Error Bounds Based Stochastic Approximations and Simulations of Hybrid Dynamical Systems

Alessandro Abate, Aaron D. Ames and Shankar Sastry
Department of Electrical Engineering and Computer Sciences
University of California at Berkeley, Berkeley, CA 94720
{aabate,adames,sastry@eecs.berkeley.edu}

Abstract— **This paper introduces, develops and discusses an integration-inspired methodology for the simulation and analysis of deterministic hybrid dynamical systems. When simulating hybrid systems, and thus unavoidably introducing some numerical error, a progressive tracking of this error can be exploited to discern the properties of the system, i.e., it can be used to introduce a stochastic approximation of the original hybrid system, the simulation of which would give a more complete representation of the possible trajectories of the system. Moreover, the error can be controlled to check and even guarantee (in certain special cases) the robustness of simulated hybrid trajectories.**

I. Background

Deterministic hybrid dynamical systems are intrinsically difficult to analyze. Unlike linear or nonlinear systems, both in continuous time (CT) or in discrete time (DT), numerous involved technicalities are needed to describe their dynamical properties; their most natural characteristics—stability or control design, for instance—are only marginally explained if compared to those of their constituent components. Computation, or verification, of their properties presents nontrivial challenges. For the sake of explanation, we could adduce this limitation to the presence of spacial guards that introduce asynchronous discrete events in the continuous flow of trajectories in each domain. We can surely claim that this represents an "analysis bottleneck"; in fact, destroys one of the fundamental properties of both CT and DT systems: robustness, i.e., the fact that *small changes in the initial condition result in small changes in the final position*.

Simulating trajectories of hybrid systems requires the use of a numerical integration techniques. This undeniably introduces approximation errors (Section III). While these errors represent a certainly undesired burden, it is possible to turn necessity into virtue by exploiting them—at the expense of tracking them. This allows for simulations which, accounting for the imprecisions they introduce, enable a more fair assessment of the actual position of the hybrid trajectory.

The first concept that stems from these considerations is that of stochastic approximations for the original deterministic hybrid systems (Section IV). We shall show how the information coming from the time-dependent integration error can be funneled towards approximating hybrid systems. The approximated system is reframed within a known stochastic setting. Limiting behaviors of this equivalent stochastic system will be established. Furthermore, some characteristic properties of this approximation will be highlighted. Simulations will hint at their usefulness.

The second concept, related to the idea of exploiting numerical error for positive benefit, is that of controlling the error bounds of the numerical integration in order to force the actual solution of the vector fields on each domain to detect the same discrete event that the numerical solution singles out. We shall show how to practically do this (Section V) in a single domain, propose an extension to the general case, while pointing out the limitations of this process. Although this appears to be a partially negative outcome, we shall explain that this is related to the presence of pathological behaviors in a hybrid system, the existence of which is indeed important to recognize, understand and give due consideration.

II. The Hybrid Systems Setting

Throughout the paper we will utilize the classical framework for deterministic hybrid systems, see [7].

Definition 1: Define a *hybrid system* as a tuple $\mathscr{H} = (Q, E, D, G, R, F)$ where

- $Q = \{1, ..., m\} \subset \mathbb{Z}$ is a finite set of *discrete states*,
- $E \subset Q \times Q$ is a set of *edges* which define relations between the domains,
- $D = \{D_i\}_{i \in Q}$ is a set of *domains* where D_i is a compact subset of $\mathbb{R}^n$,
- $G = \{G_e\}_{e \in E}$ is a set of *guards*, where $G_e \subseteq D_{\mathfrak{s}(e)}$; we assume that there exists a collection of smooth functions $\{g_e : \mathbb{R}^n \to \mathbb{R}\}_{e \in E}$ such that $G_e = \{x : g_e(x) = 0\}$ and that $g_e(x) \geq 0$ for all $x \in D_{\mathfrak{s}(e)}$,
- $R = \{R_e\}_{e \in E}$ is a set of *reset maps*, which are continuous maps from $G_e \subseteq D_{\mathfrak{s}(e)}$ to $R_e(G_e) \subseteq D_{\mathfrak{t}(e)}$,
- $F = \{f_i\}_{i \in Q}$ is a set of *vector fields* such that f_i is Lipschitz on $\mathbb{R}^n$; the solution to the ODE f_i with initial condition $x_0 \in D_i$ at time t_0 is a function from $\mathbb{R}$ to D_i, denoted by $x_i(t)$, which verifies $x_i(t_0) = x_0$.

Executions. An *execution* or *hybrid trajectory*[1] of the hybrid system $\mathscr{H}$ is a tuple $\chi = (\tau, \rho, \xi)$ where

This research is supported by the National Science Foundation (NSF award number CCR-0225610) and the grant DAAD19-03-1-0373.

[1] Here we are considering only *infinite* executions; introducing the definition of a finite execution would require unnecessary complication. For the more general definition see [7].

