Model predictive control of discrete-time hybrid systems with discrete inputs [*]

B. Potočnik, G. Mušič and B. Zupančič

Faculty of Electrical Engineering, University of Ljubljana

Tržaška 25, SI-1000 Ljubljana, Slovenia

Abstract

This paper proposes and discusses a model predictive control approach to hybrid systems with discrete inputs only. The algorithm, which takes into account a model of a hybrid system, described as an MLD (*mixed logical dynamical*) system, is based on a performance-driven reachability analysis. The algorithm abstracts the behavior of the hybrid system by building a “tree of evolution”. The nodes of the tree represent the reachable states of a process, and the branches connect two nodes if a transition exists between the corresponding states. A cost-function value is associated with each node, and based on this value the exploration of the tree is driven. As soon as the exploration of the tree is finished, the corresponding input is applied to the system and the procedure is repeated.

I. INTRODUCTION

Hybrid systems are dynamic systems that involve the interaction of continuous dynamics (modeled as differential or difference equations) and discrete dynamics (modeled by finite-state machines). Hybrid systems have been the topic of intense research activity in recent years, primarily because of their importance in applications [1]. Hybrid models are important for a number of problems in system analysis, for example, the computation of trajectories,

[*]ISA Transactions, 2005, Vol. 44 (2), pp. 199-211.

control, stability, and safety analysis.

Several control approaches were proposed in the literature. However, optimal control approaches for hybrid systems are the most promising ones at the moment, and have been thoroughly investigated in recent years [12,17]. Many results can be found in the control-engineering literature. The optimal control of hybrid systems in manufacturing is addressed in [1,13,16], where the authors combine time-driven and event-driven methodologies to solve optimal control problems. An algorithm to optimize the switching sequences for a class of switched linear problems is presented in [20], where the algorithm searches for solutions that are arbitrarily close to the optimal ones. A similar problem is addressed in [3], where the potential for numerical optimization procedures to make optimal sequencing decisions in hybrid dynamic systems is explored. A computational approach based on ideas from dynamic programming and convex optimization is presented in [17]. Piecewise linear quadratic optimal control is addressed in [23], where the use of piecewise quadratic cost functions is extended from a stability analysis of piecewise linear systems. Optimal control based on a reachability analysis, and where the inputs of the system are continuous, is addressed in [6]. A model predictive control technique is presented in [7,5]; this is able to stabilize the MLD system on desired reference trajectories, and online optimization procedures are solved through *mixed integer quadratic programming* (MIQP). Multi-parametric approaches are thoroughly investigated in [10,11,8,2], where both optimal and model predictive control approaches are discussed. The latter, in particular, has been successfully applied to many real problems [21].

In this paper we study and discuss the solution to a model predictive control for linear discrete-time hybrid systems with discrete inputs only, where the system is described as a *mixed logical dynamical* (MLD) system [7]. Many of the control approaches are limited to discrete-time hybrid systems because many complex mathematical issues are removed. In many applications the command variables are intrinsically discrete, either because such a system design is simpler or for other technological reasons. A car with an automatic gear transmission system is one example where the control system influences the car dynamics only through discrete inputs (gears).

The existing MIQP (MILP) based solutions can also be applied to systems with discrete inputs, but only in cases with a small number of discrete inputs and a small number of auxiliary internal system variables (δ and z variables in MLD terminology - see Section II). The reasons are in the computational complexity of the optimization approach. The optimization problem is solved in the extended (x, u, z, δ) space with equality constraints, which increase the size of the problem dramatically [2]. An new approach is proposed in this paper, which is more efficient than MIQP (MILP) approaches and also more general in terms of the allowable cost function.

The paper is organized as follows. In Sec. II we address the *mixed logical dynamical* and *piecewise affine* modeling frameworks, as the model predictive approaches based on these two models will be discussed in the paper. The problems of model predictive control of hybrid systems with discrete inputs and the proposed solution are addressed in Sec. III. The proposed algorithm, applied to the model of a motorboat, is discussed in Sec. IV. The conclusions are given in Sec. V.

