Sampling-Based Approximation of the Viability Kernel for High-Dimensional Linear Sampled-Data Systems

Jeremy H. Gillula
Electrical Engineering and Computer Sciences Dept.
UC Berkeley
301 Cory Hall
Berkeley, CA 94720
jgillula@eecs.berkeley.edu

Shahab Kaynama
Electrical Engineering and Computer Sciences Dept.
UC Berkeley
307 Cory Hall
Berkeley, CA 94720
kaynama@berkeley.edu

Claire J. Tomlin
Electrical Engineering and Computer Sciences Dept.
UC Berkeley
721 Sutardja Dai Hall
Berkeley, CA 94720
tomlin@eecs.berkeley.edu

ABSTRACT

Proving that systems satisfy hard input and state constraints is frequently desirable when designing cyber-physical systems. One method for doing so is to compute the viability kernel, the subset of the state space for which a control signal exists that is guaranteed to keep the system within the constraints over some time horizon. In this paper we present a novel method for approximating the viability kernel for linear sampled-data systems using a sampling-based algorithm, which by its construction offers a direct trade-off between scalability and accuracy. We also prove that the algorithm is correct, that its convergence properties are optimal, and demonstrate it on a simple example. We conclude by briefly describing additional results which are omitted due to space constraints.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search—*control theory*

Keywords

constrained sampled-data systems; viability kernel approximations; sampling-based algorithms

1. INTRODUCTION

As we enter the twenty-first century, cyber-physical systems are increasingly being used in "workspaces" which are shared with the general public and thus for which safety is critical. In order to guarantee the safety of these robots as well as the humans working with them it is vitally important that we be able to prove that these systems satisfy hard input and state constraints.

One method for doing so is to compute the *viability kernel* [2], the subset of the state space for which a control signal exists which is guaranteed to keep the system state within some hard constraints over some finite time horizon. (In the infinite horizon case this set is also known as the maximal controlled-invariant set [4].) In this paper we present a novel method for approximating the viability kernel for sampled-data systems using a sampling-based algorithm which offers a direct trade-off between scalability and accuracy. In addition to presenting the algorithm we will prove its correctness, analyze its convergence properties (and prove that they are optimal), and demonstrate the algorithm on a simple example.

HSCC'14, April 15–17, 2014, Berlin, Germany.
ACM 978-1-4503-2732-9/14/04.
http://dx.doi.org/10.1145/2562059.2562117 .

1.1 Related Work

A great deal of prior work has been done on methods for computing the viability kernel, e.g. [25, 29]. Since these particular techniques represent the viability kernel using a grid over the state space, however, they have historically been limited to use on low-dimensional systems (usually 5D or less). Efforts to overcome this limitation, although still grid-based, include imposing and/or taking advantage of the structure of the dynamics [16, 24], or using approximate dynamic programming techniques [6].

An alternative to computing the viability kernel on a grid is to follow the flow of the system while using compact set representations (similar in spirit to Lagrangian techniques for maximal reachability [10, 11, 13]). For example [15] utilizes a recursive method to under-approximate the viability kernel for continuous-time systems, and proposes a scalable piecewise ellipsoidal algorithm (based on ellipsoidal techniques for maximal reachability [17]) for linear time-invariant (LTI) dynamics.

The model-predictive control (MPC) community has also developed algorithms that enable the computation of the viability kernel for discrete-time LTI systems with polytopic constraints [4]. Due to the high computational cost of the operations involved (Minkowski sums, intersections, and vertex-to-facet enumerations of polytopes) these algorithms can only be applied to low-dimensional systems. Alternatively, [21] proposed an efficient and scalable algorithm based on support vector representations for discrete-time LTI systems.

Approximating the viability kernel can also be viewed as a search for an appropriate control Lyapunov function, subject to additional input and state constraints. In this spirit, sums-of-squares (SOS) optimization-based techniques have been proposed [22, 28, 32–35] for polynomial systems with semi-algebraic constraints, which either directly calculate an approximation of the viability kernel or can be modified to do so. Such techniques strike a tradeoff between conservatism and complexity (as determined, e.g., by the chosen degree of the SOS multipliers). A related SOS-based technique is the method of occupation measures [14, 23] that, while convex and thus scalable, can only compute an over-approximation of the desired set (which is not sufficient for safety).

Despite this rich body of literature, all of the above work has focused on either continuous- or discrete-time systems, and very little work has been done on computing the viability kernel for *sampled-data* systems, i.e. systems with continuous dynamics in which the

state of the system is measured at every time instant $t_k := k\delta$ for $k \in \mathbb{Z}_+$ and fixed sampling interval $\delta \in \mathbb{R}_+$, and for which the input is applied at the beginning of each sampling interval and kept constant until the next sampling instant. We believe that this class of systems more accurately represents modern cyber-physical systems in that most controllers are implemented on digital platforms while the system itself is analog and evolves continuously [12]. [1] Unfortunately continuous-time analysis on a system cannot provide guarantees about the behavior of the sampled-data system where the input is restricted to draw from the class of piecewise constant signals [26] and the state can only be measured at a fixed frequency. Similarly, simply discretizing the dynamics and designing control policies in discrete time would not account for the behavior of the underlying analog system between sampling instants, and could result in loss of safety. These complications make the work described above difficult to adapt to sampled-data systems. Instead our proposed algorithm handles sampled-data systems directly.

To the authors' knowledge the only other scalable work on computing the viability kernel for sampled-data systems is presented in [26], and applies to LTI systems and represents the constraints in a piecewise-ellipsoidal manner. Unfortunately due to the operations involved the results of that algorithm are extremely conservative. By contrast our approach (which is also scalable) yields a tight approximation of the viability kernel in the sense that the resulting set touches the boundary of the true viability kernel with arbitrary precision.

1.2 Definitions and Problem Formulation

In order to more formally state the problem our algorithm solves, we must first present some additional definitions and notation.

The $\|\cdot\|_p$-distance of a point $\mathbf{x} \in \mathcal{X}$ from a nonempty set $\mathcal{A} \subset \mathcal{X}$ is defined as $\mathrm{dist}_p(\mathbf{x}, \mathcal{A}) := \inf_{\mathbf{a}\in\mathcal{A}} \|\mathbf{x} - \mathbf{a}\|_p$.

We say that the vector field f is bounded on $\mathcal{A}$ if a norm $\|\cdot\|_p$ and a real number $M > 0$ exist for some norm $\|\cdot\|_p$ such that $\|f(\mathbf{x}, \mathbf{u})\|_p \leq M \ \forall(\mathbf{x}, \mathbf{u}) \in \mathcal{A} \times \mathcal{U}$. If $\mathcal{A}$ is compact, every continuous vector field f is bounded on $\mathcal{A}$.

The *Minkowski sum* of any two nonempty subsets $\mathcal{A}$ and $\mathcal{C}$ is $\mathcal{A} \oplus \mathcal{C} := \{\mathbf{a} + \mathbf{c} \mid \mathbf{a} \in \mathcal{A}, \ \mathbf{c} \in \mathcal{C}\}$; their *Pontryagin difference* (or, the *erosion* of $\mathcal{A}$ by $\mathcal{C}$) is $\mathcal{A} \ominus \mathcal{C} := \{\mathbf{a} \mid \mathbf{a} \oplus \mathcal{C} \subseteq \mathcal{A}\}$.

We denote by $\partial\mathcal{C}$ the boundary of $\mathcal{C}$, and by $\mathcal{C}^c$ its complement. $\mathcal{B}_p^n(\mathbf{x}, \alpha)$ denotes a closed p-norm ball of radius $\alpha \in \mathbb{R}_+$ about a point $\mathbf{x}$ in $\mathbb{R}^n$, and $\mathcal{S}_p^{n-1}$ the codimension one boundary $\partial\mathcal{B}_p^n(0, 1)$ of the unit ball centered at the origin.

A *ray* in $\mathbb{R}^n$ is the set of points $\vec{r} = \{\mathbf{r}_0 + s\mathbf{r}_d \mid s \in \mathbb{R}_+\}$, where $\mathbf{r}_0 \in \mathbb{R}^n$ is the *origin* of the ray, and $\mathbf{r}_d \in \mathbb{R}^n$ is a unit vector giving the *direction* of the ray.

We denote by $\mathrm{conv}(\{\mathbf{v}_0, \ldots, \mathbf{v}_N\})$ the convex hull of the points $\mathbf{v}_0, \ldots, \mathbf{v}_N$, and by $\mathrm{vol}(\mathcal{C})$ the volume of a set $\mathcal{C}$.

Finally, consider the LTI system of the form:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)) = A\mathbf{x}(t) + B\mathbf{u}(t), \quad \mathbf{x}(0) = \mathbf{x}_0 \qquad (1)$$

with a finite-dimensional normed vector state space $\mathcal{X} := \mathbb{R}^n$, state vector $\mathbf{x}(t) \in \mathcal{X}$, and control input $\mathbf{u}(t) \in \mathcal{U}$, where $\mathcal{U}$ is a compact convex subset of $\mathbb{R}^m$. A and B are constant matrices of appropriate dimensions. We are concerned with the evolution of the system over the interval $\mathbb{T} := [0, \tau]$ with arbitrary, finite time horizon $\tau \in \mathbb{R}_+$. We denote by $N_\delta := \tau/\delta \in \mathbb{Z}_+$ the number of sampling intervals in $\mathbb{T}$. Since we are concerned with sampled-data systems, the input signal draws from the set of piecewise constant functions

$$\mathscr{U}_{\mathbb{T}}^{\mathrm{pw}} := \{\mathbf{u} \colon \mathbb{T} \to \mathbb{R}^m \text{ piecewise const., } \mathbf{u}(t_k) \in \mathcal{U} \ \forall k, \ \mathbf{u}(t) = \mathbf{u}(t_k) \ \forall t \in [t_k, t_{k+1})\}. \qquad (2)$$

For every $\mathbf{x}_0 \in \mathcal{X}$ and $\mathbf{u}(\cdot) \in \mathscr{U}_{\mathbb{T}}^{\mathrm{pw}}$, we denote the (unique) trajectory of (1) by $\mathbf{x}_{\mathbf{x}_0}^{\mathbf{u}} \colon \mathbb{T} \to \mathcal{X}$ with initial condition $\mathbf{x}_{\mathbf{x}_0}^{\mathbf{u}}(0) = \mathbf{x}_0$. When clear from the context, we shall drop the subscript and superscript from the trajectory notation.

