Preprint at Journal: Nonlinear Analysis: Hybrid Systems,
Volume 3, Issue 2, May 2009, Pages 150-162

Understanding Deadlock and Livelock Behaviors in Hybrid Control Systems

Alessandro Abate[1,4], Alessandro D'Innocenzo[2,3],
Maria Domenica Di Benedetto[3], and Shankar Sastry[4]

[1] Department of Aeronautics and Astronautics,
Stanford University - USA
aabate@stanford.edu
[2] Department of Electrical and Systems Engineering,
University of Pennsylvania - USA
adinnoce@seas.upenn.edu
[3] Department of Electrical Engineering and Computer Science,
Center of Excellence DEWS, University of L'Aquila - Italy
dibenede@ing.univaq.it
[4] Department of Electrical Engineering and Computer Sciences,
University of California, at Berkeley - USA
sastry@eecs.berkeley.edu

Abstract. This paper introduces a formal definition and a categorization of Deadlock and Livelock behaviors for a general class of deterministic Hybrid Control Systems (HCS), thus extending the classical notion known for (uncontrolled) discrete transition systems. This characterization hinges on three important aspects: (1) the concept of composition (or interconnection) of HCSs; (2) the notion of control-dependent specification, and that of composition of specifications; (3) the dynamical structure of a HCS and its related behaviors. The first notion is introduced in a novel manner, by including aspects from the literature of discrete transition systems, as well as accounting for classical concepts such as that of feedback interconnection of dynamical systems. The second point allows to formally express general properties that are of interest from a systems and control theory perspective. The third part discriminates between the different and possibly pathological behaviors that are characteristic to HCSs. After commenting on the issues of Deadlock and Livelock prevention and verification, the article concludes with two case studies.

Keywords: Hybrid Systems, Control Systems, Systems Composition, Deadlock, Verification of Specifications.

1 Introduction

The concept of deadlock and its close relative, that of livelock, have been widely investigated in various branches of computer science [22], in particular with regards to discrete transition systems. Deadlock has often been regarded as a pathology and has been related to the absence of a particular liveness specification, that of forward progress [2,

4]. Much interesting work has been done on the verification of the presence of deadlock situations in concurrent software modules, or in guaranteeing its absence from the composition of these modules [6, 29].

Hybrid Systems are rather general mathematical models that connect between discrete, logical, synchronous systems and continuous, real-time, asynchronous ones [26, 27]. In particular, they can model real-time and embedded systems, as well as distributed, multi-agent and communication systems. They display interleaved and interacting discrete and continuous dynamics. They present behaviors or are endowed with properties that are "at the limit" between classical discrete transition systems and continuous dynamical models [27]. Understanding these distinctive behaviors sheds light on the structure of this general modeling framework, its capabilities, and its limitations.

Motivated by a number of case studies, this work aims at "exporting" the notions of deadlock and livelock to Hybrid Control Systems (HCS). The definition of these known concepts within the HCS framework–and in particular to the case of continuous dynamics–appears to be novel. More precisely, the objective is that of introducing a mathematically rigorous definition of such phenomena and that of providing a clear characterization of them. The ensuing task of presenting verification procedures aimed at checking the existence of these phenomena, as well as at drawing sufficient conditions for their absence, is also briefly looked at, and is to be further developed as a future research direction. Unlike a recent approach in the literature, which introduces the concept of deadlock according solely to the HS dynamics [14], this work argues that a formal definition of this notion can be correctly given considering three important aspects: that of *composition* of HCS, which we reformulate inspired by different approaches in the literature; that of *specifications*—and their composition—which we reinterpret from a perspective that is meaningful for control systems; and that of the *dynamics* that are characteristic to HCSs and, as such, more complex than those related to discrete transition systems. According to the proposed procedure, it is then possible to reinterpret dynamical behaviors that are typical of HCS within this new characterization. Attempting a definition of deadlock or livelock based exclusively on a specification of "forward progress" (see Section 4) and on the model dynamics (as with simpler discrete event systems) fails to capture a number of important aspects: the presence of controls in the hybrid model, the subtleties of its continuous dynamical behaviors, the issue of composition between systems. We stress that, as expected, the introduced con-

cepts nicely tailor back to corresponding ones in the literature of, respectively, discrete and continuous systems.

The presentation of the material matches the three important aspects discussed above, and the article is thus structured as follows. Section 2 formally introduces the concept of HCS with its semantics, and discusses structural properties of the model. The definition in particular deals with the presence of a control in the model. Section 3 defines the composition between HCSs. Section 4 introduces the notion of specification, and discusses a few possible forms that are of relevance for control systems. Unlike the classical notion of specification dealt with in the literature on automatic verification, where we are interested in an operative notion to be used with model checking procedures, in this work we highlight the presence of a control action and set up the concept of specification with the objective of defining the concepts under study in the article. Furthermore, the composition of specifications is illustrated and discussed. Given this setup, by looking at the set of possible dynamical behaviors associated to HCSs—and to their composition—Section 5 defines the notions of deadlock and livelock. Section 6 succinctly establishes a framework for the study of the issues of deadlock prevention and verification. Finally, Section 7 develops two case studies to describe how to apply and understand the notions of deadlock and livelock within the developed theoretical framework.

2 Deterministic Hybrid Control Systems: the Model

Let us start by introducing the concept of HCS. This definition is rather detailed, in order to formally pin down concepts that will be further developed in the definition of hybrid systems composition.

Definition 1 (Deterministic Hybrid Control System). *A* Deterministic Hybrid Control System *is a tuple* $\mathcal{H} = (X, U, Y, F, T)$*, where:*

- *X is the hybrid state space, composed of*
  - *$Q \subseteq \mathbb{N}$, a finite set of discrete states, or modes*
  - *$D = \{D_i\}_{i \in Q}$, the set of domains associated with each mode. D_i is a subset of $\mathbb{R}^{n_i}$, where $n_i < \infty$ may vary for different domains*
  - Init $\subseteq D$ *is the set of initial conditions*

  The space is specified as the disjoint union of the domains, i.e. $X = \cup_{i \in Q} \{q_i\} \times D_i$

– *U is the control space, $U \subseteq \mathbb{R}^{n_U}$ and bounded. The control is a function defined on the set of nonnegative reals and with values in U, $u : \mathbb{R}_0^+ \to U$, where $\mathbb{R}_0^+ = \mathbb{R}^+ \cup \{0\}$. We assume it is cadlag, i.e. piecewise-continuous from the right and with left limit, lying in $\mathbb{R}^{n_U}$*
– *Y is the observation space, $Y \subseteq \mathbb{R}^{n_Y}$. The outputs will be determined by a static "observation" function $h : X \to Y$*
– *$F = \{f_i\}_{i \in Q} : D \times U \to D$ is the set of vector fields. Each f_i is assumed to be continuous w.r.t. time and Lipschitz continuous with respect to the dependent variables. The vector field f_i characterizes the ODE $\dot{x} = f_i(x, u)$,[1] where $u(t) \in U$, for any $t \in \mathbb{R}_0^+$ and with an initial condition $x_0 \in D_i$*
– *T is the set of transition relations, composed of*
  - *$E \subseteq Q \times Q$, a set of edges. Each edge $e \in E$ is an ordered pair of modes, the first component of which is the source and is denoted by $s(e)$, while the second is the target and denoted by $t(e)$*
  - *$G = \{G_e\}_{e \in E}$, a set of guards, where $G_e : U \to 2^{D_{s(e)}}$. The guards are defined over the domains. We assume that $\forall e', e'' \in E, e' \neq e'', s(e') = s(e''), \forall u \in U, G_{e'}(u) \cap G_{e''}(u) = \emptyset$, i.e. the guards do not intersect in a domain. The guards are considered to be "forcing" the jumps, as will be stressed in the definition of execution*
  - *$R = \{R_e\}_{e \in E}$, a set of reset functions, where $\forall e \in E, R_e : D_{s(e)} \to D_{t(e)}$*

Remark 1 (Determinism of the Model). The assumptions of Lipschitz continuity for the vector fields,[2] of non-intersection between guards, on their "forcing" feature, and on the structure of the reset functions, are sufficient for ensuring the *determinism* of the model for all initial conditions in *Init*, both in the continuous dynamics and in the discrete jumps. We allowed for resets onto a guard set, which translates into admitting multiple discrete state transitions in zero-time. This does not prevent determinism, as it is also stressed in [27]. □

Remark 2 (Non-blocking Conditions). For $i \in Q$, let $G_i(u) = \bigcup_{e : s(e) = i} G_e(u)$, and define $\partial D_i = cl(D_i) \backslash int(D_i)$, where $cl(D_i)$ is the closure of D_i, while $int(D_i)$ is the interior of

[1] For the sake of precision, we remark that the state and the input are different signals; one is defined over the hybrid time set (see Definition 2), the second over the real time, which is a subset of the first. This can be extended to time-dependent vector fields.

[2] In particular, we stress this assumption with regards to the control input, which in general is allowed to be Lebesgue measurable [31], and in particular piecewise-continuous as in the present case.