II. MIXED LOGICAL DYNAMICAL AND PIECEWISE AFFINE SYSTEMS

In this section *mixed logical dynamical systems* and *piecewise affine systems* defined in discrete time are presented. The model predictive approaches that will be discussed in the following are based on these two modeling formalisms.

A. Mixed logical dynamical (MLD) systems

Hybrid systems are a combination of logic, finite-state machines, linear discrete-time dynamic systems and constraints [7]. The interaction between continuous and discrete/logic dynamics is shown in Fig. 1, where both parts are connected through interfaces. The MLD modeling framework is based on the idea of translating logic relations, discrete/logic dynamics, *A/D* (analog to digital (logic)), *D/A* conversion and logic constraints into mixed integer

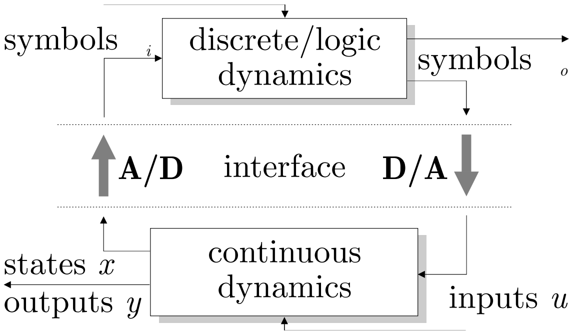

FIG. 1. Hybrid control system – discrete and continuous dynamics interact through interfaces

linear inequalities. These inequalities are combined with the continuous dynamical part, which is described by linear difference equations. The resulting MLD system is described by the following relations:

$$x(k{+}1){=}Ax(k){+}B_1u(k){+}B_2\delta(k){+}B_3z(k) \tag{1a}$$

$$y(k){=}Cx(k){+}D_1u(k){+}D_2\delta(k){+}D_3z(k) \tag{1b}$$

$$E_2\delta(k){+}E_3z(k){\leq}\ E_1u(k){+}E_4x(k){+}E_5, \tag{1c}$$

where $x(k) \in \mathbb{R}^{n_c} \times \{0,1\}^{n_b}$ is a vector of continuous and logic states, $u(k) \in \mathbb{R}^{m_c} \times \{0,1\}^{m_b}$ are the inputs, $y(k) \in \mathbb{R}^{p_c} \times \{0,1\}^{p_b}$ are the outputs and $\delta \in \{0,1\}^{r_b}$ and $z \in \mathbb{R}^{r_c}$ are the logic and continuous auxiliary variables, respectively. The inequalities (Eq. 1c) can also include physical constraints over continuous variables (states and inputs). Given the current state $x(k)$ and the input $u(k)$, the time evolution of (1) is determined by solving $\delta(k)$ and $z(k)$ from (1c), and then updating $x(k+1)$ and $y(k)$ from (1a, 1b). The MLD system (1) is assumed to be well posed if for a given state $x(k)$ and a given input $u(k)$ the inequalities (1c) have a unique solution for $\delta(k)$ and $z(k)$. Because they are so extensive, the details of the translation techniques from logic relations, discrete/logic dynamics, *A/D* (analog to digital (logic)), *D/A* conversion and logic constraints into mixed integer linear inequalities, are not given here. For a more detailed description of MLD form and translation techniques the reader is referred to [7].

