[8] E. Fridman and U. Shaked, "An improved stabilization method for linear time-delay systems," *IEEE Trans. Automat. Contr.*, vol. 47, pp. 1931–1937, Nov. 2002.
[9] ——, "Delay-dependent stability and H_∞ control: constant and time-varying delays," *Int. J. Control*, vol. 76, no. 1, pp. 48–60, 2003.
[10] ——, "Parameter dependent stability and stabilization of uncertain time-delay systems," *IEEE Trans. Automat. Contr.*, vol. 48, pp. 861–866, May 2003.
[11] T. Mori, "Criteria for asymptotic stability of linear time-delay systems," *IEEE Trans. Automat. Contr.*, vol. AC-30, pp. 158–161, Feb. 1985.
[12] J. H. Lee, S. W. Kim, and W. H. Kwon, "Memoryless H_∞ controllers for state delayed systems," *IEEE Trans. Automat. Contr.*, vol. 39, pp. 159–162, Jan. 1994.
[13] T. J. Su and C. G. Huang, "Robust stability of delay dependence for linear uncertain systems," *IEEE Trans. Automat. Contr.*, vol. 37, pp. 1656–1659, Oct. 1992.
[14] X. Li and C. E. de Souza, "Delay-dependent robust stability and stabilization of uncertain linear dealy systems: a linear matrix inequality approach," *IEEE Trans. Automat. Contr.*, vol. 42, pp. 1144–1148, Aug. 1997.
[15] Y. Gu, S. Wang, Z. C. Q. Li, and J. Qian, "On delay-dependent stability and decay estimate for uncertain systems with time-varying delay," *Automatica*, vol. 34, no. 8, pp. 1035–1039, 1998.
[16] Y. Y. Cao, Y. X. Sun, and C. W. Cheng, "Delay-dependent robust stabilization of uncertain systems with multiple state delays," *IEEE Trans. Automat. Contr.*, vol. 43, pp. 1608–1612, Nov. 1998.
[17] C. E. de Souza and X. Li, "Delay-dependent robust H_∞ control of uncertain linear state-delayed systems," *Automatica*, vol. 35, no. 7, pp. 1313–1321, 1999.
[18] P. Park, "A delay-dependent stability criterion for systems with uncertain time-invariant delays," *IEEE Trans. Automat. Contr.*, vol. 44, pp. 876–877, Apr. 1999.
[19] Y. S. Moon, P. Park, W. H. Kwon, and Y. S. Lee, "Delay-dependent robust stabilization of uncertain state-delayed systems," *Int. J. Control*, vol. 74, no. 14, pp. 1447–1455, 2001.
[20] Q. L. Han and K. Q. Gu, "On robust stability of time-delay systems with norm-bounded uncertainty," *IEEE Trans. Automat. Contr.*, vol. 46, pp. 1426–1431, Sept. 2001.
[21] J. H. Kim, "Delay and its time-derivative dependent roubst stability of time-delayed linear systems with uncertainty," *IEEE Trans. Automat. Contr.*, vol. 46, pp. 789–792, May 2001.
[22] Q. L. Han, "New results for delay-dependent stability of linear systems with time-varying delay," *Int. J. Syst. Sci.*, vol. 33, pp. 213–228, 2002.
[23] ——, "Robust stability of uncertain delay-differential systems of neutral type," *Automatica*, vol. 38, no. 4, pp. 719–723, 2002.
[24] D. Yue and S. Won, "An improvement on 'delay and its time-derivative dependent roubst stability of time-delayed linear systems with uncertainty'," *IEEE Trans. Automat. Contr.*, vol. 47, pp. 407–408, Feb. 2002.
[25] S. Boyd, L. E. Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequality in System and Control Theory*. Philadelphia, PA: SIAM, 1994.
[26] J. Hale, *Theory of Functional Differential Equations*. New York: Springer-Verlag, 1977.

Efficient Conversion of Mixed Logical Dynamical Systems Into an Equivalent Piecewise Affine Form

Alberto Bemporad

***Abstract*—For hybrid systems described by switched linear difference equations, linear threshold conditions, automata, and propositional logic conditions, described in mixed logical dynamical form, this note describes two algorithms for transforming such systems into an equivalent piecewise affine form, where equivalent means that for the same initial conditions and input sequences the trajectories of the system are identical. The proposed techniques exploit ideas from mixed-integer programming and multiparametric programming.**

***Index Terms*—Equivalent models, hybrid systems, mixed-integer programming, multiparametric programming, piecewise affine systems.**

I. Introduction

Hybrid systems provide a unified framework for describing processes evolving according to continuous dynamics, discrete dynamics, and logic rules [1]–[5]. The interest in hybrid systems is mainly motivated by the large variety of practical situations where physical processes interact with digital controllers, as for instance in embedded systems. Several modeling formalisms have been developed to describe hybrid systems [6]. Among them are the class of piecewise affine (PWA) systems [7], linear complementarity (LC) systems [8], and mixed logical dynamical (MLD) systems [9]. In particular the language HYSDEL (hybrid systems description language) [10] was developed to obtain MLD models from a high level textual description of the hybrid dynamics. Examples of real-world applications that can be naturally modeled within the MLD framework are reported in [9]–[11].

Each subclass has its own advantages. Although control and state-estimation techniques based on online mixed-integer optimization or offline multiparametric mixed-integer programming were proposed for MLD hybrid models [9], [12], most of the other hybrid techniques require a PWA formulation, such as: stability criteria [13]–[15], synthesis of explicit piecewise affine optimal controllers [16]–[18], and verification of safety properties via reachability analysis [19]–[21]. In addition, simulation of hybrid systems is much easier for PWA systems (evaluation of a PWA function per time step), than for MLD systems (one mixed-integer feasibility test per time step [9]) and LC system (one linear complementarity problem per time step).

In [6] and [22], we showed that MLD, PWA, LC, and other classes of hybrid systems are equivalent. Some of the equivalences were obtained under additional assumptions related to well-posedness (i.e., existence and uniqueness of solution trajectories) and boundedness of (some) system variables. These results are extremely important, as they allow to transfer all the analysis and synthesis tools developed for one particular class to any of the other equivalent subclasses of hybrid systems.

While the transformation of a PWA system into MLD form can be done immediately, for instance by using appropriate "big-M" techniques [9], the reverse transformation from MLD to PWA described in [22] requires the enumeration of all possible combinations of the binary

Manuscript received November 10, 2002; revised October 8, 2003. Recommended by Associate Editor L. E. Holloway. This work was supported in part by the European Project IST-2001-33520.

The author is with the Information Engineering Deptartment, the University of Siena, 53100 Siena, Italy (e-mail: bemporad@dii.unisi.it).

Digital Object Identifier 10.1109/TAC.2004.828315

variables contained in the MLD model. This note provides efficient algorithms that avoid such an enumeration and compute very efficiently the equivalent PWA form of a given MLD system. We believe that the proposed algorithms will extend the use of tools tailored for PWA systems, such piecewise quadratic Lyapunov functions for stability analysis [14], [15] and multiparametric programming for the synthesis of optimal control laws [16], to many real-life nontrivial hybrid problems, as those that can be described in the modeling language HYSDEL. A Matlab implementation of the techniques described in this note is available at http://www.dii.unisi.it/~bemporad/tools.html and are included in the Hybrid Toolbox for Matlab [23].