For a nonempty compact convex state constraint set $\mathcal{A} \subset \mathcal{X}$ (deemed *safe*), we examine the following backward construct:

DEFINITION 1 (SAMPLED-DATA VIABILITY KERNEL). *The finite-horizon sampled-data viability kernel of $\mathcal{A}$ is the set of all initial states in $\mathcal{A}$ for which there exists a control law such that the trajectories emanating from those states remain in $\mathcal{A}$ over the horizon $\mathbb{T}$:*

$$\mathrm{Viab}_{\mathbb{T}}^{\mathrm{sd}}(\mathcal{A}) := \big\{\mathbf{x}_0 \in \mathcal{A} \mid \exists \mathbf{u}(\cdot) \in \mathscr{U}_{\mathbb{T}}^{\mathrm{pw}}, \ \mathrm{s.t.}\, \mathbf{x}_{\mathbf{x}_0}^{\mathbf{u}}(t) \in \mathcal{A} \ \forall t \in \mathbb{T}\big\}. \qquad (3)$$

We note that any approximation to the viability kernel must be in the form of an under-approximation in order to guarantee safety; thus the goal of our algorithm will be to find an efficient and scalable technique that under-approximates the viability kernel.

2. THE SAMPLING-BASED REACHABILITY ALGORITHM

At a high level the algorithm we propose works by sampling from points in $\mathcal{A}$. We then verify whether each sampled point is within the viability kernel by formulating a convex program which attempts to find a piecewise constant control input that keeps the system trajectory, starting from the sampled point, inside the safe set $\mathcal{A}$ over the period $\mathbb{T}$. If such a control signal can be found, then the sampled point is inside the viability kernel. Finally, since we are working with linear systems, we use the fact that the sets $\mathcal{U}$ and $\mathcal{A}$ are convex to construct an under-approximation to the viability kernel from the convex hull of the sampled points.

2.1 Checking Point Feasibility

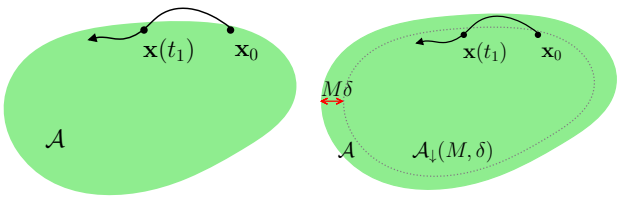
An important aspect of our algorithm is that it constructs an approximation to the viability kernel by iteratively adding to a list of points known to be just inside its boundary. To do this we need a way of checking whether or not a sampled point $\mathbf{x}_0$ is in $\mathrm{Viab}_{\mathbb{T}}^{\mathrm{sd}}(\mathcal{A})$. We will call a point *feasible* if it is inside the viability kernel, and *infeasible* otherwise.

2.1.1 From Continuous to Pointwise Feasibility

By definition if we can find a piecewise constant control signal $\mathbf{u}(\cdot)$ such that $\mathbf{x}_{\mathbf{x}_0}^{\mathbf{u}}(t) \in \mathcal{A} \ \forall t \in \mathbb{T}$, then $\mathbf{x}_0 \in \mathrm{Viab}_{\mathbb{T}}^{\mathrm{sd}}(\mathcal{A})$. Of course without an analytical representation for the trajectory $\mathbf{x}_{\mathbf{x}_0}^{\mathbf{u}}(t)$, checking that all points along the trajectory are within $\mathcal{A}$ is impossible. Thus instead of verifying that the entire trajectory is inside the safe set, we will instead verify that discrete points $\mathbf{x}_k := \mathbf{x}_{\mathbf{x}_0}^{\mathbf{u}}(t_k)$ are within $\mathcal{A}$. However, since the trajectories in between two sampling instants could escape safety even when the end points are within $\mathcal{A}$ (Figure 1(a)), we will instead verify that the points are within an appropriately constructed subset of $\mathcal{A}$ such that the entire trajectory is guaranteed to maintain safety. The following lemma describes how to construct such a subset, called $\mathcal{A}_\downarrow$.

LEMMA 1 ([15]). *Suppose that f is uniformly bounded on $\mathcal{A}$ in some norm $\|\cdot\|_{p_1}$ by M. With a sampling interval $\delta > 0$,*

[1] Note that even though some systems can be treated continuously because the controllers run at a much higher frequency than the time scales of the system dynamics, there are still many systems for which this is not the case, such as the automated anesthesia system described in [21].



(a) (b)

Figure 1: (a) Even though the sampled points x_0 and $x(t_1)$ are contained in $\mathcal{A}$, the continuous trajectory may not be. (b) Erosion of the set $\mathcal{A}$ ensures that the curvature of the continuous trajectory in between two sampling time instants does not escape safety.

define a set

$$\begin{aligned}\mathcal{A}_\downarrow(M,\delta) &:= \{\mathbf{x}\in\mathcal{A} \mid \mathrm{dist}_{p_1}(\mathbf{x},\mathcal{A}^c)\geq M\delta\}\\ &= \mathcal{A}\ominus\mathcal{B}^n_{p_1}(0,M\delta).\end{aligned} \tag{4}$$

For a given initial condition $\mathbf{x}_0$, if there exists a piecewise constant control signal $\mathbf{u}(\cdot)\in\mathscr{U}^{\mathrm{pw}}_{\mathbb{T}}$ such that $\mathbf{x}(t_k)\in\mathcal{A}_\downarrow(M,\delta)$ $\forall k\in\{0,\dots,N_\delta\}$, then $\mathbf{x}_0\in\mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})$.

PROOF. Over any sampling interval $[t_k,t_{k+1}]$ the trajectory can travel a distance of at most

$$\|\mathbf{x}(t_{k+1})-\mathbf{x}(t_k)\|_{p_1}\leq\int_{t_k}^{t_k+\delta}\|\dot{\mathbf{x}}(\lambda)\|_{p_1}\,d\lambda\leq M\delta. \tag{5}$$

Thus for any $s\in[t_k,t_{k+1})$ we have that

$$\|\mathbf{x}(s)-\mathbf{x}(t_k)\|_{p_1}\leq M(s-t_k)<M\delta. \tag{6}$$

Since $\mathbf{x}(t_k)\in\mathcal{A}_\downarrow(M,\delta)$, we know that $\mathrm{dist}_{p_1}(\mathbf{x}(t_k),\mathcal{A}^c)\geq M\delta$. Therefore,

$$\begin{aligned}\mathrm{dist}_{p_1}(\mathbf{x}(s),\mathcal{A}^c)&\geq\mathrm{dist}_{p_1}(\mathbf{x}(t_k),\mathcal{A}^c)-\|\mathbf{x}(s)-\mathbf{x}(t_k)\|_{p_1}\\ &>M\delta-M\delta=0.\end{aligned} \tag{7}$$

Thus $\mathbf{x}(t)\in\mathcal{A}$ over any closed sub-interval $[t_k,t_{k+1}]$ and therefore $\mathbf{x}(t)\in\mathcal{A}\ \forall t\in\mathbb{T}$. Hence, $\mathbf{x}_0\in\mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})$. □

We have thus reduced a continuous feasibility problem (checking if an entire trajectory is in $\mathcal{A}$) to a pointwise feasibility problem (checking if the points $\mathbf{x}_k$ are in $\mathcal{A}_\downarrow$, see Figure 1(b)).

2.1.2 Discretization Error

To solve this pointwise feasibility problem we will use forward simulation to determine whether a piecewise constant control signal exists that satisfies $\mathbf{x}(t_k)\in\mathcal{A}_\downarrow\ \forall k\in\{0,\dots,N_\delta\}$ when applied to the true sampled-data system. To do this we will use a finite-difference approximation of the dynamics, and thus need to take into account the effect of the discretization error introduced by the truncation of the Taylor series expansion of the dynamics [1].

Recall that the solution $\mathbf{x}_{k+1}=\mathbf{x}(t_{k+1})$ of the sampled-data system (1) at time t_{k+1} for any sampling interval $[t_k,t_k+\delta]$ starting from $\mathbf{x}_k=\mathbf{x}(t_k)$ using a constant input $\mathbf{u}_k$ is given by

$$\mathbf{x}_{k+1}=e^{A\delta}\mathbf{x}_k+\int_0^\delta e^{A(\delta-r)}B\mathbf{u}_k dr \tag{8}$$

$$=e^{A\delta}\mathbf{x}_k+\left(\int_0^\delta e^{A\lambda}d\lambda\cdot B\right)\mathbf{u}_k. \tag{9}$$

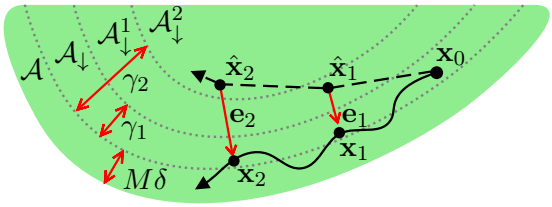


Figure 2: The discretization error $\mathbf{e}_k$ due to the mismatch between the true sampled-data state $\mathbf{x}_k=\mathbf{x}(t_k)$ at sampling time t_k and the nominal state $\hat{\mathbf{x}}_k$ of the discretized system propagates in time. If not accounted for in forward simulations, this mismatch could jeopardize the safety of the sampled-data system. By ensuring that $\hat{\mathbf{x}}_k\in\mathcal{A}^k_\downarrow$, we can guarantee $\mathbf{x}_k\in\mathcal{A}_\downarrow$ and thus guarantee safety.

Consider the Taylor series expansion of the matrix exponential

$$e^{As}=\sum_{i=0}^{\infty}\frac{(As)^i}{i!}. \tag{10}$$

We approximate the above infinite sum by a ζ-th order finite sum

$$A_s:=\sum_{i=0}^{\zeta}\frac{(As)^i}{i!}\approx e^{As},\quad \zeta<\infty \tag{11}$$

with *truncation error*

$$E_s:=e^{As}-A_s=\sum_{i=\zeta+1}^{\infty}\frac{(As)^i}{i!}. \tag{12}$$

The trajectory of the resulting finite-difference equation is

$$\hat{\mathbf{x}}_{k+1}=A_\delta\hat{\mathbf{x}}_k+\left(\int_0^\delta A_\lambda d\lambda\cdot B\right)\mathbf{u}_k. \tag{13}$$

We denote $\hat{\mathbf{x}}$ as the *nominal* state of the system.

