THE CONSISTENCY APPROACH TO THE ON-LINE PREDICTION OF HYBRID SYSTEM CONFIGURATIONS [1]

Emmanuel Benazera * **Louise Travé-Massuyès** *

* *LAAS-CNRS, 7, av. du Colonel Roche, 31077 Toulouse Cedex 4*

Abstract: The supervision of an uncertain hybrid system poses the problem of the fast prediction of its behavioral operating regions on a sampled time-line. We refer to these regions as the *configurations* of the hybrid system, that partition the state-space into cells that ease its monitoring, diagnosis and control. In this paper, configuration prediction is approached as a constraint satisfaction problem at both continuous and discrete levels. *Copyright, 2003, IFAC.*

Keywords: Uncertain Hybrid System, Prediction, Configurations, Constraint Satisfaction, Constraint Filtering.

1. INTRODUCTION

In recent years hybrid systems emerged as a framework of interest to the model-based monitoring, diagnosis and control of embedded digitally controlled systems. These devices have complex behaviors characterized by a pre-eminence of discrete switches in their dynamics. The hybrid system discrete side abstracts the physical system as a set of functional modes M and models autonomous and controlled jumps among modes. The continuous state is made of continuous variables over which differential equations govern the continuous behavior. *An uncertain hybrid model is used to estimate the physical state of a system based on observations from high frequency sensors.*

This paper poses the problem of predicting the region in which an uncertain hybrid system behaves, in sampled time, as two constraint satisfaction problems, respectively at discrete and continuous levels. To this end, we propose a generic formulation that uses the existence of the automaton abrupt discrete switches to slice the continuous state-space into a finite set of regions of behavior. Each region is referred to as a *configuration* of the hybrid system. The general idea is to make a hybrid state prediction *consistent with the grid of configurations* at each sampled time step. The paper is organized as follows: section 2 formulates the hybrid automaton and configurations; section 3 predicts the hybrid system's set of configuration by solving constraint satisfaction problems at continuous and discrete levels; section 4 computes discrete switches in sampled time; section 5 illustrates the approach with a simple model from the hybrid systems litterature.

2. FORMULATION OF HYBRID SYSTEM CONFIGURATIONS

2.1 Uncertain Hybrid Automaton

Definition 1. (Uncertain Hybrid System). An Uncertain Hybrid System H is a tuple (Π, X, T, C, Θ) such that:

- $\Pi = M \cup D \cup J$ – discrete state, respectively *mode*, *discrete* and *conditional* variables.

[1] This work is partially supported by the Centre National d'Etudes Spatiales (CNES) and ASTRIUM.

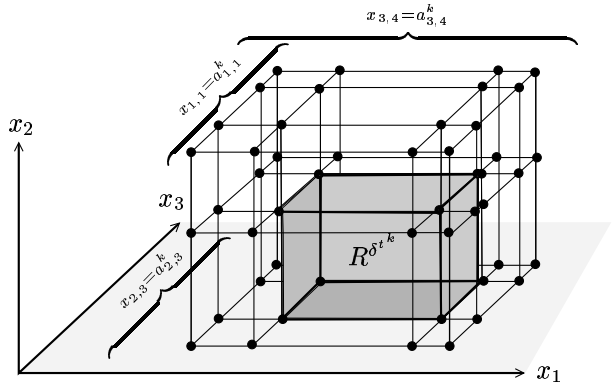

Fig. 1. 3-dimensional grid of configurations at any time-step t^k. Edges correspond to the evaluation of discrete modalities κ^j

- X – continuous state vector of size n. Any $x_i \in X$ can take real or *interval* values.
- T – finite set of transitions between modes. $\tau \in T$ is triggered by a conditional statement $\phi(\tau)$ over variables in J, and is associated to a *mapping function* l_τ that transfers X from a mode to another.
- $C = Q \cup E$ – constraints, *qualitative* and *quantitative* respectively .
- Θ – initial uncertain state of H.

We do not make any assumption on the set of continuous differential equations E. We denote Q the set of internal and *safety* constraints between discrete variables, e.g. exclusion between modes.