II. Preliminaries

A. PWA Systems

PWA systems are described by

$$\begin{aligned} x(k+1) &= A_i x(k) + B_i u(k) + f_i \\ y(k) &= C_i x(k) + D_i u(k) + g_i, \qquad \text{for } \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} \in \Omega_i \end{aligned} \tag{1}$$

where $u(k) \in \mathbb{R}^m, x(k) \in \mathbb{R}^n$, and $y(k) \in \mathbb{R}^p$ denote the input, state and output, respectively, at time $k, \Omega_i \triangleq \{ \begin{bmatrix} x \\ u \end{bmatrix} : H_{ix} x + H_{iu} u \leq K_i \}, i = 1, \ldots, s$, are convex (possibly unbounded) polyhedra in the input+state space. $A_i, B_i, C_i, D_i, H_{ix}$, and H_{iu} are real matrices of appropriate dimensions and f_i and g_i are real vectors for all $i = 1, \ldots, s$.

PWA systems have been studied by several authors (see [7], [9], [13], [14], and the references therein) as they form the "simplest" extension of linear systems that can still model nonlinear and nonsmooth processes with arbitrary accuracy and are capable of handling hybrid phenomena, such as linear-threshold events and mode switching.

A PWA system of the form (1) is called *well-posed*, if (1) is uniquely solvable in $x(k+1)$ and $y(k)$, once $x(k)$ and $u(k)$ are specified. A necessary and sufficient condition for the PWA system (1) to be well-posed over $\Omega \triangleq \bigcup_{i=1}^{s} \Omega_i$ is therefore that $x(k+1), y(k)$ are single-valued PWA functions of $x(k), u(k)$. Therefore, typically the sets Ω_i have mutually disjoint interiors, and are often defined as the partition of a convex polyhedral set Ω. In case of discontinuities of the PWA functions over overlapping boundaries of the regions Ω_i, one may ensure well-posedness either by writing some of the inequalities in the form $(H_{ix})^j x + (H_{iu})^j u < K_i^j$, where ^{j} denotes the jth row, or by shrinking some of the inequalities to the form $(H_{ix})^j x + (H_{iu})^j u \leq K_i^j - \epsilon$, where ϵ is a small number (see Remark 1). Although this would be important from a system theoretical point of view, it is not of practical interest from a numerical point of view, as "$<$" cannot be represented in numerical algorithms working in finite precision. In the following we shall neglect this issue for the sake of compactness of notation.

B. MLD Systems

In [9], a class of hybrid systems has been introduced in which logic, dynamics, and constraints are integrated, of the form

$$x(k+1) = Ax(k) + B_1 u(k) + B_2 \delta(k) + B_3 z(k) \tag{2a}$$
$$y(k) = Cx(k) + D_1 u(k) + D_2 \delta(k) + D_3 z(k) \tag{2b}$$
$$E_2 \delta(k) + E_3 z(k) \leq E_1 u(k) + E_4 x(k) + E_5 \tag{2c}$$

where $x(k) = \begin{bmatrix} x_c(k) \\ x_\ell(k) \end{bmatrix}$ is the state vector, $x_c(k) \in \mathbb{R}^{n_c}$, and $x_\ell(k) \in \{0,1\}^{n_\ell}$, $y(k) = \begin{bmatrix} y_c(k) \\ y_\ell(k) \end{bmatrix} \in \mathbb{R}^{p_c} \times \{0,1\}^{p_\ell}$ is the output vector $u(k) = \begin{bmatrix} u_c(k) \\ u_\ell(k) \end{bmatrix} \in \mathbb{R}^{m_c} \times \{0,1\}^{m_\ell}$ is the input vector, $z(k) \in \mathbb{R}^{r_c}$ and $\delta(k) \in \{0,1\}^{r_\ell}$ are auxiliary variables, A, B_i, C, D_i, and E_i denote real constant matrices, E_5 is a real vector, $n_c > 0$, and $p_c, m_c, r_c, n_\ell, p_\ell, m_\ell, r_\ell \geq 0$. Without loss of generality, we assumed that the continuous components of a mixed-integer vector are always the first. Inequalities (2c) must be interpreted componentwise. Systems that can be described by model (2) are called MLD systems. Contrary to [9], we allow here that the input vector $u(k)$ and state vector $x(k)$ may have unbounded components.

The MLD system (2) is called *completely well-posed* if $\delta(k)$ and $z(k)$ are uniquely defined by (2c) in their domain, once $x(k)$ and $u(k)$ are assigned [9]. From (2a)–(2b) this implies that also $x(k+1), y(k)$ are uniquely defined functions of $x(k), u(k)$.

The MLD formalism allows specifying the evolution of continuous variables through linear dynamic equations, of discrete variables through propositional logic statements and automata, and the mutual interaction between the two. The key idea of the approach consists of embedding the logic part in the state equations by transforming Boolean variables into 0-1 integers, and by expressing the relations as mixed-integer linear inequalities (see [9], [10], and the references therein). MLD systems are therefore capable of modeling a broad class of systems, in particular those systems that can be modeled through the hybrid system description language HYSDEL [10].

C. Equivalence of MLD and PWA Systems

Definition 1: Let Σ_1, Σ_2 be hybrid systems in state-space form, with equal state, input, and output dimensions. Σ_1 and Σ_2 are said *equivalent* if whenever their states $x_1(k) = x_2(k)$ and input vectors $u_1(k) = u_2(k)$, then the successor states $x_1(k+1) = x_2(k+1)$ and outputs $y_1(k) = y_2(k)$, where $k \in \mathbb{Z}_{\geq 0}$ and $\mathbb{Z}_{\geq 0}$ is the set of nonnegative integers.

Note that if trajectories are persistent, i.e., defined for all $k \in \mathbb{Z}_{\geq 0}$, then Definition 1 implies that starting from the same initial state and applying the same input sequence, the state and output trajectories of two equivalent systems are identical.

The following proposition has been stated in [22] and is an easy extension of the corresponding result in [9] for piecewise linear (PWL) systems (i.e., PWA systems with $f_i = g_i = 0$).

Proposition 1: Consider a well-posed PWA system (1) and let the set Ω of feasible states and inputs be bounded. Then, there exists an MLD system (2) such that (1) and (2) are equivalent.