Truncating the tail of the Taylor series expansion introduces a *discretization error* $\mathbf{e}_k$ that results in a mismatch between the true values of the system trajectory $\mathbf{x}_k=\mathbf{x}(t_k)$ at discrete time instants and the values generated by the finite-difference model, i.e. $\hat{\mathbf{x}}_k=\mathbf{x}_k-\mathbf{e}_k$. Thus to guarantee an under-approximation of the viability kernel using a pointwise feasibility check, we must take into account this error and its forward propagation in time (Figure 2).

LEMMA 2. *Suppose that there exist constants $\gamma_k\geq 0$ and a norm $\|\cdot\|_{p_2}$ such that $\|\mathbf{e}_k\|_{p_2}\leq\gamma_k\ \forall k$. Let*

$$\mathcal{A}^k_\downarrow(M,\delta,\gamma_k):=\mathcal{A}_\downarrow(M,\delta)\ominus\mathcal{B}^n_{p_2}(0,\gamma_k)\neq\emptyset. \tag{14}$$

Then we have that

$$\hat{\mathbf{x}}_k\in\mathcal{A}^k_\downarrow(M,\delta,\gamma_k)\Rightarrow\mathbf{x}_k\in\mathcal{A}_\downarrow(M,\delta). \tag{15}$$

PROOF. Since $\mathbf{e}_k\in\mathcal{B}^n_{p_2}(0,\gamma_k)$, regardless of the perturbation caused by the error, we have $\mathbf{x}_k\in\hat{\mathbf{x}}_k\oplus\mathcal{B}^n_{p_2}(0,\gamma_k)$. By enforcing $\hat{\mathbf{x}}_k\in\mathcal{A}^k_\downarrow(M,\delta,\gamma_k)$ we guarantee that

$$\mathbf{x}_k\in\left(\mathcal{A}_\downarrow(M,\delta)\ominus\mathcal{B}^n_{p_2}(0,\gamma_k)\right)\oplus\mathcal{B}^n_{p_2}(0,\gamma_k)\subseteq\mathcal{A}_\downarrow(M,\delta), \tag{16}$$

since for every two sets $\mathcal{X}$ and $\mathcal{Y}$ we have $(\mathcal{X}\ominus\mathcal{Y})\oplus\mathcal{Y}\subseteq\mathcal{X}$ (Figure 2). □

For any initial condition $\mathbf{x}_0$ we trivially have that $e_0=0$ and thus $\mathcal{A}^0_\downarrow(M,\delta,\gamma_0)=\mathcal{A}_\downarrow(M,\delta)$. We now describe a procedure to compute error bounds γ_k for $k\in\{1,\dots,N_\delta\}$ which will be used for *a priori* (offline) construction of the sets $\mathcal{A}^k_\downarrow(M,\delta,\gamma_k)$.

Using the identity $e^{As} = A_s + E_s$ in equation (9) yields

$$\begin{aligned}\mathbf{x}_k =&(A_\delta + E_\delta)\mathbf{x}_{k-1} + \left(\int_0^\delta (A_\lambda + E_\lambda)d\lambda \cdot B\right)\mathbf{u}_{k-1}\\ =&\underbrace{A_\delta \mathbf{x}_{k-1} + \left(\int_0^\delta A_\lambda d\lambda \cdot B\right)\mathbf{u}_{k-1}}_{\hat{\mathbf{x}}_k} +\\ &\underbrace{E_\delta \mathbf{x}_{k-1} + \left(\int_0^\delta E_\lambda d\lambda \cdot B\right)\mathbf{u}_{k-1}}_{\mathbf{e}_k}. \end{aligned} \tag{17}$$

Recall that for any discrete-time LTI system $\mathbf{x}_k = \Phi\mathbf{x}_{k-1} + \Psi\mathbf{u}_{k-1}$ we can rewrite the solution $\mathbf{x}_k$ in terms of the initial condition $\mathbf{x}_0$ and the past inputs as

$$\mathbf{x}_k = \Phi^k\mathbf{x}_0 + \sum_{i=0}^{k-1}\Phi^i\Psi\mathbf{u}_{k-1-i}. \tag{18}$$

Back-substituting the solutions $\mathbf{x}_{k-1}$ into $\mathbf{x}_k$, and $\hat{\mathbf{x}}_{k-1}$ into $\hat{\mathbf{x}}_k$ for every $k = 1, \dots, N_\delta$ we can rewrite $\mathbf{e}_k = \mathbf{x}_k - \hat{\mathbf{x}}_k$ as

$$\begin{aligned}\mathbf{e}_k = &\left((A_\delta + E_\delta)^k - A_\delta^k\right)\mathbf{x}_0 +\\ &\sum_{i=0}^{k-1}\Big[\left((A_\delta + E_\delta)^i - A_\delta^i\right)\int_0^\delta A_\lambda d\lambda +\\ &(A_\delta + E_\delta)^i\int_0^\delta E_\lambda d\lambda\Big]B\mathbf{u}_{k-1-i}. \end{aligned} \tag{19}$$

To compute the upper-bound γ_k on (19), invoke

i) the inequality $\left\|A^k\right\| \le \|A\|^k$ which holds for any matrix A and positive constant k,
ii) the multiplicative and triangular inequalities, and
iii) the binomial expansion for $(A_\delta + E_\delta)^k$ (which is valid since $A_sE_s = E_sA_s$ [31]).

We obtain

$$\begin{aligned}\|\mathbf{e}_k\| \le &\left\|(A_\delta + E_\delta)^k - A_\delta^k\right\|\|\mathbf{x}_0\| \\ &+\sum_{i=0}^{k-1}\left[\left\|\left((A_\delta + E_\delta)^i - A_\delta^i\right)\int_0^\delta A_\lambda d\lambda\right\| \right.\\ &\left.+\left\|(A_\delta + E_\delta)^i\int_0^\delta E_\lambda d\lambda\right\|\right]\sup_{u\in\mathcal{U}}\|B\mathbf{u}\| \end{aligned} \tag{20}$$

$$\begin{aligned}\le &\left\|\sum_{l=0}^{k}\binom{k}{l}A_\delta^l E_\delta^{k-l} - A_\delta^k\right\|\|\mathbf{x}_0\| \\ &+\sum_{i=0}^{k-1}\left[\left\|\sum_{l=0}^{i}\binom{i}{l}A_\delta^l E_\delta^{i-l} - A_\delta^i\right\|\left\|\int_0^\delta A_\lambda d\lambda\right\| \right.\\ &\left.+\|A_\delta + E_\delta\|^i\left\|\int_0^\delta E_\lambda d\lambda\right\|\right]\sup_{u\in\mathcal{U}}\|B\mathbf{u}\|. \end{aligned} \tag{21}$$

For $k > 1$ we find

$$\begin{aligned}\|\mathbf{e}_k\| \le &\sum_{l=0}^{k-1}\binom{k}{l}\left\|A_\delta^l\right\|\|E_\delta\|^{k-l}\|\mathbf{x}_0\| \\ &+\sum_{i=1}^{k-1}\left[\sum_{l=0}^{i-1}\binom{i}{l}\left\|A_\delta^l\right\|\|E_\delta\|^{i-l}\left\|\int_0^\delta A_\lambda d\lambda\right\| \right.\\ &\left.+(\|A_\delta\| + \|E_\delta\|)^i\left\|\int_0^\delta E_\lambda d\lambda\right\|\right]\sup_{u\in\mathcal{U}}\|B\mathbf{u}\|, \end{aligned} \tag{22}$$

and for $k = 1$ this inequality is

$$\|\mathbf{e}_1\| \le \|E_\delta\|\,\|\mathbf{x}_0\| + \left\|\int_0^\delta E_\lambda d\lambda\right\|\sup_{u\in\mathcal{U}}\|B\mathbf{u}\|. \tag{23}$$

The term $\int_0^\delta A_\lambda d\lambda$ is a definite integral of a finite sum. Therefore, it can be easily computed:

$$\int_0^\delta A_\lambda d\lambda = \int_0^\delta\sum_{i=0}^{\zeta}\frac{(A\lambda)^i}{i!}d\lambda = \sum_{i=0}^{\zeta}\frac{A^i\delta^{i+1}}{(i+1)!}. \tag{24}$$

Evaluating $\int_0^\delta E_\lambda d\lambda$ and $\|E_\delta\|$ is trickier. We can, however, formulate upper-bounds on their infinity norm. To do so, we use the following property [19]:

$$\|E_\delta\|_\infty \le \sum_{i=\zeta+1}^{\infty}\frac{(\|A\|_\infty\,\delta)^i}{i!} \tag{25}$$

$$\le \frac{(\|A\|_\infty\,\delta)^{\zeta+1}}{(\zeta+1)!}\cdot(1 + \varepsilon + \varepsilon^2 + \cdots) \tag{26}$$

$$\le \frac{(\|A\|_\infty\,\delta)^{\zeta+1}}{(\zeta+1)!}\cdot\frac{1}{1-\varepsilon}, \quad \varepsilon := \frac{\|A\|_\infty\,\delta}{\zeta+2}, \tag{27}$$

where the order of approximation ζ is chosen such that $\varepsilon < 1$. Denote (27) as

$$\psi_\delta := \frac{(\|A\|_\infty\,\delta)^{\zeta+1}}{(\zeta+1)!}\cdot\frac{1}{1-\varepsilon}. \tag{28}$$