2.2 States and Time

Given a hybrid system H, at the continuous level, time is explicit in the equations E, we refer to it as the *physical time*. The physical time is discretized according to the highest frequency sensor, providing H reference sampling period T_s. $X(kT_s)$ or $X(k)$ for short, specifies the value of the *continuous state-vector* at physical time kT_s. We call *abstract time* the time at the discrete level. It is dated according to the occurrence of discrete events such as a transition firing. At date t, the *discrete state* Π_t of the hybrid system is the tuple (M_t, D_t), where M_t is the mode, and D_t the vector of instances of other discrete variables in $D \cup J$. τ is said to be *enabled* w.r.t. a discrete state Π when $\Pi \models \phi(\tau)$. Abstract time dates are indexed on physical time, which informs on how long a H has been in a given discrete state. We use l as the abstract time index when needed. If $t = kT_s$, then we write the indexed date t^k. When there is no ambiguity it is simply denoted by t. The *hybrid state* s_{t^k} of H is the tuple $\big(\Pi_{t^k}, X(k)\big)$. *Uncertainty imposes to reason on a set S of hybrid states with respective modes and uncertain continuous state-vectors.*

2.3 Hybrid System Configurations

ϕ are conditions over continuous variables. Let us consider such a condition over X as an inequality of the form $x_i \leq (\geq) f(x_1, \cdots, x_{i-1}, x_{i+1}, \cdots, x_n)$, where the *condition function* $f(x_1, \cdots, x_{i-1}, x_{i+1}, \cdots, x_n)$ is any continuous function. A condition function is referenced to the variable x_i, and the set of functions $f_i^1, \cdots, f_i^q$ defines the *conditional domain* $(x_{i,1}, \cdots, x_{i,q})$ of x_i. The evaluation function of a condition function f_i^j against a continuous vector $X(k)$ at time step k delimitates $x_{i,j}$. It is noted $x_{i,j} \leq (\geq) a^k_{i,j}$, where $a^k_{i,j}$ is a real or an interval.

It comes that given a time step k, we write the conjunction of evaluated domains for all the $x_i \in X$: $G\big(X(k)\big) = \bigwedge_{i,j}(x_{i,j} \leq (\geq) a^k_{i,j})$ partitions the continuous state-space into polyhedres that form a grid. We denote a cell of this grid a *configuration*. Cells are subparts of $G\big(X(k)\big)$ as combinations of variable conditional domains. For a given set $J^* = \{J_1^*, \cdots, J_n^*\}$ of indexes, $G_{J^*}\big(X(k)\big) = \bigwedge_i\big(\bigwedge_{j \in J_i^*} x_{i,j} \leq (\geq) a^k_{i,j}\big)$ is the cell of $G\big(X(k)\big)$ delimited by J^*. The cell's volume is denoted the *configuration region*. Configuration regions depend on the condition functions, that act as constraints among the cells. Two cases arise:

- The f_i^j are all constants, and the grid is independent of the continuous state $X(k)$,
- Otherwise configuration regions depend on $X(k)$: $X(k)$ has real values only, and configuration regions are distinct, or $X(k)$ is uncertain and configuration regions overlap.

Figure 1 illustrates the case where $X(k)$ is real and has three dimensions: axis $x_{i,j} \leq (\geq) a^k_{i,j}$ are volumes that define configuration regions.

We now want to link continuous variables in X to the discrete level. This is done naturally: given $X(k)$, the $x_{i,j} \leq (\geq) a^k_{i,j}$ may enclose $X(k)$ or not, i.e. $(x_{i,j} \leq (\geq) a^k_{i,j}) \cap X(k) \neq \emptyset$ is true or not. We have discrete variables $\kappa \in J$ differ from other variables by having *their modalities rely on conditions over the continuous variables in X*. Conditional domain $(x_{i,1}, \cdots, x_{i,q})$ of x_i, is mapped onto the modalities κ_i^j of a discrete *condition variable* $\kappa_i \in \Pi_{Cond}$ in the following manner:

- At time step k, if $(x_{i,j} \leq (\geq) a_{i,j}) \cap X(k) \neq \emptyset$, then κ_i^j is true,
- False otherwise.

Constraints of exclusion among the κ_i^j are qualitative constraints in Q. We use the following notations: $fct(\kappa_i^j) = (x_{i,j} \leq (\geq) a^k_{i,j})$, its reduction to equality relation $fce(\kappa_i^j) = (x_{i,j} = a^k_{i,j})$ and $var(\kappa_i^j) = x_i$.