Remark 1: As MLD models only allow that nonstrict inequalities are included in (2c), in rewriting a discontinuous PWA system as an MLD model strict inequalities like $x(k) < 0$ must be approximated by $x(k) \leq -\varepsilon$ for some $\varepsilon > 0$ (typically the machine precision), with the assumption that $-\varepsilon < x(k) < 0$ cannot occur due to the finite number of bits used for representing real numbers (no problem exists when the PWA is continuous, where the strict inequality can be equivalently rewritten as nonstrict, or $\varepsilon = 0$); see [6] and [9] for more details. From a strictly theoretical point of view, the inclusion stated in Proposition 1 is, therefore, not exact for discontinuous PWA systems. As discussed previously, one way of circumventing such an inexactness is to allow part of the inequalities in (2c) to be strict, or to avoid strict inequalities in the definition of the PWA dynamics. On the other hand, from a numerical point of view this issue is not relevant. □

The reverse statement of Proposition 1 has been established in [22] under the condition that the MLD system is completely well-posed. A slightly different and more general proof is reported here below, as it will be an essential ingredient of the MLD-to-PWA translation algorithms described in Section III.

Definition 2: The *feasible state+input set* $\Omega \subseteq \mathbb{R}^{n_c} \times \{0,1\}^{n_\ell} \times \mathbb{R}^{m_c} \times \{0,1\}^{m_\ell}$ is the set of states+inputs pairs $(x(k), u(k))$ for which (2c) has a solution for some $\delta(k) \in \mathbb{R}^{r_\ell}$, $z(k) \in \mathbb{R}^{r_c}$.

Proposition 2: For every completely well-posed MLD system (2) there exists an equivalent well-posed PWA system (1), i.e., the feasible state+input set of (2) Ω can be partitioned into a collection of convex polyhedra $\{\Omega_i\}_{i=1}^s$, $\Omega = \bigcup_{i=1}^s \Omega_i$, and there exist 5-tuples $(A_i, B_i, C_i, f_i, g_i)$, $i = 1, \ldots, s$, such that all the trajectories $x(k), u(k), y(k)$ of the MLD system (2) also satisfy (1).

Proof: By well-posedness of system (2), given $x(k)$ and $u(k)$ there exists only one pair $(\delta(k), z(k))$ satisfying (2c), i.e., there exist two functions $F_\ell : \mathbb{R}^{n_c} \times \{0,1\}^{n_\ell} \times \mathbb{R}^{m_c} \times \{0,1\}^{m_\ell} \mapsto \{0,1\}^{r_\ell}$ and $F_c : \mathbb{R}^{n_c} \times \{0,1\}^{n_\ell} \times \mathbb{R}^{m_c} \times \{0,1\}^{m_\ell} \mapsto \mathbb{R}^{r_c}$ such that $\delta(k) = F_\ell(x(k), u(k)), z(k) = F_c(x(k), u(k)), k \in \mathbb{Z}_+$. The idea is to partition the space $\mathbb{R}^{n_c+m_c}$ of continuous states and inputs by grouping in regions Ω_i all

$$\begin{bmatrix} x_c(k) \\ u_c(k) \end{bmatrix} \in \mathbb{R}^{n_c+m_c}$$

corresponding to the same logic state $x_\ell(k) = x_{\ell i} \in \{0,1\}^{n_\ell}$, binary input $u_\ell(k) = u_{\ell i} \in \{0,1\}^{u_\ell}$, and binary vector $\delta(k) = F_\ell(x(k), u(k)) \in \{0,1\}^{r_\ell}$. Let us fix $x_\ell(k) = x_{\ell i}, u_\ell(k) = u_{\ell i}, \delta(k) \equiv \delta_i, i = 1, \ldots, 2^{n_\ell+m_\ell+r_\ell}$. The inequalities (2c) define a polyhedron $\mathcal{P}$ in $\mathbb{R}^{n_c+m_c+r_c}$. Moreover, from (2c) it is possible to extract linear relations that involve $z(k), x_c(k), u_c(k)$ (for instance, pairs of symmetric inequalities that correspond to linear equalities) and, therefore, matrices K_{4i}, K_{1i}, K_{5i} such that

$$\begin{aligned} &z(k) = K_{4i}x_c(k) + K_{1i}u_c(k) + K_{5i} \\ &\forall x(k),\ u(k) : \begin{bmatrix} x_\ell(k) \\ u_\ell(k) \\ F(x(k), u(k)) \end{bmatrix} = \begin{bmatrix} x_{\ell i} \\ u_{\ell i} \\ \delta_i \end{bmatrix} \end{aligned} \tag{3}$$

and that $\mathcal{P} \subset \mathbb{R}^{n_c+m_c+r_c}$ is a polyhedral set of dimension less than or equal to $n_c + m_c$ (for instance if $n_c = 1, m_c = 0, r_c = 1, \mathcal{P}$ would be a segment in $\mathbb{R}^2$). By substituting (3) into (2a)–(2b), and by partitioning

$$A = \begin{bmatrix} A_{cc} & A_{c\ell} \\ A_{\ell c} & A_{\ell\ell} \end{bmatrix} \quad B_1 = \begin{bmatrix} B_{1cc} & B_{1c\ell} \\ B_{1\ell c} & B_{1\ell\ell} \end{bmatrix} \quad B_2 = \begin{bmatrix} B_{2c} \\ B_{2\ell} \end{bmatrix}$$
$$B_3 = \begin{bmatrix} B_{3c} \\ B_{3\ell} \end{bmatrix} \quad C = \begin{bmatrix} C_{cc} & C_{c\ell} \\ C_{\ell c} & C_{\ell\ell} \end{bmatrix} \quad D_1 = \begin{bmatrix} D_{1cc} & D_{1c\ell} \\ D_{1\ell c} & D_{1\ell\ell} \end{bmatrix}$$
$$D_2 = \begin{bmatrix} D_{2c} \\ D_{2\ell} \end{bmatrix} \quad D_3 = \begin{bmatrix} D_{3c} \\ D_{3\ell} \end{bmatrix}$$

we obtain

$$\begin{aligned} &x_c(k+1) \\ &\quad = (A_{cc} + B_{3c}K_{4i})x_c(k) + (B_{1cc} + B_{3c}K_{1i})u_c(k) \\ &\qquad + (B_{2c}\delta_i + B_{3c}K_{5i} + A_{c\ell}x_{\ell i} + B_{1c\ell}u_{\ell i}) \end{aligned} \tag{4a}$$

$$\begin{aligned} &x_\ell(k+1) \\ &\quad = (A_{\ell c} + B_{3\ell}K_{4i})x_c(k) + (B_{1\ell c} + B_{3\ell}K_{1i})u_c(k) \\ &\qquad + (B_{2\ell}\delta_i + B_{3\ell}K_{5i} + A_{\ell\ell}x_{\ell i} + B_{1\ell\ell}u_{\ell i}) \end{aligned} \tag{4b}$$

$$\begin{aligned} &y_c(k) \\ &\quad = (C_{cc} + D_{3c}K_{4i})x_c(k) + (D_{1cc} + D_{3c}K_{4i})u_c(k) \\ &\qquad + (C_{c\ell}x_{\ell i} + D_{1c\ell}u_{\ell i} + D_{3c}K_{5i} + D_{2c}\delta_i) \end{aligned} \tag{4c}$$

$$\begin{aligned} &y_\ell(k) \\ &\quad = (C_{\ell c} + D_{3\ell}K_{4i})x_c(k) + (D_{1\ell c} + D_{3c}K_{4i})u_c(k) \\ &\qquad + (C_{\ell\ell}x_{\ell i} + D_{1\ell\ell}u_{\ell i} + D_{3\ell}K_{5i} + D_{2\ell}\delta_i) \end{aligned} \tag{4d}$$