1-4244-0210-7/06/$20.00 ©2006 IEEE

- $\tau = \{\tau_i\}_{i\in\mathbb{N}}$ with $\tau_0 = 0 \le \tau_1 \le \cdots \le \tau_j \le \cdots$ is a *hybrid time sequence* or a sequence of *switching times*,
- $\rho : \mathbb{N} \to Q$ is a *discrete evolution map*,
- $\xi = \{\xi_i\}_{i\in\mathbb{N}}$ with $\xi_i \in \bigcup_{i\in Q} D_i$ is a *sequence of initial conditions*.

Additionally, we require that $\chi = (\tau, \rho, \xi)$ must satisfy the condition that for $i \in \mathbb{N}$, such that $(\rho(i), \rho(i+1)) \in E$ and

$$\begin{aligned}\xi_i &= x_{\rho(i)}(\tau_i),\\ \tau_{i+1} &= \min\{t \ge \tau_i : x_{\rho(i)}(t) \in G_{(\rho(i),\rho(i+1))}\},\\ \xi_{i+1} &= R_{(\rho(i),\rho(i+1))}(x_{\rho(i)}(\tau_{i+1})).\end{aligned}$$

We also require that $x_{\rho(i)}(t) \in D_{\rho(i)}$ for all $t \in [\tau_i, \tau_{i+1}]$; this is quite a natural assumption.

With this definition of execution, we can introduce the notion of Zeno trajectory (for more details on this issue, please refer to [3], [7], [2]) as follows:

Definition 2: A hybrid system $\mathscr{H}$ is *Zeno* if for some execution χ of $\mathscr{H}$ there exists a finite constant τ_∞ (called the *Zeno time*) such that

$$\lim_{i\to\infty} \tau_i = \sum_{i=0}^{\infty} (\tau_{i+1} - \tau_i) = \tau_\infty.$$

The execution χ is called a *Zeno execution.*

III. Error Bounds of Numerical Integration

Let $\dot{x} = f(x)$ be an ordinary differential equation. The initial value problem (IVP) is the problem of finding a solution, $x(t)$, to the ODE on some interval $[t_0, t_F]$ subject to an initial condition $x(t_0) = x_0$; we denote such an IVP by $\mathfrak{I} = (f, [t_0, t_F], x_0)$. Since it is in general not possible to explicitly solve an IVP of this form—in fact, almost never possible—in practice, the ODE is usually numerically integrated. A numerical integration technique is an integration method that associates to the IVP, $\mathfrak{I}$, an approximate solution $\hat{x}(t)$ on $[t_0, t_F]$ such that $\hat{x}(t_0) = x_0$. Approximate solutions will be of central focus to this paper. We will consider global error bounds on the distance between the numerical solution and the actual solution. Since for most integration techniques precise global bounds are not available, we will review local error bounds and show how to obtain approximate[2] global bounds from local bounds (such as those given in Matlab).

We will suppose, as in [10], that the numerical integration method produces a solution that is accurate of order $M(t, h)$, where $M(t, h)$ is a function, continuous in both its arguments, such that $M(0, h) = 0$, $M(t, h) \to 0$ monotonically as $t \to 0$ and $M(t, h) \to 0$ monotonically as $h \to 0$ (here h is related to the integration step size). In other words, for the IVP $\mathfrak{I}$ there exists a constant $C_\mathfrak{I}$ such that

$$\|x(t) - \hat{x}(t)\| \le C_\mathfrak{I} M(t - t_0, h).$$

The bound $C_\mathfrak{I} M(t - t_0, h)$ is a *global* or *true* bound on the error. As already mentioned, most integration techniques do not control the global error, but rather the *local error*. Note that in the following, for simplicity we will let $B(t, h) = C_\mathfrak{I} M(t - t_0, h)$ and assume that the IVP is clear from context.

Most numerical integrations techniques produce a discrete set of points which approximate the actual solution; this is the situation we will work with in this paper. Specifically, for the IVP $\mathfrak{I} = (f, [t_0, t_F], x_0)$ we assume that the numerical integration technique produces a set of times t_n and points x_n, for $n = 0, \dots, k$ such that $t_0 < t_1 < \cdots t_k = t_F$ and for this integration method

$$\|x(t_n) - x_n\| \le B(t_n, h) = C_\mathfrak{I} M(t_n - t_0, h),$$

where if $h_n = t_{n+1} - t_n$ is the n^{th} step size, $h = \max\{h_n\}$. The approximate solution $\hat{x}(t)$ is obtained by interpolating linearly between the points x_n and x_{n+1} when $t \in [t_n, t_{n+1}]$.