Requirement 1. The domain of a discrete conditional variable κ_i is attached to a continuous variable x_i must cover the entire real domain of x_i.

Condition variables pave the way for the definition of a configuration that articulates continuous and discrete levels.

Definition 2. (Hybrid System Configuration). [2] A configuration for a hybrid system H is a logical conjunction $\delta = m \wedge \left(\bigwedge_j (\bigwedge_{j \in J_i^*} \kappa_i^j\right)$ where $m \in M$.

If κ_i^j is involved in the definition of a configuration δ, we say that $\kappa_i^j \in \delta$. The continuous region underlying a configuration δ is denoted R^δ, with $R^\delta = \bigcap_{\kappa_i^j \in \delta} fct(\kappa_i^j)$. We also define the region's *frontier* $fr(R^\delta) = \bigcup_{\kappa_i^j \in \delta} fce(\kappa_i^j)$. For a given k, we can write: δ true $\Leftrightarrow M \wedge \left(R^\delta \cap X(k) \neq \emptyset\right)$ true . The total number n_c of configurations is the product of modalities of both mode and conditional variables. The number of underlying regions is $n_r \leq n_c$ where n_r is the product of the modalities of conditional variables only. Changes between two time-steps may also be non-linear, but it is important to note that the logical representation of each cell stays unchanged. The next section develops a consistency-based approach to the partitioning of the state-space with configurations.

3. THE CONSISTENCY APPROACH TO CONFIGURATION PREDICTION

3.1 Hybrid state prediction in sampled time

From the work of the model-checking community on hybrid systems (Alur *et al.*, 1995), specific operators have been defined to reason forward and backward in time on hybrid systems. The sampled time hybrid state prediction can be described by making use of the forward operators.

The *forward time closure* $\langle S_{t^k} \rangle^\nearrow$ of a set of hybrid states S_{t^k} is expressed as the set of hybrid states that are reachable from $s_{t^k} \in S_{t^k}$ by letting the physical time progress:

$$s_{t^{k'}} \in \langle S \rangle_d^\nearrow \text{ iff } \exists s_{t^k} \in S \text{ and } k' = k + dT_s$$
$$s_{t^k} = \left(\Pi_t, X(k)\right) \xrightarrow{k} s_{t^{k'}} = \left(\Pi_t, X(k')\right) \quad (1)$$

where $\xrightarrow{k}$ expresses the physical time progress. Relation 1 can be computed in different ways, notably by using interval numerical methods (Henzinger *et al.*, 2000). $\langle S \rangle_0^\nearrow$ is a *static* version of the operator, i.e. that envisions the closure instantaneously. We denote states resulting from the foward time closure hybrid *pseudo-states* as at this point the prevision process is merely incomplete. For a given transition τ and a set of hybrid states $S_{t_l^k}$, the *forward transition closure* $\langle S_{t_l^k} \rangle^\tau$ is the set of hybrid states that are reachable from some state $s_{t_l^k} \in S_{t_l^k}$ by executing a transition τ:

$$s_{t_{l+1}^{k'}} \in \langle S_{t_l^k} \rangle^\tau \text{ iff}$$
$$\exists s_{t_l^k} \in S_{t_l^k} \text{ such that } s_{t_l^k} \xrightarrow{\tau} s_{t_{l+1}^{k'}} \quad (2)$$

where $\xrightarrow{\tau}$ changes the automaton mode and applies the mapping function l_τ to the continuous state ($k - k'$ is τ's delay). The hybrid system prediction over time alternates both forward operators. However, the triggering of an autonomous transition imposes to test its guard against the continuous state $X(k)$. Sometimes, the resulting truth value may be undetermined when checked against the rectangular enclosing of continuous variables, because the system (uncertain) state is *spanning over more than one configuration.* This is the reason why an additional operator, denoted *split*, is required to articulate forward time and transition closures and *produce a consistent partitioning of the state-space at each time-step.* This corresponds to bringing a solution to the following general problem.

3.2 Discrete and continuous states consistency problems

The partitioning problem we want to solve is expressed as follows [3] :

Problem 3. (Consistent Configurations Partition). Given a set of hybrid states $S_{t^{k-1}}$ at time-step t^{k-1} and the forward time closure $\langle S_{t^{k-1}} \rangle_1^\nearrow$, find a partition $\mathcal{P}_S = \{s^1, \cdots, s^m\}$ such that for any $p = 1, \cdots, m$: $\Pi_p \models \delta_p$, $X_p \subseteq R^{\delta_p}$, $\mathcal{P}_X = \{X_1, \cdots, X_m\}$ a partition of X and $S_{t^k} = \{\left(\Pi_p, X_p(k)\right)\}_{p=1,\cdots,m}$.