which, by a suitable choice of A_i, B_i, C_i, f_i, g_i, corresponds to (1) for

$$\begin{aligned} \Omega_i = \Bigg\{ \begin{bmatrix} x_c \\ u_c \end{bmatrix} &: (E_3K_{4i} - E_{4c})x_c + (E_3K_{1i} - E_{1c})u_c \\ &\leq (E_{1\ell}u_{\ell i} - E_2\delta_i - E_3K_{5i} + E_{4\ell}x_{\ell i} + E_5) \Bigg\} \\ &\times \{x_{\ell i}\} \times \{u_{\ell i}\} \end{aligned} \tag{5}$$

where $E_1 = [E_{1c}\ \ E_{1\ell}], E_4 = [E_{4c}\ \ E_{4\ell}]$. ■

Note that the well-posedness of the original MLD system implies that $x_\ell(k+1)$ and $y_\ell(k)$ in (4) are always $\{0,1\}$-valued. Note also that well-posedness of the equivalent PWA systems and, hence, that the affine maps in (4) must coincide on possible overlaps $\Omega_i \cap \Omega_j$, $i \neq j$. We also remark that, in general, the feasible state+input set of (2) $\Omega = \bigcup_{i=1}^s \Omega_i$ may be nonconvex.

Remark 2: It may happen that different combinations (x_ℓ, u_ℓ, δ) lead to the same dynamics $(A_i, B_i, C_i, f_i, g_i)$ and polyhedral cell Ω_i, for instance if the MLD system contains redundant auxiliary binary variables. In this case, duplicates should be eliminated. □

III. Translation Algorithms

For any given MLD system, Proposition 2 is constructive, as it returns the equivalent PWA system. However, it is based on the enumeration of all $2^{n_\ell+m_\ell+r_\ell}$ combinations of binary (x_ℓ, u_ℓ, δ) variables. In general, most combinations lead to empty regions Ω_i in (5), and a method that avoids the enumeration of all possibilities is therefore desirable for computation efficiency. In this note, we propose to avoid such an enumeration by using techniques of multiparametric programming [24], [25] and of mixed-integer linear programming (MILP), for which several efficient solvers exist [26], [27]. The key idea is to determine a feasible combination $(x_{\ell i}, u_{\ell i}, \delta_i)$ via MILP and generate the corresponding polyhedral cell Ω_i and dynamics $(A_i, B_i, C_i, D_i, f_i, g_i)$.

Before proceeding further, we first embed the sets Ω_i in $\mathbb{R}^{n+m}$ by treating the integer vectors x_ℓ, u_ℓ as real-valued vectors during the exploration of the state+input set. In particular, we replace the set $\{0,1\}$ with $[-1/2, 1/2) \cup [1/2, 3/2]$.

Let

$$S = \left\{ \begin{bmatrix} x \\ u \end{bmatrix} : \bar{A} \begin{bmatrix} x \\ u \end{bmatrix} \leq \bar{B} \right\} \subseteq \mathbb{R}^{n+m}$$

be a polyhedral set of states/input pairs $(x(k), u(k))$ over which we look for a PWA system (1) which is equivalent to the given MLD system,[1] and assume there exist a pair

$$(x_1, u_1) = \left(\begin{bmatrix} x_{c1} \\ x_{\ell 1} \end{bmatrix}, \begin{bmatrix} u_{c1} \\ u_{\ell 1} \end{bmatrix} \right)$$

in S such that $x_{\ell 1} \in \{0,1\}^{n_\ell}, u_{\ell 1} \in \{0,1\}^{m_\ell}$, and such that the MLD inequalities (2c) $E_2\delta_1 + E_2z_1 \leq E_1u_1 + E_4x_1 + E_5$ are satisfied for some $\delta_1 \in \{0,1\}^{r_\ell}, z_1 \in \mathbb{R}^{r_c}$. Assuming that S is bounded, in order to have (x_1, u_1) as much inside S as possible we solve the MILP

$$\begin{aligned} (x_1, u_1, \delta_1, z_1, \epsilon_1) = \arg &\max_{x,u,\delta,z,\epsilon} \epsilon \\ \text{subj. to } &E_2\delta + E_2z \leq E_1u + E_4x + E_5 \\ &\bar{A}\begin{bmatrix} x \\ u \end{bmatrix} + e_{m+n}\epsilon \leq \bar{B} \\ &\epsilon \geq 0 \\ &x_\ell \in \{0,1\}^{n_\ell} \quad u_\ell \in \{0,1\}^{m_\ell} \\ &\delta \in \{0,1\}^{r_\ell} \quad z \in \mathbb{R}^{r_c} \end{aligned} \tag{6}$$

[1]Typically, an information is available a priori on an over approximation S of Ω, as generally MLD models are obtained by HYSDEL through the application of the so-called "big-M" technique, which requires the specification of upper and lower bounds on state and input variables [9], [10], [28] (see also the example reported in [29, App. A]).

TABLE I
ALGORITHM 1: MLD-TO-PWA TRANSLATION

1. Given an MLD system (2) and a polyhedron $S \subseteq \mathbb{R}^{n+m}$;
2. $[\mathcal{D}, \mathcal{O}]$ =mld2pwa(S);
3. Eliminate possible duplicates from $[\mathcal{D}, \mathcal{O}]$;
4. For all pairs of regions $\Omega_i,\ \Omega_j \in \mathcal{O}$:
   4.1. If (continuous and logic) dynamics $D_i = D_j$ $(D_i, D_j \in \mathcal{D})$ try to compute $\Omega_i \cup \Omega_j$ according to [31];
5. End.

where $e_{n+m} = [1 \dots 1]' \in \mathbb{R}^{n+m}$. In case S is unbounded, a finite pair (x_1, u_1) can be obtained by adding the linear constraint $\epsilon \leq \epsilon_{\max}$ in (6), and by choosing a finite optimizer (x_1, u_1) in case there are multiple optima.

If the MILP (6) is infeasible, then either S is empty or the PWA dynamics is not defined over S (the latter condition may arise if the MLD system is badly posed, i.e., for all initial states $x(0)$ no input $u(0)$ and no auxiliary vectors $\delta(0), z(0)$ exist that fulfil the MLD constraints (2c) and therefore provide a successor $x(1)$ and output $y(0)$). Otherwise, let δ_1 be the corresponding optimal δ for problem (6), and for the triple $(x_{\ell 1}, u_{\ell 1}, \delta_1)$ compute the corresponding linear expression for z according to (3), the quintuple $(A_1, B_1, C_1, f_1, g_1)$ according to (4), and the corresponding region Ω_1 where such quintuple is valid according to (5), with the assumption that $x_{\ell 1}^i = 0$ is represented by $x_{\ell 1}^i \in [-1/2, 1/2], x_{\ell 1}^i = 1$ is represented by $x_{\ell 1}^i \in [1/2, 3/2]$, where $x_{\ell 1}^i$ denotes the ith component of $x_{\ell 1}$, and similarly for the components $u_{\ell 1}^i$ of the logic part of the input vector $u_{\ell 1}$. Clearly, Ω_1 is a polyhedron in $\mathbb{R}^{n+m}$ and represents the first region of the equivalent PWA system, associated with the combination of logic variables $x_{\ell 1}, u_{\ell 1}, \delta_1$.