D_i. If the following holds:

$$(\forall i \in Q, \forall u \in U), \quad \partial D_i \cap D_i \subseteq G_i(u) \ \wedge \ \partial D_i \backslash D_i \subseteq cl(G_i(u)), \tag{1}$$

then no trajectory ever exits from a domain without hitting a guard (recall that the guards are understood to be forcing the jump). The condition then states that the boundary has to belong to the guard set, if it is part of the domain, or to the closure of the guard set. The condition can be weakened, as long as we impose that the guard set is by definition a subset of the domain. Notice that the resets are always acting within a domain. □

Notice in passing that we have introduced a dependence of the guards on the continuous control, but the resets are left uncontrolled—the extension is possible and left to the interested reader.

To define the notion of execution associated to $\mathcal{H}$, we first introduce the following notion [26]:

Definition 2 (Hybrid Time Set). *A hybrid time set $\tau = \{I_k\}_{k \geq 0}$ is a finite or infinite sequence of intervals $I_k = [t_k, t'_k] \subseteq \mathbb{R}^+_0$ such that*

1. *I_k is closed if τ is infinite. I_k might be right-open if it is the last interval of a finite sequence τ*
2. *$t_k \leq t'_k$ for $k > 0$ and $t'_{k-1} = t_k$ for $k > 1$*

The length $(t'_k - t_k)$ of every interval I_k denotes the dwelling time of the hybrid flow within a discrete location, while the extrema t_k, t'_k specify the switching instants. Let us stress that the above set is ordered, hence it makes sense to use notations such as $t_k \leq t'_k$. A *hybrid trajectory*, or *hybrid flow*, is a pair (x, τ), where the first component is a function $x = (q, v) : \tau \to X$, that describes the evolution of the continuous part v and the discrete part q, which is defined on a hybrid time set τ and take values on X. Finally, a *hybrid execution* is a pair (x, τ) which can be algorithmically generated as follows:

Algorithm 1 (Hybrid Execution):

1. *Pick $(q(t_0), v(t_0)) \in$ Init, set $k = 0$, $\tau = \varnothing$*
2. *Evolve the continuous trajectory $v(t)$ according to the vector field, which depends on the exogenous control $u|_{[t_k,t)}$ and with initial condition $v(t_k)$ until a guard is hit: namely until time $t'_k \in [t_k, \infty)$ such that $v(t'_k) \in G_e\left(\lim_{t \to t'^{-}_k} u(t)\right)$, where $s(e) = q(t_k)$*
3. *If $t'_k = \infty$, add $I_k = [t_k, \infty)$ to τ and exit the algorithm*

4. *Else add* $I_k = [t_k, t'_k]$ *to* τ. *Define* $q(t_{k+1}) = t(e)$ *and* $v(t_{k+1}) = R_e(v(t'_k))$. *Increment* k *and go to line 2*

Remark 3. In what follows, we will call *event* a discrete state transition. Notice that events are only due to the intersection of an execution with a spatial guard. For the sake of generality (and in order to further tailor the model into that of discrete transition systems), in Definition 1 we have assumed a dependency of the guard on the control signal. The control is a piecewise right-continuous function: in order to rule out problems related to its discontinuity points, we have introduced in the semantical definition of an event a dependence on the limit from the left. This assumption will play a role in ruling out "cycling conditions" on the composition of two HCSs (see Remark 6). □

Within the set of hybrid executions, we shall also focus on the following subset:

Definition 3 (Zeno Executions). *Zeno executions are hybrid trajectories which are characterized by an infinite number of jumps in a finite amount of elapsed time. The hybrid time set of a Zeno trajectory has infinite cardinality and satisfies the following property:*

$$\sum_{k=0}^{\infty}(t'_k - t_k) < \infty$$

Zeno can be of two kinds [8]: chattering and genuine. The first case happens when $\exists k^* : \forall k \geq k^*, t_k = t'_k = t_{k^*}$. *The second happens if* $\forall k \geq 0, \exists k^* \geq k : (t'_{k^*} - t_{k^*}) > 0$.

The output of the hybrid system is, for each execution, a function from the hybrid time set to the output space. Since our purpose is to set up a notion of input-output interconnection, in the spirit of [28, 29], we suppose that the *interconnected* output of a HCS is a physical signal expressed by a function of the real time, rather than of the hybrid time basis. This allows to give a notion of interconnection that is asynchronous and disregards the dynamics at the level of the internal states of the HCS. Recall that a hybrid time set is defined as $\tau = \{I_k\}_{k\geq 0}$, with $I_k = [t_k, t'_k] \subset \mathbb{R}^+_0, t'_k \geq t_k, \forall k \geq 0$. We introduce the following:

Definition 4 (Interconnectable Outputs). *Given a HCS and any output execution* $y(t)$, $t \in \tau$, *the* interconnectable output *associated with* $y(t)$ *is a signal* $\tilde{y}(t), t \in \mathbb{R}^+_0$, *which is a function of the real time basis defined by a projection of* y *on* $\mathbb{R}^+_0$, *such that* $\tilde{y}(t)$ *is cadlag.*

Considered in its generality, the model in Definition 1 is a melange between the classic hybrid automaton [26] and the Lipschitz HIOA in [28, 29]. In particular, we remark that the state space is intended to be an *internal* state, while the output and the control ones are intended to be *interface/external* states. The model is similar in structure to the hybrid automaton at an internal level, while it relates to the HIOA at a higher, external level. In comparison to the HIOA model, we highlight here the presence of spatial guards, reset maps, as well as the influence of the control structure at the internal level. Motivated by a control perspective on the model, we avoid to introduce the notion of *action* (either internal, or external) as in [28, 29]. Here an action is simply the dynamic outcome of a state jump, and is possibly influenced by the exogenous control. We also do not distinguish a-priori between internal and external signals (which, in the HIOA setup, are thought to be "variables"). We have implicitly assumed that there is a relationship between the state and the output spaces by the introduction of the observation function h—being this a function, determinism is retained. If the system is *fully-observable*, we take the continuous output space Y to coincide with the largest of the domains D_i, once we have embedded all of them in $\mathbb{R}^{\max_i\{n_i\}}$—$h$ performs this embedding.

3 Composition of Hybrid Control Systems

The concept of systems *composition* may be introduced in a number of ways, depending on the characteristics and properties of the systems (discrete or continuous, causal or non-causal, to name a few), the structure of the operation (in parallel [11] or as a product [32], for instance), and the particular properties that we may want to check for (preservation of synchronicity or sequentiality, for example). In this work we consider a composition operation that can be interpreted as a form of *parallel composition*. A similar concept has been introduced, among the many places, in [6, 7, 11, 30]. Notice that the introduction of a model structure with different "layers," similar to that in [28, 29], allows to conceive the system at the level of its hidden/internal signals (the hybrid state space with its vector fields and transition relations, as in Section 2) as a black box and to simply focus on the external components (the interface components in Section 2) when performing the interconnection. This is the advantage of the interpretation as an input/output system. Unlike previous work, which simply performed parallel compositions or crude variable "sharing," inspired here by a more control-theoretical perspective we allow the connections between inputs and outputs of the systems to depend on gen-

eral functions endowed with some properties—in other words, we naturally introduce an *output feedback* framework. We are in particular interested in a definition which may further tailor into known operations in the purely discrete case (transition systems) or dynamical instance (feedback interconnection). We suppose that:

Assumption 1. *The input spaces $U_i, i = 1, 2$ are rectangular sets, i.e. $U_i = [\underline{u}_1, \bar{u}_1] \times [\underline{u}_2, \bar{u}_2] \times ... \times [\underline{u}_m, \bar{u}_m]$, where $m = dim(span(U_i))$, $\underline{u}_j, \bar{u}_j \in \mathbb{R}$ and $\underline{u}_j \leq \bar{u}_j$.*

The necessity to avoid mutual dependence between input components comes from the need to perform projections on their spaces, as it will become shortly clear. Before introducing the notion of composition of two hybrid systems we raise the following assumption:

Assumption 2 (Compatibility Conditions). *When two HCS $\mathcal{H}_1$ and $\mathcal{H}_2$ are to be composed, we assume that the two systems, considered in separation, have no shared inputs, nor shared output signals (in other words, input signals do not "split" and outputs do not "merge"). If this happens, we say that $\mathcal{H}_1$ and $\mathcal{H}_2$ are compatible.*

The first assumption is raised for consistency with the definition below. The second assumption avoids imposing conditions on the dynamics of the two systems once they get interconnected. In the literature this structural requirement is a subset of the known *compatibility* conditions, [7, 28]. We nevertheless recall that, as discussed at the end of Section 2, in general we allow for the (partial) coincidence of internal and external components of two systems, unlike [28, 29].