Local error bounds for integration techniques are common, especially in Matlab. First recall the definition of a local error bound. Let $x^{[n]}(t)$ be the solution to the IVP $\mathfrak{I}^{[n]} = (f, [t_n, t_{n+1}], x_n)$, i.e., $x^{[n]}(t)$ is the solution to $\dot{x} = f(x)$ on the interval $[t_n, t_{n+1}]$ subject to the initial condition $x^{[n]}(t_n) = x_n$. The n^{th} local error is given by

$$\text{error}_n = x^{[n]}(t_{n+1}) - x_{n+1}.$$

We can use this local error to obtain a specific instance $\widetilde{B}$ of the global error bound B by setting

$$\widetilde{B}(t_n, h) = \|\text{error}_n\| + \sum_{i=0}^{n-1} \|\text{error}_i\| e^{L(t_n - t_i)},$$

where L is the Lipschitz constant of f. It can be verified that

$$\|x(t_n) - x_n\| \le \widetilde{B}(t_n, h).$$

Similarly to what we did for the approximate trajectory $\hat{x}(t)$, let $\widetilde{B}(t, h)$ be the function obtained by linear interpolation between $\widetilde{B}(t_n, h)$ and $\widetilde{B}(t_{n+1}, h)$ when $t \in [t_n, t_{n+1}]$. Note that since $\|\text{error}_n\| \to 0$ as $h \to 0$, $\widetilde{B}(t, h) \to 0$ as $h \to 0$, which is one of the characteristics of the accuracy function described above.

The relevance of the local error is that this is the quantity that Matlab controls (and allows the user to control) when numerically solving ODEs (cf. [9]). Specifically, in Matlab the local error is estimated in each step and made to satisfy

$$\|\text{error}_n\| \le \mathrm{R}_{\text{tol}} \|x_n\| + \mathrm{A}_{\text{tol}},$$

where R_{tol} and A_{tol} are the relative error tolerance and absolute error tolerance, respectively; the default value for these quantities are 10^{-3} and 10^{-6} and can be otherwise specified by the user. Using this bound on the local error, we obtain a bound on the global error:

$$\begin{aligned}&\|x(t_n) - x_n\| \le \widetilde{B}(t_n, h)\\ &\qquad \le \sum_{i=0}^{n-1} (\mathrm{R}_{\text{tol}} \|x_i\| + \mathrm{A}_{\text{tol}}) e^{L(t_n - t_i)} + \mathrm{R}_{\text{tol}} \|y_n\| + \mathrm{A}_{\text{tol}}.\end{aligned}$$

Since R_{tol} and A_{tol} are related (and in fact determine) the step size, reducing these quantities to zero is equivalent to reducing the maximum step size to zero; in both cases the

[2] We say "approximate" here because, although in theory we can exactly obtain a formula relating global error bounds to local error bounds, in practice we must approximate parts of this formula.

global bound on the error obtained from this local bound goes to zero. For this reason, this is the global error bound that we will use in the Simulations. To this end we need the Lipschitz constant of the ODE $\dot{x} = f(x)$. We do not assume that this is known; instead, we approximate L with L_{approx} by setting

$$L_{\text{approx}} = \max_{0 \leq n \leq k} \left(\sigma_{\max}(Df(x_n)) \right),$$

where $\sigma_{\max}$ is the maximum singular value of the Jacobian matrix $Df(x_n)$.

IV. Stochastic Approximations

A. Stochastic Hybrid Systems

There are essentially four ways to introduce probability in the setting of deterministic hybrid systems: in the *edges*, in the *guards*, in the *reset maps* and in the *vector fields*. In this paper, we shall focus on uncertainty introduced in the second (and, implicitly, in the first) of these entities, and introduce the following definition:

Definition 3: Define a *stochastic hybrid system* as a tuple $\mathscr{S} = (Q, E, D, G, R, F)$ where the elements in the tuple are those of $\mathscr{H}$, except for

- $D = \{D_i\}_{i \in Q}$ is a set of *domains* where $D_i = \mathbb{R}^n$ (we need to relax the restriction on the compactness, which comes from the presence of the guards). We define $S = \bigcup_{i \in Q} \{i\} \times \mathbb{R}^n$ to be the *hybrid state space*, and we assume a Borel space $(S, \mathcal{B}(S))$ is properly defined, as in [5].
- $G = \{G_e\}_{e \in E}$, formerly a set of sets (the spatial guards), and now a set of functions, $G_e(x) = \lambda_{(i,j)}(x)$; here $\lambda_{(i,j)} : D_i \to \mathbb{R}^+$ is the *transition (or jumping) intensity*.

Assumption 1: We assume the *measurability* of $\lambda_{(i,j)}$ on D_i, and that the composed function $\lambda_{(i,j)} \circ x_i : t \to \lambda_{(i,j)}(x_i(t))$, where $x_i(t_0) = x_0 \in D_i$, is *integrable* on every bounded set $[t_0, t_0 + \epsilon)$, $\epsilon > 0$.

Remark 1: The definition is a special case of the *PDMP* by Davis, [5], where no forced jumps are allowed. Furthermore, we disallow probabilistic resets or the presence of diffusion terms in the vector fields. The fundamental assumption in [5] on the finiteness of the discrete events on bounded time intervals is implied here by construction, i.e., by the absence of the spacial guards and the continuity conditions on vector fields and reset maps.

