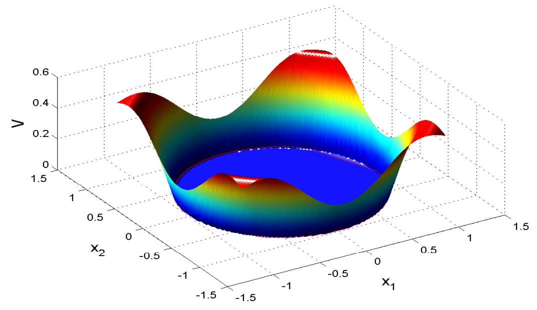
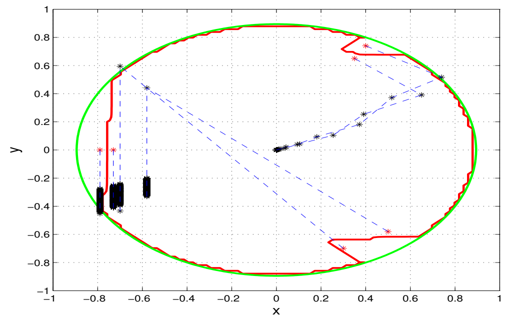


Fig. 1. An illustration of the maximal robust invariant set computed via Alg. 1 for Example 1. Above: Green and red curves denote the boundaries of the state constraint set X and the computed maximal robust invariant set, respectively. Red and black stars denote the initial states and subsequent states, respectively. The dash blue line denotes the transition between states. Below: Level sets of V obtained via the value iteration in Alg. 1.



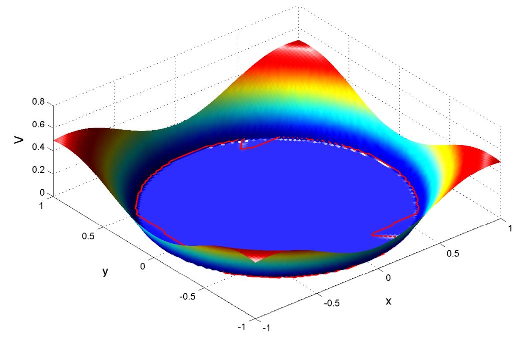


Fig. 2. An illustration of the maximal robust invariant set computed via Alg. 1 for Example 2. Above: Green and red curves denote the boundaries of the state constraint set X and the computed maximal robust invariant set, respectively. Red and black stars denote the initial states and subsequent states, respectively. The dash blue line denotes the transition between states. Below: Level sets of V obtained via the value iteration in Alg. 1.

equation based method equipped with Alg. 1 and the Bellman equation based method in [26] equipped with Alg.1, then we compare the value iteration and the policy iteration for solving (2) via tuning the parameter α based on the other parameters listed in Table I. Policy iteration, which manipulates the policy directly rather than finding it directly via the optimal value function, is also able to deal with the Bellman type equation (2). The policy iteration is presented in Alg.2 in Appendix. It is also guaranteed to converge to the unique bounded solution to (2). The convergence analysis is omitted here and can be reasoned about by following the one in, e.g., [21].

We compare the Bellman equation based method in the present paper with the one in [26] based on Examples 1 and 2. Based on the same parameter inputs listed in Table I, the value iteration to solve the Bellman type equation in [26] does not terminate after one and a half hours for both Examples 1 and 2. The underlying reason is that the value iteration for solving the Bellman type equation in [26] may not converge. Consequently, it cannot be directly used to compute the maximal robust invariant set. Also, we use the value iteration in Alg. 1 to solve the Bellman type equation (2) with $\alpha = 1$ for these two examples, the value iteration does not terminate after one and a half hours for both Examples 1 and 2 as well.

