Set-Theoretic Estimation of Hybrid System Configurations

Emmanuel Benazera and Louise Travé-Massuyès, *Senior Member, IEEE*

***Abstract*—Hybrid systems serve as a powerful modeling paradigm for representing complex continuous controlled systems that exhibit discrete switches in their dynamics. The system and the models of the system are nondeterministic due to operation in uncertain environment. Bayesian belief update approaches to stochastic hybrid system state estimation face a blow up in the number of state estimates. Therefore, most popular techniques try to maintain an approximation of the true belief state by either sampling or maintaining a limited number of trajectories. These limitations can be avoided by using bounded intervals to represent the state uncertainty. This alternative leads to splitting the continuous state space into a finite set of possibly overlapping geometrical regions that together with the system modes form configurations of the hybrid system. As a consequence, the true system state can be captured by a finite number of hybrid configurations. A set of dedicated algorithms that can efficiently compute these configurations is detailed. Results are presented on two systems of the hybrid system literature.**

***Index Terms*—Configurations, estimation, hybrid systems, numerically bounded uncertainty.**

I. Introduction

THIS paper is concerned with the state estimation of plants that are modeled as hybrid systems with uncertainty. It is targeted at the monitoring and diagnosis of these plants. Most of the modern controlled systems exhibit continuous dynamics with abrupt switches. These systems can be modeled with a mixture of discrete and continuous variables. The discrete dynamics evolve according to the switches that are represented by transitions among a set of discrete modes. The behavioral continuous dynamics are modeled within each mode, often by a set of discrete-time equations. In general, the full hybrid state remains only partially observable. Depending on the level of abstraction of the model, or because of physical or design impediments, some switches cannot directly be observed neither. The estimation of the hybrid state is the operation that reconstructs the whole hybrid state based on a stream of measurements and the knowledge of the hybrid model. This is also known as hybrid state filtering, and the module that performs this operation is called a filter. Most plants operate in uncertain environments and are not accurately known due to the presence of sensor and process uncertainties. As a consequence, transitions among modes may be nondeterministic, and continuous behavioral models may embed a representation of instrumentation and process uncertainties. It follows that modern filtering algorithms must cope with uncertainty. Probabilities and bounded sets are two main representations of uncertainty.

State estimation of hybrid systems has received increased attention in the last decade or so. However, while the systems are hybrid in nature, a first set of methods and algorithms for hybrid state estimation has remained close to continuous state estimation techniques [1]–[3]. Another cluster of approaches has mixed a heterogeneous set of techniques for continuous state estimation with qualitative reasoning [4]–[8]. Another set is formed with particle filtering methods whose focus is on the sampling of discrete transitions [9]–[11]. This group of filters has emerged as the set of most popular techniques. Basically, they apply a Bayesian belief update to stochastic hybrid systems [10]–[14]. The filter computes a posterior probability distribution function (pdf) on the continuous part of the state for each mode. Measurement likelihood w.r.t. the pdfs is used with transition probabilities to rank the possible hybrid state estimates. These methods all suffer from several weaknesses.

The main drawback is an inevitable blowup of the number of state estimates, which are also called hypotheses. It stems from the fact that the statistics that are maintained on hypotheses with the same discrete states cannot be merged without loss. The blowup is particularly intractable when the hybrid system represents faults by discrete switches that may occur at any time. Several works have explored methods for mitigating the blowup: through better use of available information by looking ahead [15] or by enumerating the first few best estimates [16]; by merging estimates [17], [18]; and hierarchical filtering [19], risk sensitive sampling [20], learning [21], forward heuristic search [14], or mixed sampling and search [22]. However, the blowup remains inevitable, and some states with low probabilities must be dropped. Unfortunately, this can lead to the loss of the true state [23].

A second problem lies in the infinite tails of the representational pdfs. In practice, the Gaussian distribution is widely used for representing the belief states due to its good statistical properties. The distribution tails are the cause of several problems by notably preventing unambiguous fault detection [24] and elimination of hypotheses. Working with truncated Gaussian pdfs [25] has been studied as an alternative, but is unattractive

Manuscript received March 21, 2008; revised July 14, 2008 and November 4, 2008. This paper was recommended by Associate Editor W. J. Wang.

The authors are with LAAS–Centre National de Recherche Scientifique (CNRS), Université de Toulouse, 31077 Toulouse Cedex 4, France (e-mail: ebenazer@laas.fr; louise@laas.fr).

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

Digital Object Identifier 10.1109/TSMCB.2009.2015280

due to the loss of the statistical properties, e.g., Bayes' rule does not yield a truncated Gaussian.[1]

Additionally, the stochastic modeling of faults is weak, since for a good part, the modeled faults have never been observed, and, thus, *a priori* numerical knowledge such as probability of occurrence is indicative at best. The reliability of the produced results can, therefore, be questioned. Nevertheless, the literature has produced a plethora of algorithms that run a recurrent and rigorous Bayesian belief update on these values and that require the computation of difficult integrands [26].

Finally, current modeling formalisms do not accept constraints that mix discrete and continuous variables. In general, constraints over discrete variables apply to operational modes, and a set of linear or nonlinear equations link continuous variables in each mode. However, in case of software systems or abstracted continuous behavior systems, qualitative descriptions are better suited [27], [28]. There is a need for constraints that formally capture dependencies between variables of different types. The absence of such constraints prevents a natural connection between variables of different types and, consequently, decouples variables that are strongly coupled in nature.

Adding up the facts, it appears that pdfs are simply badly suited to the state estimation of uncertain hybrid systems with fault models. Such considerations are not new even for continuous systems [29]. Tackling the ambiguity that plagues the stochastic filters recommends a bounded representation of uncertainty as adopted in set-theoretic approaches. Set-theoretic state estimation of linear and nonlinear systems [30]–[33] has been studied before, but not the case of hybrid systems. This paper fills this gap by developing a hybrid scheme that supports bounded uncertainty with interval models. A special look is given at the articulation of discrete and continuous dynamics in that case. Doing so aims at circumventing most of the drawbacks that have been mentioned. Bounded uncertainty yields several advantages compared to pdfs. First, it provides guaranteed results, i.e., an enclosure of the whole set of real solutions. For this reason, the use of bounded uncertainty has been popular in applications to fault detection and diagnosis, since it avoids false-positive detections [34]. Second, and most importantly, it prevents exponential blowup in the number of state estimates. The reason behind this key property is that estimates with identical discrete states can be merged with no loss of information, i.e., preserving completeness; although this comes at a price. The recursive computation of convex bounded trajectories suffers from the well-known *wrapping effect* that results from the convex enclosure at each prediction step. This is because the convex bounds provide an outer approximation of complex geometrical shapes, and their computation is thus plagued with a recursively growing error. This problem calls for aggressive optimization techniques to mitigate the error growth. Another well-known problem related to intervals is multiple incident parameters. Specific strategies like optimization over a time-sliding window may then be required [35]. Summarizing, the computational burden of a stochastic filter comes from the need of tracking a very high number of belief states, whereas that of set-theoretic hybrid state estimation lies in the computation of tight bounds. However, as this paper shows, switched systems sometimes offer a cheap way of tightening the bounds as a side effect of their chopped dynamics.

The alternative idea proposed in this paper leads to splitting the continuous state space into a finite set of possibly overlapping geometrical regions that, together with the system modes, form configurations of the hybrid system. As a consequence, the true system state can be captured by a finite number of hybrid configurations. This paper contrasts with the pure prediction performed in reachability analysis of hybrid systems [36]. First, because our estimator reconstructs the hybrid state for arbitrary continuous dynamics and switching conditions. Second, because it incrementally operates in sampled time: discrete switches that occur between two sampled time steps are reconstructed by our estimator.

Overall, this paper proposes a hybrid estimation method that aims at computing an outer approximation of the hybrid state. In Section II, this paper formalizes a hybrid modeling scheme that naturally embeds both bounded uncertainty and mixed discrete/continuous constraints over the hybrid state. Based on these two ingredients, it is shown that there exists a special form of mixed constraints that fully capture a system hybrid configuration under uncertainty. Here, a *configuration* is a mixed continuous/discrete constraint that characterizes the possible hybrid states of the system at a given point in time. Configurations are detailed in Section III. The hybrid state estimation process is developed in Section IV. It is a matured version of the work initiated in [37]. The experimental results are given in Section V.

II. Hybrid System With Unknown but Bounded Uncertainty

We represent a physical plant as a nondeterministic and uncertain hybrid discrete-time model. This representation has several key features that significantly differ from the existing formalisms. First, all continuously valued variables are assumed to be uncertain but numerically bounded. Second, the formalism uses two timescales in parallel for the discrete and continuous dynamics, respectively. This permits an unknown but finite number of instantaneous switches in the discrete dynamics to occur in-between two steps of the continuous dynamics. Third, the representation does not make any particular assumption on the conditions triggering the switches, particularly w.r.t. the continuous state of the system. Finally, the model supports both qualitative and quantitative behavioral representations. For this reason, our formalism is richer than more traditional ones such as [38] and suitable for modeling a wide range of physical components and plants. To help the reader throughout this paper, Table I sums up the main notations.

Definition 1 (Hybrid System): A hybrid system H is represented by a tuple

$$H = (X, E, Q, \mathcal{T}, L, \Theta) \tag{1}$$

[1]Interestingly, whenever some data or signal is discarded from a Gaussian distribution for falling below a threshold, the resulting data do obey a truncated Gaussian. Applying Bayes rule and approximating the resulting belief state with a new Gaussian increases the error recursively.

TABLE I
MAIN NOTATIONS

H	hybrid system.
$\mathbf{x}_{c,k}$	continuous state vector at sampled time-step k.
$\mathbf{y}_{c,k}$	observable state vector at sampled time-step k.
$\tilde{\mathbf{y}}_{c,k}$	observed state vector at sampled time-step k.
$x_{m,l}$	system mode at logical time-step l.
$\mathbf{x}_{d,l}$	discrete state vector at logical time-step l.
$\boldsymbol{\pi}_l = (\mathbf{x}_{m,l}, \mathbf{x}_{d,l})$	full discrete state vector at logical time-step l.
$\mathbf{s}_{l,k} = (\boldsymbol{\pi}_l, \mathbf{x}_{c,k})$	hybrid state vector at date (l, k).
$\mathbf{s}_{l,k}^{(p)} = (\boldsymbol{\pi}_l^{(p)}, \mathbf{x}_{c,k}^{(p)})$	p-th hybrid estimate at date (l, k).
ϕ_i^j	partial guard of transition τ^j in continuous dimension i.
g_i^j	condition function that determines the Boolean value of ϕ_i^j.
$\tilde{\mathbf{x}}_{c,k}^j$	positive domain: enabling region of transition τ^j in $\mathbf{x}_{c,k}$.
$\mathbb{X}_k$	conditional domain of $\mathbf{x}_{c,k}$.
$\mathcal{C}_k$	configuration of H at sampled time-step k.
$\mathbf{r}_{\mathcal{C}_k}$	configuration region at sampled time-step k.
$\boldsymbol{\kappa}_d^i$	conditional vector.
$\nabla\delta_k$	logical configuration region at sampled time-step k.

where $X = \{X_d, X_c\}$ is the set of discrete and continuous variables, respectively, E is the set of difference equations, Q is the set of propositional formulas, $\mathcal{T}$ is the set of transitions, L is the set of continuous mapping functions associated to transitions, and Θ's are the initial variable values.

A. Variables and States

A hybrid system H abstracts the behavior of a physical system through a set of functional modes. The system *mode* is x_m, which has domain $\{m_1, \ldots, m_{n_m}\}$. The full discrete state is noted as $\boldsymbol{\pi} = (x_m, \mathbf{x}_d)$, where $\mathbf{x}_d = [x_{d1}, \ldots, x_{dn_d}]^T$ is the vector of other discretely valued variables used to describe qualitatively abstracted continuous behavior within modes. Therefore, $X_d = \{x_m, x_{d1}, \ldots, x_{dn_d}\}$. The system mode is assumed not to be directly observable. $\mathbf{y}_d$ denotes the observable subpart of $\mathbf{x}_d$. The vector of actually observed discrete values is noted as $\tilde{\mathbf{y}}_d$. The discrete input vector is noted as $\mathbf{u}_d$.

The continuous dynamics of the system are captured by the continuous state vector $\mathbf{x}_c = [x_{c1}, \ldots, x_{cn_c}]^T$, the observation vector $\mathbf{y}_c$, and the continuously valued input vector $\mathbf{u}_c$. The vector of actually observed values is noted as $\tilde{\mathbf{y}}_c$. X_c is the set of all continuous variables. The continuous state is represented with uncertainty in a bounded form. Thus, $\mathbf{x}_c$ is an interval vector (a box) in the continuous state space. That is, $\mathbf{x}_c$ is a closed and connected rectangular subset of $\Re^{n_c}$, or equivalently, $\mathbf{x}_c \in IR^{n_c}$, where IR is the set of real-valued intervals. The hybrid state of the system is noted as $\mathbf{s} = (\boldsymbol{\pi}, \mathbf{x}_c)$.