B. Piecewise affine (PWA) systems

A discrete-time piecewise affine system is defined by the state-space equations

$$\begin{aligned} x(k+1) &= A_i x(k) + B_i u(k) + f_i \\ y(k) &= C_i x(k) + D_i u(k) + g_i \end{aligned} \qquad \text{for} \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} \in \mathcal{C}_i, \tag{2}$$

where $x(k) \in \mathbb{R}^{n_c} \times \{0,1\}^{n_b}$ is the state, $u(k) \in \mathbb{R}^{m_c} \times \{0,1\}^{m_b}$ is the input, $y(k) \in \mathbb{R}^{p_c} \times \{0,1\}^{p_b}$ is the output, at time instance k. $\{\mathcal{C}_i\}_{i=1,\dots,s}$ is a polyhedral partition of the state-input space defined by a system of inequalities $\{H_x^i x + H_u^i u \leq K^i\}$. $A_i, B_i, f_i, C_i, D_i, g_i, H_x^i, H_u^i$ and K^i are real matrices of suitable dimensions for all i. A PWA system (2) is well posed if $x(k+1)$ and $y(k)$ have a unique solution for a given state $x(k)$ and input $u(k)$, i.e., $\mathcal{C}_i \cap \mathcal{C}_j = \varnothing \ \forall i \neq j$.

For a more detailed description of PWA systems, the reader is referred to [24,15] and the references therein.

III. MODEL PREDICTIVE CONTROL

Predictive control amounts to finding the control sequence $V_0^{H-1} = \{v(0), ..., v(h), ..., v(H-1)\}$ in a horizon H, which transfers the initial state $x(k|k)$ as close as possible to the final state x_f in a horizon time $T = H \cdot T_s$ (T_s is the sampling time), thus minimizing a performance index. Only the first sample of the optimal control sequence V_0^{H-1} is actually applied to the plant at time step k. At time $k+1$ a new sequence is evaluated to replace the previous one.

In [10,11] the author presents the following constrained model predictive problem. Considering the *constrained* PWA *system* (CPWA):

$$\begin{aligned} x(k+1) &= A_i x(k) + B_i u(k) + f_i \\ y(k) &= C_i x(k) + D_i u(k) + g_i \end{aligned} \quad \text{for} \begin{bmatrix} x(k) \\ u(k) \end{bmatrix} \in \tilde{\mathcal{C}}_i = \{\tilde{H}_x^i x(k) + \tilde{H}_u^i u(k) \leq \tilde{K}^i\}, \tag{3}$$

and the model predictive control approach, we can formulate the following model predictive problem:

$$J^*(x(k|k)) \triangleq J(x(k|k), V_0^{H-1}) = \min_{V_0^{H-1}} \|x(k+H|k)-x_f\|_P^p + \\ +\sum_{h=0}^{H-1} \|x(k+h|k)-x_f\|_Q^p + \|v(h)-v_f\|_R^p \tag{4a}$$

$$\begin{aligned} \text{subj. to} \qquad x(k+h+1|k) &= A_i x(k+h|k)+B_i u(k+h|k)+f_i \\ y(k+h|k) &= C_i x(k+h|k)+D_i u(k+h|k)+g_i \\ \text{if } \begin{bmatrix} x(k+h|k) \\ u(k+h|k) \end{bmatrix} &\in \tilde{\mathcal{C}}_i, \end{aligned} \tag{4b}$$

where $\|x\|_Q^p$ represents the p–norm for $p = 1, 2, \infty$. Q, R and P are weighting matrices with the following properties $Q = Q' \geq 0$, $R = R' > 0$, $P \geq 0$ for $p = 2$ and are full column rank for $p = 1, \infty$. $h = 0, ..., H-1$ is the prediction step and $V_0^{H-1} = \{v(0), ..., v(H-1)\}$ is the optimal input sequence defined by the optimization algorithm. Considering the predictive approach, only the first element in the sequence V_0^{H-1} is used, i.e., $u(k) = v(0)$.