Given a set of hybrid pseudo-states, we want to partition it into configuration cells whose volumes include the entire continuous vector. The result is an arranged set of hybrid states that is the prediction of the system's behavior. Numerical methods will help partition the continuous space into regions, while constraint satisfaction techniques will find discrete states that correspond to these regions. Problem 3 is hence decomposed in the two following subproblems.

[2] We have a configuration include the automaton mode for compacting the representation. It is nevertheless possible to treat them separately.

[3] time index is omitted when the reasoning takes place at a single time-step. In this case, X denotes $X(k)$.

Problem 4. (Discrete State Consistency). Find δ_p such that it is consistent with the hybrid system's automaton, discrete state and safety constraints:

$$H \cup \Pi_p \cup Q \cup \delta_p \qquad (3)$$

is consistent.

Such a problem is easily solved with any constraint satisfaction engine. We note $sat(\delta_p, H \cup \Pi_p \cup Q)$ the operator that solves it.

Problem 5. (Continuous State Consistency). Given a configuration δ_p, find the configuration grid edges that are consistent with $fr(R^{\delta_p})$, defined by the $\kappa_i^j \in \delta_p$:

$$\begin{cases} (x_1 = f_1^{j_1}(x_2, \cdots, x_n)) \\ \quad \vdots \\ (x_n = f_n^{j_n}(x_1, \cdots, x_{n-1})) \end{cases} \qquad (4)$$

Evaluating functions in relation 4 leads to a set of axis-parallel hyperplanes (of equation $x_i = a$, a some real) that define the grid's edges in the n-dimensional space [4]. Such a problem can be solved with standard filtering or branch-and-bound techniques. Independently from the technique, we note $filter(X, F)$ an operator that solves this problem, where F is the set of the f_i^j. It returns a continuous state X_p.

We now complete the sampled time hybrid system state prediction cycle, before we position an algorithm that solves both problems above.

3.3 Splitting the hybrid state with configurations

Let us construct the operator *split*, that applies to a set S of hybrid pseudo-states and returns a correct partitioning. For a given state $s \in S$, when a truth value of a condition over a continuous variable x_i is undetermined, we want to find the possible configurations (problem 4) and branch on the solutions, then refine consequently the bounds on all continuous variables in every explored configuration (problem 5). Algorithm 1 merges a filtering algorithm that satisfies the constraints among the grid cells, and a constraint satisfaction engine that searches for consistent configurations. The algorithm starts from the current configuration δ_0 and returns a set of hybrid states with consistent configurations. Second step checks for the x_is whose bounds may fall outside the starting configuration region R^{δ}. Third step solves relation 3 by searching over the modalities κ_i^j related to x_i, and fourth step solves relations 4. The algorithm is applied until there are no more variable values in s that fall outside R^{δ_p}. For each hybrid state s, it results into a collection of new hybrid states S_p. By extending the operator to all states $s \in S$, we note $\mathcal{P}_S = split(S)$ and $\mathcal{P}_X = split(X)$ its projection on the continuous state-space.

[4] The total number of such hyperplanes in the grid is the number of axis-parallel edges necessary to define n_r rectangles of dimension n.