B. Time and Dynamics

1) Continuous Dynamics: Every mode is associated to a unique continuous evolution model. The continuous behavior of the physical system is modeled by a finite set of difference equations in E with uncertain but bounded parameters. In each mode, x_m corresponds to a subset of discrete-time equations of the following standard form, assuming a sampling period T_s:

$$\mathbf{x}_{c,k} = f(\mathbf{x}_{c,k-1}, \mathbf{u}_{c,k-1}, \mathbf{w}_{c,k-1}, x_m) \tag{2}$$

$$\mathbf{y}_{c,k} = h(\mathbf{x}_{c,k}, \mathbf{v}_{c,k}, x_m) \tag{3}$$

where (2) is the state equation, (3) is the measurement equation, k is the discrete-time index, and $\mathbf{w}_c = [w_{c1}, \ldots, w_{cn_w}]^T$ and $\mathbf{v}_c = [v_{c1}, \ldots, v_{cn_v}]^T$ represent the process and measurement noise vectors, respectively, and are assumed to be independent. This uncertainty and the parameters defining f and h are assumed to be unknown but numerically bounded. In particular, this means that $\|\mathbf{w}_c\|_\infty \leq \epsilon_w$ and $\|\mathbf{v}_c\|_\infty \leq \epsilon_v$, where ϵ_w and ϵ_v are known positive scalars. $\|.\|_\infty$ denotes the ∞-norm such that $\|\mathbf{w}_c\|_\infty = \max_i |w_{ci}|$, $i = 1, \ldots, n_w$.

What we denote the *sampled timescale* is the timeline that is explicit in (2) and (3). The sampled time step k thus labels the kth sampling period between continuous instants $T_s(k-1)$ and $T_s k$. $\mathbf{x}_{c,k}$ and $\mathbf{y}_{c,k}$ are the valuations of the continuous state and the output at sampled time step k.

2) Discrete Dynamics: A need for an abstracted qualitative representation of behavior was discussed in Section I. Behaviors that are naturally expressed by means of discrete variables, like those of embedded software, also need to be represented. Thus, at a discrete level, these descriptions are written in propositional logic by a set of time-independent propositional formulas Q over discrete variables of X_d.

What we denote as the *logical timescale* marks the sequence of changes in the discrete dynamics of the system. With $\boldsymbol{\pi}_l = (x_{m,l}, \mathbf{x}_{d,l})$, we specify the discrete state at logical time step l. The switches from one mode to another are represented by *transitions*. Transition τ switches H from mode $x_{m,l}$ to mode $x_{m,l+1}$. $\mathcal{T}$ is the set of $n_{\mathcal{T}}$ transitions of H. Transitions are of the following different types.

1) *Autonomous* transitions are triggered by conditions over the continuous state. These conditions are referred to as *guards* and noted $\phi : \mathbf{x}_c \to \{0, 1\}$. Section III conducts an in-depth analysis of guards.
2) *Commanded* transitions are triggered by discrete commands $\mathbf{u}_d$.
3) *Unpredictable* transitions have no guards and can trigger anytime, for instance, fault transitions.

A transition is said to be *enabled* whenever its guard is realized. Nondeterminism arises from the possibility of having multiple transitions enabled simultaneously. When enabled, a transition *triggers* a mode change. After a transition τ has triggered and switched the system mode from $x_{m,l}$ to $x_{m,l+1}$, the continuous state $\mathbf{x}_{c,k}$ becomes $l_\tau(\mathbf{x}_{c,k})$, where l_τ is denoted as the transition *mapping function*.

Transitions are assumed to be instantaneous. However, when abstracting certain behaviors using a hybrid model, it appears that transitions may have nonnegligible duration. The present framework supports the triggering of a transition after a certain delay has expired. Importantly, the transition triggering remains instantaneous. Thus, the duration of a transition is really to be understood as a delay, that is, a certain number d of sampled time steps before an enabled transition does trigger and

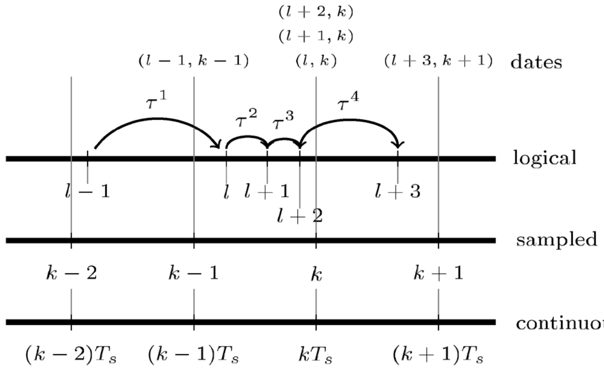

Fig. 1. Discrete and continuous parallel timescales. Transitions are instantaneous but are represented by arrows from the previous logical time step to the time step at which they trigger (e.g., τ^1 triggers at l). Dates synchronize the timescales at every sampled time point.

does lead to a different mode. Assuming that transition τ has its autonomous guard enabled in $x_{c,k}$, it triggers d-sampled time steps later, and the continuous arrival state is given by $l_\tau(x_{c,k+d})$. In the rest of this paper, we assume that $d = 0$ with no loss of generality.

3) Discrete and Continuous Parallel Timescales: As mentioned above, our representation uses two discretized timescales in parallel on top of the continuous timescale: the sampled and the logical timescales. As a consequence, changes in the discrete dynamics are not assumed to take place at a particular sampled time step, but can occur in-between two sampled time steps. However, hybrid states need to be synchronized in time. Because the sampled time evolves according to a fixed sampling period T_s, the logical time is synchronized with the sampled time, and not the opposite. In consequence, the logical time is always associated to the first sampled time step that follows a switch (see Fig. 1). Note that for this reason, an instantaneous switch is always triggered after its occurrence on the physical system. In this context, (l, k) is a *date* for the system, and $\mathbf{s}_{l,k}$ denotes the hybrid state at logical time step l and sampled time step k. We assume that a finite but unknown number of switches can occur between two sampled time steps. In this case, hybrid states are indexed by dates whose sampled indexes are the same, but with different logical indexes (see time step k in Fig. 1). In this formulation, the execution (solution trajectory) of the proposed class of hybrid systems is a succession of hybrid states at established dates. The execution corresponding to the succession of dates in Fig. 1 is written as $s_{l-1,k-2}, s_{l-1,k-1} \xrightarrow{\tau^1} s_{l,k} \xrightarrow{\tau^2} s_{l+1,k} \xrightarrow{\tau^3} s_{l+2,k} \xrightarrow{\tau^4} s_{l+3,k+1}$.

C. Example

Example 1 (Thermostat System): The temperature x of a room is controlled by a thermostat that keeps it between $x^{\min}$ and $x^{\max}$ degrees by switching a heater on and off. The system is modeled as a hybrid system H. $X_d = \{x_m\}$ with domain $\{m_1 = \text{off},\ m_2 = \text{on},\ m_3 = \text{stuck on},\ m_4 = \text{stuck off}\}$. $\mathbf{x}_c$ is reduced to the temperature x of the room, and $\mathbf{u}_c$ is reduced to the input u. The continuous dynamics of the system are modeled by the first-order differential equation $\dot{x} = D(u - x)$, where D is a multiplying factor. We model $E = \{E_{m_1}, E_{m_2}, E_{m_3}, E_{m_4}\}$ with $E_{m_1} = E_{m_4}$ such that $u = \bar{x}$ (i.e., the temperature outside the room), and $E_{m_2} = E_{m_3}$ such that $u = h$ (i.e., the heater constant whose value is uncertain but bounded). In discretized form, the dynamics are given by the following recurrent equation in standard form (2): $x_k = ax_{k-1} + bu_{k-1}$, with $a = 1 - DT_s$, and $b = DT_s$, assuming a sampling period T_s. Q is empty, and $\mathcal{T} = \{\tau^1, \tau^2, \tau^3, \tau^4\}$, where $\tau^1 : m_2 \xrightarrow{\phi^1 = 1 \text{ if } (x \geq x^{\max})} m_1$, $\tau^2 : m_1 \xrightarrow{\phi^2 = 1 \text{ if } (x \leq x^{\min})} m_2$, $\tau^3 : m_2 \xrightarrow{\phi^3 = 1 \text{ if } (x \geq x^{\max})} m_3$, and $\tau^4 : m_1 \xrightarrow{\phi^4 = 1 \text{ if } (x \leq x^{\min})} m_4$. Notice that $\phi^1 = \phi^3$, and $\phi^2 = \phi^4$. L associates the identity function to every transition.

III. Set-Theoretic Hybrid Configurations

This section formalizes the concept of configuration of a hybrid system. A canonical form of a transition guard is given. It leads to the definition of a configuration as a rectangular bounded region that enables a possibly empty set of transitions. Another contribution is the logical abstraction of a configuration that articulates the discrete and continuous dynamics of the hybrid system. This formulation paves the way for the estimation algorithms in Section IV.

A. Transition Guards

Commanded transition triggering is conditioned over the discretely valued inputs $\mathbf{u}_d$, but these conditions are directly expressed as constraints at the discrete level and do not require specific processing. Autonomous transitions require more attention.

Definition 2 (Autonomous Transition Guard): The guard of an autonomous transition τ^j is noted as $\phi^j : \mathbf{x}_c = (x_{c1}, \ldots, x_{cn})^T \rightarrow \{0, 1\}$. $\phi^j(\mathbf{x}_c)$ can be expressed as a set of inequalities in the canonical form given in the if condition of (5). The inequalities referring to a given state variable x_{ci} define the partial guard $\phi_i^j(\mathbf{x}_c)$ as

$$\phi_i^j(\mathbf{x}_c) = \bigwedge_\alpha \phi_{i_\alpha}^j(\mathbf{x}_c) \tag{4}$$

$$\phi_{i_\alpha}^j(\mathbf{x}_c) = \begin{cases} 1, & \text{if } x_{ci} \lesseqgtr g_{i_\alpha}^j(x_{c1}, \ldots, x_{ci-1}, x_{ci+1}, \ldots, x_{cn}) \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

where $g_{i_\alpha}^j : \mathbf{x}_c \rightarrow \Re$ is referred to as a *condition function*, and $\lesseqgtr$ stands either for "$\leq$" or "$\geq$."

The index i_α identifies one specific condition function in the set of condition functions referring to transition τ^j and variable x_{ci}. Note that no assumption is made on the form of the condition functions.[2] For the sake of clarity, in the rest of this paper, we make two simplifying assumptions. First, we assume that the set of condition functions is either empty or

[2]The inequality canonical form does not limit the expressiveness of the framework. Complex inequalities can always be manipulated to be brought back to this form, possibly by introducing new variables.

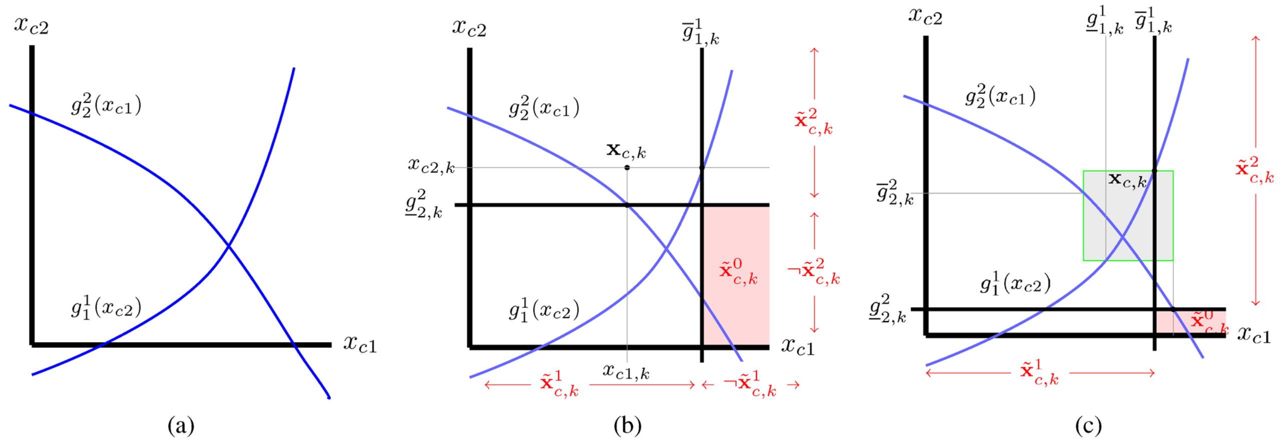

Fig. 2. Example 2: generic 2-D situation with guards $\phi_1^1 : x_{c1} \leq g_1^1(x_{c2})$ and $\phi_2^2 : x_{c2} \geq g_2^2(x_{c1})$. The positive and negative subdomains are computed from conditional functions g_1^1 and g_2^2 taken at $\mathbf{x}_{c,k}$, or at its corners when it is a box. The upper bounds to conditional domains $\overline{g}_i^j(\mathbf{x}_{c,k})$ are abbreviated as $\overline{g}^j_{i,k}$. A similar abbreviation is used for lower bounds. They yield $\mathbb{X}_k = [\tilde{\mathbf{x}}^0_{c,k}, \tilde{\mathbf{x}}^1_{c,k}, \tilde{\mathbf{x}}^2_{c,k}]$. (a) Functional representation of the guards. (b) Real-valued $\mathbf{x}_{c,k}$. (c) Uncertain $\mathbf{x}_{c,k}$ is a hyperrectangle.

of cardinality 1 for every x_{ci} and τ^j. In other words, there is at most one inequality referring to a variable x_{ci} associated to a partial guard ϕ_i^j. Second, we assume that $\phi_i^j(\mathbf{x}_c) = 1$ whenever the set of condition functions is empty (i.e., g_i^j is not specified). This allows us to write $\phi^j(\mathbf{x}_c) = \bigwedge_{i=1}^{n_c} \phi_i^j(\mathbf{x}_c)$.