Similarly we have that

$$\begin{aligned}\left\|\int_0^\delta E_\lambda d\lambda\right\|_\infty &\le \int_0^\delta\|E_\lambda\|_\infty d\lambda \le \int_0^\delta\sum_{i=\zeta+1}^{\infty}\frac{\|A\|_\infty^i\lambda^i}{i!}d\lambda\\ &= \sum_{i=\zeta+1}^{\infty}\int_0^\delta\frac{\|A\|_\infty^i\lambda^i}{i!}d\lambda = \sum_{i=\zeta+1}^{\infty}\frac{\|A\|_\infty^i\delta^{i+1}}{(i+1)!}\\ &\le \frac{\|A\|_\infty^{\zeta+1}\delta^{\zeta+2}}{(\zeta+2)!}\cdot\left(1 + \eta + \eta^2 + \cdots\right) \end{aligned} \tag{29}$$

$$\le \frac{\|A\|_\infty^{\zeta+1}\delta^{\zeta+2}}{(\zeta+2)!}\cdot\frac{1}{1-\eta}, \ \ \eta := \frac{\|A\|_\infty\,\delta}{\zeta+3} \tag{30}$$

(and note that since $\eta < \varepsilon, \varepsilon < 1 \Rightarrow \eta < 1$). Denote (30) as

$$\phi_\delta := \frac{\|A\|_\infty^{\zeta+1}\delta^{\zeta+2}}{(\zeta+2)!}\cdot\frac{1}{1-\eta}. \tag{31}$$

Substituting (24), (27), and (30) in (23) and (22) for the infinity norm we obtain a conservative upper-bound on the error:

$$\|\mathbf{e}_1\|_\infty \le \psi_\delta\,\|\mathbf{x}_0\|_\infty + \phi_\delta\sup_{u\in\mathcal{U}}\|B\mathbf{u}\|_\infty =: \tilde{\gamma}_1 \tag{32}$$

and, for $k > 1$,

$$\begin{aligned}\|\mathbf{e}_k\|_\infty \le &\sum_{l=0}^{k-1}\binom{k}{l}\left\|A_\delta^l\right\|_\infty\psi_\delta^{k-l}\|\mathbf{x}_0\|_\infty \\ &+\sum_{i=1}^{k-1}\left[\sum_{l=0}^{i-1}\binom{i}{l}\left\|A_\delta^l\right\|_\infty\psi_\delta^{i-l}\left\|\sum_{j=0}^{\zeta}\frac{A^j\delta^{j+1}}{(j+1)!}\right\|_\infty \right.\\ &\left.+\left(\|A_\delta\|_\infty + \psi_\delta\right)^i\phi_\delta\right]\sup_{u\in\mathcal{U}}\|B\mathbf{u}\|_\infty \\ =:&\ \tilde{\gamma}_k. \end{aligned} \tag{33}$$

Using $\tilde{\gamma}_k$ in (32)–(33) in Lemma 2 will allow us to check for feasibility of a given point $\mathbf{x}_0$ via only the nominal (finite-difference

approximation) system, while ensuring that safety will not be violated due to discretization error.

REMARK 1. *The bound $\tilde{\gamma}_k$ is asymptotically tight in the sense that for any k, $\lim_{\zeta\to\infty}\tilde{\gamma}_k = 0$.*

In practice, the chosen order of approximation ζ must be large enough such that it ensures convergence of the power series in (26) (and thus (29)) as well as non-emptiness of the eroded sets $\mathcal{A}^k_\downarrow(M,\delta,\tilde{\gamma}_k)$ in Lemma 2.

2.1.3 Verifying Feasibility via Forward Simulation

We can now use the discretized model

$$\hat{\mathbf{x}}_{k+1} = A_\delta \hat{\mathbf{x}}_k + \underbrace{\left(\int_0^\delta A_\lambda d\lambda \cdot B\right)}_{B_\delta}\mathbf{u}_k \tag{34}$$

to determine feasibility of a given initial condition $\mathbf{x}_0$ without worrying about the impact on safety of the discretization error or the behavior of the continuous trajectory of (1) in between sampling times. If there exists a sequence of controls $\{\mathbf{u}_k\}$ so that the nominal states $\hat{\mathbf{x}}_k$ of the closed-loop system belong to the precomputed sets $\mathcal{A}^k_\downarrow(M,\delta,\tilde{\gamma}_k)$ described above, then via Lemmas 2 and 1 the trajectories of the original sampled-data system will never exit $\mathcal{A}$.

We can construct the prediction equation as

$$\begin{bmatrix}\hat{\mathbf{x}}_0\\ \hat{\mathbf{x}}_1\\ \hat{\mathbf{x}}_2\\ \vdots\\ \hat{\mathbf{x}}_{N_\delta}\end{bmatrix} = \underbrace{\begin{bmatrix} I\\ A_\delta\\ A_\delta^2\\ \vdots\\ A_\delta^{N_\delta}\end{bmatrix}}_{G}\mathbf{x}_0 + \underbrace{\begin{bmatrix} 0 & 0 & \dots & 0\\ B_\delta & 0 & \dots & 0\\ A_\delta B_\delta & B_\delta & \dots & 0\\ \vdots & \vdots & \ddots & \vdots\\ A_\delta^{N_\delta-1}B_\delta & A_\delta^{N_\delta-2}B_\delta & \dots & B_\delta\end{bmatrix}}_{H}\underbrace{\begin{bmatrix}\mathbf{u}_0\\ \mathbf{u}_1\\ \vdots\\ \mathbf{u}_{N_\delta-1}\end{bmatrix}}_{U}, \tag{35}$$

and formulate the finite horizon feasibility program

$$\min_U \quad 0 \tag{36a}$$
$$\text{s.t.} \quad U \in \mathcal{U}^{N_\delta} \tag{36b}$$
$$\hat{\mathbf{x}}_k \in \mathcal{A}^k_\downarrow(M,\delta,\tilde{\gamma}_k), \quad k=0,\dots,N_\delta \tag{36c}$$
$$\begin{bmatrix}\hat{\mathbf{x}}_0 & \cdots & \hat{\mathbf{x}}_{N_\delta}\end{bmatrix}^T = G\mathbf{x}_0 + HU. \tag{36d}$$

THEOREM 1. *If $\exists U^*$ satisfying* (36)*, then $\mathbf{x}_0 \in \mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})$.*

PROOF. The proof follows directly from Lemmas 1 and 2. For a fixed input sequence U the prediction equation (35) generates a forward simulation of the trajectory of the finite-difference system (34) over the horizon $[0, N_\delta]$ corresponding to the continuous time horizon $[0, N_\delta\delta] = \mathbb{T}$. Constraint (36b) ensures that this input sequence is point-wise admissible, while constraint (36c) restricts $\hat{\mathbf{x}}_k$ so that the trajectory of the sampled-data system (1) evaluated at t_k belongs to $\mathcal{A}_\downarrow(M,\delta)$ since via Lemma 2, $\hat{\mathbf{x}}_k \in \mathcal{A}^k_\downarrow(M,\delta,\tilde{\gamma}_k) \Rightarrow \mathbf{x}_k = \mathbf{x}(t_k) \in \mathcal{A}_\downarrow(M,\delta)$. Lemma 1 then automatically guarantees $\mathbf{x}(t) \in \mathcal{A}\ \forall t \in [0,\tau]$ which implies $\mathbf{x}_0 \in \mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})$. □

In terms of practical implementation, if $\mathcal{U} = \{\mathbf{u} \in \mathbb{R}^m \mid \mathbf{u}_{\min} \le \mathbf{u} \le \mathbf{u}_{\max}\}$ for some limits $\mathbf{u}_{\min}$ and $\mathbf{u}_{\max}$, and the sets $\mathcal{A}^k_\downarrow(M,\delta,\tilde{\gamma}_k)$ are represented by polytopes $\{\mathbf{x} \mid P_k\mathbf{x} \le \mathbf{p}_k\}$, then to determine if a given point $\mathbf{x}_0$ is feasible we can simply check for feasibility of the linear convex program

$$\min_U \quad 0 \tag{37a}$$
$$\text{s.t.} \quad \mathbf{1}_{N_\delta}\otimes\mathbf{u}_{\min} \le I_{N_\delta m}U \le \mathbf{1}_{N_\delta}\otimes\mathbf{u}_{\max} \tag{37b}$$
$$P_k\hat{\mathbf{x}}_k \le \mathbf{p}_k, \quad k=0,\dots,N_\delta \tag{37c}$$
$$\begin{bmatrix}\hat{\mathbf{x}}_0 & \cdots & \hat{\mathbf{x}}_{N_\delta}\end{bmatrix}^T = G\mathbf{x}_0 + HU, \tag{37d}$$

where $\otimes$ denotes the Kronecker product.[2]

In the algorithm below we will encapsulate Theorem 1 to determine the feasibility of a given sample point $\mathbf{x}_0$ in the subroutine FEASIBLE. Its computational complexity is proportional to the complexity of the program (36), which in turn depends only on the shape of the sets $\mathcal{U}$ and $\mathcal{A}$.

2.2 Additional Subroutines

In addition to the feasibility-checking subroutine FEASIBLE described above, our algorithm will also make use of the following subroutines which we describe in less detail due to their more straightforward nature.

- FIND-INTERSECTION-ON-BOUNDARY$(\mathcal{C},\vec{r})$ – Input: A convex compact set $\mathcal{C} \subset \mathcal{X}$, and a ray $\vec{r}$ with origin $\mathbf{r}_0 \in \mathcal{C}$. Returns a point $\mathbf{x}$ along the ray $\vec{r}$ on ∂C. We note that since $\mathcal{C}$ is convex and compact, and since $\mathbf{r}_0 \in \mathcal{C}$, there is exactly one such point $\mathbf{x}$. Runs in time proportional to the number of faces of $\mathcal{C}$ if $\mathcal{C}$ is a polytope, and constant time if $\mathcal{C}$ is an ellipsoid.
- SAMPLE-RAY$(\mathbf{x})$ – Input: A point $\mathbf{x} \in \mathbb{R}^n$. First samples a point $\mathbf{r}_d \in \mathcal{S}_2^{n-1}$ at random[3] from the Euclidean unit ball in $\mathbb{R}^n$, then returns the ray $\vec{r} = \{\mathbf{x} + s\mathbf{r}_d\}$, i.e. a ray with random direction and origin $\mathbf{x}$. Runs in time linear in n.

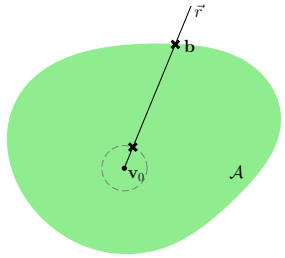
We will also use BISECTION-FEASIBILITY-SEARCH$(\mathbf{a},\mathbf{b},\mathcal{A})$ as defined in Algorithm 1, which searches along the line segment between $\mathbf{a}$ and $\mathbf{b}$ to find a point $\mathbf{c}$ such that $\mathrm{dist}_p(\mathbf{c}, \partial\,\mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})) < \epsilon$.