The jumping intensity $\lambda_{(i,j)}(x_i(t))$ on D_i induces a jumping probability $\mathbb{P}^i_j(t)$, for all $j \in Q$ such that $(i,j) \in E$ and $t \geq t_0$; for j such that $(i,j) \notin E$, $\mathbb{P}^i_j(t) = 0$. This allows us to define a discrete kernel over the set Q, $\mathbf{P}^i(t) = \left(\ \mathbb{P}^i_j(t) \ \right)_{j \in Q}$ for each domain D_i (when $t \geq t_0$).

A *stochastic execution* is defined, similarly to the deterministic case, as follows:

Definition 4: An execution of $\mathscr{S}$ is defined through a tuple $\chi^{\mathscr{S}} = (\tau, \rho, \xi)$, where

- $\tau = \{\tau_i\}_{i \in \mathbb{N}}$ with $\tau_0 = 0 < \tau_1 < \cdots < \tau_j < \cdots$ is a sequence of *stopping times*,
- $\rho : \mathbb{N} \to Q$ is a *discrete evolution map*,
- $\xi = \{\xi_i\}_{i \in \mathbb{N}}$ with $\xi_i \in \bigcup_{i \in Q} D_i$ is a *sequence of initial conditions*.

Additionally, we require that $\chi^{\mathscr{S}} = (\rho, \tau, \xi)$ must satisfy the condition that for $i \in \mathbb{N}$,

- $x_{\rho(i)}(\tau_i) = \xi_i$,
- τ_{i+1} is a stopping time associated with the following survivor function:
$$S(t) = \exp\left(- \int_{\tau_i}^{t - \tau_i} \sum_{j \in Q, \, (\rho(i), j) \in E} \lambda_{(\rho(i), j)}(s) ds \right),$$
- $\rho(i+1)$ is chosen via the transition kernel $\mathbf{P}^{\rho(i)}(t)$, and more precisely via $\mathbb{P}^{\rho(i)}_{(\ \cdot\ )}(\tau_{i+1})$,
- $\xi_{i+1} = R_{(\rho(i), \rho(i+1))}(x_{\rho(i)}(\tau_{i+1}))$.

Because we are assuming that $\tau_i < \tau_{i+1}$, we can define a function $q : \bigcup_{i \in \mathbb{N}} [\tau_i, \tau_{i+1}) \to Q$, where $q(t) = \rho(i)$ for $t \in [\tau_i, \tau_{i+1})$. Moreover, it is possible to introduce a $x : \bigcup_{i \in \mathbb{N}} [\tau_i, \tau_{i+1}) \to \mathbb{R}^n$, where $x(t) = x_{\rho(i)}(t)$, if $t \in [\tau_i, \tau_{i+1})$. Finally, we state that an *execution* of $\mathscr{S}$, given a tuple $\chi^{\mathscr{S}} = (\tau, \rho, \xi)$ verifying the properties above, is a stochastic process $(q(t), x(t)) \in S$ based on the tuple $\chi^{\mathscr{S}}$.

The following holds (the proof is a modification of that in [5], as discussed in Remark 1):

Proposition 1: *The stochastic hybrid system $\mathscr{S}$ introduced in Def. 3 and with executions constructed as in Def. 4 is endowed with the Markov property and admits an explicit process generator.*

B. Introducing Stochastic Approximations

In this section, we shall focus on the original hybrid system $\mathscr{H}$ and describe how to approximate it with a stochastic counterpart $\mathscr{S}_h$, where h relates to the integration step size.

1) Transforming Guards: Suppose that for the i^{th} domain, D_i, there are K_i guard functions, i.e. $K_i = |\{e \in E : e = (i, \cdot)\}|$. In this case, the set guard functions on D_i can be ordered as $\{g_{(i,j)}\}_{j=1}^{K_i}$. We can assume without loss of generality that these functions are of the form $g_{(i,j)}(x) = x_j$ where x_j is the j^{th} element of x. Let f_i be the vector field of the hybrid system on domain $D_i \subseteq \mathbb{R}^n$; this claim is justified by the observation (see Shampine in [8]) that it is possible to transform a system into a form so that the above assumption is satisfied. This is achieved by defining new variables $z_j = g_{(i,j)}(x), \forall j = 1, \ldots, K_i$. A new domain $\tilde{D}_i$, a new vector field $\tilde{f}_i$ and a new set of guard functions on this domain $\{\tilde{g}_{(i,j)}\}_{j=1}^{K_i}$ is then defined as follows: let $\tilde{D}_i = \mathbb{R}^{K_i} \times D_i$ and for an element (z, x) in this new domain then $\tilde{f}_i$ is defined by

$$(\tilde{f}_i(z,x))_j = \begin{cases} \frac{\partial g_{(i,j)}(x)}{\partial x} \cdot f_i(x), & \text{if} \quad 1 \leq j \leq K_i \\ (f_i(x))_{j - K_i}, & \text{if} \quad K_i + 1 \leq j \leq K_i + n \end{cases}$$

Finally, the guard functions on this domain are given by $\tilde{g}_{(i,j)}(z,x) = z_j$. It is clear that the behavior of the solution, especially with respect to its discrete events, is the same for the transformed system and the original system.