Unpredictable transitions are modeled with guards such that $\phi^j = 1$, independent of $\mathbf{x}_c$. When the model contains guards as disjunctions of inequalities, these can be broken into guards over several transitions and modes. Admittedly, the modeling of a discrete switch as a transition whose guard is made of e disjunctions of inequalities necessitates a total of 2^e modes.

τ^j is said to be enabled in the hybrid state $\mathbf{s} = (\boldsymbol{\pi}, \mathbf{x}_c)$ whenever $\phi^j(\mathbf{x}_c) = 1$. When enabled, the triggering of the transition is an instantaneous transfer of the hybrid state to another state (possibly identical) at the next logical time step. This operation is detailed in Section IV along with the hybrid state estimator. The rest of this section studies the structure of the continuous space as constrained by the autonomous transition guards.

B. *Grid of Configurations*

At sampled time step k, the evaluation of transition guards against a continuous vector $\mathbf{x}_{c,k}$ is done through the evaluation of the condition functions $g_i^j(\mathbf{x}_{c,k})$. Each inequality referring to a condition function indeed splits the domain of $\mathbf{x}_{c,k}$ in two subdomains.

1) $\tilde{\mathbf{x}}^j_{c,k} = \{(x_{c1}, \ldots, x_{cn_c})^T | \phi^j(\mathbf{x}_{c,k}) = 1\}$: The region that satisfies the inequalities or *positive* subdomain. $\tilde{\mathbf{x}}^j_{c,k}$ denotes the region in which transition τ^j is enabled at sampled time step k.
2) The region that does not satisfy the inequality or negative subdomain, which is noted as $\neg\tilde{\mathbf{x}}^j_{c,k} = \Re^{n_c} - \tilde{\mathbf{x}}^j_{c,k}$ (complementary set of $\tilde{\mathbf{x}}^j_{c,k}$).

As $\mathbf{x}_{c,k}$ defines a box in $\Re^{n_c}$, the values of $g_i^j(\mathbf{x}_{c,k})$ are bounded intervals of the form $[\underline{g}_i^j(\mathbf{x}_{c,k}), \overline{g}_i^j(\mathbf{x}_{c,k})]$. Thus, $\tilde{\mathbf{x}}^j_{c,k}$ and $\neg\tilde{\mathbf{x}}^j_{c,k}$ are interval vectors of dimension n_c, the scalar bounds of which take value $\underline{g}_i^j(\mathbf{x}_{c,k})$, $\overline{g}_i^j(\mathbf{x}_{c,k})$, $-\infty$, or $+\infty$. Considering all the autonomous transitions, this formulation leads to splitting the continuous space into several overlapping subregions. The set of positive and negative subdomains for $\mathbf{x}_{c,k}$ for all the autonomous transitions is used to build what we refer to as the conditional domain of $\mathbf{x}_{c,k}$.

Definition 3 (Conditional Domain): Given a hybrid system H, the conditional domain of $\mathbf{x}_{c,k}$ at k is given by $\mathbb{X}_k = [\tilde{\mathbf{x}}^0_{c,k}, \tilde{\mathbf{x}}^1_{c,k}, \ldots, \tilde{\mathbf{x}}^{n_T}_{c,k}]$, where we have the following:

1) $\tilde{\mathbf{x}}^j_{c,k}$ is the positive subdomain for every transition τ^j, $j = 1, \ldots, n_T$ of H;
2) $\tilde{\mathbf{x}}^0_{c,k} = \bigcap_{j=1}^{n_T}(\neg\tilde{\mathbf{x}}^j_{c,k})$ is the region that satisfies no partial guard.

Example 1 (Continued): The model has two guards over four transitions. Guards depend on temperature $\mathbf{x}_c = x$ only. Then, $\mathbb{X}_k = [\tilde{x}^0_k, \tilde{x}^1_k, \tilde{x}^2_k, \tilde{x}^3_k, \tilde{x}^4_k]$ with $\tilde{x}^0_k =]x^{\min}, x^{\max}[$, $\tilde{x}^1_k = \tilde{x}^3_k =]-\infty, x^{\min}]$, and $\tilde{x}^2_k = \tilde{x}^4_k = [x^{\max}, +\infty[$.

Example 2: Consider a hybrid system H with x_m taking its value in domain $\{m_1, m_2, m_3\}$, $\mathbf{x}_c = [x_{c1}, x_{c2}]^T$, and $\mathcal{T} = \{\tau^1, \tau^2\}$ with $\tau^1 : m_1 \xrightarrow{\phi^1} m_2$, $\tau^2 : m_1 \xrightarrow{\phi^2} m_3$, and $\phi^1 = \phi_1^1 : \begin{cases} 1, & \text{if } x_{c1} \leq g_1^1(x_{c2}) \\ 0, & \text{otherwise} \end{cases}$, $\phi^2 = \phi_2^2 : \begin{cases} 1, & \text{if } x_{c2} \geq g_2^2(x_{c1}) \\ 0, & \text{otherwise} \end{cases}$.

Initially, H is in mode m_1. Fig. 2 shows the conditional domain for this generic 2-D example in two situations: when $\mathbf{x}_{c,k}$ is real valued and when $\mathbf{x}_{c,k}$ is a box. In both cases, the conditional domain is given by

$$\mathbb{X}_k = [\tilde{\mathbf{x}}^0_{c,k}, \tilde{\mathbf{x}}^1_{c,k}, \tilde{\mathbf{x}}^2_{c,k}] = \begin{bmatrix} \tilde{x}^0_{c1,k} & \tilde{x}^1_{c1,k} & \tilde{x}^2_{c1,k} \\ \tilde{x}^0_{c2,k} & \tilde{x}^1_{c2,k} & \tilde{x}^2_{c2,k} \end{bmatrix}$$
$$= \begin{bmatrix}]\overline{g}^1_{1,k}, +\infty[&]-\infty, \overline{g}^1_{1,k}] &]-\infty, +\infty[\\ ]-\infty, \underline{g}^2_{2,k}[&]-\infty, +\infty[& [\underline{g}^2_{2,k}, +\infty[\end{bmatrix}$$

where $\underline{g}^j_{i,k}$ abbreviates $\underline{g}_i^j(\mathbf{x}_{c,k})$. Note that when $\mathbf{x}_{c,k}$ is real valued, $\underline{g}^j_{i,k} = \overline{g}^j_{i,k}$.

$\mathbb{X}_k$ concretizes the split[3] of the continuous space defined by the autonomous transition guards at time step k. Note that $\mathbb{X}_k$ evolves and is reshaped according to the continuous state vector at each time step. Geometrically, the bounds of $\tilde{\mathbf{x}}^j_{c,k}$ define edges that split the continuous state space into overlapping volumes shaped by boxes. Later developments require the definition of the bounds of these boxes. The lower bound of $\mathbb{X}_k$ is written as $\underline{\mathbb{X}}_k$, and the upper bound is written as $\overline{\mathbb{X}}_k$.

Every combination of elements of $\mathbb{X}_k$ corresponds to a subregion of the continuous state space in which some transitions are enabled and some are not. These regions are in the form of bounded boxes that support the concept of *configuration* of the hybrid system H. A configuration corresponds to a possible situation of the hybrid system in terms of simultaneously enabled and nonenabled transitions. Due to the boxed shape of the regions, the set of all configurations is organized in a grid that evolves with time, which is dubbed as the *grid of configurations*.

Definition 4 (Configuration): A configuration $\mathcal{C}_k$ of the hybrid system H at time step k is defined as follows.

1) A *configuration region* $\mathbf{r}_{\mathcal{C}_k}$ that is a box in the continuous state space that confines a region that simultaneously enables a possibly empty subset of transitions of $\mathcal{T}$.
2) A *configuration function* $\delta_{\mathcal{C}_k}$ that is a Boolean function that tells whether there exist points of the continuous state $\mathbf{x}_{c,k}$ that belong to the configuration region or not.
3) A *configuration-enabling set* $\mathcal{T}^e_{\mathcal{C}_k}$ that indicates which transition(s) is (are) enabled in the configuration region.

A configuration $\mathcal{C}_k$ is hence defined by a tuple $(\mathbf{r}_{\mathcal{C}_k}, \delta_{\mathcal{C}_k}, \mathcal{T}^e_{\mathcal{C}_k})$.

Definition 5 (Configuration Region): At time step k and for continuous vector $\mathbf{x}_{c,k}$, consider for every $i = 1, \ldots, n_c$ a unit[4] vector $\boldsymbol{\beta}_i$ of size $n_T + 1$. $\{\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_{n_c}\}$ form a set of projection vectors that extract a combination of transition partial guards, one per continuous dimension. Then, a configuration region is the volume defined by

$$\mathbf{r}_{\mathcal{C}_k} = \left[\mathbb{X}_{k,[1,.]}\boldsymbol{\beta}_1, \ldots, \mathbb{X}_{k,[n_c,.]}\boldsymbol{\beta}_{n_c}\right]^T \tag{6}$$

where $\mathbb{X}_{k,[i,.]}$ yields the ith line of matrix $\mathbb{X}_k$.

Using bounds of the conditional domain, we write the configuration region's frontier as the lowermost and uppermost vertices of the region's hyperrectangle. They are given by

$$\bar{\mathbf{r}}_{C_k} = \left[\underline{\mathbb{X}}_{k,[1,.]}\boldsymbol{\beta}_1, \ldots, \underline{\mathbb{X}}_{k,[n_c,.]}, \boldsymbol{\beta}_{n_c}\right]^T \cup \left[\overline{\mathbb{X}}_{k,[1,.]}\boldsymbol{\beta}_1, \ldots, \overline{\mathbb{X}}_{k,[n_c,.]}\boldsymbol{\beta}_{n_c}\right]^T. \tag{7}$$

Different configuration regions may overlap. A consequence is that some configurations may be subsumed by some set of other configurations and then be left aside. In example 2, any region obtained with $\boldsymbol{\beta}_1 = \begin{bmatrix}0\\0\\1\end{bmatrix}$ and/or $\boldsymbol{\beta}_2 = \begin{bmatrix}0\\1\\0\end{bmatrix}$ is subsumed by regions obtained with other vectors. By extension, we say that a configuration $\mathcal{C}_i$ is subsumed by a configuration $\mathcal{C}_j$ when the enabling set of $\mathcal{C}_i$ is also enabled by $\mathcal{C}_j$, i.e., $\mathcal{T}^e_{\mathcal{C}_i} \subseteq \mathcal{T}^e_{\mathcal{C}_j}$, and the configuration region of the second is included in that of the first, i.e., $\mathbf{r}_{C_j} \subset \mathbf{r}_{C_i}$. However, mostly, this is a byproduct of the formulation. In practice, such configurations are easily avoided (see Section III-C).

Definition 6 (Configuration Function): At time step k and for continuous vector $\mathbf{x}_{c,k}$, the configuration function $\delta_{\mathcal{C}_k}$ of the hybrid system H is a Boolean function from $\mathbf{x}_{c,k} \to \{0,1\}$ given by

$$\delta_{\mathcal{C}_k} = \begin{cases} 1, & \text{if } \mathbf{r}_{\mathcal{C}_k} \cap \mathbf{x}_{c,k} \neq \emptyset \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

When $\delta_{\mathcal{C}_k} = 1$, the configuration region $\mathbf{r}_{\mathcal{C}_k}$ (and by extension, the configuration $\mathcal{C}_k$ itself) is said to be *enabled*. Checking $\mathbf{x}_{c,k}$ against the configuration regions of the grid, hence, allows one to determine which transition(s) are enabled at time step k.

Definition 7 (Configuration-Enabling Set): The configuration-enabling set $\mathcal{T}^e_{\mathcal{C}_k}$ is the set of transitions τ^j whose guards are such that $\phi^j(\mathbf{r}_{\mathcal{C}_k} \cap \mathbf{x}_{c,k}) = 1$. It is empty whenever $\delta_{\mathcal{C}_k} = 0$.

Example 2 (Continued): Assume that $\mathbf{x}_{c,k}$ is a box [see Fig. 2(c)]. This example has four not subsumed configurations $\mathcal{C}^{(p)}_k$, $p = 1, \ldots, 4$. They are defined by the following.