Given a set W, subset of a subspace $\mathcal{W} \subseteq \mathbb{R}^n$ with $dim(\mathcal{W}) = m \leq n$, let $M_{\mathcal{W}}$ be an invertible matrix such that for any $w \in \mathcal{W}$, $M_{\mathcal{W}} w = [w^T, 0^T]^T$. Such a matrix always exists. Assume that W and $\mathcal{W}$ are rectangular (see Assumption 1), that is of the form $\mathcal{W} = a_1 span\{w_1\} \times a_2 span\{w_2\} \times ... \times a_n span\{w_n\}$, where $a_i \in \{0, 1\}$ and w_i are the canonical base vectors of $\mathbb{R}^n$. Introduce the operator $\pi|_A(B)$, which yields the projection of the set B over the space A. A (parallel) *composition procedure* between two systems is introduced as follows:

Definition 5 (HCS Composition). *Given two compatible HCS $\mathcal{H}_1 = (X_1, U_1, Y_1, F_1, T_1)$ and $\mathcal{H}_2 = (X_2, U_2, Y_2, F_2, T_2)$, a composition procedure $\|_\Sigma$ is specified by $\Sigma = \{\mathcal{W}_1 \times \mathcal{W}_2, g_1 \times g_2\}$, i.e. a set of rectangular subspaces $\mathcal{W}_i \subseteq span(U_i), i = 1, 2$, and the maps $g_1 : Y_2 \to \pi|_{\mathcal{W}_1}(U_1), g_2 : Y_1 \to \pi|_{\mathcal{W}_2}(U_2)$. The operation $\|_\Sigma$ yields $\mathcal{H}^\Sigma = \mathcal{H}_1 \|_\Sigma \mathcal{H}_2 = (X^\Sigma, U^\Sigma, Y^\Sigma, F^\Sigma, T^\Sigma)$, which is a new HCS made up of the following:*

- $X^\Sigma = X_1 \times X_2$, $\text{Init}^\Sigma = \text{Init}_1 \times \text{Init}_2$ *are the hybrid state space and initial conditions*
- $U^\Sigma = \pi|_{(\mathcal{W}_1^\perp \times \mathcal{W}_2^\perp)}(U_1 \times U_2)$ *is the input space (here* $\mathcal{W}_\cdot^\perp$ *is the orthogonal of* $\mathcal{W}_\cdot$*.)*
- $Y^\Sigma = Y_1 \times Y_2$ *is the output space*
- $F^\Sigma = \{f^\Sigma_{(q_1,q_2)}\}_{(q_1,q_2)\in Q_1\times Q_2}$ *is the vector field. For any* $(x_1, x_2) \in X^\Sigma$ *and* $(u_1, u_2) \in U^\Sigma$, $f^\Sigma_{(q_1,q_2)}((x_1, x_2), (u_1, u_2)) = f_{q_1}(x_1, M^{-1}_{\mathcal{W}_1}[u_1^T, g_1(h_2(x_2))^T]^T) \times f_{q_2}(x_2, M^{-1}_{\mathcal{W}_2}[u_2^T, g_2(h_1(x_1))^T]^T)$
- T^Σ *is the set of transition relations, composed of*
  - $E^\Sigma = E_1 \times E_2$*, is the set of edges*
  - $G^\Sigma = \{G^\Sigma_e\}_{e\in E^\Sigma}$*, is the set of guards, where for any* $(u_1, u_2) \in U^\Sigma$ *and any* $(e_1, e_2) \in E^\Sigma$ *the guard set* $G^\Sigma_{(e_1,e_2)}$ *is implicitly defined by* $(x_1, x_2) \in G^\Sigma_{(e_1,e_2)}$*, where* $(x_1, x_2) \in (G_1)_{e_1}(M^{-1}_{\mathcal{W}_1}[u_1^T, g_1(h_1(x_2))^T]^T) \times (G_2)_{e_2}(M^{-1}_{\mathcal{W}_2}[u_2^T, g_2(h_2(x_1))^T]^T)$
  - $R^\Sigma = \{R_e\}_{e\in E}$*, is the reset function, where* $\forall e = (e_1, e_2) \in E^\Sigma, R^\Sigma_e = (R_1)_{e_1} \times (R_2)_{e_2}$

Proposition 1 (Closure). *Given a pair of compatible HCS* $\mathcal{H}_1$ *and* $\mathcal{H}_2$ *and a composition* $\Sigma = \{\mathcal{W}_1 \times \mathcal{W}_2, g_1 \times g_2\}$*, the system* $\mathcal{H}^\Sigma = \mathcal{H}_1 \|_\Sigma \mathcal{H}_2$ *is an HCS.*

Proof. It follows by construction. The conditions raised on the structure of the HS $\mathcal{H}_1$ and $\mathcal{H}_2$, as well as on Σ, do not require additional assumptions to derive the conclusion on $\mathcal{H}^\Sigma$. □

The following remarks contain a number of specific comments on the introduced notion.

Remark 4 (Input-Output Signals Interconnection). The interconnecting functions g_1, g_2 that in part characterize the composition Σ turn a transformation of part of the original output spaces into part of the original input spaces. Notice that we have not assumed that the input and output spaces have the same dimension: the dimensionality is handled directly by the interconnecting functions g_1, g_2. In particular, such interconnections map the output signal to a subset of the input space. More precisely, the functions g_1, g_2 are defined on a subset of the output spaces, i.e. $dom(g_1) \subseteq Y_2$ and $dom(g_2) \subseteq Y_1$: their dimensions indicate the number of signals employed in the interconnection. The dimension of the codomain $dim(\mathcal{W}_i)$ defines the number of signals actually connected. The way they get connected to the input signals is further specified by the subspaces $\mathcal{W}_i$ in an intuitive way. More general compositions, that allow the output signals to be "shuffled" (rather than just sequentially clustered), can be obtained by appropriately redefining the matrices $M_{\mathcal{W}_i}$. The new input set for the composition is the projection of the cartesian product of the two original input sets onto the space of the "unused signals," i.e. of those inputs that have accepted no feedback. Clearly, in the full-feedback

case, the codomains of the g_i shall be the whole input spaces U_i, thus providing a fully dynamic (and, thus, uncontrolled) composition. Finally, let us stress that it is possible to write $g = g_1 \times g_2$ as a cartesian product between functions. □

Remark 5 (Events on the composed system). The events of the composed system are specified "asynchronously." In other words, while the dynamics of the two models mutually affect each other, the events happen within each single system, possibly at the same time, with no need to be reciprocally synchronized. □

Remark 6 (Asynchronicity, Absence of Cyclic Constraints). Both $\mathcal{H}_1$ and $\mathcal{H}_2$ are asynchronous by definition and their composition $\mathcal{H}^{\Sigma}$ is asynchronous as well. The composition can indeed be viewed from the outside as an interconnection of two dynamical systems. This does not exclude the presence of pathological behaviors (Zeno or blocking, for instance), which arises at an internal level. For now, we do not impose any a-priori condition to exclude this, unlike previous literature, where the focus was on ensuring infinite forward time progress under systems composition [6, 7, 28, 29]. As discussed, the discrete events affecting the signals of each system are allowed to happen without any imposed synchronization, even if we have allowed the internal (state) and external (outputs) signals to possibly partially overlap. This is due to the semantics of the events, which depend on the value of the control at the limit (see Algorithm 1). In the literature [28, 29], this "cyclic constraints" have been artificially avoided by splitting up internal and external variables. Another method to prevent these conditions has been that of imposing some ordering, or sequentiality, between the signals in the loop. However, this would prevent the introduction in generality of the concept of dynamic feedback. □

Remark 7 (Commutativity and Associativity of the Composition). The composition is trivially *commutative* by the way it is defined. The *associative property* is verified in the following sense: given three HCS $\mathcal{H}_1$, $\mathcal{H}_2$, $\mathcal{H}_3$, if it is legitimate to compose them as $(\mathcal{H}_1 \| _{\Sigma_1} \mathcal{H}_2) \| _{\Sigma_2} \mathcal{H}_3$, then it is also possible to perform the composition $\mathcal{H}_1 \| _{\Sigma_1} (\mathcal{H}_2 \| _{\Sigma_2} \mathcal{H}_3)$ and it holds true that $(\mathcal{H}_1 \| _{\Sigma_1} \mathcal{H}_2) \| _{\Sigma_2} \mathcal{H}_3 = \mathcal{H}_1 \| _{\Sigma_1} (\mathcal{H}_2 \| _{\Sigma_2} \mathcal{H}_3)$. Notice that we have not specified the details of the above composition procedures, but just assumed that there exist interconnecting maps that respect the domain dimensionality, as well as bounds ensuring that the composition of the three systems can be performed in a unique way. Showing that it is possible, and indeed equivalent, to perform the composition in the other order is only a matter of calculations. We remark that this property ensures the

generality of Definition 5 in that it can be repeated more than once without worrying about the order. This will also be exploited in the ensuing sections when reasoning about composing specifications. Another issue that precedes the above property is the seek of conditions that ensure that the composition between a number of HCS is structurally allowed. An intuitive necessary condition for this fact to hold for a composition between a set $\mathcal{H}_i, i \in \mathcal{I}$ through $\Sigma = \{\Sigma_{j,k}, j,k \in \mathcal{I}\}$, is that the codomains of all the interconnecting maps to any particular input space do not intersect : $\forall i \in \mathcal{I}, \forall j,k : \{\Sigma_{j,i}, \Sigma_{k,i}\} \in \Sigma, \mathcal{W}_j \cap \mathcal{W}_k = \varnothing$. □

The following can be derived from Remark 1 and the way the composition in Definition 5 is performed.