Algorithm 1 Determines an ϵ-accurate intersection of $\partial\,\mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})$ on the line segment between $\mathbf{a}$ and $\mathbf{b}$.

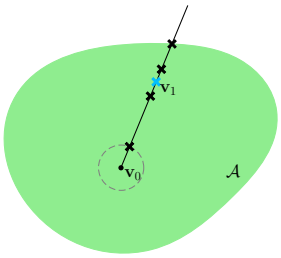
```
function BISECTION-FEASIBILITY-SEARCH(a, b, A)
    c ← a + (b − a)/2
    if FEASIBLE(c, A) then
        if dist_p(b, c) < ε then
            return c
        else
            return BISECTION-FEASIBILITY-SEARCH(c, b, A)
        end if
    else
        return BISECTION-FEASIBILITY-SEARCH(a, c, A)
    end if
end function
```

[2] A similar quadratically-constrained program can be formulated if the state or input constraints are instead represented by ellipsoids. Additionally, if the system being analyzed is a switched system then the LP (37) becomes a Mixed-Integer LP. Both extensions are straightforward, and are omitted due to space constraints.

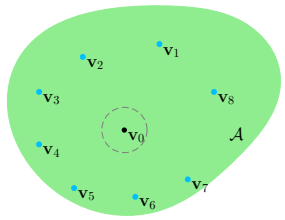
[3] Any probability distribution may be used; for our implementation in the examples that follow we used the uniform distribution.



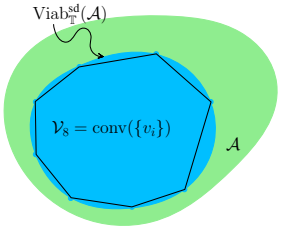
(a) Sample a ray $\vec{r}$ at random, and find $\mathbf{b}$, its intersection with $\mathcal{A}$ (lines 4 and 5).



(b) Perform a bisection search along $\vec{r}$ between $\mathbf{v}_0$ and $\mathbf{b}$ to find $\mathbf{v}_1 \in \text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$, where $\text{dist}_p(\mathbf{v}_1, \partial\,\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})) < \epsilon$ (line 6).



(c) Repeat this procedure for N samples.



(d) Return the polytope $\mathcal{V}_N := \text{conv}(\{\mathbf{v}_0, \ldots, \mathbf{v}_N\})$.

Figure 3: Illustration of Algorithm 2.

2.3 The Algorithm

Given the subroutines described above, our algorithm for computing a polytopic under-approximation of the viability kernel for an LTI sampled-data system is given by Algorithm 2. It proceeds as follows (see Figure 3). We assume as input a description of the safe constraint set $\mathcal{A}$, as well as some initial point $\mathbf{v}_0 \in \text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$.[4] We then construct a polytopic approximation of $\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$ by iteratively sampling a direction $\vec{r}$ (line 4); finding the point $\mathbf{b} \in \mathbb{R}^n$ where the ray with that direction (centered at $\mathbf{v}_0$) intersects the boundary of the safe set $\mathcal{A}$ (line 5); and then performing a bisection search between $\mathbf{v}_0$ and $\mathbf{b}$ until we find a point $\mathbf{v}_i$ such that $\mathbf{v}_i \in \text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$ and $\text{dist}_p(\mathbf{v}_i, \partial\,\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})) < \epsilon$ in some desired norm $\|\cdot\|_p$ (see Algorithm 1). By taking the convex hull of these points $\{\mathbf{v}_0, \ldots, \mathbf{v}_N\}$ we arrive at a polytope $\mathcal{V}_N \subseteq \text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$ which converges to $\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$ (in a manner we will formalize after we have proved the algorithm's correctness).

3. ALGORITHM PROPERTIES

3.1 Correctness

THEOREM 2. *Given convex sets $\mathcal{A}$ and $\mathcal{U}$ and an initial point $\mathbf{v}_0 \in \text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$, and let $\mathcal{V}_N = \text{POLYTOPIC-APPROX}(\mathcal{A}, \mathbf{v}_0, N)$, then $\mathcal{V}_N \subseteq \text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$.*

PROOF. We use proof by induction. First, it is obvious that for $N = 0$, $\mathcal{V}_0 = \text{POLYTOPIC-APPROX}(\mathcal{A}, \mathbf{v}_0, 0) = \text{conv}(\{\mathbf{v}_0\}) \subseteq \text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$, since we are given that $\mathbf{v}_0 \in \text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$.

[4]For linear systems we have that if $0 \in \mathcal{A}$ and $0 \in \mathcal{U}$, then $0 \in \text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$, and thus we can use $\mathbf{v}_0 = 0$. However, if the origin is not inside the constraint set, then we assume we can sample points $\mathbf{x}$ at random over the interior of $\mathcal{A}$ and check if they are inside $\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$ using the subroutine $\text{FEASIBLE}(\mathbf{x}, \mathcal{A})$.

Algorithm 2 Computes a polytopic under-approximation of $\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$ with at most $N+1$ vertices

1: **procedure** POLYTOPIC-APPROX($\mathcal{A}, \mathbf{v}_0, N$)
2: $\quad\mathcal{V}_0 \leftarrow \{\mathbf{v}_0\}$
3: $\quad$**for** $i = 1$ to N **do** $\quad\triangleright\ N$ samples
4: $\quad\quad\vec{r} \leftarrow \text{SAMPLE-RAY}(\mathbf{v}_0)$
5: $\quad\quad b \leftarrow \text{FIND-INTERSECTION-ON-BOUNDARY}(\mathcal{A}, \vec{r})$
6: $\quad\quad\mathbf{v}_i \leftarrow \text{BISECTION-FEASIBILITY-SEARCH}(\mathbf{v}_0, \mathbf{b}, \mathcal{A})$
7: $\quad\quad\mathcal{V}_i \leftarrow \text{conv}(\mathcal{V}_{i-1} \cup \{\mathbf{v}_i\})$
8: $\quad$**end for**
9: $\quad$**return** $\mathcal{V}_N$
10: **end procedure**

Next assume that $\mathcal{V}_N = \text{conv}(\{\mathbf{v}_0, \ldots, \mathbf{v}_N\}) \subseteq \text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$, and that we have $\mathbf{v}_{N+1} = \text{BISECTION-FEASIBILITY-SEARCH}(\mathbf{v}_0, \mathbf{b}, \mathcal{A})$ for some point $\mathbf{b} \in \partial\mathcal{A}$ (as returned by FIND-INTERSECTION-ON-BOUNDARY). Then since BISECTION-FEASIBILITY-SEARCH only returns points which are inside $\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$, we have $\mathbf{v}_{N+1} \in \text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$. Let $\mathcal{V}_{N+1} = \text{conv}(\mathcal{V}_N \cup \{\mathbf{v}_{N+1}\})$. Now since $\mathcal{V}_{N+1}$ is convex, $\forall \mathbf{x}_0 \in \mathcal{V}_{N+1}$, $\exists \mathbf{x}_0' \in \mathcal{V}_N$ and $\exists \theta \in [0,1]$ such that $\mathbf{x}_0 = \theta\mathbf{x}_0' + (1-\theta)\mathbf{v}_{N+1}$. For $\mathbf{x}_0'$ we know (by the induction hypothesis) that $\exists \mathbf{u}_{x_0'}(\cdot) \in \mathscr{U}^{\text{pw}}_{\mathbb{T}}$ such that

$$\mathbf{x}_{\mathbf{x}_0'}^{\mathbf{u}_{\mathbf{x}_0'}}(t) = e^{At}\mathbf{x}_0' + \int_0^t e^{A(t-r)}B\mathbf{u}_{\mathbf{x}_0'}(r)dr \in \mathcal{A} \quad \forall t \in \mathbb{T}. \tag{38}$$

For $\mathbf{v}_{N+1}$ we also know $\exists \mathbf{u}_{\mathbf{v}_{N+1}}(\cdot) \in \mathscr{U}^{\text{pw}}_{\mathbb{T}}$ such that

$$\mathbf{x}_{\mathbf{v}_{N+1}}^{\mathbf{u}_{\mathbf{v}_{N+1}}}(t) = e^{At}\mathbf{v}_{N+1} + \int_0^t e^{A(t-r)}B\mathbf{u}_{\mathbf{v}_{N+1}}(r)dr \in \mathcal{A} \ \forall t \in \mathbb{T}. \tag{39}$$

Let $\mathbf{u}_{\mathbf{x}_0}(\cdot) = \theta\mathbf{u}_{\mathbf{x}_0'}(\cdot) + (1-\theta)\mathbf{u}_{\mathbf{v}_{N+1}}(\cdot) \in \mathscr{U}^{\text{pw}}_{\mathbb{T}}$, then

$$\begin{aligned}\mathbf{x}_{\mathbf{x}_0}^{\mathbf{u}_{\mathbf{x}_0}}(t) &:= e^{At}\mathbf{x}_0 + \int_0^t e^{A(t-r)}B\mathbf{u}_{\mathbf{x}_0}(r)dr \\ &= e^{At}\left(\theta\mathbf{x}_0' + (1-\theta)\mathbf{v}_{N+1}\right) + \\ &\qquad \int_0^t e^{A(t-r)}B\left(\theta\mathbf{u}_{\mathbf{x}_0'}(r) + (1-\theta)\mathbf{u}_{\mathbf{v}_{N+1}}(r)\right)dr \\ &= \theta\mathbf{x}_{\mathbf{x}_0'}^{\mathbf{u}_{\mathbf{x}_0'}}(t) + (1-\theta)\mathbf{x}_{\mathbf{v}_{N+1}}^{\mathbf{u}_{\mathbf{v}_{N+1}}}(t) \in \mathcal{A} \quad \forall t \in \mathbb{T}\end{aligned} \tag{40}$$

since $\mathcal{A}$ and $\mathcal{U}$ are convex and compact. Thus $\mathbf{u}_{\mathbf{x}_0}(\cdot)$ is safety-preserving and $\mathbf{x}_0 \in \text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$. Because $\mathbf{x}_0$ was chosen arbitrarily in $\mathcal{V}_{N+1}$, we conclude that $\mathcal{V}_{N+1} \subseteq \text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$. □

3.2 Convergence

One of the striking features of our algorithm is that it is random; additional points on the boundary of the polytopic approximation of $\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$ are generated based on random sampling. This is necessary because $\partial\,\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$ is unknown *a priori* so it is impossible to know deterministically what points to sample in order to construct a polytope that converges to $\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$ as quickly as possible. In fact any algorithm which deterministically chooses such points could be presented with a safe set $\mathcal{A}$ and system dynamics for which the algorithm would converge arbitrarily poorly. This result comes from the literature of estimating the volume of convex bodies using a *separation oracle*, which is a function that accepts as input a convex set and a point, and returns whether or not that point is inside the convex set. [5] More specifically it can be shown

[5]Such an oracle is primarily used when the representation of the body is too complicated to be tractable for analytical volume cal-

that for any algorithm that deterministically queries a separation oracle a polynomial number of times in order to build a polytopic approximation, the error (the difference in volume between the approximation and the true set) could be exponential in the number of dimensions [3]. Random algorithms, on the other hand, can perform provably better [7], and we use a proof from the literature [30] to prove our algorithm's asymptotic convergence. First, however, we must introduce several new definitions.