1) Configuration regions: $\mathbf{r}_{\mathcal{C}^{(1)}_k} = [\mathbb{X}_{k,[1,.]}\boldsymbol{\beta}_1, \mathbb{X}_{k,[2,.]}\boldsymbol{\beta}_2]^T = (\tilde{\mathbf{x}}^0_{c,k})^T$ obtained with $\boldsymbol{\beta}_1 = \boldsymbol{\beta}_2 = \begin{bmatrix}1\\0\\0\end{bmatrix}$; $\mathbf{r}_{\mathcal{C}^{(2)}_k} = (] -\infty, \overline{g}^1_{1,k}],]-\infty, \underline{g}^2_{2,k}[)^T$ obtained with $\boldsymbol{\beta}_1 = \begin{bmatrix}0\\1\\0\end{bmatrix}$ and $\boldsymbol{\beta}_2 = \begin{bmatrix}1\\0\\0\end{bmatrix}$; $\mathbf{r}_{\mathcal{C}^{(3)}_k} = (]\overline{g}^1_{1,k}, +\infty[, [\underline{g}^2_{2,k}, +\infty[)^T$ obtained with $\boldsymbol{\beta}_1 = \begin{bmatrix}1\\0\\0\end{bmatrix}$ and $\boldsymbol{\beta}_2 = \begin{bmatrix}0\\0\\1\end{bmatrix}$; and $\mathbf{r}_{\mathcal{C}^{(4)}_k} = (] -\infty, \overline{g}^1_{1,k}], [\underline{g}^2_{2,k}, +\infty[)^T$ obtained with $\boldsymbol{\beta}_1 = \begin{bmatrix}0\\1\\0\end{bmatrix}$ and $\boldsymbol{\beta}_2 = \begin{bmatrix}0\\0\\1\end{bmatrix}$.
2) Configuration functions $\delta_{\mathcal{C}^{(p)}_k}$, $p = 1, \ldots, 4$ with $\delta_{\mathcal{C}^{(1)}_k} = \delta_{\mathcal{C}^{(2)}_k} = 0$ and $\delta_{\mathcal{C}^{(3)}_k} = \delta_{\mathcal{C}^{(4)}_k} = 1$.
3) Configuration-enabling sets: $\mathcal{T}^e_{\mathcal{C}^{(1)}_k} = \mathcal{T}^e_{\mathcal{C}^{(2)}_k} = \emptyset$; $\mathcal{T}^e_{\mathcal{C}^{(3)}_k} = \{\tau^2\}$; $\mathcal{T}^e_{\mathcal{C}^{(4)}_k} = \{\tau^1, \tau^2\}$: the situation is nondeterministic since τ^1 and τ^2 are simultaneously enabled.

C. Condition Variables

Configurations relate subregions of the continuous space to the enabling of transitions, which are discrete events. Thus, configurations are a natural articulation between the continuous and discrete dynamics. However, at this stage of the formulation, configurations have not yet been directly related to the modes.

The difficulty is that one mode may be consistent with several configurations of the hybrid system. Thus, in the thermostat example, the *on* mode is consistent with both $x \in]-\infty, x^{\min}]$ and $x \in]x^{\min}, x^{\max}[$. The opposite is also true since one configuration may be consistent with several modes. In the same example, modes *on* and *off* are both consistent with

[3] We enforce the term "split" over the term "partition" to acknowledge the possibly overlapping regions of the conditional domain.

[4] Here, a vector in which a single element is 1 and all the others are 0.

$x \in]x^{\min}, x^{\max}[$. In the following, we show how to relate the configurations to the modes. The final aim is to give a formal basis for the estimation algorithm to circumvent the full enumeration of all possible combinations of modes and configurations. What is sought is thus an articulation of the configurations with the modes.

The solution comes quite naturally. The idea is to reflect the enabled configurations at the discrete level. The enabled configurations can be expressed within the discrete state through a set of projection unit vectors: the $\boldsymbol{\beta}_i$ that define the configuration regions (6). However, relation (8) considers only those regions that intersect $\mathbf{x}_{c,k}$. The solution becomes finding the subset of vectors $\boldsymbol{\beta}_i$ that define those configuration regions that satisfy (8) and including them into the discrete representation of the state.

To differentiate them from other vectors, these unit solution vectors are noted as $\boldsymbol{\kappa}_d^i = [\kappa_{d0}^i, \kappa_{d1}^i, \ldots, \kappa_{dn_T}^i]^T$ for every $i = 1, \ldots, n_c$. Every κ_{dj}^i has domain $\{0, 1\}$, and we refer to it as a *conditional variable* since it refers to which portion of the conditional domain does enable a configuration. $\boldsymbol{\kappa}_d^i$ is dubbed as a *conditional vector*.

Definition 8 (Conditional Vectors): Given H and its continuous state $\mathbf{x}_{c,k}$, the conditional vectors $\boldsymbol{\kappa}_d^1, \ldots, \boldsymbol{\kappa}_d^{n_c}$ are unit vectors such that $[\mathbb{X}_{k,[1,.]}\boldsymbol{\kappa}_d^1, \ldots, \mathbb{X}_{k,[n_c,.]}\boldsymbol{\kappa}_d^{n_c}]^T \cap \mathbf{x}_{c,k} \neq \emptyset$, $i = 1, \ldots, n_c$.

Given $\mathbf{x}_{c,k}$ as a box, there exist many different combinations of conditional vectors. Every combination extracts an enabled configuration from $\mathbb{X}_k$. Finally, we permit additional constraints among $\boldsymbol{\kappa}_d^i$ and other discrete variables of X_d to be specified in Q. This allows discrete variables other than modes to depend on the continuous state values. Additionally, configurations that are subsumed can be avoided. These configurations arise from conditional vectors that extract dimensions that are unconstrained by the condition functions of some transitions. Constraining the Boolean values of the associated condition variables eliminates these solution vectors. See the example below.

Example 2 (Continued): Consider $\mathbb{X}_k = [\tilde{\mathbf{x}}_{c,k}^0, \tilde{\mathbf{x}}_{c,k}^1, \tilde{\mathbf{x}}_{c,k}^2]$ defined earlier. $\tilde{x}_{c1,k}^2 = \tilde{x}_{c2,k}^1 =]-\infty, +\infty[$. Thus, any configuration region obtained with solution vectors $\boldsymbol{\kappa}_d^1 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ and/or $\boldsymbol{\kappa}_d^2 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ is subsumed. The constraints to exclude these solution vectors are in Q.

Example 1 (Continued): Given $\mathbf{x}_c = x$ and hence $\mathbb{X}_k = [\tilde{x}_k^0, \tilde{x}_k^1, \tilde{x}_k^2, \tilde{x}_k^3, \tilde{x}_k^4]$ defined earlier, the thermostat system uses one vector $\boldsymbol{\kappa}_d = [\kappa_{d0}, \kappa_{d1}, \kappa_{d2}, \kappa_{d3}, \kappa_{d4}]^T$. Assuming that $]x^{\min}, x^{\max}[\subseteq x_k$, then $\boldsymbol{\kappa}_d = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$, and $\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$ are the five conditional solution vectors such that $\mathbb{X}_k \boldsymbol{\kappa}_d \cap x_k \neq \emptyset$.

Conditional variables pave the way for the definition of a *logical configuration* that articulates the continuous and discrete states and dynamics.

D. Logical Configuration

What is referred to as a *logical configuration* is simply the expression of a configuration at the discrete level. The useful feature is that logical configurations directly relate to hybrid system modes.

Definition 9 (Logical Configuration): Given a hybrid system H and its continuous state $\mathbf{x}_{c,k}$ at time step k, a logical configuration of H is noted as the logical conjunction

$$\nabla \delta_k = x_m \wedge \left[\bigwedge_{i=1}^{n_c} \left[\bigwedge_{j=0}^{n_T} \left(\kappa_{dj}^i = \xi_j \right) \right] \right]$$

where

$$\xi_j = \begin{cases} 1, & \text{if } \kappa_d^i \text{ is the } j\text{th unit vector} \\ 0, & \text{otherwise.} \end{cases}$$

Example 2 (Continued): In Fig. 2(c), the system is in mode m_1. We have seen that $\mathcal{C}_k^{(3)}$ and $\mathcal{C}_k^{(4)}$ are enabled. The conditional vectors of interest are thus $\boldsymbol{\kappa}_d^1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$ and $\boldsymbol{\kappa}_d^2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$; $\boldsymbol{\kappa}_d^1 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$ and $\boldsymbol{\kappa}_d^2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$, respectively. This leads to two logical configurations $\nabla \delta_k^{(3)}$ and $\nabla \delta_k^{(4)}$ of the form $\nabla \delta_k^{(p)} = (x_m = m_1) \wedge [\bigwedge_{i=1}^2 [\bigwedge_{j=0}^2 (\kappa_{dj}^i = \xi_j)]]$.

IV. Hybrid State Estimation

Given a set of commands and observations at every time step, the set-theoretic estimation of hybrid states consists of predicting a set of hybrid state candidates and rejecting those that do not predict the observations. In consequence, most operations are concerned with prediction. The problem of prediction is its cost, since many predicted states may end up being rejected. It is thus essential to eliminate impossible candidates as early as possible. Prediction consists of a loop at each sampled time step: continuous prediction, discrete state prediction, and continuous state transfer, until there are no more enabled changes in the discrete dynamics. It follows that early elimination of state candidates is possible at every loop step. While continuous state elimination simply requires an inclusion test of the observations, discrete state elimination requires a full consistency check that is more demanding. However, this task has connections with a set of techniques referred to as the consistency-based approach to diagnosis [39]. These techniques use the constraints in the models to limit the state candidates to be considered [23], [40]. They can prune out candidates at each step that standard filters would keep in their set of estimates. In consequence, our algorithms rely on these techniques to manage discrete state consistency. To further mitigate the number of candidates, our estimation scheme shows how the modeling of uncertainty in a bounded form allows us to merge estimates with identical discrete state. This proves to be a decisive advantage of state estimation based on uncertain but bounded models over state estimation based on stochastic models. In addition, our estimator includes a procedure that estimates several fast successive switches in discrete dynamics in-between two sampled time steps. Here, again, a bounded uncertainty is key to allowing this feature.

A. Hybrid State Prediction in Sampled Time

1) Forward Time Prediction: A prediction of the hybrid state is obtained with a forward predictive operator [41].

Definition 10 (Forward Time Prediction): The forward time prediction $\langle S_{l,k-1}\rangle_\gamma^{\nearrow}$ of a set $S_{l,k-1}$ of hybrid states at logical time step l and sampled time step $k-1$ is the set of hybrid states that are reachable from $S_{l,k-1}$ by letting the sampled time progress over γ sampled steps. For a single hybrid state, $\mathbf{s}_{l,k-1} = (\boldsymbol{\pi}_l, \mathbf{x}_{c,k-1})$, $\boldsymbol{\pi}_l = (x_{m,l}, \mathbf{x}_{d,l})$

$$\langle \mathbf{s}_{l,k-1}\rangle_1^{\nearrow} = \big\{\mathbf{s}_{l,k} = (\boldsymbol{\pi}_l, \mathbf{x}_{c,k}) | \mathbf{x}_{c,k} = f(\mathbf{x}_{c,k-1}, \mathbf{u}_{c,k-1}, \mathbf{w}_{c,k-1}, x_{m,l})\big\} \quad (9)$$

and $\langle S_{l,k-1}\rangle_\gamma^{\nearrow}$ is the repetition of $\langle S_{l,k-1}\rangle_1^{\nearrow}$, γ times, over all $\mathbf{s}_{l,k-1} \in S_{l,k-1}$.

There are many ways for relation (9) to be efficiently computed. The difficulty is that the box $\mathbf{x}_{c,k}$ keeps growing with the number of steps γ. This is because the rectangular approximation at each step introduces an error that is reapproximated by successive steps and thus rapidly amplified. This phenomenon is known as the *wrapping effect*. In general, convex optimization techniques help mitigate this explosion of uncertainty. In the current implementation, interval numerical methods similar to those in [36] are used.

While the mechanics of transition triggering are described later, here, it is enough to mention that two cases arise: 1) whenever no transition is enabled by the forward time prediction, then the observations $\tilde{\mathbf{y}}_{c,k}$ can be used to prune impossible candidates; and 2) when a transition is enabled, observations cannot be used immediately since they may have been produced by a behavior that is different from that of the current mode and model. Case 1 corresponds to applying set-theoretic filtering techniques to forward time prediction. Linear and nonlinear filters have been described [30]–[33]. In the case of nonlinear systems, the produced bounded estimates can be approximated by a variety of geometrical shapes, ellipsoids [33], rectangles [30], [32], and polytopes [42]. These filters can be utilized to control the quality of the forward time prediction. In the following, it is assumed that the produced shapes are rectangular boxes, but the approach can be extended to other shapes as well.[5]

2) Forward Transition Prediction: A prediction of the discrete switches is obtained with a second forward predictive operator.

[5]With the limitation that intersection with the grid of configurations may not conserve certain shapes.

Definition 11 (Forward Transition Prediction): Given transition τ and a set of hybrid states $S_{l,k}$, the *forward transition prediction* $\langle S_{l,k}\rangle^\tau$ is the set of hybrid states that are reachable from some state $\mathbf{s}_{l,k} \in S_{l,k}$ by executing a transition τ. For $\mathbf{s}_{l,k} = (\boldsymbol{\pi}_l, \mathbf{x}_{c,k})$, with $\boldsymbol{\pi}_l = (x_{m,l}, \mathbf{x}_{d,l})$, if τ is enabled, then

$$\langle \mathbf{s}_{l,k}\rangle^\tau = \big\{\mathbf{s}_{l+1,k} = \left(\boldsymbol{\pi}_{l+1}, \mathbf{x}'_{c,k}\right) | x_{m,l} \xrightarrow{\tau} x_{m,l+1} \text{ and } \mathbf{x}'_{c,k} = l_\tau(\mathbf{x}_{c,k})\big\} \quad (10)$$

where $\boldsymbol{\pi}_{l+1} = (x_{m,l+1}, \mathbf{x}_{d,l+1})$ such that $Q \cup \boldsymbol{\pi}_{l+1}$ is consistent.