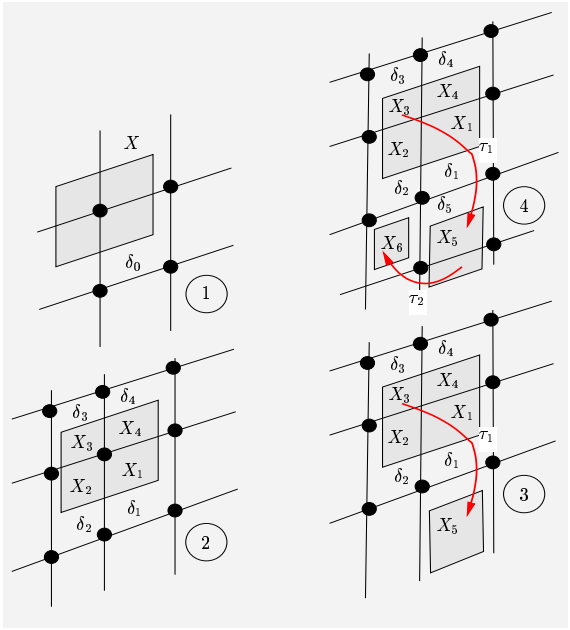

Fig. 2. Search for configurations

1: $\delta = \delta_0$
2: While $\exists\ x_i$ such that $x_i \not\subseteq R^{\delta_0}$.
3: $\forall\ \kappa_i^j$, do $sat(\delta_p, H \cup \Pi_p \cup Q \cup \kappa_i^j)$.
4: $\forall\ \delta_p$, do $X_p = filter\left(X, \bigwedge_{\kappa_i^j \in \delta_p} fct(\kappa_i^j)\right)$.
$\forall p$, $\delta = \delta_p$, go to step 2.

Algorithm 1: Splitting the state-space: $split(s)$.

On figure 2, step 2 visualizes a partition of X. In our present implementation we use the filtering technique in (Hyvonen, 1992) to solve relations 4 as it works well on non-linear constraints, and the boolean satisfaction engine in (Williams and Nayak, 1997) as it allows fast jumps over configurations. When partitioning is completed, transition can be fired from any of the δ_p locations.

Proposition 6. Each hybrid state s_p that results from a split is such that $X_p \subseteq R^{\delta_p}$.

Proof For a given variable x_i, we note $x_i \in X$, $x_i^p \in X_p$ and $var(\kappa_i^j) = x_i$. Satisfying relation 4 leads to $x_i^p = \bigcap_j fct(\kappa_i^j) \cap x_i$. Considering all variables in X, we have:

$$X_p = \bigcap_i \Big(\bigcap_j fct(\kappa_i^j) \cap x_i \Big) = \bigcap_{i,j} fct(\kappa_i^j) \bigcap_i x_i = R^{\delta_p} \cap X \qquad (5)$$

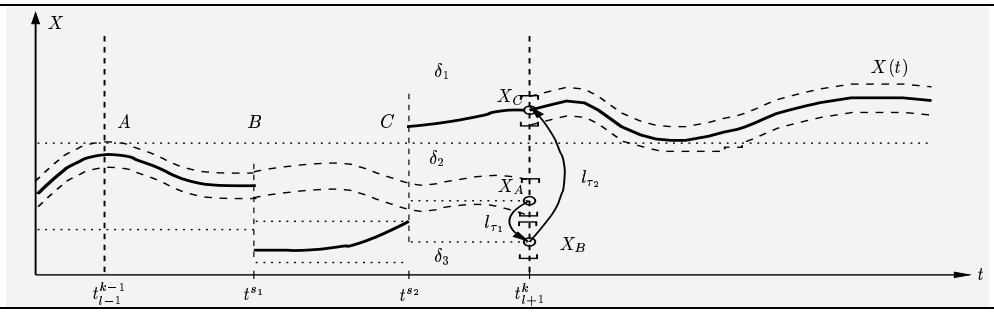

Fig. 3. Multiple switches lead up to time-step t_{l+1}^k, making up for fast switches at t^{s_1}, t^{s_2} between two physical time-steps, over 3 configurations.

All values of X_p are then inside (or equal) to R^{δ_p}. □

Proposition 7. X_p, $p = 1, \cdots, m$, that results from a split of a set of hybrid states S are such that $X_1 \cup \cdots \cup X_m = X$.

Proof First, *completeness* of the partitioning comes from requirement 1, that ensures the entire state-space is partitioned with configurations: $X \subseteq \bigcup_p R^{\delta_p}$. From relation 5: $\bigcup_p X_p = \bigcup_p R^{\delta_p} \cap X$. It comes that $\bigcup_p X_p = X$. □

4. SWITCHING IN SAMPLED TIME

The final step toward the prediction of the valid configurations of the hybrid system is the triggering of the enabled transitions. The sampled-time approach rises two problems: first, *a mode switch is always computed a small period of time after the real switch occurred* (it poses the problem of how to compute X_5 on figure 2, step 3); second, *multiple successive switches may occur during a single sampled time interval* (step 4).