Proposition 2 (Determinism of the Composition). *Given a pair of deterministic HCS $\mathcal{H}_1$ and $\mathcal{H}_2$ and a composition $\Sigma = \{\mathcal{W}_1 \times \mathcal{W}_2, g_1 \times g_2\}$, if the maps g_i are almost-everywhere continuous, then the hybrid system $\mathcal{H}^\Sigma = \mathcal{H}_1 \|_\Sigma \mathcal{H}_2$ is also deterministic.*

Proof. The continuity hypotheses imply that the piecewise-continuous inputs will properly affect the other vector fields—this, coupled with the boundedness of the input signals, ensures that the solutions of the vector fields will again exist and be unique [31]. The absence of cyclic constraints also prevents non-deterministic trajectories from existing. The conclusion follows from the comments in Remark 1. □

More general compositions can be described with small changes to the setup given in this section. For instance, it is possible to define feedbacks endowed with "self-loops" within a single HCS. Futhermore, it is also interesting to look at time-varying interconnections, which may introduce a "switching" composition. This may introduce interesting, but possibly pathologic phenomena, such as Zeno [18].

4 Specifications and their Composition

In this section we consider the notion of "specification," which is defined for trajectories on the observation space. In general, specifications may be defined via temporal logic formulae for real-time systems. We are not considering general classes of temporal logic formulae such as those defined in Computation Tree Logic (CTL) [13] and Timed CTL [5]. In this work, we are not interested in the problem of automatic verification of these specifications by model checking algorithms: rather than focusing on an operative/logical definition, we introduce the notion with the goal of defining the

concepts of deadlock and livelock or HCS, which requires the investigation of the problem of composition of specifications. We will focus on four simple temporal properties of interest and, due to the dependence of the models on a control, we will introduce existential or universal quantifiers on the control signals. This choice allows to express specifications that are general enough to cover a range of important problems in control theory. Let $\mathcal{M} \subseteq Y$ and recall that U is the set of control inputs for a HCS $\mathcal{H}$. We further denote $\mathcal{U}$ as the set of cadlag functions $u : \mathbb{R}_0^+ \to U$, defined on the non-negative reals and with values in U. We set up the following four specifications as template problems for our study:

1. *reachability*: $\varphi_R(\mathcal{U}, \mathcal{M}) \coloneqq \exists u^* \in \mathcal{U}, \exists t^* < \infty : y(t^*) \in \mathcal{M}$
2. *attractivity*: $\varphi_A(\mathcal{U}, \mathcal{M}) \coloneqq \forall u \in \mathcal{U}, \exists t^* < \infty : y(t^*) \in \mathcal{M}$
3. *invariance*: $\varphi_I(\mathcal{U}, \mathcal{M}) \coloneqq \forall u \in \mathcal{U}, \forall t \geq 0 : y(t) \in \mathcal{M}$
4. *viability*: $\varphi_V(\mathcal{U}, \mathcal{M}) \coloneqq \exists u^* \in \mathcal{U}, \forall t \geq 0 : y(t) \in \mathcal{M}$

The known *liveness* property, which is the object of much investigation in the literature on transition systems, can be reinterpreted within the first two of the above properties. In particular, the requirement of *forward progress* has been studied [6, 7, 22, 2] for its practical implications and connections with phenomena such as Zeno or blocking behavior. The last two of the above four properties instead can be thought of as *safety* specifications. With regards to the quantification on the control, the first and fourth instances can be intended as *control synthesis* problems (i.e., we search for a "witness" related to a particular specification), while the second and third as *verification* problems. If the system is purely dynamic, then the quantification over the controls is to be disregarded. In this case as expected 1=2 and 3=4, and we are exclusively interested in the validity of the proposition.

Notice that the above definitions have been given in generality with respect to the signal $y(t) \in Y$, which is the output of the control-dependent dynamics of $\mathcal{H}$, and which is defined on the hybrid time set τ, according to Definition 2. Because of the hypothesis on the determinism of the model (see Remark 1), given an initial condition and a time-dependent control, the generated hybrid trajectory is unique.
We write $x_0 \models_{\mathcal{H}} \varphi(\mathcal{U}, \mathcal{M})$ (and say that x_0 satisfies $\varphi(\mathcal{U}, \mathcal{M})$ for $\mathcal{H}$) if the specification under study (which again depends on the sets $\mathcal{U}$ and $\mathcal{M}$) is satisfied by executions with initial condition $x_0 \in X$. We are then interested in the set of initial conditions that satisfy a given specification: more formally, we introduce the following subset of the

set of initial conditions:

$$\mathcal{X}_{\mathcal{H}} = \{x_0 \in \text{Init} : x_0 \models_{\mathcal{H}} \varphi(\mathcal{U}, \mathcal{M})\}.$$

We write $\mathcal{H} \models \varphi(\mathcal{U}, \mathcal{M})$ (and say that the system $\mathcal{H}$ satisfies $\varphi(\mathcal{U}, \mathcal{M})$) if $\mathcal{X}_{\mathcal{H}} = \text{Init}$. Let us now consider two hybrid systems $\mathcal{H}_1$ and $\mathcal{H}_2$, two corresponding specifications $\varphi_1(\mathcal{U}_1, \mathcal{M}_1)$, $\varphi_2(\mathcal{U}_2, \mathcal{M}_2)$ and a composition procedure Σ. We are interested in composing the two specifications φ_1, φ_2.[3] Similar to [1], we define:

Definition 6 (Composition of Specifications). *Given two compatible HCS $\mathcal{H}_1, \mathcal{H}_2$ with specifications $\varphi_1(\mathcal{U}_1, \mathcal{M}_1)$, $\varphi_2(\mathcal{U}_2, \mathcal{M}_2)$ and a composition procedure Σ, the composed specification $\varphi^{\Sigma}(\mathcal{U}^{\Sigma}, \mathcal{M}_1 \times \mathcal{M}_2) = \varphi_1(\mathcal{U}_1, \mathcal{M}_1) \|_{\Sigma} \varphi_2(\mathcal{U}_2, \mathcal{M}_2)$ for the HCS $\mathcal{H}^{\Sigma}$ is the conjunction $\varphi_1(\mathcal{U}_1, \mathcal{M}_1) \wedge \varphi_2(\mathcal{U}_2, \mathcal{M}_2)$, modulo proper substitutions of its entities according to the composition maps g_1, g_2 that characterize Σ.*[4]

4.1 Preservation of Specifications under Composition of HCSs

The composed system $\mathcal{H}^{\Sigma} = \mathcal{H}_1 \|_{\Sigma} \mathcal{H}_2$ can be investigated from the perspective of the preservation of specifications defined on the original systems $\mathcal{H}_1, \mathcal{H}_2$. Let us start by looking at the existence of initial conditions for the composed system $\mathcal{H}^{\Sigma}$ that are associated to trajectories that verify the composed specification φ^{Σ}:

$$\mathcal{X}_{\mathcal{H}^{\Sigma}} = \{(x_1^0, x_2^0) \in \text{Init}^{\Sigma} = \text{Init}_1 \times \text{Init}_2 : (x_1^0, x_2^0) \models_{\mathcal{H}^{\Sigma}} \varphi^{\Sigma}\}. \tag{2}$$

Furthermore, rather than verifying the specification φ^{Σ} directly on the composed HCS $\mathcal{H}^{\Sigma}$, as in equation (2), we may be interested in checking whether some verified properties on the original systems (say φ_1 and φ_2 on $\mathcal{H}_1$ and $\mathcal{H}_2$) are retained through a certain composition procedure (say Σ). This approach is motivated by the definition of the concepts of deadlock and livelock, to be introduced shortly, which is associated with pathological behaviors at the level of the composition. More precisely, let us introduce the following set:

$$\bar{\mathcal{X}}_{\mathcal{H}^{\Sigma}} := \{(x_1^0, x_2^0) \in \mathcal{X}_{\mathcal{H}_1} \times \mathcal{X}_{\mathcal{H}_2} : (x_1^0, x_2^0) \models_{\mathcal{H}^{\Sigma}} \varphi^{\Sigma}\}.$$

[3] In the following, we may omit the specification of the spaces $\mathcal{U}, \mathcal{M}$ for a specification φ when redundant.

[4] In other words, it may be necessary to substitute some control variables appearing in the formulas for $\varphi_1(\mathcal{U}_1, \mathcal{M}_1), \varphi_2(\mathcal{U}_2, \mathcal{M}_2)$ according to the composition functions g_1, g_2. Notice that in [1], a space embedding is also necessary in the case of composition of specifications defined on heterogeneous spaces. Definitions 1 and 5 render this unnecessary.