2) Forming Transition Functions: With a set of guard functions satisfying the above assumption, i.e., $g_{(i,j)}(x) = x_j$, we can define the *transition functions*. First, we need some definitions. Let $\hat{x}_i(t)$ be the approximate solution to $x_i(t)$ on the domain D_i. We will use this approximate solution to define a set of transition functions $\pi_{ij}(t)$. Let $\mathbb{S}^n(r, x)$ denote an n-dimensional sphere of radius r centered at x and suppose that this sphere intersects the hyperplane $H = \{x_1 = 0\}$. Then if

$$\sigma = \min\left(\text{dist}(\mathbb{S}^n(r, x) \cap H, \partial\mathbb{S}^n(r, x))\right),$$

the volume of the sphere lying "below" the plane H is given by

$$\text{V}(r, \sigma) = \int_0^\sigma V_{n-1}(\sqrt{r^2 - (z^2 - r^2)})dz,$$

where $V_{n-1}(r)$ is the volume of an $(n-1)$-dimensional sphere of radius r. Now let

$$\begin{aligned} G^+_{(i,j)} &= \{x : g_{(i,j)}(x) > 0\}, \\ G^-_{(i,j)} &= \{x : g_{(i,j)}(x) < 0\}, \\ G^0_{(i,j)} &= \{x : g_{(i,j)}(x) = 0\}. \end{aligned}$$

Using this, define the transition function π_{ij} for the guard $g_{(i,j)}$ and error bound $B(t, h)$ on the numerical integration technique as

$$\pi_{ij}(t, h) = \begin{cases} 1 \quad \text{if} \quad \mathbb{S}^n(B(t, h), \hat{x}_i(t)) \subset G^+_{(i,j)} \\ 0 \quad \text{if} \quad \mathbb{S}^n(B(t, h), \hat{x}_i(t)) \subset G^-_{(i,j)} \\ \frac{\text{V}(B(t,h), \sigma_{(i,j)}(t,h))}{V_n(B(t,h))} \quad \text{if} \quad \begin{matrix} \mathbb{S}^n(B(t, h), \hat{x}_i(t)) \\ \cap G^0_{(i,j)} \neq \emptyset \end{matrix} \end{cases}$$

where here

$$\begin{aligned} \sigma_{(i,j)}(t, h) &= \min\begin{pmatrix} \text{dist}\{\mathbb{S}^n(B(t, h), \hat{x}(t)) \cap G^0_{(i,j)}, \\ \partial\mathbb{S}^n(B(t, h), \hat{x}(t))\} \end{pmatrix} \\ &= |\hat{x}_i(t) - B(t, h)|. \end{aligned}$$

This completes the construction of the transition functions. In the case when we fix h, we will denote the transition functions by $\pi_{ij}(t) := \pi_{ij}(t, h)$.

3) Defining Transition Probabilities: With reference to the setting of hybrid systems, given a numerical integration method (which we assume to have fixed integration step), a starting time τ_i and with an initial condition $x_0 \in D_{\rho(i)}$ on a domain $\rho(i) \in Q$, we can explicitly derive through time an error bound around the numerical solution. At any point in time $\tau_i \geq t \geq \tau_{i+1}$ the real solution of the ODE lies somewhere inside a sphere centered around $\hat{x}_{\rho(i)}(t)$. If the numerical solution approaches one of the boundaries of our domain, then this sphere may intersect it. We give a special interpretation to this occurrence: *at every point in time, the probability that the actual solution switches from the current domain to that identified by a guard is given by the proportion of the volume sphere centered around the numerical solution at that time that is beyond the guard itself.* More precisely, we claim the following: given the initial conditions τ_i, x_0 and $\rho(i)$, and obtaining a numerical solution by application of an integration method, we have the following knowledge on the actual discrete portion of the hybrid trajectory, for $t \geq \tau_i$:

$$\mathbb{P}^{\rho(i)}_j(t) = \Pr(\tau_{i+1} = t, q(\tau_{i+1}) = j | q(\tau_i) = \rho(i)) = \pi_{ij}(t).$$

Remark 2: The reader should ponder over the probabilistic interpretation of the actual trajectory through the error bounds: clearly, it implicitly assigns a uniform distribution over the entire volume of the sphere. This is certainly an arbitrary choice, but arguably the most general and the most intuitively natural one given the shape of the error bounds and our knowledge of the actual position of the trajectory.