By consistent, we mean that $x_{m,l+1}$ and $\mathbf{x}_{d,l+1}$ together satisfy all the formulas in Q.

3) Hybrid State Prediction: The hybrid system prediction over time alternates both forward operators. As seen earlier, multiple transitions can simultaneously be enabled. This is due to the fact that the box $\mathbf{x}_{c,k}$ can span over several configuration regions. A consequence is that different points of $\mathbf{x}_{c,k}$ happen to enable and trigger different transitions, thus leading the system from its current state to different modes and states. Given a forward time prediction, the aim of the estimation process is to transfer each point of the continuous state at date (l,k) to the possibly multiple mode(s) it belongs to at date $(l+1,k)$. The solution is to produce a split of $\mathbf{x}_{c,k}$ such that the produced fragments fit the grid of configurations. The enabled transitions can then trigger from such state fragments, and the forward transition prediction yields the new set of modes of the system along with the set of continuous estimates.

B. Hybrid Consistency Problems

Given a set of hybrid states $S_{l,k-1}$ at date $(l,k-1)$ and the forward time prediction $S_{l,k} = \langle S_{l,k-1}\rangle_1^{\nearrow}$, the problem of intersecting $\mathbf{x}_{c,k}$ with the grid of configurations comes to the finding of a split $P_{l,k} = \{\mathbf{s}_{l,k}^{(1)}, \ldots, \mathbf{s}_{l,k}^{(n_p)}\}$ such that for every $p = 1, \ldots, n_p$, $\mathcal{C}_k^{(p)}$ is a configuration, with $\mathbf{x}_{c,k}^{(p)} \subseteq \mathbf{r}_{\mathcal{C}_p}$, and $\boldsymbol{\pi}_l^{(p)} \cup Q \cup \nabla\delta_k^{(p)}$ is consistent. This is done in two steps. Given a predicted hybrid state $\mathbf{s}_{l,k}$, $\mathbf{x}_{c,k}$ is used to find $\mathbb{X}_k$ and the conditional vectors $\boldsymbol{\kappa}_d^i$. Those vectors yield the logical configurations $\nabla\delta_k^{(p)}$ that are consistent with $\mathbf{s}_{l,k}$. An initial set of conditional vectors is easily obtained by iterating the continuous dimensions and checking whether $\mathbb{X}_k$ intersects $\mathbf{x}_{c,k}$. Further checking against Q yields the reduced set of logical configurations that are possible under the set of qualitative constraints. Impossible configurations are eliminated. The second step takes the remaining logical configurations and computes the configuration regions out of the predicted $\mathbf{x}_{c,k}$. Recall that every configuration region is shaped by a system of inequalities over the condition functions g_i^j in (5). These inequalities form a constraint network among continuous variables. Therefore, the change of one variable-bounded value often affects the range of other variables. By iterating a constraint filtering process over all continuous variables, the focus narrows down onto the only possible continuous states. The double logical/continuous formulation of configurations from Section III is key as it permits the pruning of impossible estimates at both levels.

Basically, the first pruning step takes place at a discrete level, and the second pruning step takes place at the continuous level. Information is passed through the logical configurations.

1) Discrete State Consistency: Given a hybrid system H and a prediction $\mathbf{s}_{l,k} = (\boldsymbol{\pi}_l, \mathbf{x}_{c,k})$, then $\{(\boldsymbol{\pi}_l^{(p)}, \nabla\delta_k^{(p)})\}$, $p = 1, \ldots, n_p$, are such that we have the following:

1) They are consistent with $\mathcal{Q}$

$$\boldsymbol{\pi}_l^{(p)} \cup Q \cup \nabla\delta_k^{(p)} \text{ is consistent.} \tag{11}$$

2) $\boldsymbol{\pi}_l^{(p)} = (x_{m,l}, \mathbf{x}_{d,l}^{(p)})$, so that the mode estimate $x_{m,l}$ is that of $\mathbf{s}_{l,k}$, since no transition has triggered yet.

The conditional vectors $\boldsymbol{\kappa}_d^i$ determine a set of logical configurations. A subset of those is selected by solving relation (11). This can be done with a constraint satisfaction engine. Solutions to (11) are logical configurations along with discrete state estimates $\boldsymbol{\pi}_l^{(p)}$. This operation is noted as $\{(\boldsymbol{\pi}_l^{(p)}, \nabla\delta_k^{(p)})\}_{p=1,\ldots,n_p} = sat(\mathbf{s}_{l,k}, Q)$, where sat denotes the constraint satisfaction engine. In the present implementation, the Boolean satisfaction engine described in [43] is used. A wide range of other techniques is applicable.

2) Continuous State Consistency: Given a configuration $\mathcal{C}_k^{(p)}$ at the continuous level, the subregion $\mathbf{x}_{c,k}^{(p)}$ of $\mathbf{x}_{c,k}$ that is consistent with the configuration region is given by

$$\mathbf{x}_{c,k}^{(p)} = \mathbf{x}_{c,k} \cap \mathbf{r}_{\mathcal{C}_k^{(p)}}. \tag{12}$$

Computing $\mathbf{x}_{c,k}^{(p)}$ is more difficult than it seems. Recall that $\mathbf{r}_{\mathcal{C}_k^{(p)}}$ is equal to $[\mathbb{X}_{k,[1,.]}\boldsymbol{\kappa}_d^1, \ldots, \mathbb{X}_{k,[n_c,.]}\boldsymbol{\kappa}_d^{n_c}]^T$, where the $\boldsymbol{\kappa}_d^i$'s are given by $\nabla\delta_k^{(p)}$. Every unit vector $\boldsymbol{\kappa}_d^i$ extracts a positive or negative subdomain from $\mathbb{X}_k$. Every subdomain is obtained by evaluating a condition function g_i^j, where j is given by the entry equal to 1 of unit vector $\boldsymbol{\kappa}_d^i$. To satisfy (12), the points of the box $\mathbf{x}_{c,k}^{(p)}$ must satisfy all the condition functions that determine $\mathbf{r}_{\mathcal{C}_k^{(p)}}$.

However, a variable x_{ci} can be coupled with some other variables $x_{ci'}$ through g_i^j. This means that tightening the bounds of x_{ci} has an effect on $x_{ci'}$'s bounds. This problem can be seen as the task of filtering a set of bounded variables x_{ci} with a set of inequalities over those same variables. Such a problem can be solved with a slightly revised version of standard filtering or branch-and-bound techniques. Indeed, in general, these techniques do not handle inequalities but only equality constraints [44]. The algorithmic solution in Table II is a variant of the constraint propagation system in [44] that handles inequalities. Prior to detailing the algorithm, admissibility and consistency are to be distinguished.

1) A condition function g_i^j is said to be admissible for $\mathbf{x}_{c,k}$ iff there exists at least a point of $\mathbf{x}_{c,k}$ such that the inequality based on $g_i^j(\mathbf{x}_{c,k})$ is satisfied.
2) $\mathbf{x}_{c,k}$ is said to be consistent with $g_i^j(\mathbf{x}_{c,k})$ when the inequality based on g_i^j is satisfied for all points in $\mathbf{x}_{c,k}$.

The algorithm in Table II finds $\mathbf{x}_{c,k}^{(p)}$ such that it is consistent with all of the condition functions g_i^j that determine $\mathbf{r}_{\mathcal{C}_k^{(p)}}$. The algorithm constrains all the variables that appear in the condition functions g_i^j drawn from an input logical configuration $\nabla\delta_k^{(p)}$. It does so until each condition function is either satisfied or inadmissible. The operator described by the algorithm is dubbed $filter(\nabla\delta_k^{(p)}, \mathbf{x}_{c,k})$. Its result is a continuous state fragment $\mathbf{x}_{c,k}^{(p)}$.

TABLE II
FINDING CONSISTENT CONTINUOUS STATES: $filter(\nabla\delta_k^{(p)}, \mathbf{x}_{c,k})$

Require: $\nabla\delta_k^{(p)}$, $\mathbf{x}_{c,k}$.
1: Agenda = $\{g_i^j \mid \boldsymbol{\kappa}_d^i$ is the j-th unit vector, $i = 1, \cdots, n_c\}$.
2: $\mathbf{x}_{c,k}^{(p)} = \mathbf{x}_{c,k}$.
3: **while** Agenda not empty **do**
4: Select a g_i^j in Agenda.
5: Recompute positive subdomain $\tilde{\mathbf{x}}_{c,k}^j$, i.e. find the $x'_{ci,k} \leqslant g_i^j(x_{c1,k}^{(p)}, .., x_{ci-1,k}^{(p)}, x_{ci+1,k}^{(p)}, .., x_{cn,k}^{(p)})$ such that $\phi_i^j(\mathbf{x}_{c,k}^{(p)}) = 1$. When $j = 0$, the negative subdomain is recomputed instead.
6: $Int = x'_{ci,k} \cap x_{ci,k}^{(p)}$.
7: **if** $Int = \emptyset$ **then**
8: g_i^j is inadmissible for $\mathbf{x}_{c,k}^{(p)}$.
9: Reject $\nabla\delta_k^{(p)}$.
10: **return** $\emptyset$.
11: **if** $x_{ci,k}^{(p)} \subseteq x'_{ci,k}$ **then**
12: Remove g_i^j from Agenda.
13: **else**
14: Add $\{g_{i'}^j \mid \boldsymbol{\kappa}_d^{i'}$ is the j-th unit vector, $i' \neq i\}$ to the Agenda.
15: $\mathbf{x}_{c,k}^{(p)} \leftarrow Int$.
16: **return** $\mathbf{x}_{c,k}^{(p)}$.

TABLE III
SPLITTING THE CONTINUOUS SPACE: $split(\mathbf{s}_{l,k})$

Require: $\mathbf{s}_{l,k}$.
1: $P_{l,k} = \{\}$.
2: Find the combinations of $\boldsymbol{\kappa}_d^i$ for all $i = 1, \cdots, n_c$ that define the $\nabla\delta_k^{(q)}$, $q = 1, \cdots, n_q$.
3: Compute $\{(\boldsymbol{\pi}_l^{(p)}, \nabla\delta_k^{(p)})\}_{p=1,\cdots,n_p} = sat(\mathbf{s}_{l,k}, Q)$, $n_p \leq n_q$.
4: **for all** $\nabla\delta_k^{(p)}$ **do**
5: $\mathbf{x}_{c,k}^{(p)} = filter(\nabla\delta_k^{(p)}, \mathbf{x}_{c,k})$.
6: **if** $\mathbf{x}_{c,k}^{(p)} \neq \emptyset$ **then**
7: $\mathbf{s}_{l,k}^{(p)} = (\boldsymbol{\pi}_l^{(p)}, \mathbf{x}_{c,k}^{(p)})$.
8: $P_{l,k} \leftarrow P_{l,k} \cup \mathbf{s}_{l,k}^{(p)}$.
9: **return** $P_{l,k}$.

C. Splitting the Hybrid State With Configurations

The operator that articulates sat and $filter$, i.e., the discrete and continuous consistency operators, respectively, is referred to as $split$. $split$ applies to a set $S_{l,k}$ of hybrid states and returns another set $P_{l,k}$ (see Table III). The algorithm takes a predicted hybrid state $\mathbf{s}_{l,k}$ as input.

Example 2 (Continued): Starting from the configurations obtained in Fig. 2(c), Fig. 3(a) and (b) shows the split of the continuous space for enabled configurations $\mathcal{C}_k^{(3)}$ and $\mathcal{C}_k^{(4)}$, respectively. The logical configurations are $\nabla\delta_k^{(3)}$ and $\nabla\delta_k^{(4)}$ defined earlier. The $filter$ operator applied to each configuration reduces $\mathbf{x}_{c,k}$ by using partial guard g_1^1 (step 5, Table II). In both cases, evaluating $g_2^2(\mathbf{x}_{c,k})$ does not further reduce $\mathbf{x}_{c,k}$. The results are then $\mathbf{x}_{c,k}^{(3)}$ and $\mathbf{x}_{c,k}^{(4)}$.