The following containment relationships hold valid: $\bar{\mathcal{X}}_{\mathcal{H}^\Sigma} \subseteq \mathcal{X}_{\mathcal{H}^\Sigma} \subseteq \text{Init}_1 \times \text{Init}_2$. Let us remark that $\mathcal{X}_{\mathcal{H}^\Sigma}$ is the subset of initial conditions of $\mathcal{H}^\Sigma$ that satisfy the composed specification φ^Σ. $\bar{\mathcal{X}}_{\mathcal{H}^\Sigma}$, instead, is the subset of the sets of initial conditions $\mathcal{X}_{\mathcal{H}_1} \times \mathcal{X}_{\mathcal{H}_2}$ which, taken separately, satisfy the single specifications φ_1, φ_2 for $\mathcal{H}_1, \mathcal{H}_2$, and which also satisfy the composed specification φ^Σ for $\mathcal{H}^\Sigma$. We are finally able to define the set:

$$\tilde{\mathcal{X}}_{\mathcal{H}^\Sigma} := (\mathcal{X}_{\mathcal{H}_1} \times \mathcal{X}_{\mathcal{H}_2}) \backslash \bar{\mathcal{X}}_{\mathcal{H}^\Sigma}. \tag{3}$$

This is the set of initial conditions which, taken separately (i.e. within each of the original systems), would satisfy the corresponding properties for that single system, but which (considered on the composed system) do not satisfy the composed specification φ^Σ for $\mathcal{H}^\Sigma$. Notice that, if the specifications φ_1, φ_2 are expressed with the universal quantifier over the control functions (that is, they are of the form $\forall u_i \in \mathcal{U}_i, i = 1, 2$), then $\bar{\mathcal{X}}_{\mathcal{H}^\Sigma} = \mathcal{X}_{\mathcal{H}_1} \times \mathcal{X}_{\mathcal{H}_2}$, and thus $\tilde{\mathcal{X}}_{\mathcal{H}^\Sigma} = \emptyset$. This happens because the interconnection simply restricts the set of available input signals for the composed system. Thus, if a universal property holds for any of the control input signals, it also holds for a subset of them. This suggests to restrict the attention in the following to the case where either φ_1 or φ_2 handle the controls u_i with an existential quantifier (i.e. $\exists u_i \in \mathcal{U}_i, i = 1, 2$). In other words, as anticipated earlier in this Section, we shall focus on *control synthesis* problems, rather than on verification problems (which are usually tackled with model checking procedures). In the next Section, we claim that the elements of $\tilde{\mathcal{X}}_{\mathcal{H}^\Sigma}$ are initial states of $\mathcal{H}^\Sigma$ that are associated with "pathological" behaviors, when looked at from the perspective of the verification of the composed specification φ^Σ, resulting from the interconnection Σ.

It is then within the set $\tilde{\mathcal{X}}_{\mathcal{H}^\Sigma}$ that we will be looking for deadlock and livelock executions. The actual definition of these executions will hinge on the dynamical properties of the generated executions, as described in the next Section 5.

5 Formal Definition of Deadlock and Livelock for HCS

Let us start by introducing the following known invariance concept, tailored to the HCS case:

Definition 7 (Hybrid Invariant Set). *Consider a HCS $\mathcal{H}$. Select a particular control over time. A hybrid set $\mathcal{I} \subseteq X$ is defined to be* invariant *if the following holds for the hybrid trajectory (x, τ), solution of $\mathcal{H}$ and originating from the chosen control: if $\exists\, t^* \in \tau : x(t^*) \in \mathcal{I}$, then $x(t) \in \mathcal{I}, \forall t \geq t^*$.*

From a dynamical standpoint, the concepts of deadlock, or livelock, are intuitively related to the idea of a trajectory being "constrained" or "stalled" in a region of the state space. This bounding condition is then further described with regards to the presence or absence of continuing motion within the region. We then distinguish the pathological trajectories associated with $\tilde{\mathcal{X}}_{\mathcal{H}^\Sigma}$ as follows:

Definition 8 (Deadlock and Livelock for HCS). *Consider a HCS $\mathcal{H}^\Sigma = \mathcal{H}_1 \|_\Sigma \mathcal{H}_2$, and the associated set $\tilde{\mathcal{X}}_{\mathcal{H}^\Sigma}$ as in* (3)*. The items in $\tilde{\mathcal{X}}_{\mathcal{H}^\Sigma}$ can be categorized into the following, possibly overlapping sets:*

$$\tilde{\mathcal{X}}_{\mathcal{H}^\Sigma} = \tilde{\mathcal{X}}^d_{\mathcal{H}^\Sigma} \cup \tilde{\mathcal{X}}^l_{\mathcal{H}^\Sigma} \cup \tilde{\mathcal{X}}^e_{\mathcal{H}^\Sigma}. \tag{4}$$

More precisely, they belong to either of the following:

- $\tilde{\mathcal{X}}^d_{\mathcal{H}^\Sigma} \cup \tilde{\mathcal{X}}^l_{\mathcal{H}^\Sigma} = \{x \in \tilde{\mathcal{X}}_{\mathcal{H}^\Sigma} : x(0) = x, \exists\, \mathcal{I} \subset X^\Sigma, t^* \geq 0 : x(t^*) \in \mathcal{I}\}$*, the set of initial conditions associated with trajectories that enter an invariant set $\mathcal{I}$*
- $\tilde{\mathcal{X}}^e_{\mathcal{H}^\Sigma} = \tilde{\mathcal{X}}_{\mathcal{H}^\Sigma} \backslash (\tilde{\mathcal{X}}^d_{\mathcal{H}^\Sigma} \cup \tilde{\mathcal{X}}^l_{\mathcal{H}^\Sigma})$*, the complement of the above set in $\tilde{\mathcal{X}}_{\mathcal{H}^\Sigma}$, the set of initial conditions that correspond to trajectories that do not encounter an invariant set in X^Σ*

The above two-point categorization is exhaustive over the set of executions. Furthermore, the first set $\tilde{\mathcal{X}}^d_{\mathcal{H}^\Sigma} \cup \tilde{\mathcal{X}}^l_{\mathcal{H}^\Sigma}$ is precisely composed of:

- $\tilde{\mathcal{X}}^d_{\mathcal{H}^\Sigma} = \{x \in \tilde{\mathcal{X}}_{\mathcal{H}^\Sigma} : x(0) = x, \exists\, \mathcal{I} \subset X^\Sigma, \exists\, t^* \geq 0 : x(t^*) \in \mathcal{I} \wedge \forall t \geq t^*, \{x(t) = x(t^*)\} \vee \{x(t)$ *is undefined*$\}\}$*, the set of initial conditions associated with* deadlock *executions: these are defined by absence of motion in finite time ("stalling" situation)*
- $\tilde{\mathcal{X}}^l_{\mathcal{H}^\Sigma} = \{x \in \tilde{\mathcal{X}}_{\mathcal{H}^\Sigma} : x(0) = x, \exists\, \mathcal{I} \subset X^\Sigma, t_1^* \geq 0 : \forall t \geq t_1^*, x(t) \in \mathcal{I}, \nexists t_2^* \geq 0 : \forall t \geq t_2^*, \dot{x}(t) = 0\}$*, the set of initial conditions associated with a* livelock *situation: these are characterized by endless motion, either in their continuous or discrete component.*

The above three sets are non overlapping if the composition is fully dynamic, namely if $U^\Sigma = \emptyset$. Notice that in the definition of $\tilde{\mathcal{X}}^d_{\mathcal{H}^\Sigma}$ the set $\mathcal{I}$ could simply coincide with the point $x(t^*)$, while for the livelock case this set has to be non-trivial. The definition above hinges on a purely *dynamical* level. This represents the last point, after that of *composition* and that of *specification*, which is regarded in this work as a necessary point to introduce the notions of deadlock and livelock in the framework of HCS.

Let us now study how known types of dynamics in a HCS are categorized within the above definition (this list does not claim to cover all the set of possible hybrid trajectories):

- Deadlock Situations:
  - *blocking conditions*: states for which no "next" state is defined.
  - *stable equilibria in finite time*: equilibria in the continuous dynamics that are reached by a reset operation
  - *chattering Zeno*: the discrete component infinitely jumps instantaneously between different domains, while the continuous component remains unchanged (Definition 3)
  - *genuine Zeno*: the hybrid trajectory performs an infinite number of transitions in a finite amount of time, with non-trivial continuous motion (Definition 3)
- Livelock Situations:
  - *stable equilibria in infinite time*: equilibria for the continuous dynamics
  - *limit cycles*, both in the continuous and the discrete dynamics

It is easy to add to the above that the set $\tilde{\mathcal{X}}^e_{\mathcal{H}^\Sigma}$ is associated with *diverging trajectories*: such trajectories are characterized by finite or infinite escape time, either in the continuous, or in the discrete component.

Remark 8 (On Zeno Phenomena). Zeno behaviors are peculiar phenomena that occur to trajectories in HCS models and which truly highlight their structural characteristics. Even in the case of the proposed characterization, it is interesting to stress that their characterization is "in between" that of deadlock and livelock. They are in fact similar to a blocking condition, in that they are not defined for all the (hybrid) time set, while adhering to the second group as they are endowed with an "infinite motion". We categorized them within the deadlock situations because of the practical outcome involving their presence, that of "stalling" a program that simulates them. □

6 Perspectives on Prevention and Verification of Deadlock and Livelock

The next step after the characterization of the notions of deadlock and livelock for HCS is that of deriving conditions on the original systems (both considered in separation) and on the composition operator, which can enforce the absence of deadlock/livelock behaviors on the composition, possibly without actually computing it (we shall talk

about *prevention*). Furthermore, it is also key to be able to check for the presence of these behaviors directly on the composition (*verification* problem).