4) From Probabilities to Intensities: Considering the time-varying transition kernel $\mathbb{P}^{\rho(i)}_j(t), t \geq \tau_i$. With some calculation it can be shown that it is right continuous along time; it thus appears natural to take its right derivative and reason in terms of *transition intensities*, as introduced in Def. 3. These new entities are introduced by:

$$\lambda_{(\rho(i),j)}(t) = \lim_{s \downarrow t} \frac{\mathbb{P}^{\rho(i)}_j(s) - \mathbb{P}^{\rho(i)}_j(t)}{s - t}, \ s \geq t \geq \tau_i.$$

It should be clear that the framework is easily prone to work in the case of numerous guards per each domain, addressing the fundamental intricacy underlying the relative spacial position of the guards in the deterministic setting.

C. Properties

We are now in a position to discuss some of the properties of the stochastic approximation of the deterministic hybrid system.

1) Limiting Equivalence: The first important property, is that in the limit $\mathscr{H}$ and $\mathscr{S}_h$ agree as follows.

Theorem 1: *Given a hybrid system $\mathscr{H}$, the non-trivial stochastic hybrid system $\mathscr{S}_h$, dependent on a parameter h (the integration step), verifies the following:*

$$\lim_{h \to 0} \mathscr{S}_h = \mathscr{H}.$$

In other words, the stochastic hybrid trajectory will be, at the limit, equivalent to the deterministic one.

Remark 3: The reader should notice that Assumption 1 is always verified if $h > 0$. It is indeed implied that the value of the intensities is bounded, $h > 0 \Rightarrow \lambda_{(i,j)}(x) < \infty, \forall x \in D_i, j \in Q, (i, j) \in E$.

2) Excluding Zeno: Another important property of $\mathscr{S}_h$ is that it is not Zeno, even if the original hybrid system was.

Theorem 2: *Given a hybrid system $\mathscr{H}$, the non trivial stochastic hybrid system $\mathscr{S}_h$ admits no Zeno behavior, for all $h > 0$.*

D. Simulations

Despite the sophistication of the available numerical integration tools, it should be clear that the presence of pathological behaviors in a hybrid system due to the interconnections of its domains may lead to faulty simulations; that unless we can solve analytically for the vector fields in each domain. Introducing numerical error bounds, of which we have exact knowledge, and interpreting them through distributions describing probabilistically the position of the exact trajectory, allows us to execute multiple simulations and extrapolate the behavior of the actual solution from the outputs. This section presents simulations that display how the stochastic hybrid systems obtained from the original deterministic one handle some pathological conditions.

1) Grazing: Let us define "grazing" as the situation where a hybrid trajectory osculates a guard. The essential question related to this kind of dynamics is what the behavior of the real solution is; the answer can be given probabilistically. Fig. 1 shows a trajectory from a clockwise circular vector field in two dimensions, which grazes the $x_1 = 1$ surface. Propagating the hypercone, we check the transition probability through time: it tends to the value $1/2$ the closer the trajectory gets to the surface, while being smaller at points in space further away from it.

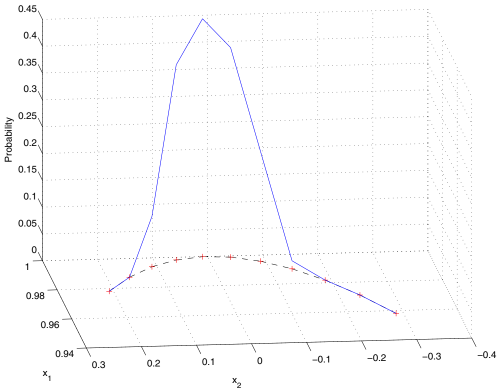

Fig. 1. Variation in time of the Transition Probabilities, for a trajectory grazing the surface $x_1 = 1$.

2) Handling Zeno: As also discussed in [1], the trajectories of the stochastic hybrid system $\mathscr{S}_h$ will be defined for all the time, despite the presence of Zeno behavior in the original hybrid system. The well known *two-tanks example* (refer to [11] for a model description) is one classical instance of this phenomenon. The plots in Fig. 2 show how, by decreasing the step size in the simulation, the switching events are increasingly better approximated. In Fig. 3 we plot the cone along which the probability sphere is obtained.

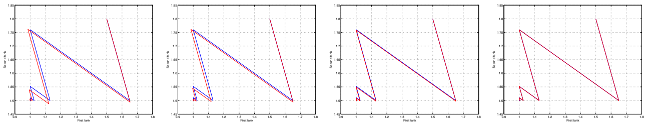

Fig. 2. Crossing detection and Zeno handling via stochastic approximations (red trajectories); the smaller the step size in the simulations, the closer the approximation to the numerical solution (blue trajectory).

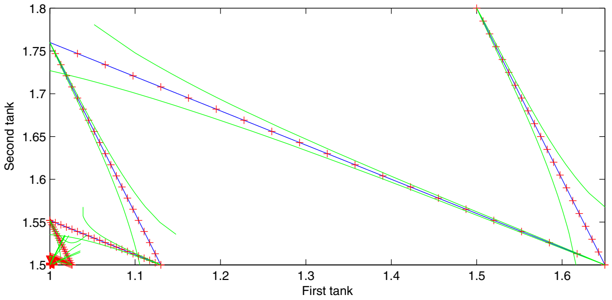

Fig. 3. The *two tanks* system: propagation of the error cone defined through global bounds.