In the following sections, we detail two possible algorithms for determining the remaining polyhedral regions and the corresponding dynamics of the equivalent PWA form of the given MLD dynamics.

A. Recursive Algorithm

The nonconvex rest $S \setminus \Omega_1 \subset \mathbb{R}^{n+m}$ is partitioned into convex polyhedral cells $R_j, j = 1, \dots, p_0$, in accordance with the following theorem (cf. ([25, Th. 3]).

Theorem 1: Let $P \subseteq \mathbb{R}^{n+m}$ be a polyhedron, and let $\Theta = \{\begin{bmatrix} x \\ u \end{bmatrix} \in P : G \begin{bmatrix} x \\ u \end{bmatrix} \leq g\}$ be a nonempty polyhedral subset of P, where $G \in \mathbb{R}^{p \times (n+m)}$. Also, let

$$R_j = \left\{ \begin{bmatrix} x \\ u \end{bmatrix} \in P\colon \begin{array}{ll} G^j \begin{bmatrix} x \\ u \end{bmatrix} > g^j & \\ G^h \begin{bmatrix} x \\ u \end{bmatrix} \leq g^h & \forall h < j \end{array} \right\}$$
$$j = 1, \dots, p_0$$

where G^j denotes the jth row of G and g^j denotes the jth entry of g. Then, i) $P = (\cup_{j=1}^{p} R_j) \cup \Theta$; ii) $\Theta \bigcap R_j = \emptyset$ for all j and $R_j \cap R_h = \emptyset$ for all $j \neq h$, i.e., $\{\Theta, R_1, \dots, R_p\}$ is a partition of P.

After partitioning the rest $S \setminus \Omega_1$, we proceed recursively: We choose for each region R_i a new triple $(x_{\ell 1}, u_{\ell 1}, \delta_1)$ by solving the MILP (6) with $\bar{A}, \bar{B}$ such that $\{\begin{bmatrix} x \\ u \end{bmatrix} : \bar{A} \begin{bmatrix} x \\ u \end{bmatrix} \leq \bar{B}\} = R_i$. If the MILP is infeasible, region R_i is discarded. Otherwise, if the optimal solution $(x_{\ell 1}, u_{\ell 1}, \delta_1)$ provides a new combination, (3)–(5) are computed to calculate a new affine dynamics and polyhedral cell Ω_i. Then, Theorem 1 is applied with $P = R_i, \Theta = R_i \cap \Omega_i$ (in order to minimize the number p_0 of regions R_i generated at each recursion, before applying Theorem 1 it is convenient to remove all redundant inequalities from the representation of Θ, which requires the solution of q_i linear programs, where q_i is the number of linear inequalities defining R_i), and the algorithm proceeds recursively.

In order to avoid finding the same combination (x_ℓ, u_ℓ, δ) twice during the recursion, and therefore improve the performance of the algorithm, the "no-good" constraint [30]

$$(x_\ell, u_\ell, \delta) \neq (x_{\ell j}, u_{\ell j}, \delta_j) \tag{7}$$

is imposed when determining a new combination (x_ℓ, u_ℓ, δ), for all combinations $(x_{\ell j}, u_{\ell j}, \delta_j)$ already found during the recursion. It is easy to check that (7) is equivalent to the linear inequality constraint

$$\sum_{i=1}^{n_\ell} \left(2x_{\ell j}^i - 1\right) x_\ell^i + \sum_{i=1}^{m_\ell} \left(2u_{\ell j}^i - 1\right) u_\ell^i + \sum_{i=1}^{r_\ell} \left(2\delta_j^i - 1\right) \delta^i \leq \left(\sum_{i=1}^{n_\ell} x_{\ell j}^i + \sum_{i=1}^{m_\ell} u_{\ell j}^i + \sum_{i=1}^{r_\ell} \delta_j^i \right) - 1 \tag{8}$$

which is included in (6).

At the end of the recursion, in a postprocessing operation, in order to reduce the number of polyhedral cells in the PWA system we check all pairs of regions in the state+input space $\mathbb{R}^{n+m}$ where the affine dynamics (for both the continuous and logic components of the state and output vectors) are the same, and try to compute their union, provided that the union is a convex set [31].

Algorithm 1 reported in Table I and Algorithm 1-1 reported in Table II summarize the recursive procedure for translating a given MLD system into a PWA system that is equivalent in the sense of Definition 1; see Fig. 1.

B. Sequential Algorithm

Rather then following the recursive approach of Algorithm 1, all regions $\Omega_i, i = 1, \dots, s$ can be obtained by repeatedly solving the MILP (6) with the addition of the constraint (8), for all $j = 1, \dots, i-1$. To this end, Algorithm 1-1 should be replaced by Algorithm 1-2 reported in Table III.

We remark that after the PWA form has been generated, an optimized MLD with minimum number of integer variables can be easily obtained by efficiently coding the PWA dynamics as described in [9], for instance by using $\lceil \log_2 s \rceil$ integer variables, where $s \leq 2^{n_\ell + m_\ell + r_\ell}$ is the number of regions in the PWA partition generated by Algorithm 1-1 or Algorithm 1-2.

C. Convergence and Complexity

Since the number of possible combinations of binary variables is bounded by $2^{n_\ell + m_\ell + r_\ell}$, and since at each recursion of Algorithm 1-1 or at each iteration of Algorithm 1-2 the bound is decreased by one because of constraint (8), it is immediate to prove that both algorithms terminate after a finite number of iterations. In particular, the tree associated with the recursive structure of Algorithm 1-1 has a maximum depth of $2^{n_\ell + m_\ell + r_\ell}$. Although in the worst case the complexity of both algorithms is clearly exponential in the number of binary variables, on average it is typically significantly better than the algorithm suggested

TABLE II
ALGORITHM 1-1: RECURSIVE MLD-TO-PWA TRANSLATION

5. Function $[\mathcal{D}, \mathcal{O}]$ =mld2pwa(S);
 5.1. Solve the MILP (6), (8);
 5.2. If the MILP is infeasible, return $[\emptyset, \emptyset]$;
 5.3. Otherwise, let $(x_{\ell j}, u_{\ell j}, \delta_j)$ be the optimal solution to the MILP;
 5.4. Compute the matrices $A_j, B_j, f_j, C_j, D_j, g_j$ of the linear dynamics from (3), (4), and the corresponding region $\Omega_j = \{[\begin{smallmatrix} x \\ u \end{smallmatrix}] : H_{jx}x + H_{ju}u \leq K_j\}$ from (5), where $x^i_{\ell j} = 0$ is replaced by $x^i_{\ell j} \in [-1/2, 1/2]$, $x^i_{\ell j} = 1$ by $x^i_{\ell j} \in [1/2, 3/2]$, and similarly for $u^i_{\ell j}$;
 5.5. Remove redundant inequalities from $\Theta = S \cap \Omega_j$;
 5.6. Generate new polyhedral cells $R_1, \ldots, R_k$ by applying Theorem 1 with $P = S$;
 5.7. For $i = 1, \ldots, p$:
  5.7.1. $[\mathcal{D}_i, \mathcal{O}_i]$=mld2pwa($R_i$);
 5.8. Let $\mathcal{D}_j = \{(A_j, B_j, f_j, C_j, D_j, g_j)\}$, $\mathcal{O}_j = \{(H_{jx}, H_{ju}, K_j)\}$;
 5.9. Return $[\cup_{i=0}^k \mathcal{D}_i, \cup_{i=0}^k \mathcal{O}_i]$.