In this section, we start to explore the *prevention* and *verification* problems, and we suggest that it is possible to relate them to problems already studied in the literature.

In order to *prevent* deadlock in a composition of two systems, we need to restrict the set of executions of the original HCSs. More precisely, it is sufficient to exclude specific dynamic behaviors related to deadlock situations: blocking, finite time stability, and Zeno.

For instance, when investigating the prevention of blocking executions in a composition, it can be shown that it is sufficient to raise non-blocking conditions on the original systems (as stated in Remark 2, equation (1)). Similarly, we can investigate conditions to prevent finite-time stable trajectories that may be related to deadlock behaviors in the composition. Such conditions will involve the interplay between guard sets, reset maps, and equilibrium points. Furthermore, we can investigate conditions to prevent Zeno behaviors in the composition. A number of approaches have appeared in the literature [28, 29], in part inspired by older work [6, 7], which entail the presence of exclusively non-blocking and non-Zeno executions for the HCS. These results hinge on the knowledge that the two original systems are Zeno-free. Necessary conditions for their existence have been discussed in the literature [20, 34, 19].

It is straightforward to speculate that the issue of verification of deadlock behaviors on a HCS is at best computationally hard, if not undecidable. In this paper, we limit the study to hinting at the main computational issues and we relate them to the adjacent literature. According to Definition 8, a verification procedure ought to focus on the presence of non-trivial *invariant regions* on the composed HCS, and check if such regions belong to the set of executions of the system.

In general, finding invariant sets is a computationally heavy task [10], let alone in the hybrid case [3]. The reason for this can be traced back to the variety of possible dynamical behaviors, especially switching ones. Procedurally, we can either resort to the back-propagation of equilibria or $\omega-$limit sets, or to the forward-propagation of sets from initial conditions. The literature on computational approaches to reach set computation for HSCs is quite rich [12, 9, 24, 25, 33, 16, 17, 21]. This intensive research has given birth to several verification tools for reachability analysis on hybrid systems, such as *d/dt*, *MATISSE*, *CheckMate*, the *Ellipsoidal Toolbox*, and many others. For a more exhaustive review, the reader is directed to `http://wiki.grasp.upenn.edu/~graspdoc/wiki/hst/`

7 Case Studies

Deadlock Example: Three cars crossing an intersection

Consider the interconnection of three hybrid models $\mathcal{H}_i, i \in \{1,2,3\}$, each of which describes the dynamics of a car moving along a road. The road has a stop sign. The model describes the dynamics associated with the position of the car, and is hence one dimensional. The stop sign is assumed to be positioned at the origin of the axis. The i^{th} car is controlled through its velocity—the control action depends on four qualitatively different states in the spatial domain. In the first (before and away from the intersection), the vehicle speeds toward the intersection, until it reaches the second region, where it decelerates until hitting the intersection. There (third region), it may move on and accelerates until getting out of a buffer zone, where (fourth region) the control steers the car away from the intersection. The HCS of each car is described by $\mathcal{H} = (X, U, Y, F, T)$. X is composed of the discrete state space $Q = \{1,2,3,4\}$ and the collection of domains $D = \{D_1, D_2, D_3, D_4\}$ where $D_1 = (-\infty, -c]$, $D_2 = [-c, 0]$, $D_3 = [0, d]$ and $D_4 = [d, +\infty)$, with the positive constants c, d. $U = [0, N], N < \infty$, is the control space, and $Y = \mathbb{R}$ is the observation space. We suppose that the output coincides with the state, i.e. we are in the case of full observability. $F = \{f_1, f_2, f_3, f_4\}$, where, considering a "small" constant $\delta > 0$,

$$f_1(x,u) = u + \delta, \quad f_2(x,u) = -\frac{u}{c}(x - \delta), \quad f_3(x,u) = \frac{u}{d}(x + \delta), \quad f_4(x,u) = u + \delta.$$

T is the set of transition relations, composed of $E = \{e_1, e_2, e_3\}$ with $e_1 = (1,2)$, $e_2 = (2,3)$ and $e_3 = (3,4)$. $G = \{G_e\}_{e \in E}$ is the set of guards, where for any $u \in U$, $G_{e_1}(u) = \{-c\}$, $G_{e_2}(u) = \{0\}$ and $G_{e_3}(u) = \{d\}$. $R = \{R_e\}_{e \in E}$, where $R_e(x) = x$, $\forall e \in E$.

We assume that the hybrid model of each car coincides with the HCS $\mathcal{H}$, i.e. $\mathcal{H}_i = (X_i, U_i, Y_i, F_i, T_i) = \mathcal{H}$, $i = 1,2,3$. The initial condition $x_i^o \in Init_i = (D_1)_i$. The control signals are cadlag functions. Notice that the structure of the vector fields with their simple dependence on the control signals, as well as the identity of the reset maps, defines the state trajectory, and hence the output, on the real time. It is hence not necessary to introduce the maps l, which turn output signals to interconnectable ones.

We specify for each system a reachability property, which denotes the possibility for the car to reach a certain point "way ahead of the stop sign":

$$\varphi_R(\mathcal{U}_i, K) := \exists u_i^* : [0,\infty) \to U_i, \exists\, 0 \le t^* < \infty : x_i(t^*) = K > d.$$

As above we have denoted with $\mathcal{U}_i$ the set of control functions in U_i. It should be clear that considering all the three systems, singularly taken, for each initial condition there exists a control (discrete and continuous) such that the system satisfies the reachability specification. More precisely,

$$\forall u_i(t) \in (0, N], t \geq 0, \forall x_i^o \in Init_i, x_i^o \models_{H_i} \varphi_R, \text{ hence } \mathcal{H}_i \models \varphi_R.$$

We can then conclude that $\mathcal{X}_{\mathcal{H}_i} = Init_i$. Notice that the left closure of the sets U_i make a different "attractivity" specification (universally quantified over the controls) be not verified.

We interconnect the three systems in order to model the intersection of a three-way stop (see Figure 1). As customary for the majority of real world traffic laws, we impose a "yield to the right" rule. Introduce the indicator function

$$\mathbf{1}_{[-c,0]}(x) = \begin{cases} 1, \text{ if } x \in [-c, 0]; \\ 0, \text{ else.} \end{cases}$$

Notice that the interconnection is "directed," in that a cycle between the systems is formed. More specifically, the connection between $\mathcal{H}_i$ and $\mathcal{H}_{(i+1)\text{mod}3}$ is specified by $\Sigma_{i,(i+1)\text{mod}3} = \{\mathbb{R} \times \varnothing, g_{i,(i+1)\text{mod}3} \times \tilde{g}\}, i \in \{0, 1, 2\}$, where $\tilde{g} = g_{(i+1)\text{mod}3,i}$ is undefined because $\mathcal{W}_{(i+1)\text{mod}3}$ is the empty set (again, because of the orientation of the interconnection). The following function defines the structure of the interconnected inputs:

$$u_i = g_{i,(i+1)\text{mod}3}(y_{(i+1)\text{mod}3}) = u_i \mathbf{1}_{[-c,0]^C}(x_{(i+1)\text{mod}3}) = \begin{cases} 0, \text{ if } y_{(i+1)\text{mod}3} \in [-c, 0]; \\ u_i, \text{ else.} \end{cases}$$

We have denoted with $[-c, 0]^C$ the complement of $[-c, 0]$. Given the structure of the $\Sigma_{i,(i+1)\text{mod}3}, i \in \{0, 1, 2\}$, the interconnection is associative. The three models are then composed with no particular order: define $\mathcal{H}^\Sigma = (((\mathcal{H}_1 ||_{\Sigma_{2,1}} \mathcal{H}_2) ||_{\Sigma_{3,(2,1)}} \mathcal{H}_3) ||_{\Sigma_{1,(3,(2,1))}} \mathcal{H}_1)$. Notice that, while having fully utilized the inputs of the three systems by connecting them, the use of the identity feedback map allows to obtain a pairwise composed system which is still partially controlled. More precisely, if the state of $\mathcal{H}_{(i+1)\text{mod}3}$ is in $[-c, 0]^C$, then $U^{\Sigma_{i,(i+1)\text{mod}3}} = U_i$. If instead the state of $\mathcal{H}_{(i+1)\text{mod}3}$ is in $[-c, 0]$, then the composition is dynamical (see Figure 2).

Assume now that the system parameters are the same for the three systems. Whenever the initial conditions of the three cars is the same, i.e. $x_1^0 = x_2^0 = x_3^0$ and the cars apply the same control policy until the intersection is reached, a deadlock situation appears. In fact the three cars shall stop at the intersection, each of them "waiting" for the next one on its right to proceed. This is a *deadlock* situation. By modifying the

width (parameter c) of the buffer zone, we can tune the "spatial sensitivity" that yields possible deadlock situations. From another perspective, for any combination of the parameters and a predefined choice of the control policy for each model, there exists a set of initial conditions that is associated with a deadlock. If we look at the system in its entirety, from a dynamical standpoint the obtained deadlock condition corresponds to finite-time stability.