V. Guaranteeing Simulations of Hybrid Systems

A. Exact Event Detection through Global Numerical Integration Bounds

Consider a hybrid system $\mathscr{H}$, as introduced in Sec. III, with the additional assumption that each domain is delimited by its guards; this fact can be encompassed by the following condition:

$$\forall\, i \in Q,\;\; D_i = \{x : g_{(i,j)}(x) \geq 0, j \in Q, (i,j) \in E\}.$$

For the IVP $\mathfrak{I} = (f, [t_0, t_F], x_0)$ on domain D_i, an event occurs whenever the trajectory, starting at time t_0 and position x_0, intersects any of the $G_{(i,j)}$. Ideally, we can say that this happens at time t^e if $t^e = \min_{j\in Q}\{t > t_0 : \exists j \in Q : g_{(i,j)}(x_i(t)) = 0\}^3$ and at the point $x^e = x_i(t^e) \in G_{(i,j)}$. Unfortunately, oftentimes we are only able to calculate an event-time $\hat{t}^e$ based on the outcome of the IVP above, $\hat{t}^e = \min_{j\in Q}\{t > t_0 : g_{(i,j)}(\hat{x}_i(t)) = 0\}$; we also introduce the point $\hat{x}^e = \hat{x}(\hat{t}^e_i)$.

Based on a solution $\hat{x}_i(t)$ of the IVP $\mathfrak{I}$, parameterized by the integration step h, and its corresponding global error $B(t,h)$, we can propagate forward an *error cone* $\mathfrak{C}_h(t), t \geq t_0$:

$$\mathfrak{C}_h(t) = \bigcup_{t_0 \leq s \leq t} \{x \in D_i : \|x - \hat{x}_i(s)\| \leq B(s,h)\}$$

This subset of D_i has the property of containing the actual solution of the vector field, $x_i(t) \in \mathfrak{C}_h(t), t \geq t_0$. Similar to the idea developed in Section IV, it is interesting to look at the possible intersection of $\mathfrak{C}_h$ with the set of guards $\{G_{(i,j)} : j \in Q, (i,j) \in E\}$. In particular, utilizing the quantity $\mathbb{S}^n(B(t,h), \hat{x}_i(t))$, we shall focus on the set

$$\mathfrak{J}_h = \{j \in Q : \mathfrak{C}_h(t) \cap G_{(i,j)} \neq \emptyset, t_0 \leq t \leq \hat{t}^g, (i,j) \in E\},$$

where

$$\hat{t}^g = \min\{t > t_0 : \mathbb{S}^n(B(t,h), \hat{x}_i(t)) \cap D_i = \emptyset\} > \hat{t}^e.$$

The introduction of the quantities above allows us to claim the following:

Proposition 2: *For the IVP $\mathfrak{I}$ on domain D_i with an integration step $h > 0$, the actual trajectory $x(t)$ will detect the same event as the approximate trajectory $\hat{x}(t)$ (the approximate solution of $\mathfrak{I}$) if there exists an h such that $\hat{t}^e < \infty$, $\hat{t}^g < \infty$ and $\mathfrak{J}_h$ is a singleton.*

Remark 4: The condition on $\hat{t}^e$ refers to the actual existence of an event, that on $\hat{t}^g$ forces the whole error cone to

[3] For the sake of precision, we assume the quantity t_F in $\mathfrak{I}$ is large enough to make sense of the event, i.e. $t_F > t^e$.

exit the domain of interest, while the last on $\mathcal{J}_h$ tallies the number of intersections with possibly different guards. The second condition can be overridden by considering a small enough h.

Clearly, much cannot be said about the exact value of t^e or x^e, except that the difference $|t^e - \hat{t}^e|$ will be finite and that a bound can be established on the distance $\|x^e - \hat{x}^e\|$. Propagating forward the idea to the subsequent domain, we observe that the actual solution will be contained in the union of the cones starting from the intersection of $\mathcal{C}_h$ and the guard $G_{(i,\mathcal{J}_h)}$.

The concept of *metrics on trajectories*, [6], has been introduced in the literature of discrete events systems to describe the (finite) difference between the evolution of two hybrid trajectories. Our concept goes is much in the same vain, deriving conditions for computing this difference.