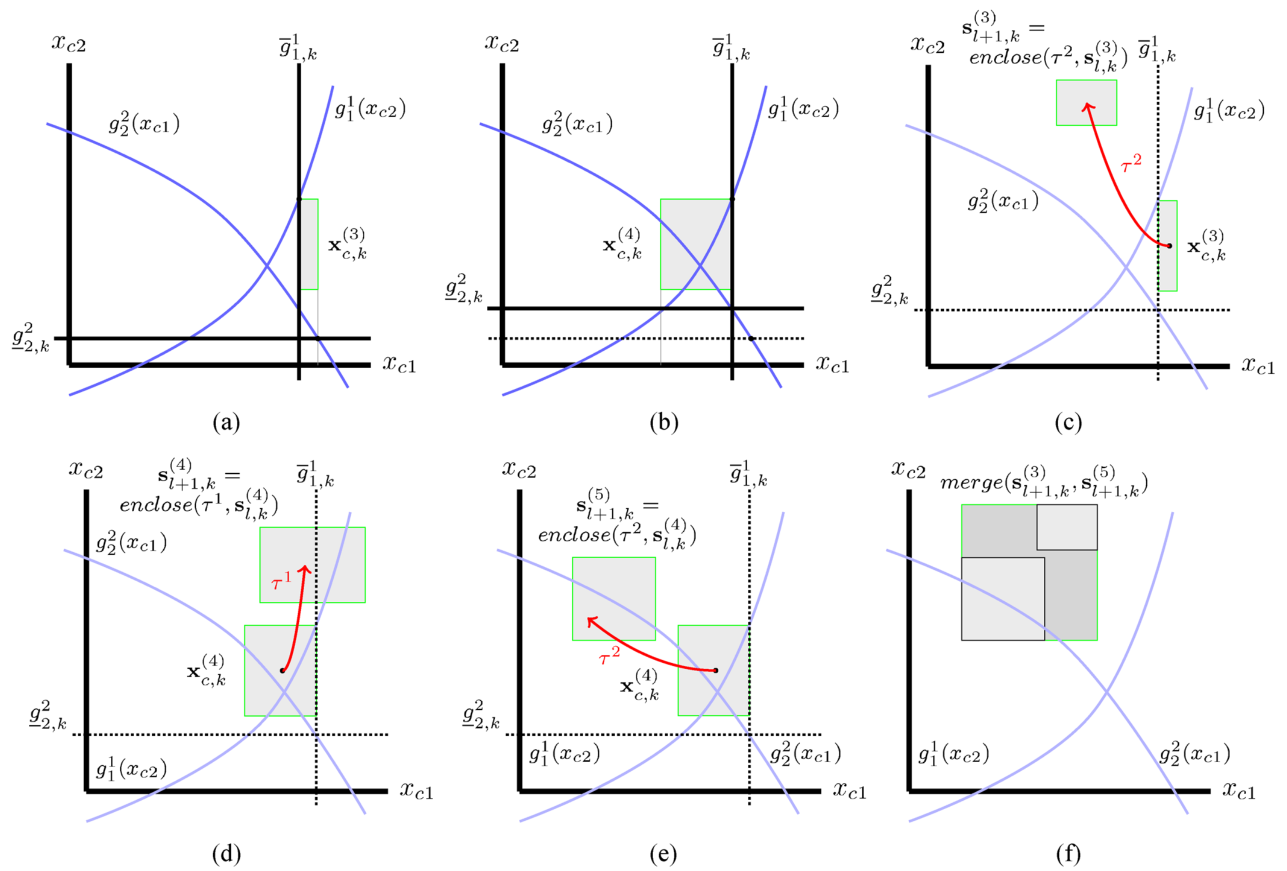

Fig. 3. Example 2. (a) and (b) Continuous state split. On (b), note that the configuration domain has changed: the split with $\overline{g}^1_{1,k}$ affects the value of $\underline{g}^2_{2,k}$. The dotted line shows the previous boundary. (c), (d), and (e) Enclosure and switches according to $\mathcal{T}^e_{\mathcal{C}^{(3)}_k} = \{\tau^2\}$ and $\mathcal{T}^e_{\mathcal{C}^{(4)}_k} = \{\tau^1, \tau^2\}$. Only the late switch is represented. $\mathbf{s}^{(3)}_{l+1,k} = enclose(\tau^2, \mathbf{s}^{(3)}_{l,k})$, $\mathbf{s}^{(4)}_{l+1,k} = enclose(\tau^1, \mathbf{s}^{(4)}_{l,k})$, and $\mathbf{s}^{(5)}_{l+1,k} = enclose(\tau^2, \mathbf{s}^{(4)}_{l,k})$. (f) Merging step. The estimates obtained in (c) and (e) have identical mode m_3. In consequence, their continuous estimates can be merged. (a) Split with configuration $\mathcal{C}^{(3)}_k$. (b) Split with configuration $\mathcal{C}^{(4)}_k$. (c) Triggering of τ^2 from $\mathcal{C}^{(3)}_k$. (d) Triggering of τ^1 from $\mathcal{C}^{(4)}_k$. (e) Triggering of τ^2 from $\mathcal{C}^{(4)}_k$. (f) Merging of (c) and (e).

In all cases, remark that the union of the continuous state fragments yields the originally predicted state. That is, the $\mathbf{x}^{(p)}_{c,k}$, $p = 1, \ldots, n_p$, that result from the split of a state $\mathbf{x}_{c,k}$ are such that $\bigcup_{p=1}^{n_p} \mathbf{x}^{(p)}_{c,k} = \mathbf{x}_{c,k}$. Formally, this is because the conditional domain $\mathbb{X}_k$ of $\mathbf{x}_{c,k}$ contains the positive and negative subdomains $\tilde{\mathbf{x}}^j_{c,k}$ and $\neg\tilde{\mathbf{x}}^j_{c,k}$ for all transitions τ^j. Therefore, the entire continuous state space is covered by configuration regions, and both $\mathbf{x}_{c,k} \subseteq \bigcup_{p=1}^{n_p} \mathbf{r}_{\mathcal{C}^{(p)}_k}$ and $\bigcup_{p=1}^{n_p} \mathbf{x}^{(p)}_{c,k} = \bigcup_{p=1}^{n_p} (\mathbf{r}_{\mathcal{C}^{(p)}_k}) \cap \mathbf{x}_{c,k}$ [from relation (12)] hold.

However, the hybrid states produced by a split are rarely optimal: some hybrid states are, in fact, not reachable by the system. This is due to a lack of constraints between modes and conditional variables in logical configuration equations. In example 1, hybrid state $s_k = (x_m = on \wedge x_k)$ with $x_k \geq x^{\max}$ is unreachable but predicted at some point: the thermostat cannot be turned on, and the temperature can be over the upper threshold $x^{\max}$. The problem is complex, as these configurations represent the so-called *mythical states* [28], [45]–[47], i.e., *instantaneous* states between normal states when a discontinuous change takes place. In a mythical state, the variables do not satisfy all of the system constraints. This happens to be the case of the state s_k above, since a transition to the mode *off* is enabled but has not triggered yet. The problem is that it is not clear whether these states represent very short but real instances, or whether they are artifacts of the representation and reasoning procedures. For this reason, the *split* operator is said to be complete but unsound.

D. Switching in Sampled Time

When the split is completed, some of the configuration-enabling sets are not empty. The two final steps of the estimation process are thus the triggering of the enabled transitions and the use of available observations. The triggering of transitions at a sampled time raises the following two problems.

1) A transition triggering is always considered a small period of time after the real switch has occurred. A consequence is that $\mathbf{x}_{c,k}$ computed at time step k is not guaranteed to capture the real behavior of the system.
2) Multiple successive switches may occur during a single sampled time interval.

TABLE IV
APPLIES TRANSITION τ ENABLED AT TIME STEP (l, k) : $enclose(\tau, \mathbf{s}_{l,k}^{(p)})$

Require: $\mathbf{s}_{l,k}^{(p)} = (\boldsymbol{\pi}_l^{(p)}, \mathbf{x}_{c,k}^{(p)})$, enabled transition τ.
1: Late switch: $\mathbf{s}'_{l+1,k} = (\boldsymbol{\pi}_{l+1}^{(p)}, \mathbf{x}'_{c,k}) = \langle \mathbf{s}_{l,k}^{(p)} \rangle^{\tau}$.
2: Early switch: $\mathbf{s}^*_{l+1,k-1} = (\boldsymbol{\pi}_{l+1}^{(p)}, \mathbf{x}^*_{c,k-1})$ with $\mathbf{x}^*_{c,k-1} = l_\tau(\bar{\mathbf{x}}_{c,k-1})$ and $\bar{\mathbf{x}}_{c,k-1} = \mathbf{x}_{c,k-1}^{(p)} \cup \bar{\mathbf{r}}_{\mathcal{C}_{k-1}^{(p)}}$.
3: Prediction after the early switch: $s^*_{l+1,k} = \langle s^*_{l+1,k-1} \rangle_1^{\nearrow}$.
4: update: $\mathbf{x}_{c,k}^{(p)} = [\min(\mathbf{x}^*_{c,k}, \mathbf{x}'_{c,k}), \max(\mathbf{x}^*_{c,k}, \mathbf{x}'_{c,k})]$.
5: **return** $\mathbf{s}_{l+1,k}^{(p)} = (\boldsymbol{\pi}_{l+1}^{(p)}, \mathbf{x}_{c,k}^{(p)})$.

1) Guaranteed Enclosure at Switching Points: The problem arises from the triggering of a transition in-between two sampled time steps. At time step $k-1$, no transition is enabled. Prediction produces a set of hybrid states at time step k. The $split$ operator applies and splits the continuous state according to candidate configurations. As a result, assume that some configurations are found to enable transitions at time step k, and consider an enabled transition τ. On the physical system, τ has triggered somewhere between sampled time steps $k-1$ and k. However, prediction proceeds by computing a late switch at time step k. Let $\mathbf{x}_{c,k'}$, $k-1 < k' \leq k$, be the continuous state at the unknown continuous time instant $k'T_s$ at which the transition has triggered on the physical system and where $k' \in \Re$. In general, $\mathbf{x}_{c,k'} \not\subseteq \mathbf{x}_{c,k}$, so switching at k misses the transfer of some continuous regions.

A solution is proposed to transfer the continuous state from one mode to another, which guarantees to capture the true behavior of the system. It computes an early switch at $k-1$, in addition to the late switch at k. Under the assumption that the continuous evolution of the system is monotonous between two sampled time steps, unionizing the continuous vectors obtained from both switches yields an enclosure of the true physical state of the system. In practice, due to high sampling rates, the assumption above is realistic and found in another body of works [36]. Table IV details the operator $enclose(\tau, \mathbf{s}_{l,k}^{(p)})$ that applies a transition τ to a state fragment $\mathbf{s}_{l,k}^{(p)}$ and transfers the continuous state from $\mathbf{s}_{l,k}^{(p)}$ to $\mathbf{s}_{l+1,k}^{(p)}$. The algorithm returns $\mathbf{s}_{l+1,k}^{(p)}$ that is guaranteed to capture the true state of the system under the assumption above. The sole subtle operation of the algorithm is step 2, which *virtually* enables a switch at time step $k-1$. This is required since τ cannot be enabled at time step $k-1$, as if it were, it would have triggered at that time step. Therefore, τ has to be virtually enabled at $k-1$. This is achieved by triggering τ from the union of $\mathbf{x}_{c,k-1}^{(p)}$ and the frontier $\bar{\mathbf{r}}_{\mathcal{C}_{k-1}^{(p)}}$ of the configuration region that enables τ. Note that the algorithm requires working on a temporal window of at least two sampled time steps, and that both $\mathbf{r}_{\mathcal{C}_{k-1}^{(p)}}$ and $\mathbf{x}_{c,k-1}^{(p)}$ must remain accessible in memory.

Example 2 (Continued): Fig. 3(c)–(e) shows the triggering of the enabled transitions τ^1 and τ^2. On these figures, only the late switch is represented. We have $\mathbf{s}_{l+1,k}^{(3)} = enclose(\tau^2, \mathbf{s}_{l,k}^{(3)})$, $\mathbf{s}_{l+1,k}^{(4)} = enclose(\tau^1, \mathbf{s}_{l,k}^{(4)})$, and $\mathbf{s}_{l+1,k}^{(5)} = enclose(\tau^2, \mathbf{s}_{l,k}^{(4)})$.

TABLE V
$switch(S_{l,k})$ OPERATOR

Require: $S_{l,k}$, $\xi = 0$.
1: **while** $\exists \tau$ enabled in $\mathbf{s}_{l,k}$ **do**
2: $\quad \mathbf{s}_{l+\xi+1,k} = enclose(\tau, \mathbf{s}_{l+\xi,k})$,
3: $\quad \xi \leftarrow \xi + 1$.
4: $\quad S_{l+\xi,k} = split(\mathbf{s}_{l+\xi,k})$.
5: $S_{l+\xi,k} = split(S_{l+\xi,k})$
6: $clear(S_{l+\xi,k}, \tilde{\mathbf{y}}_{d,k}, \tilde{\mathbf{y}}_{c,k})$.
7: $merge(S_{l+\xi,k})$.
8: **return** pruned $S_{l+\xi,k}$.

TABLE VI
$clear(S_{l,k})$ OPERATOR

Require: $S_{l,k}$, $\tilde{\mathbf{y}}_{c,k}$, $\tilde{\mathbf{y}}_{d,k}$.
1: **for all** $\mathbf{s}_{l,k} \in S_{l,k}$ **do**
2: $\quad$ **if** $\tilde{\mathbf{y}}_{c,k} \not\subseteq \mathbf{y}_{c,k}$ or $\tilde{\mathbf{y}}_{d,k} \wedge \mathbf{x}_{d,k}$ is inconsistent **then**
3: $\quad\quad$ Remove $\mathbf{s}_{l,k}$ from $S_{l,k}$.
4: **return** $S_{l,k}$.

2) Multiple Successive Switches: When more than one switch occurs between two sampled time steps, each switch is successively predicted with the $enclose$ operator. Operator $switch$ in Table V handles multiple switches in-between two sampled time steps. The number of successive switches is noted as ξ. At step 6, $clear$ is the operator that prunes out the states that do not intersect the observations (Table VI). At step 7, AQ1 the operator $merge$ optimizes the final partition by merging hybrid states whenever this is possible. These two operators are detailed in the two next paragraphs. A condition for $switch$ to terminate is that the hybrid system's behavior *excludes infinitely many switches occurring in-between two sampled time steps*. Whenever this condition is fulfilled, the algorithm in theory always terminates. In practice, however, the $enclose$ operator yields conservative bounds and adds up to the natural nonconvergence of numerical uncertainty. In consequence, the occurrence of infinitely many switches cannot be ruled out, but a theoretical analysis is beyond the scope of this paper.