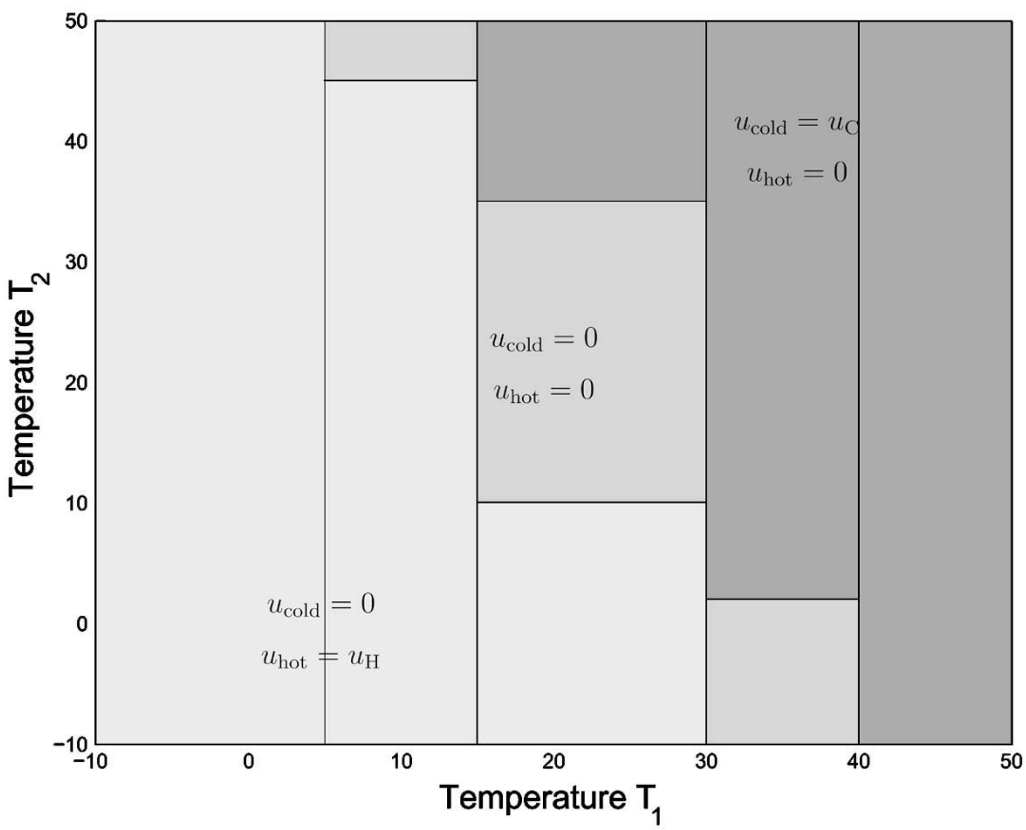

Fig. 1. PWA system equivalent to the MLD model described in the example (the partition does not depend on the input $T_{\rm amb}$). Same level of gray means same value of $u_{\rm cold}, u_{\rm hot}$.

in Proposition 2, where all possible combinations of binary variables are enumerated [22].

Algorithm 1-1 and Algorithm 1-2 have pros and cons, and in general one cannot assess the superiority of one over the other. Algorithm 1-2 enumerates all combinations (x_ℓ, u_ℓ, δ) leading to nonempty cells Ω_i. As observed in Remark 2, different combinations may lead to duplicates, and therefore Algorithm 1-2 would generate all duplicates, while Algorithm 1-1, because of the subpartitioning induced by step 5.6., may avoid generating duplicates. On the other hand, Algorithm 1-1 may execute several infeasible MILPs before stopping the search for new cells.

IV. AN EXAMPLE

Assume there are two bodies B_1, B_2 in a room, let T_1, T_2 be their temperatures, and let $T_{\rm amb}$ be the room temperature (units are omitted here, as the parameters have no particular meaning in this example). We say that B_1 is hot if $T_1 \geq T_{h1}$, cold if $T_1 \leq T_{c1}$, very hot if $T_1 \geq T_{\rm vh1}$, very cold if $T_1 \leq T_{\rm vc2}$, and that B_2 is hot if $T_2 \geq T_{h2}$, cold if $T_1 \leq T_{c2}$, very hot if $T_2 \geq T_{\rm vh2}$, very cold if $T_1 \leq T_{\rm vc2}$. The room is equipped with a heater delivering thermal power $u_{\rm hot}$ and an air conditioning system draining thermal power $u_{\rm cold}$. These are turned on/off according to the following rules: The heater is on if B_1 is cold,

TABLE III
ALGORITHM-1-2: SEQUENTIAL MLD-TO-PWA TRANSLATION

5. Function $[\mathcal{D}, \mathcal{O}]$ =mld2pwa(S);
 5.1. $j \leftarrow 0$;
 5.2. While the MILP (6) is feasible,
  5.2.1. $j \leftarrow j+1$
  5.2.2. Let $(x_{\ell j}, u_{\ell j}, \delta_j)$ be the optimal solution to (6);
  5.2.3. Compute $\mathcal{D}_j = \{(A_j, B_j, f_j, C_j, D_j, g_j)\}$, $\mathcal{O}_j = \{(H_{jx}, H_{ju}, K_j)\}$ as in step 5.4. of Algorithm 1;
  5.2.4. Add constraint (8) in (6);
 5.3. Return $[\cup_{i=0}^{j} \mathcal{D}_i, \cup_{i=0}^{j} \mathcal{O}_i]$.

or B_2 is cold and B_1 is not hot, or B_2 is very cold, but never if B_1 is very hot; the air conditioning is on if B_1 is hot, or B_2 is hot and B_1 is not cold, of B_2 is very hot, but never if B_1 is very cold; otherwise, heater and air conditioning are off.