Next, we compare the value iteration in Alg. 1 and the policy iteration in Alg. 2 via varying α. Like the value iteration in Alg. 1, the policy iteration in Alg. 2 also takes uniform grids for state and disturbance spaces, and the corresponding computational state spaces are also restricted to $[-1.1, 1.1] \times [-1.1, 1.1]$ and $[-1, 1] \times [-1, 1]$ for Examples 1 and 2, respectively. We first take $\alpha = 0.005, 0.01, 0.05, 0.1, 0.2, 0.3, 0.5$ as instances to make comparisons. The maximal robust invariant sets for Example 1 computed via Alg. 1 and Alg. 2 with these values are almost the same as that presented in Fig. 1[1]. This statement is also applicable to Example 2. The corresponding computation times are summarized in Fig. 3. Fig. 3 indicates that the computation time for both the value iteration and the policy iteration increases as α goes up. Also, the policy iteration is more efficient than the value iteration generally when α takes the values aforementioned. However, it does not indicate that the policy iteration outperforms the value iteration for any $\alpha \in (0, 1)$. As uncovered in Remark 4, smaller α results in faster convergence of the value iteration. Also, as indicated in Remark 5, when the convergence rate α is approaching the accuracy tolerance ϵ, the iteration number in Alg. 1 is reducing. We take $10^{-16}, 10^{-14}, 10^{-12}$ as instances to make further comparisons between the value iteration in Alg. 1 and the policy iteration in Alg. 2. Analogously, the computed maximal robust invariant sets for Example 1 and Example 2 are also almost the same as those in Fig. 1 and Fig. 2, respectively. The corresponding computation times are listed in Table II, which reflects that the value iteration nevertheless outperforms the policy iteration when the convergence rate α is close enough to the given accuracy tolerance ϵ.

Finally, it is worth remarking here that the main contribution of our method lies in the reduction of a non-convex problem of computing maximal robust invariant sets to a problem of solving a single mathematical equation (2). Although numeric computation based methods such as the value iteration and the policy iteration for solving the equation (2) give us approximate solutions (with arbitrary accuracy) rather than the exact solution generally, the estimated maximal robust invariant sets are still useful in practice. For instance, they provide insights into the boundaries of maximal robust invariant sets, facilitating both the conservativeness gauge of the widely computed inner-approximations of maximal robust invariant sets and the gain of a feasible initial state enabling the system to respect the specified state constraints. Besides, if the discrete-time nonlinear system (1) is restricted to be the polynomial type, i.e., $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{d})$ is a polynomial or piecewise polynomial in $\boldsymbol{x}$ and $\boldsymbol{d}$, and the state constraint set X is semi-algebraic, the equation (2) facilitates the construction of a semi-definite program for computing inner-approximations of maximal robust invariant sets. The semi-definite program can be obtained by following the reasoning in [26] and is not the focus of this work.

[1] 'almost the same' means that the computed maximal robust invariant sets showing on the graph are indistinguishable.

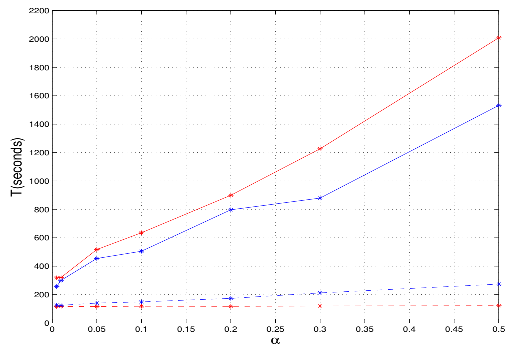


Fig. 3. An illustration of the effect of relatively large α on the computation times. The red and blue lines denote the relationship between the computation time and α in the value iteration method for Example 1 and 2, respectively. The red and blue dashed lines denote the relationship between the computation time and α in the policy iteration method for Example 1 and 2, respectively. The red and blue stars denote the computations times corresponding to $\alpha = 0.005, 0.01, 0.05, 0.1, 0.2, 0.3, 0.5$ for Examples 1 and 2, respectively.