3) Recursive Estimation: Finally, the estimation of the states of a hybrid system H is captured by a sequence $\rho : S_0, \ldots, S_{l,k}, \ldots$ that verifies

$$S_0 = switch\left(split\left(\langle \Theta \rangle_0^{\nearrow}\right)\right) \tag{13}$$

$$S_{l+\xi,k+\gamma} = switch\left(split\left(\langle S_{l,k} \rangle_\gamma^{\nearrow}\right)\right). \tag{14}$$

Relation (13) initializes the hybrid states starting from the system initial conditions Θ. The computation of the recursive relation (14) alternates forward time and transition predictions through *splits* and *switches*, and results in an updated set of hybrid states every $\gamma \neq 0$ sampled time steps.

E. Hybrid State Estimation

The $clear$ operator prunes out all the state estimates $\mathbf{s}_{l,k}$ such that the prediction $\mathbf{y}_{c,k}$ does not enclose the observations $\tilde{\mathbf{y}}_{c,k}$, or that do not predict observations $\tilde{\mathbf{y}}_{d,k}$. Note that the measurement noise is already taken into account in (3), so that $\tilde{\mathbf{y}}_{c,k}$ is a real-valued vector of $\Re^{n_y}$.

TABLE VII
$Merge(S_{l,k})$ OPERATOR

Require: $S_{l,k}$.
1: Group the $\mathbf{s}_{l,k} \in S_{l,k}$ into $\{S_{l,k}^{(1)}, \cdots, S_{l,k}^{(n_q)}\}$ such that all $\mathbf{s}_{l,k}^{(i,j)} \in S_{l,k}^{(i)}$ have the same discrete state estimate $\boldsymbol{\pi}_l^{(i)}$.
2: **for** $i = 1, \cdots, q$ **do**
3: $\mathbf{s}_{l,k}^{i,1} = (\boldsymbol{\pi}_l^i, \bigcup_{j=1}^{n_q} \mathbf{x}_{c,k}^{(i,j)})$.
4: For all $j > 1$, remove all $\mathbf{s}_{l,k}^{(i,j)}$ from $S_{l,k}$.
5: **return** $S_{l,k}$.

F. Merging Identical Discrete Estimates

Most approaches to the estimation of hybrid states apply Bayesian belief updates to a stochastic hybrid system. These techniques have to deal with an exponential blowup in the number of possible hybrid states. We can witness a similar effect in our case since the $switch$ operator generates a growing number of states at each time step. Adding up to the growing uncertainty that is due to the box approximation of the forward prediction operator, the growth rate of new state estimates rapidly increases. In general, this is the reason why modern estimators simultaneously track several hybrid state hypotheses. However, inevitably, the number of states exponentially grows with time as more hypotheses become likely.

The main advantage of our approach is that it permits the merging of similar trajectories without loss. Consider merging the uncertainty on two states $\mathbf{s}_{l,k}^{(1)}$ and $\mathbf{s}_{l,k}^{(2)}$: the question is how to merge the two continuous vector estimates $\mathbf{x}_{c,k}^{(1)}$ and $\mathbf{x}_{c,k}^{(2)}$. It is easily achieved by unionizing the variable estimated bounds. The sole condition for the merging is that the discrete states $\boldsymbol{\pi}_{l,k}^{(1)}$ and $\boldsymbol{\pi}_{l,k}^{(2)}$ are identical. The $Merge$ operator is given by Table VII. When splits and switches augment the number of hybrid state estimates at each time step, the merging step does reduce this number substantially. In general, this allows the estimation procedure to mitigate the explosion of modes and to maintain a finite, almost constant, number of hybrid estimates.

Example 2 (Continued): There are three estimates, which are represented in Fig. 3(c)–(e). In Fig. 3(c), τ^2 has transferred the system state to $\mathbf{s}_{l+1,k}^{(3)} = (m_3, \mathbf{x}_{c,k}^{(3)})$. In Fig. 3(d), τ^1 has transferred the system state to $\mathbf{s}_{l+1,k}^{(4)} = (m_2, \mathbf{x}_{c,k}^{(4)})$. In Fig. 3(e), τ^2 has transferred the system state to $\mathbf{s}_{l+1,k}^{(5)} = (m_3, \mathbf{x}_{c,k}^{(5)})$. $\mathbf{s}_{l+1,k}^{(3)}$ and $\mathbf{s}_{l+1,k}^{(5)}$ can then be merged. This is shown in Fig. 3(f).

Merging the uncertainty is particularly efficient to counter the effects of the occurrence of multiple similar splits and switches, which are a consequence of the temporal uncertainty due to variable bounds. In general, the uncertainty on the continuous state translates into the occurrence of the same transition switch over several time steps. Such situations are common and lead to the production of many estimates with an identical discrete state within just a few time steps. Using a merge operator, it takes just a few more time steps to produce a single estimate instead. However, the actual implementation behind the $\bigcup$ operation on line 3 of Table VII yields a conservative outer approximation of the merged estimates in the shape of a hypercube. This operation introduces an error, because, in general, the union of hypercubes does not yield a hypercube. In our example, the error is visible in Fig. 3(f).

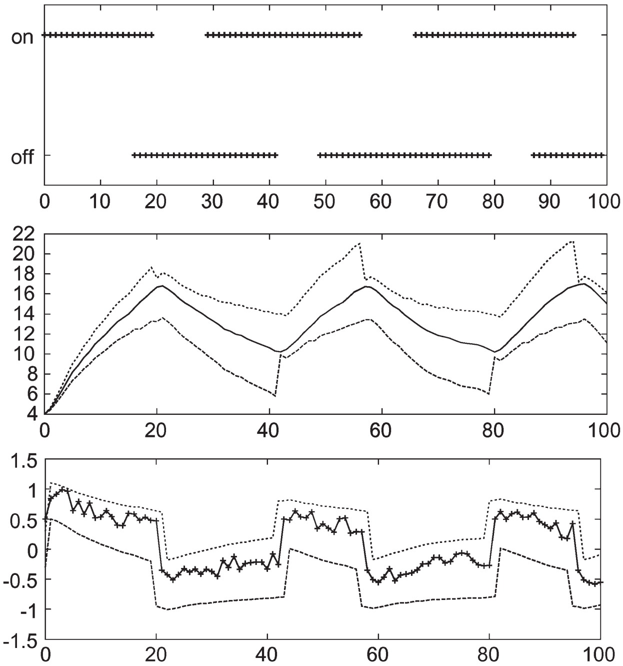

Fig. 4. Set-theoretic estimation of the hybrid state of a thermostat system (Example 1). (Top figure) Mode estimation. (Middle figure) Temperature (in Celsius). (Bottom figure) Temperature variation. All figures: X-axis is time.

Thus, in practice, the merging process enlarges the estimated bounds and reduces the number of estimates. However, the bounds remain guaranteed to enclose the true behavior of the system. However, the additional error carried by the bounds does affect the soundness of the estimator, which produces estimates that would not be reachable otherwise. A consequence is that in practice, our hybrid estimation process is complete but unsound.

V. RESULTS

A preliminary version of the presented filter was implemented in C++ as part of a hybrid system diagnosis platform.

A. Case Studies

Fig. 4 shows the result of a run on our thermostat example. In addition to the thermostat example, the state estimation scheme presented in this paper has been applied to the bi-tanks water regulation system in [48]. This system maintains an outflow of water to a virtual consumer. It models two water tanks, three valves, and a pump. As such, the model totals 1350 possible modes, each of which represents a combination of functional modes for all components in the system. Results on running our estimator on these two systems follow.

B. General Performances

We have studied the computation time of the estimate as well as the number of state estimates maintained by our filter. The results are reported in Figs. 5 and 6. Fig. 5 illustrates the double advantage of state estimation based on models with bounded uncertainty over Bayesian filtering. First, the highest number of

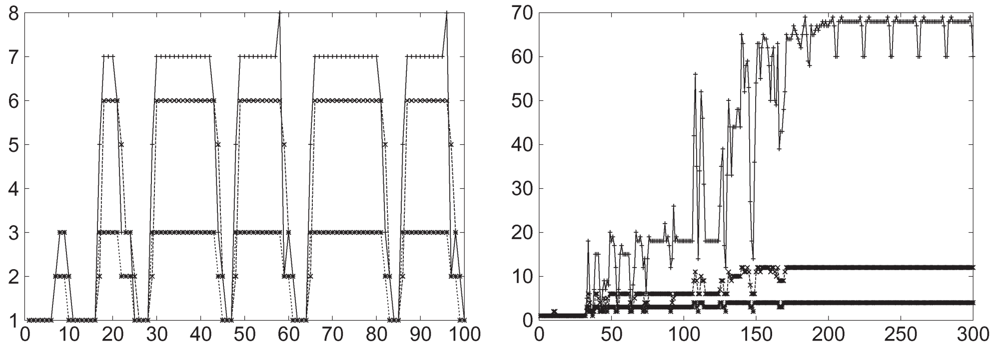

Fig. 5. Number of estimates before and after the merging step. (Left) Thermostat. (Right) Bi-tanks. (Top curve) Hybrid estimates before merging. (Middle curve) Continuous estimates before merging. (Lower curve) Hybrid estimates after merging.

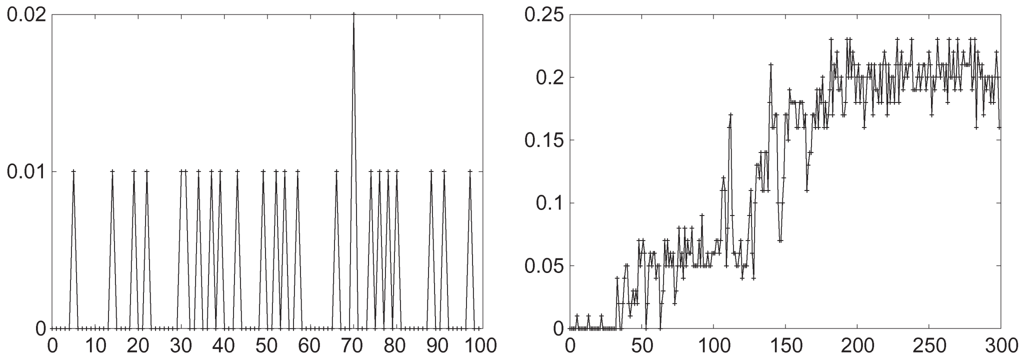

Fig. 6. Computation time per sampled time step (in seconds). (Left) Thermostat. (Right) Bi-tanks.

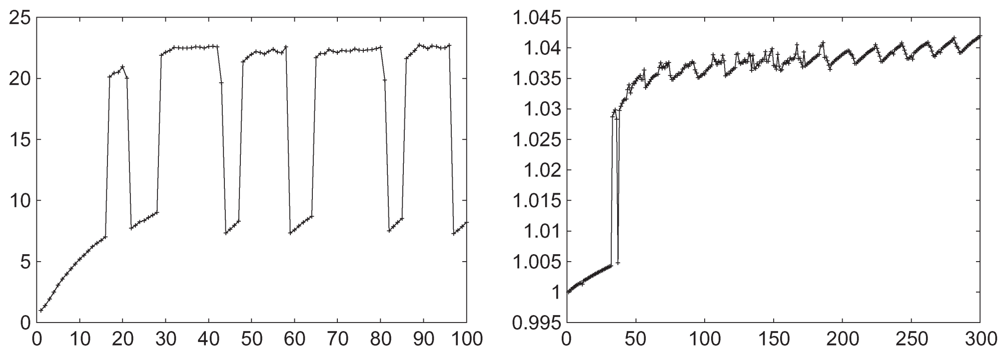

Fig. 7. Relative growth of the bounded uncertainty $\|\mathbf{x}_{c,k}\|/\|\mathbf{x}_{c,0}\|$ over time. (Left) Thermostat. (Right) Bi-tanks.

estimated hybrid states is before the merging step and remains low. For the bi-tanks, this number is around 70, that is, at worst 5% of all the possible states. Second, the merging step drastically reduces the total number of hybrid estimates down to five estimates in the worst case for the bi-tanks. It appears that the computation time is best correlated with the number of state estimates before the merging step is applied (see Fig. 6). Note that comparison with stochastic filters is not directly feasible.

C. *Uncertainty*

The discrete switches in a system's dynamics have an effect on the number of state estimates. Based on the same runs as before, we aimed to elucidate the effect of bounded uncertainty on state estimation. Since bounds do not converge, uncertainty is expected to grow unconditionally with time. Fig. 7 reports that the uncertainty is growing steadily, but is mitigated by the switches in the continuous dynamics. This property is explained by the switching mechanism presented in this paper. Each switch can help decrease uncertainty in the continuous state vector: by splitting the continuous state, a switch discards a subregion of the continuous state space. However, the uncertainty grows again invariably until the next switch occurs.

This behavior again contrasts with the stochastic hybrid filters that can shift and focus a probability distribution around subregions of the continuous state space but cannot scale their number of estimates accordingly.