The dynamical equations of the model are described by the difference equations

$$\frac{T_i(k+1) - T_i(k)}{T_s} = -\alpha_i(T_i(k) - T_{\rm amb}(k)) + k_i(u_{\rm hot}(k) - u_{\rm cold}(k)), \qquad i = 1, 2 \quad (9)$$

where $\alpha_1 = 1, \alpha_2 = 0.5, k_1 = 0.8, k_2 = 0.4, T_s = 0.5$. By introducing binary variables $\delta_{hi}, \delta_{ci}, \delta_{vhi}, \delta_{vci} \in \{0,1\}, i = 1, 2$, the logic relations of the model can be expressed as

$$[\delta_{hi} = 1] \leftrightarrow [T_i \geq T_{hi}] \quad (10a)$$

$$[\delta_{ci} = 1] \leftrightarrow [T_i \leq T_{ci}] \quad (10b)$$

$$[\delta_{vhi} = 1] \leftrightarrow [T_i \geq T_{vhi}] \quad (10c)$$

$$[\delta_{vci} = 1] \leftrightarrow [T_i \leq T_{vci}], \qquad i = 1, 2 \quad (10d)$$

$$u_{\rm hot} = \begin{cases} u_{\rm H}, & \text{if } [\delta_{\rm vh1} = 0] \wedge ([\delta_{c1} = 1] \vee ([\delta_{c2} = 1] \\ & \quad \wedge [\delta_{h1} = 0])) \\ 0, & \text{otherwise} \end{cases}$$

$$u_{\rm cold} = \begin{cases} u_{\rm C}, & \text{if } [\delta_{\rm vc1} = 0] \wedge ([\delta_{h1} = 1] \vee ([\delta_{h2} = 1] \\ & \quad \wedge [\delta_{c1} = 0])) \\ 0, & \text{otherwise} \end{cases} \quad (11)$$

where "$\wedge$" denotes the logic "and," "$\vee$" the logic "or," $T_{h1} = 30, T_{c1} = 15, T_{h2} = 35, T_{c2} = 10, T_{\rm vh1} = 40, T_{\rm vc1} = 5, T_{\rm vh2} = 45, T_{\rm vc2} = 2$, and $u_{\rm H} = 2, u_{\rm C} = 2$ are constant power heating and cooling levels, respectively.

Model (9)–(11) is described in HYSDEL, as reported in Appendix I, and the corresponding MLD model has $n_c = 2$ continuous states $(T_1, T_2), n_\ell = 0$ logic states, $m_c = 1$ continuous input $(T_{\rm amb})$, $m_\ell = 0$ binary inputs, $r_\ell = 10$ auxiliary binary variables [eight thresholds in (10) plus two for the "or" conditions in (11)], two auxiliary variables $(u_{\rm hot}, u_{\rm cold})$, and 32 mixed-integer inequalities. The total number of binary variables is $n_\ell + m_\ell + r_\ell = 10$, which gives a worst-case number of possible regions in the PWA system equal to $2^{10} = 1024$. Let $S = \{[T_1\ T_2\ T_{\rm amb}]' : -10 \leq T_1 \leq 50, -10 \leq T_2 \leq 50, -50 \leq T_{\rm amb} \leq 50\}$ be the set of states+inputs over which the MLD system is defined, and over which we want to obtain an equivalent PWA model. By running Algorithm 1-1, we obtain a PWA equivalent consisting of 9 regions, computed in 1.87 s in Matlab 6.5 on a Pentium III 800 MHz machine using the MILP solver GLPK 4.1 [27]. The same PWA system is obtained in 1.59 s using Algorithm 1-2. The enumerative algorithm of [22] requires 6.75 s (clearly, one should expect that the superiority of Algorithms 1-1, 1-2 increases with the number of binary variables in the MLD system). Further examples are reported in [23] and [29].

V. CONCLUSION

We have described an efficient algorithm for translating hybrid systems expressed as mixed logical dynamical system into an equivalent piecewise affine system, where equivalence means that the same initial conditions and inputs produce identical state and output trajectories. We believe that the result is very useful to apply several techniques available for PWA systems such as stability analysis, verification, simulation, and in particular controller synthesis techniques [16]–[18], to relatively complex hybrid systems composed by linear dynamics, automata, propositional logic, linear threshold conditions, and IF–THEN–ELSE rules, such as those described by the modeling language HYSDEL [10].

APPENDIX I
HYSDEL DESCRIPTION

```
SYSTEM heatcool {
INTERFACE {
  STATE { REAL T1 [-10,50];
      REAL T2 [-10,50]; }
  INPUT {  REAL Tamb [-50,50]; }
PARAMETER {
  REAL Ts = .5; /* sampling time, seconds */
  REAL alpha1 = 1, alpha2 = 0.5;
  REAL k1 = .8, k2 = .4;
  REAL Thot1 = 30, Tcold1 = 15;
  REAL Thot2 = 35, Tcold2 = 10;
  REAL Tvhot1 = 40, Tvcold1 = 5;
  REAL Tvhot2 = 45, Tvcold2 = 2;
  REAL Uc = 2, Uh = 2;
  }
}
IMPLEMENTATION {
  AUX { REAL uhot, ucold;
    BOOL hot1, hot2, cold1, cold2;
    BOOL vhot1, vcold1, vhot2, vcold2;
    }
  AD { hot1 = T1 >= Thot1; hot2 = T2 >= Thot2;
      cold1 = T1 <= Tcold1; cold2 = T2 <= Tcold2;
      vhot1 = T1 >= Tvhot1; vcold1 = T1 <= Tvcold1;
      vhot2 = T2 >= Tvhot2; vcold2 = T2 <= Tvcold2; }
  DA { uhot = {IF ~vhot1 & (cold1 | (cold2 & ~hot1)
          | vcold2) THEN Uh ELSE 0};
      ucold = {IF ~vcold1 & (hot1 | (hot2 & ~cold1)
          | vhot2) THEN Uc ELSE 0}; }
  CONTINUOUS { T1 = T1 + Ts * (-alpha1 * (T1 - Tamb)
            +k1 * (uhot - ucold));
      T2 = T2 + Ts * (-alpha2 * (T2 - Tamb)
          +k2 * (uhot - ucold)); }
  }
}
```

ACKNOWLEDGMENT

The author would like to thank F. D. Torrisi and M. Heemels for stimulating discussions about some of the ideas reported in this note, and the Associate Editor Prof. L. E. Holloway for his careful comments.