4.1 Guaranteed enclosing at switching point

The first problem imposes us to compute a correct continuous vector on every switch, i.e. that it must be correctly stamped w.r.t. the sampled time, and guaranteed to encompass all behaviors, w.r.t. the initial conditions Θ. Our solution to correctly transfer the continuous state from one mode to another uses the fact that it is indisputable that the switch has occurred between previous and current time-steps. The correct rectangular enclosing is thus obtained by computing a switch at both *previous* and *current* time-steps. This is however made possible only under the following hypothesis: supposing a switch (forward transition closure) led to hybrid state $s_{t_{l+1}^k}$, *the function from E that describes the continuous evolution of the system is monotonous between t_l^{k-1} and t_{l+1}^k*. In practice, due to high sampling rates, this hypothesis is realistic. Algorithm 2 describes the operator $enclose\big(\tau, X_p(k)\big)$ that transfers a continuous state $X_p(k) \in \mathcal{P}_X$ according to an enabled transition τ and returns a correctly updated vector X_p. The only tricky part is the second step that compacts the continuous state at t_l^{k-1} to the part of its configuration frontier that matches τ's guard [5] : this *virtually* enables a switch at this time-point. Finally, note that the algorithm requires working on a temporal window at least two steps long.

1: $X'(k) = l_\tau\big(X_p(k)\big)$, $X_p(k) \in \mathcal{P}_X = split\big(X(k)\big)$.
2: $X'(k-1) = l_\tau\big(X^{fr}(k-1)\big)$ such that, δ X's configuration: $X^{fr}(k-1) = \phi(\tau) \cap fr(R^\delta)$
3: $s''_{t_{l+1}^k} = \langle s'_{t_{l+1}^{k-1}} \rangle_1^{\nearrow}$ and $s''_{t_{l+1}^k} = \langle s'_{t_{l+1}^k} \rangle_0^{\nearrow}$.
4: update X_p: $\forall x_i \in X_p(k)$,

$$x_i = \Big(\min\big(x_i''(k-1)\big), \max\big(x_i''(k)\big)\Big)$$

Algorithm 2: Applies transition τ, enabled at time-step t_{l+1}^k: $enclose\big(\tau, X_p(k)\big)$.

4.2 Multiple successive switches

Figure 3 visualizes the occurrence of two switches at t^{s_1}, t^{s_2} between time-steps t_{l-1}^{k-1} and t_{l+1}^k. The correct enclosing should start being computed at time-step t_{l-1}^k and should ensure that the dynamics that result from the *hidden* switches are caught back. The solution to this problem lies in the computation of the right enclosings X_A, X_B, and X_C, if the monotony hypothesis is verified on every functional piece of behavior ($[A, B], [B, C]$). In this particular case, it guarantees that switching from $X_A \not\subseteq \delta_2$ leads to $X_B = enclose(\tau_1, X_A) \not\subseteq \delta_3$, so switching again, to $C = enclose(\tau_2, X_B) \subseteq \delta_1$. The *switch* operator of algorithm 3 refines the hybrid state until there are no more enabled transition to fire. The number of successive switches

[5] Such a continuous state is thus made of fragments of X's configuration δ's frontier.

is noted w and the operator $merge(\mathcal{P}_S)$ optimizes the final partition by merging hybrid states that are such that $\Pi_p = \Pi_{p'}$ into $s^p = \big(\Pi_p, X_p(k) \cup X_{p'}(k)\big)$, and deletes $s^{p'}$. A condition for the algorithm to terminate is that the hybrid system's behavior excludes *infinitely many switches occurring at the same point in time*. In that case, the algorithm always terminates because the number of configurations is finite. Finally, a run for H is a finite or infinite sequence $\rho : S_0 \cdots S_{t_l} \cdots$, that verifies:

1: while $\exists\ \tau$ enabled, do $S_{t_l^k} = \langle S_{t_{l-1}^k} \rangle^\tau$
   - $\forall\ s \in S_{t_l^k}$, $X(k) = enclose\big(\tau, X(k)\big)$,
   - $S_{t_l^k} = split(S_{t_l^k})$.
2: $S_{t_{l+w}^k} = split(S_{t_{l+w}^k})$
3: $\mathcal{P}_S = S_{t_{l+w}^k}$ then $\mathcal{P}_S = merge(\mathcal{P}_S)$.