The model predictive control problem (4) can be solved by applying the MLD modeling framework [4]. Considering the equivalent MLD system (1) of the PWA system (3) (the equivalence is given in [18]), the problem (4) can be rewritten as:

$$J^*(x(k|k)) \triangleq J(x(k|k), V_0^{H-1}) = \min_{V_0^{H-1}} \|x(k+H|k)-x_f\|_P^p + \\ +\sum_{h=0}^{H-1} \|x(k+h|k)-x_f\|_Q^p + \|v(h)-v_f\|_R^p \tag{5a}$$

$$\begin{aligned} \text{subj. to } x(k+h+1|k) &= Ax(k+h|k)+B_1 v(h)+B_2\delta(k+h|k)+B_3 z(k+h|k) \\ E_2\delta(k+h|k)+E_3 z(k+h|k) &\leq E_1 v(h)+E_4 x(k+h|k)+E_5. \end{aligned} \tag{5b}$$

A. Multi-parametric approach

In this section the multi-parametric solution to the predictive control problem will be presented. The aim is to express the approach's strengths and weaknesses when dealing with hybrid systems that have discrete inputs.

The problem (5) can be formulated as a *Mixed Integer Quadratic Program* (MIQP) when $p = 2$ norm is used [7], or a *Mixed Integer Linear Program* (MILP) when $p = 1, \infty$ norm is used:

$$\begin{array}{ll} \min\limits_{\mathcal{V}} & \mathcal{V}'H_1\mathcal{V} + \mathcal{V}'H_2x(k|k) + x'(k)H_1x(k|k) + f_1'\mathcal{V} + f_2'x(k|k) + c \\ \text{subj. to} & G\mathcal{V} \leq S + Fx(k|k), \end{array} \tag{6}$$

where $\mathcal{V} = [\Omega^T,\, \Delta^T,\, \Xi^T]^T$ with $\Omega = [v(0)^T, ..., v(H{-}1)^T]^T$ (note that Ω represents the input sequence V_0^{H-1}), $\Delta = [\delta(0)^T, ..., \delta(H{-}1)^T]^T$, $\Xi = [z(0)^T, ..., z(H{-}1)^T]^T$, and H_1, H_2, H_3, f_1, f_2, c are matrices of suitable dimensions. H_1, H_2, H_3 are null matrices if problem (6) is an MILP [4].

Given the value of the current (initial) state $x(k|k)$, the MIQP (or MILP) (6) can be solved to obtain the optimal input sequence $\mathcal{V}$, i.e., $\Omega \Rightarrow V_0^{H-1}$. By applying multi-parametric programming [14,8,10,11], where the current (initial) state $x(k|k)$ is considered as a parameter, the explicit form of the optimal state feedback $u(x(k|k))$ of the problem (6) can be obtained. For $p = 2$ norm the optimization problem is treated as multi-parametric MIQP (mp-MIQP), while for $p = 1, \infty$ norm the optimization problem can be treated as multi-parametric MILP (mp-MILP). The solution to the mp-MILP (mp-MIQP) is the explicitly defined control law of the form [2]:

$$u(x(k|k)) = F_i^0 x(k|k) + G_i^0 \;\text{ if }\; x(k) \in \mathcal{P}_i^0, \tag{7}$$

where P_i^0, $i = 1, ..., N^0$, is a polyhedral partition of the feasible set of the states $x(k|k)$ at time step k, and is different for $p = 1, \infty$ or $p = 2$ norm [10,11].

By looking at the definition of the optimization vector $\mathcal{V}$ in (6) and considering the current state $x(k|k)$ as a parameter, we can see that the mp-MILP (MIQP) approach solves the problem in extended $(x, v(u), \delta, z)$ space, which is numerically very difficult to handle and increases the size of the problem drastically. The reason is two fold. Firstly, the complexity of the optimization problem grows exponentially with the number of binary variables. Therefore, including binary variables δ, which are actually explicitly defined by the MLD system, state x and input v, into the optimization problem is not efficient. Secondly, the continuous variables z actually represent equality constraints that were translated into two sets of inequalities (the consequence of the MLD modeling framework [7]) and which are also explicitly defined by the MLD system, state x and input v. In [2] an efficient algorithm for computing the solution for the $p = 1, \infty$ norm case and in [10,11] for $p = 2$ norm is given. Both approaches solve the problem in extended $(x, v(u))$ space and all the problems presented previously are removed. The algorithms are based on a dynamic programming approach and mp-LP(QP) solvers. The limitation of the multi-parametric approaches mentioned above is that they are restricted to hybrid systems where the states $x(k)$ and the inputs $u(k)$ are defined in $x(k) \in \mathbb{R}^{n_c}$ and $u(k) = v(0) \in \mathbb{R}^{m_c}$, respectively. In spite of these limitations the approaches were successfully applied to many real problems [21].