First, let $\widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})$ be the subset of the viability kernel our algorithm actually approximates, i.e.

$$\widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A}) := \{\mathbf{x}_0 \in \mathcal{A}_{\downarrow}(M,\delta) \mid \exists \mathbf{u}(\cdot) \in \mathscr{U}^{\mathrm{pw}}_{\mathbb{T}}, \forall k \in \{0,\dots,N_\delta\}, \mathbf{x}(t_k) \in \mathcal{A}_{\downarrow}(M,\delta)\}. \tag{41}$$

While the true viability kernel $\mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})$ encompasses all initial conditions for which a piecewise constant control keeps $\mathbf{x}(t) \in \mathcal{A}$, the set $\widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})$ only encompasses initial conditions for which $\mathbf{x}(t_k) \in \mathcal{A}_{\downarrow}(M,\delta)$ (which is a sufficient — but not necessary — condition to imply $\mathbf{x}(t) \in \mathcal{A}$; cf. Lemma 1). Thus, $\widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A}) \subseteq \mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})$. Define the volumetric error between these two sets as

$$e_{M\delta} := \mathrm{vol}(\mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})) - \mathrm{vol}(\widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})). \tag{42}$$

and note that it depends only on the term $M\delta$, due to the definition of $\mathcal{A}_{\downarrow}(M,\delta)$ in (4).

Of course for a given choice of ζ for the discretization (Section 2.1.2) and ϵ in the bisection search (Algorithm 1), the points our algorithm samples will not be precisely on the boundary $\partial\widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})$, so it will also be useful to define $\widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)$ as the subset of $\widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})$ which our algorithm approximates as $N \to \infty$. Thus using the extended notation $\mathcal{V}_N^{\zeta,\epsilon} = \textsc{Polytopic-Approx}(\mathcal{A},\mathbf{v}_0,N)$ for a given choice of ζ and ϵ:[6]

$$\widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon) := \lim_{N\to\infty} \mathcal{V}_N^{\zeta,\epsilon} \tag{43}$$

$$\lim_{\substack{\zeta\to\infty\\ \epsilon\to 0}} \widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon) = \widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A}). \tag{44}$$

Finally we will need the key lemma which describes the *convergence rate* of random algorithms for estimating the boundary of a convex body, which we restate from [30], omitting some of the details that are unnecessary for our purposes:

LEMMA 3 ([30]). *Let $\mathcal{C}$ be a convex body in $\mathbb{R}^n$ with $\partial\mathcal{C}$ C^2, let $h : \partial\mathcal{C} \to \mathbb{R}_+$ be a probability density function defined on $\partial\mathcal{C}$, let $\mathbb{P}_h$ be the probability measure defined by h, and let $\mathbb{E}(h,N)$ be the expected volume of the convex hull of N points chosen randomly on $\partial\mathcal{C}$ with respect to $\mathbb{P}_h$. Then*

$$\lim_{N\to\infty} \left(\mathrm{vol}(\mathcal{C}) - \mathbb{E}(h,N)\right) N^{\frac{2}{n-1}} = c_n(\mathcal{C}), \tag{45}$$

where $c_n(\mathcal{C})$ is a constant which depends only on the dimension n, the distribution of h, and the shape of $\mathcal{C}$ (more specifically its Gauss-Kronecker curvature).

The key point of this lemma is that any algorithm which builds a polytopic approximation to a convex body by randomly sampling points on its boundary will converge with a rate that is proportional to $1/N^{\frac{2}{n-1}}$. Additionally, [30] also shows that this convergence rate is optimal (up to a numerical constant) with respect to all algorithms which build polytopic approximations via boundary sampling.

With this lemma and these definitions in hand, we can state the convergence of our algorithm.

PROPOSITION 1 (RATE OF CONVERGENCE). *Let $e_{\mathrm{vol}}(N,\zeta,\epsilon)$ represent the volumetric error between the viability kernel and the result of our algorithm, minus the error $e_{M\delta}$ between the true viability kernel and $\widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})$, i.e.*

$$e_{\mathrm{vol}}(N,\zeta,\epsilon) := \mathrm{vol}(\mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A})) - \mathrm{vol}(\mathcal{V}_N^{\zeta,\epsilon}) - e_{M\delta}. \tag{46}$$

Then our algorithm converges as

$$\lim_{\substack{N\to\infty\\ \zeta\to\infty\\ \epsilon\to 0}} e_{\mathrm{vol}}(N,\zeta,\epsilon) N^{\frac{2}{n-1}} = c_n(\mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A}), M\delta), \tag{47}$$

where $c_n(\mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A}), M\delta)$ is a constant dependent on the dimension n, the shape of the viability kernel, and the value $M\delta$.

Proposition 1 asserts that the volumetric error between the outcome of our algorithm and the true viability kernel asymptotically converges at the exponential rate of $c_n(\mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A}))/N^{\frac{2}{n-1}}$ to a numerical constant (due to the sampled-data nature of the system).

To prove this proposition we first need to introduce the following lemma, which introduces a fictitious source of error $\epsilon_{\mathrm{smooth}}$ between the result of our algorithm and the viability kernel:

LEMMA 4 ([8, 9]). *For every compact convex set $\mathcal{C}$, there exists a compact convex set $\tilde{\mathcal{C}} \subseteq \mathcal{C}$ whose boundary $\partial\tilde{\mathcal{C}}$ is C^2, and $\mathrm{vol}(\mathcal{C}) - \mathrm{vol}(\tilde{\mathcal{C}}) = e_{\mathrm{smooth}}$ for some arbitrarily small positive scalar constant e_{smooth}.*

We use Lemma 4 to define a convex body $\widetilde{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)$ that is a C^2 approximation of $\widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)$ such that

$$\mathrm{vol}(\widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)) - \mathrm{vol}(\widetilde{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)) = e_{\mathrm{smooth}}. \tag{48}$$

We will also require the following lemma (which we state without proof):

LEMMA 5. *Any compact convex set $\mathcal{C}$ with $\mathbf{v}_0 \in \mathcal{C}$ is homeomorphic to $\mathcal{B}_2^n(\mathbf{v}_0,1)$ [5], and in particular taking $\mathcal{C} = \widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)$, we can define the invertible mapping $\mathbf{m} : S_2^{n-1}(\mathbf{v}_0) \to \partial\widehat{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)$ as $\mathbf{m}(\mathbf{r}_d) \equiv \textsc{Bisection-Feasibility-Search}(\mathbf{v}_0, \textsc{Find-Intersection-on-Boundary}(\mathcal{C},\vec{r}),\mathcal{C})$, where $\vec{r}$ has origin $\mathbf{v}_0 \in \mathcal{C}$ and direction $\mathbf{r}_d$.*

We are now ready to prove Proposition 1.

PROOF OF PROPOSITION 1. By Lemma 3, if we have a probability density function $\tilde{g}$ that samples from $\partial\widetilde{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)$, then we have

$$\lim_{N\to\infty} \left(\mathrm{vol}(\widetilde{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)) - \mathbb{E}(\tilde{g},N)\right) N^{\frac{2}{n-1}} = \tilde{c}_n(\widetilde{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon))) \tag{49}$$

for some constant $\tilde{c}_n(\widetilde{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon))$ which depends only on the dimension n, the distribution $\tilde{g}$, and the shape of $\widetilde{\mathrm{Viab}}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)$.

culations, or as in our case (where $\textsc{Feasible}(\mathbf{x},\mathcal{A})$ plays the role of the separation oracle) when the representation of the body is not actually known in form besides the oracle.

[6]Note that Theorem 2, restated in terms of the extended notation, asserts that $\mathcal{V}_N^{\zeta,\epsilon} \subseteq \mathrm{Viab}^{\mathrm{sd}}_{\mathbb{T}}(\mathcal{A}), \forall \zeta,\epsilon,N$.