B. Controlling the Error

As discussed, the quantities $B(\cdot, h)$, $\mathcal{C}_h(\cdot)$ and $\mathcal{J}_h$ depend on the choice of the integration step h. It can be checked that $\mathcal{C}_h(\cdot)$, similarly to the global error $B(\cdot, h)$, verifies the same property. It is intuitive then that by controlling the integration error h it is possible to force the numerical approximation to display the same discrete behavior of the actual hybrid trajectory. In other words, we can "guarantee" that the evolution of the real hybrid trajectory will be described "closely enough" by that of the numerical hybrid solution. The following algorithm implements the idea:

Algorithm 1: Given a HS $\mathscr{H}$, with domains delimited by guards, and a hybrid initial condition (q_0, x_0) at time $t_0 = 0$, Init $:= x_0$; $i := 0$;

```
Repeat
  - For all $\xi \in$ Init,
      – Set up the IVP $\mathcal{I}_\xi = (f_i, [t_0, \tau], \xi)$;
      – Solve it and, under the assumptions of Prop. 2, compute $\mathcal{C}_h^\xi(\hat{t}^g)$ and $\mathcal{J}_h^\xi$.
  - End (For).
  - If $\bigcup_{\xi \in \text{Init}} J_h^\xi$ is a singleton (call it $\nu$),
      – Init $:= \partial \left( \bigcup_{\xi \in Init} \{ \mathcal{C}_h^\xi(\hat{t}^g) \cap G_{(i,\nu)} \} \right)$;
      – $i := \nu$.
  - Else
      – Decrease $h := h'$, where $h' = h/\alpha, \alpha > 1$;
      – Restart.
  - End (If).
Until True.
```

Despite the theoretical correctness of the algorithm (it can be possibly modified to terminate in order to accomodate for finite horizon simulations), it is not a computationally tractable one; for instance, propagating forward sets of initial conditions is a tricky issue, with the notable exception of two dimensional systems.

This procedure, which introduces a notion of robustness for the HS setting over a finite time horizon, is *sound*, but not *complete*. In particular, it cannot be implemented if the HS undergoes some pathological condition. For instance, a deadlock would arise in the presence of a Zeno trajectory, which is infinite in the number of discrete events and requires increasingly infinite precision in simulation (and characterization) along the finite time interval. Furthermore, if it ever happens that the actual trajectory hits an intersection between two guards, decreasing the integration step will never prevent the error cone from crossing both guards. Similarly, if the trajectory osculates a guard, further refinement would not prevent the cone from intersecting the guard only on one side.

VI. Future Directions

The authors are currently investigating ways of improving the idea of stochastic approximations, and generalizing the concept of guaranteed simulations. On both levels, it would be interesting to obtain tighter precise bounds on the currently rather conservative global error; employing a tighter but approximate bound would be feasible in the case of stochastic approximations. Focusing on the simulation aspects of the stochastic hybrid system obtained via approximation, the authors are looking into algorithms for special classes of SHS (cf. for instance [4]). Another research effort is being directed towards finding conditions that ensure some precise notion of "robustness" for hybrid trajectories; this requires the generalization of the concept of IVP's to a hybrid setting. This notion can be interpreted as finding a *"condition number"* for a specific hybrid system, i.e., a function expressing enough to quantify the presence of pathological interconnections in the hybrid system (refer to [1] for more details).

References

[1] A. Abate, A. D. Ames, and S. Sastry, "A stochastic approximation for hybrid systems," in *Prooceedings of the 24th American Control Conference, Portland, OR*, 2005.

[2] A. D. Ames, A. Abate, and S. Sastry, "Sufficient conditions for the existence of zeno behavior," in *Prooceedings of the 44th IEEE Conference on Decision and Control and European Control Conference ECC, Seville*, 2005.

[3] A. D. Ames and S. Sastry, "Blowing up affine hybrid systems." in *Prooceedings of the 43rd IEEE Conference on Decision and Control, The Bahamas*, 2004.

[4] H. Blom and E. Bloem, "Particle filtering for stochastic hybrid systems," in *Prooceedings of the 43rd IEEE Conference on Decision and Control, The Bahamas*, 2004.

[5] M. Davis, *Markov Models and Optimization.* Chapman and Hall/CRC, 1993.

[6] T. Henzinger and J. Raskin, "Robust undecidability of timed and hybrid systems," in *3rd Int. Workshop HSCC, Pittsburgh, PA*, ser. LNCS, B. Krogh and N. Lynch, Eds., vol. 1790. Springer Verlag, 2000, pp. 145–159.

[7] J. Lygeros, K. Johansson, S. Simic, J. Zhang, and S. Sastry, "Dynamical properties of hybrid automata," *IEEE Transactions on Automatic Control*, vol. 48, no. 2-18, 2003.

[8] L. F. Shampine and I. Gladwell, "Reliable solution of special event location problems for odes," *ACM Trans. Math. Software*, vol. 17, no. 1, pp. 11–25, 1991.

[9] L. F. Shampine and M. W. Reichelt, "The matlab ode suite," *SIAM J. Sci. Comput.*, vol. 18, 1997.

[10] L. F. Shampine and S. Thompson, "Event location for ordinary differential equations," *Comput. Math. Appl.*, vol. 39, no. 5-6, pp. 43–54, 2000.

[11] J. Zhang, K. H. Johansson, J. Lygeros, , and S. Sastry, "Zeno hybrid systems," *International Journal of Robust and NonLinear Control, vol. 11, pp. 435-451*, 2001.