When dealing with hybrid systems that also contain binary states or binary inputs, i.e., $x(k) \in \mathbb{R}^{n_c} \times \{0, 1\}^{n_b}$ and $u(k) \in \mathbb{R}^{m_c} \times \{0, 1\}^{m_b}$, the efficient multi-parametric methods mentioned above are not applicable and the original mp-MILP (mp-MIQP) complexity remains. Due to the drastic increase in the size of the problem when solving the problem in extended $(x, v(u), \delta, z)$ space, a different approach has to be used. Such an alternative approach for the case where the hybrid system can be influenced through binary (discrete) inputs only will be discussed in the following.

B. Performance driven approach

The algorithm that will be introduced in the following is based on a performance-driven reachability analysis [22]. The algorithm abstracts the behavior of the hybrid system by building a "tree of evolution". A cost-function value is associated with each node of the tree, and based on this value the exploration of the tree is driven. As soon as the tree exploration is finished, the corresponding input is applied to the system and the procedure is repeated.

1. Complexity of the control problem with discrete inputs

The solution to a control problem at every time step k is the control sequence $V_0^{H-1} = \{v_b(0), \ldots, v_b(H-1)\}$, where $u(k) = v_b(0)$ represents the discrete input to the system at step k. The subscript b is used to emphasize the fact that we are dealing with binary inputs only. If the system has m_b discrete inputs and no continuous inputs, that means $v_b(h) \in \{0,1\}^{m_b}$ and $V_0^{H-1} \in \{0,1\}^{m_b \cdot H}$. Because all the inputs are binary, there are $2^{m_b \cdot H}$ possible combinations for V_0^{H-1}. Hence the optimization problem is NP-hard and the computational time required to solve the problem grows exponentially with the problem size.

2. Optimization using a reachability analysis

As a result of the constraints (1c) all combinations of inputs are not, in general, feasible. One way to rule out infeasible inputs is to use a reachability analysis. Such an approach for hybrid systems with continuous inputs is presented in [6] and [9]. The same idea can also be used with hybrid systems that have discrete inputs, as will be shown in what follows.

Through a reachability analysis it is possible to extract the reachable states of the system; although enumerating all of them would not be effective, as many of them will be far away from the optimal trajectory. Therefore, it is reasonable to combine a reachability analysis with procedures that can detect reachable states not leading to the optimal solution and remove them from the exploration procedure. The whole procedure is a kind of *branch and bound* strategy. The procedure involves the generation of a tree of evolution, as will be described in the following.

3. Reachability analysis

Let $x(k+h|k)$ be the state at step $k+h$ ($h=0,...,H-1$). The reachability analysis computes all the possible states, $x^i(k{+}h{+}1|k)$, that are reachable in the next time step. If the system has m_b discrete inputs, then 2^{m_b} possible next states may exist. However, because of the constraints (1c), only a smaller number of states can actually be reached. The reachable states can be computed using the algorithm for reach-set computation described in [9].