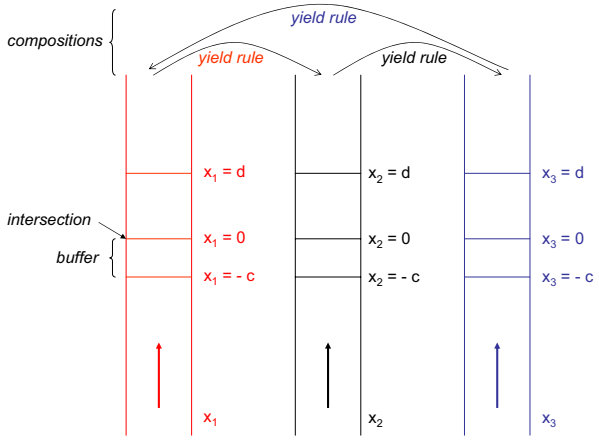

Fig. 1. Composition of the three dynamical control systems according to specific rules.

We suggest here that deadlock may be avoided if a certain level of coordination is attained between the three cars. More formally, a *centralized* control strategy is needed. If instead a (deterministic) decentralized approach is considered (that is, based just on the local information of the neighbor to the right), the above discussion should convince that deadlock cannot be avoided. In this last instance, a possible different approach would be that of *randomization* of the control strategy—this would require the introduction of a stochastic framework, which we postpone to future endeavors. As a related example, in communication systems, random-access algorithms are used for packet collision avoidance in a sender-receiver (i.e., decentralized) ethernet with shared resources. The topic of deadlock resolution is more formally studied in a ongoing research effort.

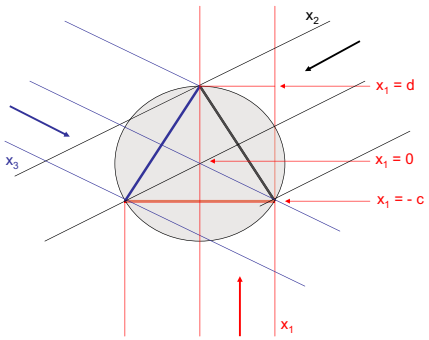

Fig. 2. Three cars crossing an intersection.

Livelock Example: Collision-Avoidance Maneuver for Decentralized Agents

Let us consider a dynamical model $\mathcal{H}$ for an autonomous agent, for instance an unmanned aerial vehicle (UAV). The objective of the UAV is that of tracking, in isolation, a point $x_0 = [(x_0)_1, (x_0)_2]^T$ in a two-dimensional plane. Let us assume in generality that Init $= D = \mathbb{R}^2$. The simple controlled dynamics of the UAV depend on the variable $x = [x_1, x_2]^T$, and are described by:

$$\begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 \end{pmatrix} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix},$$

where $u = [u_1, u_2]^T \in [-K, K]^2 \subset \mathbb{R}^2, K < \infty$, is an open-loop direct control on the velocity of the UAV. We can express the UAV goal as the following reachability specification:

$$\varphi_R(\mathcal{U}, x_0) := \exists u^* : [0, \infty) \to U, \exists\, 0 \leq t^* < \infty : x(t^*) = x_0.$$

The output space of the system $\mathcal{H}$ corresponds to its state space, $Y = X = D$. As the system under study is purely dynamical, the transition set is $T = \varnothing$. Let us consider a bounded, twice-continuously differentiable function $V : \mathbb{R}^2 \to \mathbb{R}^+$, such that $\nabla V(x_0) = [0, 0]^T$, and $\nabla^2 V(x)$ is negative definite, for all $x \in \mathbb{R}^2$. These conditions are sufficient to ensure that x_0 is a global maximum of V. Introduce the quantity $k = \max_{x \in \mathbb{R}^2} \|\nabla V(x)\|$, where $\|\cdot\|$ is any vector norm. The tracking objective can be easily achieved by selecting the following feedback control structure:

$$\begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \mu \begin{pmatrix} \frac{\partial V}{\partial x_1} \\ \frac{\partial V}{\partial x_2} \end{pmatrix}, \ \mu \in \left[-\frac{K}{k}, \frac{K}{k} \right].$$

This control tracks the point x_0 via a gradient ascent algorithm. Although our example is very simple, it effectively points out that a livelock can arise when using *navigation functions* [23] for the decentralized control of robots.

Let us now consider the "composition" of two such UAVs, $\mathcal{H}_1$ and $\mathcal{H}_2$. The connection is symmetric ($\Sigma_{1,2} = \Sigma_{2,1}$), and is specified by $\Sigma = \{[-K, K]^2, g_1 \times g_2\}$. Both UAVs are endowed with the same control strategy, which is decentralized as it does not exploit any mutual information of the state of the other controller. Assume further that both UAV have the same reachability specification, possibly with different end points x_0: $\varphi_1(\mathcal{U}_1, x_0^1), \varphi_2(\mathcal{U}_2, x_0^2)$. The composition between $\mathcal{H}_1$ and $\mathcal{H}_2$ is intended to reflect a collision-avoidance strategy. This strategy dictates to abruptly "veer to the right" if the distance between the two UAV gets to be smaller than a given constant radius $0 < r < \infty$. Let us refer to the variables $x^1 = [x_1^1, x_2^1]^T$ for $\mathcal{H}_1$ and $x^2 = [x_1^1, x_2^1]^T$ for $\mathcal{H}_2$, the controls u^1, u^2, and introduce the function

$$h(x^1, x^2) = \begin{cases} 1, & \text{if } d(x^1, x^2) \leq r, \\ 0, & \text{else,} \end{cases}$$

where $d(\cdot, \cdot)$ is a distance operator on $\mathbb{R}^2$. Consider the interconnected system

$$\begin{cases} \dot{x}^1 = \tilde{u}^1 \\ \dot{x}^2 = \tilde{u}^2 \end{cases}$$

defined by the inputs

$$\tilde{u}^1 = g_1(x^1, x^2) = u^1(1 - h(x^1, x^2)) + h(x^1, x^2)|V(x^1)| \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{pmatrix} x_2^1 \\ x_1^1 \end{pmatrix}$$
$$\tilde{u}^2 = g_2(x^1, x^2) = u^2(1 - h(x^1, x^2)) + h(x^1, x^2)|V(x^2)| \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{pmatrix} x_2^1 \\ x_1^1 \end{pmatrix}.$$

This interconnection yields a controlled model (regular point-tracking mode) if the two UAVs are distanced by more than r, while it results in a purely dynamical model (collision-avoidance mode) if the opposite holds. Notice that the interconnected system can be formally reframed as a HCS by the intuitive introduction of a guard set encoding the condition on the distance between the two UAVs. We leave this exercise to the interested reader.

The use of such a decentralized control (recall the similar situation for the deadlock in the previous example) may result in a livelock situation. Figure 3 describes the output of a simulation, where each UAV tracks a different end-point (in cyan color). Both controls depend on two Gaussian functions, centered around the specific end-point. The trajectories, in red and magenta, originate from the green points. The plots are both

in the two dimensional plane, as well as on the surface of the corresponding function. Notice that the UAV temporarily engage in a collision-avoidance situation, which is then resolved. In other words, while the two UAV tangentially influence each other, they end up achieving their goal.

Figure 4 instead shows the composition of two systems with the same objective point, $x_0^1 = x_0^2 = x_0$. A single Gaussian is then used to devise the control function. This condition results in a situation of livelock. The plots of the trajectories (in red and magenta) are in two dimensions (left), as well as lifted on the Gaussian function (right). The UAV's start from the green points, and aim at the point colored in cyan.

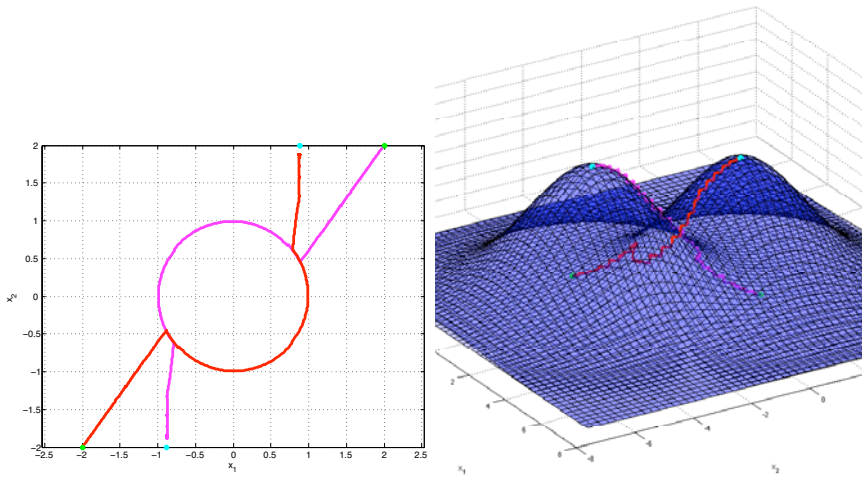

Fig. 3. Non-pathological behavior in a collision-avoidance maneuver for two UAVs.

Similar to the deadlock example, the presence of a livelock situation is related to the use of a decentralized control structure.