Now let $h(\mathbf{x})$ be the probability distribution used by SAMPLE-RAY($\mathbf{v}_0$) to generate samples on the unit sphere. Then since the mapping $\mathbf{m}(\mathbf{r}_d)$ from Lemma 5 returns points on $\widehat{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)$, we can use $\mathbf{m}$ to perform a change of variables to define a new probability density function $g(\mathbf{m}(\mathbf{r}_d))$ on $\partial\widehat{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)$ [27]. Then by Lemma 4, we can take $e_{\text{smooth}} \to 0$, and thus $\widetilde{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon) \to \widehat{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)$ and $\tilde{g} \to g$. Thus we can write (49) as:

$$\lim_{N\to\infty} \left(\text{vol}(\widehat{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)) - \mathbb{E}(g,N)\right) N^{\frac{2}{n-1}} = \tilde{c}_n(\widehat{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon))), \quad (50)$$

which, since $\mathbb{E}(g,N) = \text{vol}(\mathcal{V}_N^{\zeta,\epsilon})$, becomes

$$\lim_{N\to\infty} \left(\text{vol}(\widehat{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon)) - \text{vol}(\mathcal{V}_N^{\zeta,\epsilon})\right) N^{\frac{2}{n-1}} = \tilde{c}_n(\widehat{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A},\zeta,\epsilon))). \quad (51)$$

Then by (44), taking the limit as $\zeta \to \infty$ and $\epsilon \to 0$ on both sides of (51) gives:

$$\lim_{\substack{N\to\infty \\ \zeta\to\infty \\ \epsilon\to 0}} \left(\text{vol}(\widehat{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})) - \text{vol}(\mathcal{V}_N^{\zeta,\epsilon})\right) N^{\frac{2}{n-1}} = \tilde{c}_n(\widehat{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})). \quad (52)$$

Now, by (42), we can replace $\text{vol}(\widehat{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A}))$ to get

$$\lim_{\substack{N\to\infty \\ \zeta\to\infty \\ \epsilon\to 0}} \left(\text{vol}(\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})) - e_{M\delta} - \text{vol}(\mathcal{V}_N^{\zeta,\epsilon})\right) N^{\frac{2}{n-1}} = \tilde{c}_n(\widehat{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})). \quad (53)$$

Since the shape of $\widehat{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$ depends only on the shape of $\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})$ and the value of $M\delta$, we can write $\tilde{c}_n(\widehat{\text{Viab}}^{\text{sd}}_{\mathbb{T}}(\mathcal{A})) := c_n(\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A}), M\delta)$ for some appropriate function c_n, and we arrive at

$$\lim_{\substack{N\to\infty \\ \zeta\to\infty \\ \epsilon\to 0}} e_{\text{vol}}(N,\zeta,\epsilon) N^{\frac{2}{n-1}} = c_n(\text{Viab}^{\text{sd}}_{\mathbb{T}}(\mathcal{A}), M\delta). \quad (54)$$

□

It is worth emphasizing once again that due to the results in [30], this convergence rate is optimal (up to a multiplicative constant depending on the probability density function used for sampling); no other algorithm which approximates the viability kernel by sampling from its boundary will be able to converge at a faster rate.

3.3 Computational Complexity & Scalability

The run time complexity of our algorithm (for a fixed number of time sampling intervals N_δ) is $O(N\log(d)\Phi(n))$, where

- N is the number of samples/vertices,
- d is the "diameter" of the set $\mathcal{A}$, and
- Φ is the complexity of the feasibility program (36) as a function of the state dimension n.

That is, the algorithm runs in time linear in the number of samples N, logarithmic in the diameter d of $\mathcal{A}$ due to the bisection search, and proportional to Φ in the complexity of the appropriate feasibility program (36). For instance with polytopic constraints the feasibility problem is an LP and thus the run time complexity of the proposed algorithm is super-quadratic but sub-cubic in n. This is a direct improvement over existing techniques for approximating the sampled-data viability kernel.

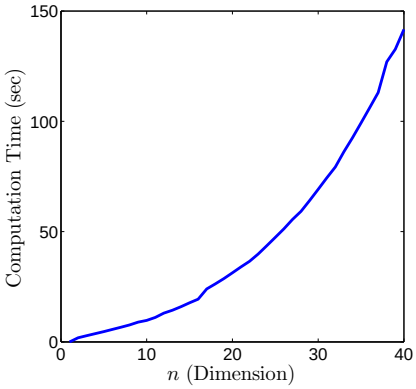


Figure 4: Run time of the proposed algorithm for a chain of n-integrators.

We note that the spatial complexity of the algorithm is still exponential in the number of samples N as the dimension n increases: The convergence rate (47) says that to keep the same approximation accuracy as $n \to kn$, we would need an increase of $N \to N^{\frac{kn-1}{n-1}}$ in the number of samples. However, the fact that we only store samples on the boundary of the viability kernel to describe that set (as opposed to storing a grid of the entire set $\mathcal{A}$ and possibly beyond) requires significantly less memory than conventional approaches such as the level-set method presented in [26]. Additionally, the computed approximation is far more accurate (and possibly more scalable) than the piecewise ellipsoidal technique also presented in [26]. In essence our algorithm strikes a balance between accuracy and scalability that to the best of our knowledge no other algorithm has yet achieved for sampled-data systems.

To examine the scalability of our algorithm empirically, consider the chain of n-integrators $\partial^n x/\partial t^n = u$ with input and state constraints $\mathcal{U} = [-0.15, 0.15]$ and $\mathcal{A} = \{\mathbf{x} \mid \|\mathbf{x}\|_\infty \leq 0.5\}$. The trajectory $\mathbf{x}(\cdot)$ is measured every $\delta = 0.05$ s and safety is to be maintained over $\mathbb{T} = [0,1]$. We use a discretization order of $\zeta = 4$, bisection accuracy of $\epsilon = 0.01$ and employ YALMIP [20] to implement (36) and MPT [18] for simple operations with polytopes. All of these parameters are kept constant as we increase the dimension n and the number of samples $N = 2n$ to examine the scalability of our algorithm. The results are shown in Figure 4. The algorithm was implemented in MATLAB version 7.13 (R2011b) and tested on an Intel Core i7 at 2.9 GHz with 16 GB RAM running 64-bit Windows 7 Professional (without optimizing the code for speed). As would be expected given that the feasibility problem is a linear program for this example, the resulting runtime is a super-quadratic but sub-cubic function in n.

4. NUMERICAL EXAMPLE

To further examine the performance of the algorithm, we will examine how it performs on a trivial system, the double-integrator:

$$\dot{\mathbf{x}} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u. \quad (55)$$

Consider a similar setting as before, where $\mathcal{U} = [-0.15, 0.15]$, $\mathcal{A} = \{\mathbf{x} \mid \|\mathbf{x}\|_\infty \leq 0.5\}$, the sampling interval is $\delta = 0.05$ s, and safety is to be maintained over $\mathbb{T} = [0,1]$. The bound M on the vector field is simply calculated as 0.5 in the infinity norm. Using $\zeta = 4$th order discretization and $\epsilon = 0.01$-accurate bisection

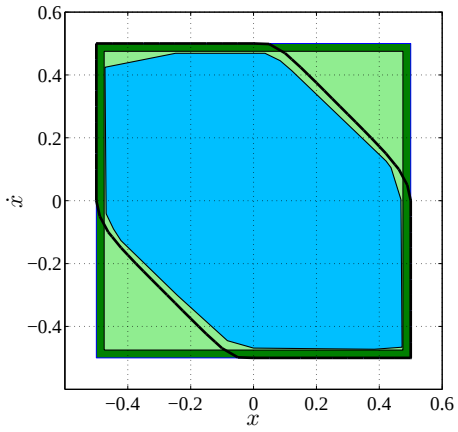


Figure 5: Polytopic under-approximation $\mathcal{V}_N$ (the inner-most set in light blue color) of $\mathrm{Viab}_{\mathbb{T}}^{\mathrm{sd}}(\mathcal{A})$ for the double-integrator using $N = 44$ samples via Algorithm 2. The outer-most set (dark green) is $\mathcal{A}$, separated from the eroded set $\mathcal{A}_{\downarrow}(M, \delta)$ (light green) by a margin determined by $M\delta$. The sampled-data level-set approximation [26] is also shown (outlined by thick black lines), though this approximation is not conservative; cf. [26, Section 5.4.4]).

search we approximate $\mathrm{Viab}_{\mathbb{T}}^{\mathrm{sd}}(\mathcal{A})$ by the set $\mathcal{V}_N$ via $N = 44$ randomly generated samples in approximately 28 seconds (Figure 5).

A quantitative upper-bound on the total volumetric error between $\mathcal{V}_N$ and the true viability kernel can be formulated using the following procedure.[7] Let L be the number of faces in $\mathcal{V}_N$. Consider the facet representation of $\mathcal{V}_N$, i.e. $\mathcal{V}_N = \{\mathbf{x} \mid F_{\mathcal{V}_N}\mathbf{x} \leq \mathbf{f}_{\mathcal{V}_N}\}$ with

$$F_{\mathcal{V}_N} = \begin{bmatrix} F_{\mathcal{V}_{N\,1}} \\ F_{\mathcal{V}_{N\,2}} \\ \vdots \\ F_{\mathcal{V}_{N\,L}} \end{bmatrix} \in \mathbb{R}^{L\times n}, \quad \mathbf{f}_{\mathcal{V}_N} = \begin{bmatrix} f_{\mathcal{V}_{N\,1}} \\ f_{\mathcal{V}_{N\,2}} \\ \vdots \\ f_{\mathcal{V}_{N\,L}} \end{bmatrix} \in \mathbb{R}^{L}. \tag{56}$$

Let $F_{\mathcal{V}_N\setminus i} \in \mathbb{R}^{(L-1)\times n}$ and $\mathbf{f}_{\mathcal{V}_N\setminus i} \in \mathbb{R}^{L-1}$ denote $F_{\mathcal{V}_N}$ and $\mathbf{f}_{\mathcal{V}_N}$ with their i-th rows removed:

$$F_{\mathcal{V}_N\setminus i} := \begin{bmatrix} F_{\mathcal{V}_{N\,1}} \\ \vdots \\ F_{\mathcal{V}_{N\,i-1}} \\ F_{\mathcal{V}_{N\,i+1}} \\ \vdots \\ F_{\mathcal{V}_{N\,L}} \end{bmatrix}, \quad \mathbf{f}_{\mathcal{V}_N\setminus i} := \begin{bmatrix} f_{\mathcal{V}_{N\,1}} \\ \vdots \\ f_{\mathcal{V}_{N\,i-1}} \\ f_{\mathcal{V}_{N\,i+1}} \\ \vdots \\ f_{\mathcal{V}_{N\,L}} \end{bmatrix}. \tag{57}$$

Suppose we also rewrite $\mathcal{A}$ using its facet representation as $\mathcal{A} = \{\mathbf{x} \mid F_{\mathcal{A}}\mathbf{x} \leq \mathbf{f}_{\mathcal{A}}\}$. Then an overestimate of the error can defined as

$$\mathrm{err} = \sum_{i=1}^{L} \left(\mathrm{vol}\left(\left\{ x \mid \begin{bmatrix} F_{\mathcal{V}_N\setminus i} \\ F_{\mathcal{A}} \end{bmatrix} x \leq \begin{bmatrix} \mathbf{f}_{\mathcal{V}_N\setminus i} \\ \mathbf{f}_{\mathcal{A}} \end{bmatrix} \right\} \right) - \mathrm{vol}(\mathcal{V}_N) \right), \tag{58}$$

i.e. for every possible face, we compute what the difference in volume would be if we removed it (while still restricting the approximation to be inside $\mathcal{A}$), and then sum up these differences. This gives a (very) conservative estimate of what the maximum possible error could be, in that it computes an over-estimate of the maximum possible volume of a convex body with the points $\{\mathbf{v}_i\}$ on its boundary. Note that this error is not guaranteed to decrease monotonically as the number of vertices increases, though we expect it to converge in the limit as $N \to \infty$. Figure 6 plots this upper-bound for the double-integrator example (averaged over 10 runs).