REFERENCES

[1] P. J. Antsaklis, "A brief introduction to the theory and applications of hybrid systems," *Proc. IEEE*, vol. 88, pp. 879–886, July 2000.
[2] J. Lygeros, C. Tomlin, and S. Sastry, "Controllers for reachability specifications for hybrid systems," *Automatica*, vol. 35, no. 3, pp. 349–370, 1999.
[3] M. S. Branicky, "Studies in hybrid systems: modeling, analysis, and control," Ph.D. dissertation, Mass. Inst. Technol., Cambridge, MA, 1995.
[4] K. Gokbayrak and C. G. Cassandras, "A hierarchical decomposition method for optimal control of hybrid systems," in *Proc. 38th IEEE Conf. Decision Control*, Phoenix, AZ, Dec. 1999, pp. 1816–1821.
[5] S. Hedlund and A. Rantzer, "Optimal control of hybrid systems," in *Proc. 38th IEEE Conf. on Decision and Control*, Phoenix, AZ, Dec. 1999, pp. 3972–3976.
[6] W. P. M. H. Heemels, B. De Schutter, and A. Bemporad, "Equivalence of hybrid dynamical models," *Automatica*, vol. 37, no. 7, pp. 1085–1091, July 2001.
[7] E. D. Sontag, "Nonlinear regulation: The piecewise linear approach," *IEEE Trans. Automat. Contr.*, vol. AC-26, pp. 346–358, Apr. 1981.
[8] W. P. M. H. Heemels, J. M. Schumacher, and S. Weiland, "Linear complementarity systems," *SIAM J. Appl. Math.*, vol. 60, no. 4, pp. 1234–1269, 2000.
[9] A. Bemporad and M. Morari, "Control of systems integrating logic, dynamics, and constraints," *Automatica*, vol. 35, no. 3, pp. 407–427, Mar. 1999.
[10] F. D. Torrisi and A. Bemporad, "HYSDEL—A tool for generating computational hybrid models," *IEEE Trans. Contr. Syst. Technol.*, vol. 12, pp. 235–249, Mar. 2004.
[11] ——, "Discrete-time hybrid modeling and verification," in *Proc. 41th IEEE Conf. Decision Control*, Orlando, FL, 2001, pp. 2899–2904.
[12] A. Bemporad, F. Borrelli, and M. Morari, "Piecewise linear optimal controllers for hybrid systems," in *Proc. Amer. Control Conf.*, Chicago, IL, June 2000, pp. 1190–1194.
[13] R. DeCarlo, M. Branicky, S. Pettersson, and B. Lennartson, "Perspectives and results on the stability and stabilizability of hybrid systems," *Proc. IEEE*, vol. 88, pp. 1069–1082, July 2000.
[14] M. Johannson and A. Rantzer, "Computation of piece-wise quadratic Lyapunov functions for hybrid systems," *IEEE Trans. Automat. Contr.*, vol. 43, pp. 555–559, Apr. 1998.
[15] G. Ferrari-Trecate, F. A. Cuzzola, D. Mignone, and M. Morari, "Analysis of discrete-time piecewise affine and hybrid systems," *Automatica*, vol. 38, pp. 2139–2146, 2002.
[16] A. Bemporad, F. Borrelli, and M. Morari, "On the optimal control law for linear discrete time hybrid systems," in *Hybrid Systems: Computation and Control*. ser. Lecture Notes in Computer Science, M. Greenstreet and C. Tomlin, Eds. New York: Springer-Verlag, 2002, 2289, pp. 105–119.
[17] E. C. Kerrigan and D. Q. Mayne, "Optimal control of constrained, piecewise affine systems with bounded disturbances," in *Proc. 42th IEEE Conf. Decision Control*, Las Vegas, NV, Dec. 2002, pp. 1552–1557.
[18] D. Q. Mayne and S. Rakovic, "Optimal control of constrained piecewise affine discrete time systems using reverse transformation," in *Proc. 42th IEEE Conf. Decision Control*, Las Vegas, NV, Dec. 2002, pp. 1546–1551.
[19] E. Asarin, O. Bournez, T. Dang, and O. Maler, "Reachability analysis of piecewise-linear dynamical systems," in *Hybrid Systems: Computation and Control*. ser. Lecture Notes in Computer Science, B. Krogh and N. Lynch, Eds. New York: Springer-Verlag, 2000, vol. 1790, pp. 20–31.
[20] A. Chutinan and B. H. Krogh, "Verification of polyhedral invariant hybrid automata using polygonal flow pipe approximations," in *Hybrid Systems: Computation and Control*. ser. Lecture Notes in Computer Science, F. Vaandrager and J. van Schuppen, Eds. New York: Springer-Verlag, 1999, vol. 1569, pp. 76–90.
[21] A. Bemporad, F. D. Torrisi, and M. Morari, "Optimization-based verification and stability characterization of piecewise affine and hybrid systems," in *Hybrid Systems: Computation and Control*. ser. Lecture Notes in Computer Science, B. Krogh and N. Lynch, Eds. New York: Springer-Verlag, 2000, vol. 1790, pp. 45–58.
[22] A. Bemporad, G. Ferrari-Trecate, and M. Morari, "Observability and controllability of piecewise affine and hybrid systems," *IEEE Trans. Automat. Contr.*, vol. 45, pp. 1864–1876, Oct. 2000.
[23] A. Bemporad, *Hybrid Toolbox—User's Guide*, Dec. 2003.
[24] T. Gal, *Postoptimal Analyzes, Parametric Programming, and Related Topics*, 2nd ed. Berlin, Germany: de Gruyter, 1995.
[25] A. Bemporad, M. Morari, V. Dua, and E. N. Pistikopoulos, "The explicit linear quadratic regulator for constrained systems," *Automatica*, vol. 38, no. 1, pp. 3–20, 2002.
[26] *CPLEX 9.0 User Manual*, ILOG, Inc. Gentilly Cedex, France, 2003.
[27] A. Makhorin. (2003) GLPK (GNU Linear Programming Kit) User's Guide. [Online]. Available: http://www.gnu.org/software/glpk/glpk.html
[28] H. P. Williams, *Model Building in Mathematical Programming*, 3rd ed. New York: Wiley, 1993.
[29] A. Bemporad, "An efficient technique for translating mixed logical dynamical systems into piecewise affine systems," in *Proc. 42th IEEE Conf. Decision Control*, 2002, pp. 1970–1975.
[30] J. N. Hooker, *Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction*. New York: Wiley, 2000.
[31] A. Bemporad, K. Fukuda, and F. D. Torrisi, "Convexity recognition of the union of polyhedra," *Comput. Geom.*, vol. 18, pp. 141–154, 2001.

Control of a Planar Underactuated Biped on a Complete Walking Cycle

Ahmed Chemori and Antonio Loría

***Abstract*—We address the problem of stabilizing a planar biped robot on a complete walking cycle. Our approach is based on singling out the three fundamental phases of motion of a biped: single and double-support, separated (sequentially) by an impact "instantaneous" phase. We propose control laws to drive the robot for a finite time during each phase, while ensuring certain robustness *vis-a-vis* the impacts which are treated as external perturbations.**

***Index Terms*—Bipod robots, underactuated mechanical systems.**

I. INTRODUCTION

Biped robots have gained an increasing interest in the last few years. From a control viewpoint the problem of making a biped have a dynamically stable walk (i.e., to follow a reference trajectory or path) is interesting due to the complexity of the model: it consists on a set of constrained differential equations and a discrete-time map which induces discontinuity in the solutions. Furthermore, the structure of the system changes depending on the phase of motion "loosing" or "gaining" degrees of freedom. Many approaches have been proposed in the literature to address the problem of stable dynamic walking. See for instance

Manuscript received November 20, 2002; revised May 27, 2003 and July 22, 2003. Recommended by Associate Editor Z.-O. Jiang.

A. Chemori is with the INPG, Laboratoire d'Automatique de Grenoble, 38402 St. Martin d'Hères, France.

A. Loría is with the C.N.R.S, Laboratoire de Signaux et Systèmes, 91192 Gif sur Yvette, France (e-mail: loria@lss.supelec.fr).

Digital Object Identifier 10.1109/TAC.2004.828314