α		10^{-16}	10^{-14}	10^{-12}
Ex. 1	$T_{\mathtt{VI}}$	88.10	89.26	87.50
	$T_{\mathtt{PI}}$	107.25	112.12	107.07
Ex. 2	$T_{\mathtt{VI}}$	87.05	90.73	87.98
	$T_{\mathtt{PI}}$	106.49	110.68	107.03

TABLE II
The effect of small α on computation times for both the value iteration and policy iteration. $T_{\mathtt{VI}}$ and $T_{\mathtt{PI}}$: computation times (seconds) in solving (2) using the value iteration in Alg. 1 and policy iteration in Alg. 2, respectively.

V. Conclusion and Future Work

In this paper we studied the maximal robust invariant set estimation for discrete-time perturbed nonlinear systems. We for the first characterized the maximal robust invariant set as the zero level set of the unique bounded solution to a Bellman type equation. Value iteration and policy iteration can be used to solve such equation with appropriate number of state and disturbance variables. Two examples demonstrated the performance of our Bellman equation based method.

In near future we would extend our method to the computation of robust invariant sets for state-constrained hybrid systems subject to competing inputs (control and disturbance).

ACKNOWLEDGMENTS

This work has been supported through grants by NSFC under grant No. 61872341, 61836005, 61625206, 61732001 and the CAS Pioneer Hundred Talents Program.

REFERENCES

[1] C. O. Aguilar and A. J. Krener. Numerical solutions to the Bellman equation of optimal control. *Journal of Optimization Theory and Applications*, 160(2):527–552, 2014.

[2] J.-P. Aubin, A. M. Bayen, and P. Saint-Pierre. *Viability theory: new directions*. Springer Science & Business Media, 2011.

[3] M. Bardi and I. Capuzzo-Dolcettae. *Optimal control and viscosity solutions of Hamilton-Jacobi-Bellman equations*. Springer Science & Business Media, 1997.

[4] D. Bertsekas. Infinite time reachability of state-space regions by using feedback control. *IEEE Trans. Autom. Control.*, 17(5):604–613, 1972.

[5] F. Blanchini and S. Miani. *Set-theoretic methods in control*. Springer, 2008.

[6] G. Chesi. Estimating the domain of attraction for non-polynomial systems via lmi optimizations. *Automatica*, 45(6):1536–1541, 2009.

[7] P. Grieder, S. Raković, M. Morari, and D. Mayne. Invariant sets for switched discrete time systems subject to bounded disturbances. *IFAC Proceedings Volumes*, 38(1):115–120, 2005.

[8] M. Jones and M. M. Peet. A generalization of bellman's equation with application to path planning, obstacle avoidance and invariant set estimation. *https://arxiv.org/abs/2006.08175*, 2020.

[9] H. K. Khalil. Nonlinear systems. *Upper Saddle River*, 2002.

[10] Y. Li and J. Liu. Invariance control synthesis for switched nonlinear systems: An interval analysis approach. *IEEE Trans. Autom. Control.*, 63(7):2206–2211, 2018.

[11] J. Lunze and F. Lamnabhi-Lagarrigue. *Handbook of hybrid systems control: theory, tools, applications*. Cambridge University Press, 2009.

[12] M. S. Mahmoud and M. G. Singh. *Discrete systems: analysis, control and optimization*. Springer Science & Business Media, 2012.

[13] J. N. Maidens, S. Kaynama, I. M. Mitchell, M. M. Oishi, and G. A. Dumont. Lagrangian methods for approximating the viability kernel in high-dimensional systems. *Automatica*, 49(7):2017–2029, 2013.

[14] K. Margellos and J. Lygeros. Hamilton–Jacobi formulation for reach–avoid differential games. *IEEE Trans. Autom. Control.*, 56(8):1849–1861, 2011.

[15] I. M. Mitchell. A toolbox of level set methods. *UBC Department of Computer Science Technical Report TR-2007-11*, 2007.

[16] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin. A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games. *IEEE Trans. Autom. Control.*, 50(7):947–957, 2005.

[17] C. A. Rabbath and N. Léchevin. *Discrete-time control system design with applications*. Springer Science & Business Media, 2013.

[18] S. Rakovic, P. Grieder, M. Kvasnica, D. Mayne, and M. Morari. Computation of invariant sets for piecewise affine discrete time systems subject to bounded disturbances. In *CDC'04*, pages 1418–1423, 2004.