[7] Although this procedure is in general not scalable, it will work for this simple example.

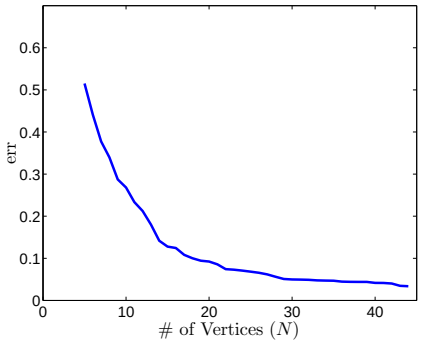


Figure 6: The upper-bound on the total volumetric error (58) between $\mathcal{V}_N$ and $\mathrm{Viab}_{\mathbb{T}}^{\mathrm{sd}}(\mathcal{A})$ for the double-integrator as a function of the number of samples N (averaged over 10 runs).

5. CONCLUSIONS, ONGOING, & FUTURE WORK

We have presented a new algorithm for computing the viability kernel for LTI sampled-data systems. This algorithm, which takes a novel sampling-based approach to calculating the viability kernel, allows the user to explicitly balance computation resources with desired accuracy. In addition to proving the algorithm's correctness, convergence properties, and complexity, we also demonstrated its use on a simple example.

One important area we are actively investigating is the extension to systems with bounded disturbances (i.e. calculating the *discriminating kernel*). One possible approach would involve further erosion of the safe sets used to check feasibility during forward simulation in the same way as the error due to discretization of the continuous system. This sort of robustification of our algorithm would also enable our system to handle nonlinear dynamics by bounding the error between the nonlinear system and its linear approximation, and treating this error as another disturbance.

There are also several areas of ongoing work for which we have results, but cannot present here due to space constraints. We have recently developed a more scalable way of over-approximating the viability kernel, again in a tight manner, by making use of support functions. By taking the difference in volume between this over-approximation and our under-approximation of the viability kernel, we have been able to construct a better approximation of the error in our results. By approximating the gradient of this error we are able to better guide the random sampling we use to generate new directions along which to constrain the under-approximation, which may result in faster convergence to the true viability kernel while making use of fewer samples. Using these improvements we have been able to apply our algorithm to a practical high-dimensional system (a 12D quadrotor model) with promising results.

We have also developed control synthesis methods based on the results of our under-approximation which construct control signals (either open-loop or feedback-based). Finally, we have also developed a sufficient condition which we can use to determine if the under-approximation of the viability kernel generated by our algorithm is control-invariant (and thus can be used as the terminal constraint in an MPC framework).

6. ACKNOWLEDGMENTS

This research was funded in part by the US National Science Foundation under CPS:ActionWebs (CNS-931843), by the US Office of Naval Research under the HUNT (N0014-08-0696) and SMARTS (N00014-09-1-1051) MURIs and by grant N00014-12-1-0609, and by the US Air Force Office of Scientific Research under the CHASE MURI (FA9550-10-1-0567).

7. REFERENCES

[1] M. Althoff, O. Stursberg, and M. Buss. Reachability analysis of linear systems with uncertain parameters and inputs. In *Proc. IEEE Conf. Decis. Contr.*, pages 726–732, 2007.

[2] J.-P. Aubin. *Viability Theory*. Systems and Control: Foundations and Applications. Birkhäuser, Boston, 1991.

[3] I. Bárány and Z. Füredi. Computing the volume is difficult. *Discrete Comput. Geom.*, 2(1):319–326, 1987.

[4] F. Blanchini and S. Miani. *Set-Theoretic Methods in Control*. Springer, 2008.

[5] G. E. Bredon. *Topology and Geometry*, volume 139. Springer, 1993.

[6] P.-A. Coquelin, S. Martin, and R. Munos. A dynamic programming approach to viability problems. In *Proc. IEEE Symp. Adapt. Dyn. Program. Reinf. Learn.*, pages 178–184, 2007.

[7] M. Dyer. Computing the volume of convex bodies: a case where randomness provably helps. *Probabilistic combinatorics and its applications*, 44:123–170, 1991.

[8] M. Ghomi. The problem of optimal smoothing for convex functions. *Proc. Am. Math. Soc.*, 130(8):2255–2260, 2002.

[9] M. Ghomi. Optimal Smoothing for Convex Polytopes. *Bull. London Math. Soc.*, 36(4):483–492, July 2004.

[10] A. Girard and C. Le Guernic. Efficient reachability analysis for linear systems using support functions. In *Proc. IFAC World Congr.*, Seoul, Korea, July 2008.

[11] A. Girard, C. Le Guernic, and O. Maler. Efficient computation of reachable sets of linear time-invariant systems with inputs. In J. Hespanha and A. Tiwari, editors, *Proc. Hybrid Syst.: Comput. Contr.*, pages 257–271. Springer-Verlag, 2006.

[12] G. Goodwin, J. Aguero, M. Cea Garridos, M. Salgado, and J. Yuz. Sampling and sampled-data models: The interface between the continuous world and digital algorithms. *IEEE Contr. Syst. Mag.*, 33(5):34–53, 2013.

[13] Z. Han and B. H. Krogh. Reachability analysis of nonlinear systems using trajectory piecewise linearized models. In *Proc. Am. Contr. Conf.*, pages 1505–1510, Minneapolis, MN, 2006.

[14] D. Henrion and M. Korda. Convex computation of the region of attraction of polynomial control systems. *preprint arXiv:1208.1751*, 2012.

[15] S. Kaynama, J. Maidens, M. Oishi, I. M. Mitchell, and G. A. Dumont. Computing the viability kernel using maximal reachable sets. In *Proc. Hybrid Syst.: Comput. Contr.*, pages 55–63, Beijing, 2012.

[16] S. Kaynama and M. Oishi. A modified Riccati transformation for decentralized computation of the viability kernel under LTI dynamics. *IEEE Trans. Autom. Contr.*, 58(11):2878–2892, 2013.

[17] A. B. Kurzhanski and P. Varaiya. Ellipsoidal techniques for reachability analysis: internal approximation. *Syst. Contr. Lett.*, 41:201–211, 2000.

[18] M. Kvasnica, P. Grieder, M. Baotić, and M. Morari. Multi-Parametric Toolbox (MPT). In R. Alur and G. J. Pappas, editors, *Proc. Hybrid Syst.: Comput. Contr., LNCS 2993*, pages 448–462, Berlin, Germany, 2004. Springer.

[19] M. Liou. A novel method of evaluating transient response. *Proceedings of the IEEE*, 54(1):20–23, 1966.

[20] J. Löfberg. YALMIP: a toolbox for modeling and optimization in MATLAB. In *Proc. Comput. Aided Contr. Syst. Des.*, pages 284–289, 2004.

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

[22] A. Majumdar and R. Tedrake. Robust online motion planning with regions of finite time invariance. In E. Frazzoli, T. Lozano-Perez, N. Roy, and D. Rus, editors, *Algorithmic Found. Robot. X*, pages 543–558. Springer Berlin Heidelberg, 2013.

[23] A. Majumdar, R. Vasudevan, M. M. Tobenkin, and R. Tedrake. Technical report: Convex optimization of nonlinear feedback controllers via occupation measures. *preprint arXiv:1305.7484*, 2013.

[24] I. M. Mitchell. Scalable calculation of reach sets and tubes for nonlinear systems with terminal integrators: a mixed implicit explicit formulation. In *Proc. Hybrid Syst.: Comput. Contr.*, pages 103–112, Chicago, IL, 2011. ACM.

[25] 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. Contr.*, 50(7):947–957, July 2005.

[26] I. M. Mitchell, S. Kaynama, M. Chen, and M. Oishi. Safety preserving control synthesis for sampled data systems. *Nonlinear Anal.: Hybrid Syst.*, 10:63–82, 2013.

[27] J. Pitman. *Probability*. Springer-Verlag, New York, 1993.

[28] S. Prajna and A. Jadbabaie. Safety verification of hybrid systems using barrier certificates. In *Proc. Hybrid Syst.: Comput. Contr.*, pages 477–492, 2004.

[29] P. Saint-Pierre. Approximation of the viability kernel. *Appl. Math. Optim.*, 29(2):187–209, Mar 1994.

[30] C. Schütt and E. Werner. Polytopes with Vertices Chosen Randomly from the Boundary of a Convex Body. In V. Milman and G. Schechtman, editors, *Geom. Aspects Funct. Anal.*, volume 1807 of *Lect. Notes Math.*, pages 241–422. Springer Berlin, 2003.

[31] C. Standish. Truncated Taylor series approximation to the state transition matrix of a continuous parameter finite Markov chain. *Linear Algebra Appl.*, 12(2):179–183, 1975.

[32] W. Tan and A. Packard. Searching for control lyapunov functions using sums of squares programming. In *Annual Allerton Conf.*, pages 210–219, 2004.

[33] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts. LQR-trees: Feedback motion planning via sums-of-squares verification. *Int. J. Robot. Res.*, 29(8):1038–1052, 2010.

[34] M. Tobenkin, I. Manchester, and R. Tedrake. Invariant funnels around trajectories using sum-of-squares programming. In *Proc. IFAC World Congr.*, volume 18, pages 9218–9223, Milano, Italy, 2011.

[35] U. Topcu, A. Packard, and P. Seiler. Local stability analysis using simulations and sum-of-squares programming. *Automatica*, 44(10):2669–2675, 2008.