4. Tree of evolution

By exploiting the reachability analysis technique we are able to abstract the possible evolution of the system over a horizon of H steps into a tree of evolution [6] as shown in Fig. 2. The nodes of the tree represent reachable states, and the branches connect two nodes if a transition exists between corresponding states. Each branch has an associated discrete input applied to the system causing the transition. For a given root node $\mathcal{N}_0$, representing the current (initial) state $x(k|k)$, the reachable states $x^i(k+h|k)$ are computed and inserted into the tree as nodes $\mathcal{N}_i$, while the corresponding discrete inputs $v_b^i(h)$ are associated with the corresponding branches connecting two nodes. $i \in \{1,2,...\}$ represents the successive index of the nodes and branches inserted into the tree. To each new node $\mathcal{N}_i$ a cost value J_i is associated. The search for the optimal control sequence is propagated from a new starting node, whose selection is based on the associated cost value J_i. As soon as a new starting node is selected, new reachable states are computed. The construction of the tree of evolution proceeds according to a depth-first strategy until one of the following conditions occurs:

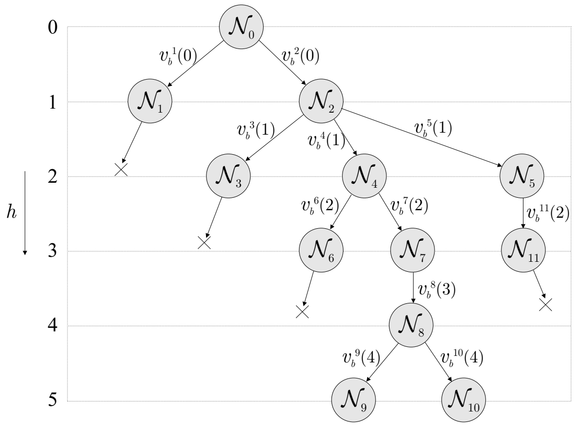

FIG. 2. The tree of evolution

- The step horizon limit is reached ($h = H$).

- The value of the cost function at the current node is greater than the current optimal value ($J_i \geq J_{opt}$, where initially $J_{opt} = \infty$).

A node that satisfies one of the above conditions is labeled as explored. If a node satisfies the first condition, the associated value of the cost function J_i becomes the current optimal value ($J_{opt} = J_i$), and the control sequence V_0^{H-1} that leads from the initial node $\mathcal{N}_0$ to the current node $\mathcal{N}_i$ becomes the current optimizer. The exploration continues until there are no more unexplored nodes in the tree and the temporary control sequence V_0^{H-1} becomes the optimal one, the input $u(k) = v_b(0)$ is applied as an input to the system and the whole optimization procedure is repeated at $k + 1$.

5. Cost function and node selection criterion

The selection of the cost function and the node selection criterion have a great influence on the size of the tree of evolution and, indirectly, on the time efficiency of the control algorithm. The best node-selection criterion is to propagate the tree of evolution in a direction that minimizes the value of the cost function. At the same time the cost value J_i associated with a node is used to detect nodes that are not going to lead to the optimal solution. To achieve that, the cost function must have certain properties that we describe below.

In the case of a classical optimal control problem we typically choose a cost function of the form:

$$J(x(k|k), V_0^{H-1}) = \|x(k{+}H|k){-}x_f\|_P^p + \sum_{h=0}^{H-1} \|x(k{+}h|k){-}x_f\|_Q^p {+} \|v_b(h){-}v_f\|_R^p \tag{8}$$

where $\|x\|_Q^p$ represents the p–norm for $p = 1, 2, \infty$, and where Q, R and P are weighting matrices with the same properties as in (4a).

For a node $\mathcal{V}_i$ inserted into the tree of evolution at the prediction step $h < H$ the associated cost value (based on (8)) is

$$J_i(x(k|k), V_0^{h-1}, h) = \sum_{j=0}^{h} \|x(k{+}j|k){-}x_f\|_Q^p {+} \sum_{j=0}^{h-1} \|v_b(j){-}v_f\|_R^p. \tag{9}$$