[19] S. Rakovic, E. Kerrigan, K. Kouramas, and D. Mayne. Invariant approximations of the minimal robust positively invariant set. *IEEE Transactions on Automatic Control*, 50(3):406–410, 2005.

[20] S. V. Raković and M. Fiacchini. Approximate reachability analysis for linear discrete time systems using homothety and invariance. *IFAC Proceedings Volumes*, 41(2):15327–15332, 2008.

[21] M. S. Santos and J. Rust. Convergence properties of policy iteration. *SIAM Journal on Control and Optimization*, 42(6):2094–2115, 2004.

[22] P. Trodden. A one-step approach to computing a polytopic robust positively invariant set. *IEEE Trans. Autom. Control.*, 61(12):4100–4105, 2016.

[23] B. Xue, M. Fränzle, and N. Zhan. Inner-approximating reachable sets for polynomial systems with time-varying uncertainties. *IEEE Transactions on Automatic Control*, 2019.

[24] B. Xue, Q. Wang, N. Zhan, and M. Fränzle. Robust invariant sets generation for state-constrained perturbed polynomial systems. In *HSCC'19*, pages 128–137, 2019.

[25] B. Xue, N. Zhan, and Y. Li. A characterization of robust regions of attraction for discrete-time systems based on bellman equations. In *IFAC'20*, pages 6468–6475. IFAC, 2020.

[26] B. Xue, N. Zhan, Y. Li, and Q. Wang. Robust non-termination analysis of numerical software. In *SETTA'18*, pages 69–88. Springer, 2018.

APPENDIX

The policy iteration for addressing (2) with $\alpha \in (0,1)$ is described in Alg. 2.

Algorithm 2 The policy iteration for solving (2)

```
1) Set $V_0(\boldsymbol{x}) := 0$ over $\boldsymbol{x} \in \mathbb{R}^n$ and $k := 0$, and decide on a grid $\Lambda = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_N\}$ on $X$ for the state variable $\boldsymbol{x}$, and a grid $\Delta = \{\boldsymbol{d}_1, \dots, \boldsymbol{d}_M\}$ in $D$ for the disturbance variable $\boldsymbol{d}$.
2) Choose disturbances $\hat{\pi} = (\boldsymbol{d}^1, \dots, \boldsymbol{d}^N)$;
3) Choose a value in $(0,1)$ for $\alpha$;
4) Choose an accuracy tolerance $\epsilon > 0$;
5) For each $\boldsymbol{x}_i \in \Lambda$, $i = 1, \dots, N$, compute
       $$\boldsymbol{x}'_i = \boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{d}^i),$$
   then compute an interpolated value function at each $\boldsymbol{x}'_i$ : $\tilde{V}_k(\boldsymbol{x}'_i)$ and compute
       $$V_{k+1}(\boldsymbol{x}_i) = \max\{\alpha \tilde{V}_k(\boldsymbol{x}'_i), \max_{j \in \{1,\dots,n_0\}} h'_j(\boldsymbol{x}_i)\}.$$
6) If $\max_{\boldsymbol{x} \in \Lambda} |V_{k+1}(\boldsymbol{x}) - V_k(\boldsymbol{x})| < \epsilon$, go to step 7); otherwise, $k := k+1$ and go back to 5);
7) $\hat{\pi}' := \hat{\pi}$ and for each $\boldsymbol{x}_i \in \Lambda$, $i = 1, \dots, N$,
       $$\boldsymbol{d}^i = \arg\max_{\boldsymbol{d}_j \in \Delta} V_l(\boldsymbol{f}(\boldsymbol{x}_i, \boldsymbol{d}_j)).$$
   $\hat{\pi} := (\boldsymbol{d}^1, \dots, \boldsymbol{d}^N)$ and go to step 8);
8) If $\hat{\pi}' = \hat{\pi}$, obtain the final solution $V(\boldsymbol{x})$ as $V(\boldsymbol{x}) \approx V_k(\boldsymbol{x})$. Otherwise, go back to 5).
```