VI. Conclusion

This paper has presented a set-theoretic alternative to the estimation of hybrid systems. It has highlighted the benefits of the approach compared to the dominant estimation scheme that utilizes continuous probability distributions to represent uncertainty. At the core of this paper are the configurations and logical configurations that articulate the discrete and continuous knowledge levels and permit dedicated algorithms to prune impossible estimates at each level. Because bounds do not converge, and due to a conservative merging of estimates, the outer approximation of the continuous state is expected to grow unconditionally with time. Potential solutions include the application of aggressive optimization techniques that produce tighter bounds, and the use of more expressive geometrical shapes. In application to large systems, the computational burden of the next state expansion can prove prohibitive. As a solution, transition selection through sampling or forward search can be implemented as for stochastic hybrid filters at the cost of losing completeness. More research should concentrate on bridging stochastic model-based estimators and their set-theoretic counterpart. In general, a pdf badly mixes with bounded spaces. Thus, the uniform distribution proves unproductive, because it is not closed under standard operations. However, some pieces of work have been produced [49], and a comparison of stochastic and set-theoretic estimation procedures for continuous systems can be found in [50]. This issue is undoubtedly a promising research direction for the future.

Acknowledgment

The authors would like to thank the anonymous reviewers for their valuable help in improving this paper.

References

[1] X. Li and Y. Bar-Shalom, "Multiple-model estimation with variable structure," *IEEE Trans. Autom. Control*, vol. 41, no. 4, pp. 478–493, Apr. 1996.

[2] P. Hanlon and P. Maybeck, "Multiple-model adaptive estimation using a residual correlation Kalman filter bank," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 36, no. 2, pp. 393–406, Apr. 2000.

[3] I. Hwang, H. Balakrishnan, and C. J. Tomlin, "State estimation for hybrid systems: Applications to aircraft tracking," *Proc. Inst. Elect. Eng.—Control Theory Appl.*, vol. 153, no. 5, pp. 556–566, Sep. 2006.

[4] S. McIlraith, G. Biswas, D. Clancy, and V. Gupta, "Towards diagnosing hybrid systems," in *Proc. 10th Int. Workshop Principles Diagnosis DX-99*, 1999. AQ2

[5] S. Narasimhan and G. Biswas, "Efficient diagnosis of hybrid systems using models of the supervisory controller," in *Proc. 12th Int. Workshop Principles Diagnosis DX-01*, Italy, 2001. Sansicario, Via Lattea.

[6] S. Narasimhan, R. Dearden, and E. Benazera, "Combining particle filter and consistency-based approaches for monitoring and diagnosis of stochastic hybrid systems," in *Proc. 15th Int. Workshop Principles Diagnosis DX-04*, 2004. AQ3

[7] S. Narasimhan and L. Brownston, "Hyde—A general framework for stochastic and hybrid model-based diagnosis," in *Proc. 18th Int. Workshop Principles Diagnosis DX-07*, 2007, pp. 162–169.

[8] S. Narasimhan and G. Biswas, "Model-based diagnosis of hybrid systems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 37, no. 3, pp. 348–361, May 2007.

[9] A. Doucet, N. de Freitas, K. Murphy, and S. Russel, "Rao-blackwellized particle filtering for dynamic Bayesian networks," in *Proc. 18th Int. Conf. Uncertainty Artif. Intell.*, 2002, pp. 176–183.

[10] F. Hutter and R. Dearden, "The Gaussian particle filter for diagnosis of non-linear systems," in *Proc. 13th Int. Workshop Principles Diagnosis DX-03*, 2003, pp. 65–70.

[11] S. Funiak and B. Williams, "Multi-modal particle filtering for hybrid systems with autonomous mode transitions," in *Proc. 13th Int. Workshop Principles Diagnosis DX-03*, 2003. AQ4

[12] U. Lerner, R. Parr, D. Koller, and G. Biswas, "Bayesian fault detection and diagnosis in dynamic systems," in *Proc. AAAI/IAAI*, 2000, pp. 531–537.

[13] U. Lerner, B. Moses, M. Scott, S. McIlraith, and D. Koller, "Monitoring a complex physical system using a hybrid dynamic Bayes net," in *Proc. UAI*, 2002, pp. 301–310.

[14] M. Hofbaur and B. Williams, "Hybrid estimation of complex systems," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 34, no. 5, pp. 2178–2191, Oct. 2004.

[15] N. de Freitas, R. Dearden, F. Hutter, R. Morales-Menendez, J. Mutch, and D. Poole, "Diagnosis by a waiter and a Mars explorer," *Proc. IEEE—Special Issue on Sequential State Estimation*, vol. 92, no. 3, pp. 455–468, Mar. 2004.

[16] O. Martin, "Diagnosis as approximate belief state enumeration for probabilistic concurrent constraint automata," in *Proc. 20th Nat. Conf. Artif. Intell.*, 2005. AQ5

[17] H. Blom and Y. Bar-Shalom, "The interacting multiple model algorithm for systems with Markovian switching coefficients," *IEEE Trans. Autom. Control*, vol. 33, no. 8, pp. 780–783, Aug. 1998.

[18] L. Johnston and V. Krishnamurthy, "An improvement to the interacting multiple model (IMM) algorithm," *IEEE Trans. Autom. Control*, vol. 49, no. 12, pp. 2909–2923, Dec. 2001.

[19] V. Verma, S. Thrun, and R. Simmons, "Variable resolution particle filter," in *Proc. 18th Int. Joint Conf. Artif. Intell.*, 2003, pp. 976–984.

[20] S. Thrun, J. Langford, and V. Verma, "Risk sensitive particle filters," in *Proc. NIPS*, 2001, pp. 961–968.

[21] C. Plagemann, D. Fox, and W. Burgard, "Efficient failure detection on mobile robots using particle filters with Gaussian process proposals," in *Proc. 20th Int. Joint Conf. Artif. Intell.*, 2007, pp. 2185–2190.

[22] L. Blackmore, S. Funiak, and B. Williams, "Combining stochastic and greedy search in hybrid estimation," in *Proc. 20th Nat. Conf. Artif. Intell.*, 2005, pp. 282–287.

[23] P. Nayak and J. Kurien, "Back to the future for consistency-based trajectory tracking," in *Proc. AAAI*, Austin, TX, 2000, pp. 370–377.

[24] E. Benazera and S. Narasimhan, "An extension to the Kalman filter for an improved detection of unknown behavior," in *Proc. 24th Amer. Control Conf.*, Jun. 2005, pp. 1039–1041.

[25] F. Cozman and E. Krotkov, "Truncated Gaussians as tolerance sets," Robot. Inst., Carnegie Mellon Univ., Pittsburgh, PA, Tech. Rep. CMU-RI-TR-94-35, Sep. 1994.

[26] M. Hofbaur and B. Williams, "Mode estimation of probabilistic hybrid systems," in *Proc. HSCC*, 2002, vol. 2289, pp. 253–266.

[27] D. S. Weld and J. D. Kleer, *Readings in Qualitative Reasoning About Physical Systems*. San Mateo, CA: Morgan Kaufmann, 1990.

[28] B. J. Kuipers, *Qualitative Reasoning: Modeling and Simulation With Incomplete Knowledge*. Cambridge, MA: MIT Press, 1994.

[29] P. Combettes, "The foundations of set theoretic estimation," *Proc. IEEE*, vol. 81, no. 2, pp. 182–208, Feb. 1993.

[30] M. Milanese and A. Vicino, "Estimation theory for nonlinear models and set membership uncertainty," *Automatica*, vol. 47, no. 2, pp. 403–408, Mar. 1991.

[31] A. Vicino and G. Zappa, "Sequential approximation of feasible parameter sets for identification with set membership uncertainty," *IEEE Trans. Autom. Control*, vol. 41, no. 6, pp. 774–785, Jun. 1996.

[32] L. Jaulin, "Interval constraint propagation with application to bounded-error estimation," *Automatica*, vol. 36, no. 10, pp. 1547–1552, 2000.

[33] U. Hanebeck, "Recursive nonlinear set-theoretic estimation based on pseudo ellipsoids," in *Proc. IEEE Conf. Multisensor Integr. Intell. Syst.*, 2001, pp. 159–164.

[34] C. Combastel, "A state bounding observer for uncertain non-linear continuous-time systems based on zonotopes," in *Proc. IEEE CDC*, Seville, Spain, Dec. 2005, pp. 7228–7234.

[35] J. Armengol, J. Vehi, L. Travé-Massuyès, and M. Sainz, "Interval model-based fault detection using multiple sliding time windows," in *Proc. 4th SAFEPROCESS*, Budapest, Hungary, 2000, pp. 168–173.

[36] T. A. Henzinger, B. Horowitz, R. Majumdar, and H. Wong-Toi, "Beyond HYTECH: Hybrid systems analysis using interval numerical methods," in *Proc. HSCC*, 2000, pp. 130–144.

[37] E. Benazera and L. Travé-Massuyès, "The consistency approach to the on-line prediction of hybrid system configurations," in *Proc. IFAC Conf. Anal. Des. Hybrid Syst.*, 2003.

[38] M. Branicky, V. Borkar, and S. Mitter, "A unified framework for hybrid control: Model and optimal control theory," *IEEE Trans. Autom. Control*, vol. 43, no. 1, pp. 31–45, Jan. 1998.

[39] W. Hamscher, L. Console, and J. D. Kleer, *Readings in Model-Based Diagnosis*. San Mateo, CA: Morgan Kaufmann, 1992.

[40] B. C. Williams and P. Nayak, "A model-based approach to reactive self-configuring systems," in *Proc. AAAI*, Portland, OR, 1996, pp. 971–978.

[41] R. Alur, C. Courcoubetis, N. Halbwachs, T. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine, "The algorithmic analysis of hybrid systems," in *Proc. 11th Int. Conf. Anal. Optim. Discrete Event Syst.*, 1995, pp. 331–351.

[42] J. Shamma and K. Tu, "Approximate set-valued observers for nonlinear systems," *IEEE Trans. Autom. Control*, vol. 42, no. 5, pp. 648–658, May 1997.

[43] B. C. Williams and P. Nayak, "Fast context switching in real-time reasoning," in *Proc. AAAI*, Providence, RI, 1997, pp. 50–56.

[44] E. Hyvonen, "Constraint reasoning based on interval arithmetic: The tolerance propagation approach," *Artif. Intell.*, vol. 58, no. 1–3, pp. 71–112, Dec. 1992.

[45] J. D. Kleer and J. S. Brown, "A qualitative physics based on confluences," *Artif. Intell.*, vol. 24, no. 1–3, pp. 7–83, Dec. 1984.

[46] T. Nishida and S. Doshita, "Reasoning about discontinuous change," in *Proc. AAAI*, 1987, pp. 643–648.

[47] Y. Iwasaki, A. Farquhar, V. Saraswat, D. Bobrow, and V. Gupta, "Modeling time in hybrid systems: How fast is 'instantaneous'?" in *Proc. IJCAI*, 1995, pp. 1773–1780.

[48] B. Heiming and J. Lunze, "Definition of the three-tank benchmark problem for controller reconfiguration," in Proc. Eur. Control Conf., 1999.

[49] U. Hanebeck, J. Horn, and G. Schmidth, "On combining statistical and set-theoretic estimation," *Automatica*, vol. 35, no. 6, pp. 1101–1109, Jun. 1999.

[50] G. Hager, S. Engelson, and S. Atiya, "On comparing statistical and set-based methods in sensor data fusion," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1993, pp. 352–358.

Emmanuel Benazera received the M.S. degree in applied mathematics and social sciences from the University Paris-Dauphine, Paris, France, in 1999 and the Ph.D. degree in computer science from Paul Sabatier University, Toulouse, France, in 2003.

He was a Visiting Scientist with the NASA Ames Research Center, Mountain View, CA, for two years before he joined the Robotics Department, DFKI/University of Bremen, Bremen, Germany, for a year. He is currently with LAAS–Centre National de Recherche Scientifique (CNRS), Université de Toulouse, Toulouse. His main research interests include automated diagnosis, decision and planning under uncertainty, and hybrid systems.

Louise Travé-Massuyès (SM'93) received the Ph.D. degree in control and the Engineering degree (specialized in control, electronics, and computer science) from the Institut National des Sciences Appliques (INSA), Toulouse, France, in 1984 and 1982, respectively, and the Habilitation Diriger des Recherches degree from Paul Sabatier University, Toulouse, in 1998.

She is currently the Research Director of the Centre National de Recherche Scientifique (CNRS), LAAS, Toulouse, where she has been the scientific leader of the Diagnosis, Supervision and Control Research Group for several years. She is responsible for the scientific area of modeling, control, and system optimization that oversees four research groups in the field of continuous and discrete automatic control. Her main research interests are in qualitative and model-based reasoning and applications to dynamic systems supervision and diagnosis. She has been particularly active in bridging the AI and control engineering model-based diagnosis communities as the leader of the BRIDGE Task Group, MONET European Network. She has been responsible for several industrial and European projects and published more than 100 papers in international conference proceedings and scientific journals.

Dr. Travé-Massuyès is a Senior Member of the IEEE Computer Society. She was the recipient of an award from the Union des Groupements d'Ingenieurs de la Region Midi-Pyrnes.