Algorithm 3: $switch(S)$ operator

$$S_0 = switch\big(split(\langle\Theta\rangle_0^{\nearrow})\big) \quad (6)$$

$$S_{t_{l+w}^{k+d}} = switch\big(split(\langle S_{t_l^k}\rangle_d^{\nearrow})\big) \quad (7)$$

Equation 6 initializes the components hybrid states according to their initial values Θ. The computation of the recursive equation 7 alternates forward time and transition closures through *splits* and *switches*, and results into a set of hybrid states at each physical time-step.

5. APPLICATION AND DISCUSSION

To illustrate our approach to the monitoring of a physical system in sampled time, we predict the configurations of a simple thermostat model from the hybrid systems litterature, on a 1Hz basis. The system models a room whose temperature x is controlled by a thermostat. It maintains x between 10 and 17 degrees by switching itself respectively to mode *on* or *off*. The temperature outside the room is of 4 degrees along the simulation. The exact rate of increase or decrease of the temperature is unknown, but bounded by known constants. x has 3 associated discrete modalities $x \leq 10$, $10 < x < 17$, $x \geq 17$ and 2 modes, that give 6 configurations, 4 of which are effective. Ineffective configurations can be pruned with available observations or properly defined safety constraints. Figure 4 visualizes a run of the thermostat system. Switching thresholds subdivise the continuous space at each time-step. The thermostat starts in configuration $\delta_1 : mode = on \wedge x \leq 10$, then oscillates between $\delta_2 : mode = on \wedge 10 < x < 17$ and $\delta_3 : mode = off \wedge 10 < x < 17$. The filtering effect is visible on the figure as smoothing the prediction on upper and lower thresholds.

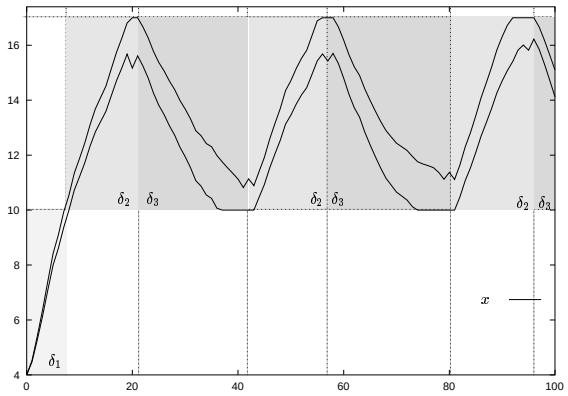

Fig. 4. Thermostat system prediction: temperature is made consistent with configurations.

This paper has presented a method for the online prediction of a hybrid system's configurations. Configurations shelter functional rectangular portions of behavior and ease the supervision, diagnosis and control of physical systems. In (Benazera *et al.*, 2002) we used the set of algorithms presented in this paper to build an autonomous supervisor for concurrent hybrid systems. An advantage of our approach is that the configurations prediction treats well any type of transition guard (e.g. non-linear functions). It is important to note however, that the configurations grid usually overnumbers the hybrid system effective behavioral regions. In a near future, we expect to apply that approach to the control and reconfiguration of physical systems.

REFERENCES

Alur, R., C. Courcoubetis, N.Halbwachs, T.A. Henzinger, P.-H. Ho, X. Nicollin, A.Olivero, J.Sifakis and S.Yovine (1995). The algorithmic analysis of hybrid systems. In: *Proceedings of the 11th International Conference on Analysis and Optimization of Discrete Event Systems*. pp. 331–351.

Benazera, E., L. Travé-Massuyès and P. Dague (2002). State tracking of uncertain hybrid concurrent systems. In: *Proceedings of the Thirteenth Internatinal Workshop on Principles of Diagnosis (DX-02), Semmering, Austria*. pp. 106–114.

Henzinger, T. A., B. Horowitz, R. Majumdar and H. Wong-Toi (2000). Beyond HYTECH: Hybrid systems analysis using interval numerical methods. In: *HSCC*. pp. 130–144.

Hyvonen, E. (1992). Constraint reasoning based on interval arithmetic: The tolerance propagation approach. *Artificial Intelligence* **58**(1-3), 71–112.

Williams, B. C. and P. Nayak (1997). Fast context switching in real-time reasoning. In: *Proceedings of AAAI-97, Providence, Rhode Island*.