8 Conclusions and Future Work

This work has introduced the notion of deadlock and livelock behaviors for deterministic Hybrid Control Systems. The formal categorization of the notions of the notions of deadlock and livelock to the hybrid framework extend known concepts in the literature. It is important to notice that the ideas introduced in the paper tailor to known ones from the literature on discrete transition systems, HIOA, or dynamical systems.

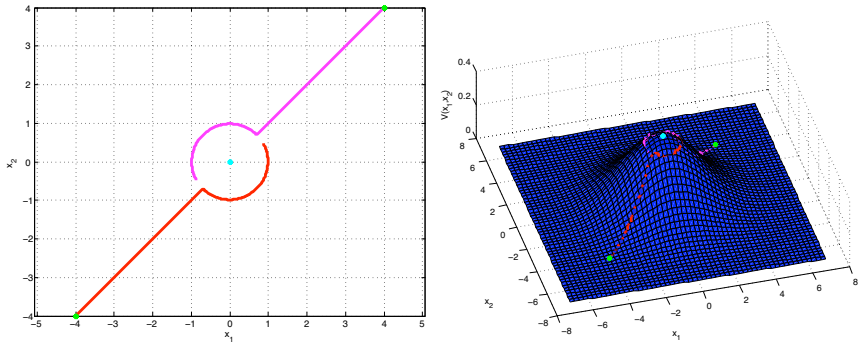

Fig. 4. Livelock situation resulting from a collision-avoidance maneuver for two UAVs.

Deadlock and Livelock *verification, prevention,* and *resolution* are topics that only find a limited space in this paper, and which the authors are currently formally investigating. In particular, the use of randomization in the control strategies is a technique that is commonly used to resolve pathological behaviors, and which can be framed within the proposed approach, at the expense of introducing a probabilistic framework.

The authors are also working on a number of extensions of the presented results. The concept of composition is prone to be generalized, and the issue of *deep composition* [15], that is of a composition procedure preserving certain properties by construction, clearly connects with the presented work when the absence of deadlock or livelock is the specification to be exported.

Acknowledgments

The authors would like to thank Giordano Pola for the initial contributions to the project. The work has been partially supported by the NSF grant CCR-0225610 and STREP project n. TREN/07/FP6AE/S07.71574/037180 iFLY.

References

1. M. Abadi and L. Lamport. Composing specifications. In *REX Workshop on Stepwise Refinement of Distributed Systems*, Mook, NL, May 1989.
2. Martin Abadi and *al.* Preseving liveness: Comments on "safety and liveness from a methodological point of view. *Information Processing Letters*, 40-3:141–142, 1991.
3. A. Abate, A. Tiwari, and S. Sastry. Box invariance for biologically-inspired dynamical systems. In *46th IEEE Conference on Decision and Control and European Control Conference*, pages 5162–5167, New Orleans, LA, 2007.

4. Bowen Alpern and Fred Schneider. Defining liveness. *Information Processing Letters*, 21:181–185, 1985.
5. R. Alur, C. Courcoubetis, T.A. Henzinger, and P. Ho. Hybrid automaton: an algorithmic approach to the specification and verification of hybrid systems. In R.L. Grossman, A. Nerode, A.P. Ravn, and H. Rishel, editors, *Hybrid Systems*, LNCS 736, pages 209–229. Springer Verlag, 1993.
6. R. Alur and T. Henzinger. Reactive modules. In *Proceedings of the 11th IEEE Symposium on Logics in Computer Science (LICS)*, pages 207–218, 1996.
7. R. Alur and T. Henzinger. Modularity for timed and hybrid systems. In *Proceedings of the 8th International Conference on Concurrency Theory (CONCUR 97), LNCS 1243*, pages 74–88, 1997.
8. A. D. Ames, A. Abate, and S. Sastry. Sufficient conditions for the existence of Zeno behavior. In *Proceedings of the 44th IEEE Conference on Decision and Control and European Control Conference*, pages 696–701, Seville, SP, 2005.
9. E. Asarin, O. Bournez, T. Dang, and O. Maler. Approximate reachability analysis of piecewise linear dynamical systems. In *Hybrid Systems: Computation and Control, Pittsburgh, USA*, Lecture Notes in Computer Science. Springer Verlag, March 2000.
10. F. Blanchini. Set invariance in control. *Automatica*, 35:1747–1767, 1999.
11. S. Bornot and J. Sifakis. On the composition of hybrid systems. In *Hybrid Systems: Computation and Control, Berkeley, USA*, volume 1386 of *Lecture Notes in Computer Science*, pages 69–83. Springer Verlag, 1998.
12. A. Chutinan and B. Krogh. Computing polyhedral approximations to flow pipes for dynamic systems. In *Proceedings of the* 37^{th} *IEEE Conference on Decision and Control, Tampa, FL*, pages 2089–2094, December 1998.
13. Edmund M. Clarke, Orna Grumberg, and Doron A. Peled. *Model Checking*. MIT Press, Boston, 2000.
14. Jennifer Davoren and Paulo Tabuada. On simulations and bisimulations of general flow systems. In A. Bemporad, A. Bicchi, and G. Buttazzo, editors, *Hybrid Systems: Computation and Control*, LNCIS 4416, pages 145–158. Springer Verlag, 2007.
15. L. de Alfaro, T. A. Henzinger, and R. Jhala. Compositional methods for probabilistic systems. In *CONCUR – Concurrency Theory*, LNCS 2154, pages 351–365. Springer Verlag, 2001.
16. A. Girard. Reachability of uncertain linear systems using zonotopes. In M. Morari and L. Thiele, editors, *Hybrid Systems: Computation and Control*, volume 3414 of *Lecture Notes in Computer Science*, pages 291–305. Springer Verlag, 2005.
17. Zhi Han and B. H. Krogh. Reachability analysis of large–scale affine systems using low–dimensional polytopes. In J. Hespanha and A. Tiwari, editors, *Hybrid Systems: Computation and Control*, volume 3927 of *Lecture Notes in Computer Science*, pages 287–301. Springer Verlag, 2006.
18. M. Heymann, F. Lin, G. Meyer, and S. Resmerita. Analysis of Zeno behaviors in hybrid systems. In *Proceedings of the 41st IEEE Conference on Decision and Control*, Las Vagas, NV, 2002.
19. M. Heymann, F. Lin, G. Meyer, and S. Resmerita. Analysis of Zeno behaviors in a class of hybrid systems. *IEEE Transactions on Automatic Control*, 50(3):376–383, 2005.
20. K. H. Johansson, M. Egerstedt, J. Lygeros, and S Sastry. On the regularization of Zeno hybrid automata. *Systems and Control Letters*, 38:141–150, 1999.
21. A.A. Julius, G. Fainekos, M. Anand, I. Lee, and G.J. Pappas. Robust test generation and coverage for hybrid systems. In A. Bemporad, A. Bicchi, and G. Buttazzo, editors, *Hybrid Systems: Computation and Control, To appear*, LNCIS 4416. Springer Verlag, 2007.
22. E. Kindler. Safety and liveness properties: A survey. *Bulletin of the European Association for Theoretical Computer Science*, 53:268–272, October 1994.

23. D.E. Koditschek and E. Rimon. Robot navigation functions on manifolds with boundary. *Advances in Applied Mathematics*, 11(4):412–442, 1990.
24. A. B. Kurzhanski and P. Varaiya. Ellipsoidal techniques for reachability analysis. In N. Lynch and B.H. Krogh, editors, *Hybrid Systems: Computation and Control*, Lecture Notes in Computer Science 1790, pages 202–214. Springer Verlag, 2000.
25. G. Lafferriere, G. J. Pappas, and S. Yovine. Symbolic reachability computations for families of linear vector fields. *Journal of Symbolic Computation*, 32(3):231–253, September 2001.
26. J. Lygeros. On reachability and minimum cost optimal control. *Automatica*, 40 - 6:317–327, 2004.
27. J. Lygeros, K. Johansson, S. Simic, J. Zhang, and S. Sastry. Dynamical properties of hybrid automata. 48(2-18), 2003.
28. Nancy Lynch, Roberto Segala, and Frits Vaandrager. Hybrid I/O automata revisited. In M.D. Di Benedetto and A. Sangiovanni-Vincentelli, editors, *Hybrid Systems: Computation and Control*, LNCIS 2034, pages 403–417. Springer Verlag, 2001.
29. Nancy Lynch, Roberto Segala, and Frits Vaandrager. Hybrid I/O automata. *Information and Computation*, 185(1):105–157, 2003.
30. Joseph Sifakis. The compositional specification of timed systems. In *Computer-Aided Verification*, Trento, IT, 1999.
31. E. Sontag. *Mathematical Control Theory: Deterministic Finite Dimensional Systems, 2nd edition*. Springer Verlag, New York, 1998.
32. P. Tabuada, G. Pappas, and P. Lima. Compositional abstractions of hybrid control systems. *Journal of Discrete Event Dynamical Systems*, 14(2):203–238, 2004.
33. H. Yazarel and G. J. Pappas. Geometric programming relaxations for linear system reachability. In *Proceedings of the 2004 American Control Conference, Boston, MA*, June 2004.
34. J. Zhang, K. H. Johansson, J. Lygeros, and S. Sastry. Zeno hybrid systems. *International Journal of Robust and Nonlinear Control*, 11(5):435–451, 2001.