It is reasonable to detect the nodes $\mathcal{N}_i$ that do not lead to the optimal solution at the step instance $h < H$ (H is horizon) by comparing $J_i(h)$ to J_{opt}. When the cost value becomes greater than the current optimal one ($J_i \geq J_{opt}$) we want to ensure that by continuing the exploration no better solution than the current one can be found. To achieve this the cost function (9) has to be monotonically increasing with the prediction step. The cost function (8), (9) fulfils the mentioned condition for $p = 1, 2, \infty$ norm, i.e.,

$$J_i(x(k|k), V_0^{h}, h+1) - J_i(x(k|k), V_0^{h-1}, h) = \tag{10a}$$

$$= \sum_{j=0}^{h+1} \|x(k{+}j|k){-}x_f\|_Q^p {+} \sum_{j=0}^{h} \|v_b(j){-}v_f\|_R^p - \sum_{j=0}^{h} \|x(k{+}j|k){-}x_f\|_Q^p {-} \sum_{j=0}^{h-1} \|v_b(j){-}v_f\|_R^p = \tag{10b}$$

$$= \|x(k{+}h{+}1|k){-}x_f\|_Q^p + \|v_b(h){-}v_f\|_R^p \geq 0 \qquad \text{for } h = 1, ..., H-2. \tag{10c}$$

and at the horizon, the corresponding difference is $\|x(k{+}H|k){-}x_f\|_P^p {+} \|v_b(H{-}1){-}v_f\|_R^p \geq 0$.

In the case of a time optimal control problem we must choose a different cost function. This may be of the form:

$$J_i(x(k|k), V_0^{h-1}, h) = f\big(x(k|k), V_0^{h-1}\big) + g(h), \qquad h = 1, ..., H, \tag{11}$$

where $f\big(x(k|k), V_0^{h-1}\big)$ is a rewarding function (the "quality" of the control) with the following property

$$f\big(x(k|k), V_0^{h+1}\big) - f\big(x(k|k), V_0^{h}\big) \leq 0, \tag{12}$$

while $g(h)$ is a penalty function (elapsed time) with the following property

$$g(h+1) - g(h) > 0. \tag{13}$$

To maintain the requirement for the cost function (11) to be monotonically increasing with the prediction step we must choose such a cost function ($f(\cdot)$ and $g(\cdot)$) that

$$J_i(x(k|k), V_0^h, h+1) - J_i(x(k+h|k), V_0^{h-1}, h) \geq 0 \tag{14a}$$

$$\text{i.e. } \big(f(x(k|k), V_0^h) + g(h+1)\big) - \big(f(x(k|k, V_0^{h-1})) + g(h)\big) \geq 0. \tag{14b}$$

IV. EXAMPLE: DISCRETELY OPERATED MOTORBOAT

The presented model predictive control approach was applied to the model of a motorboat presented in [19]. The model is a simplified version of the submarine model presented in [25]. The model of the motorboat is nonlinear and is presented as:

$$\begin{bmatrix} \dot{v} \\ \dot{\vartheta} \\ \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} -v + u_m \\ 0.15 \cdot v \cdot u_k \\ v \cdot \cos\vartheta \\ -v \cdot \sin\vartheta \end{bmatrix}, \tag{15}$$

where v is the sailing speed, ϑ is the yaw angle, x and y represent the position of the motorboat, u_m is the force produced by the motor and u_k is the rudder position. The scheme is shown in Fig. 3.

The control system can drive the motorboat using discrete inputs $u_m \in \{0, 1\}$ and $u_k \in \{-10, 0, 10\}$. The possible combinations of the inputs are presented in Table I.

The system can be separated into two parts, the first representing the dynamics and the second representing the position of the motorboat. The *first part* can be represented by the

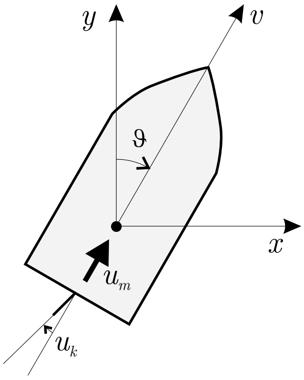

FIG. 3. Motorboat scheme

TABLE I. Possible discrete inputs

Inputs	1	2	3	4	5	6
u_m	0	0	0	1	1	1
u_k	-10	0	10	-10	0	10

three discrete-time *piecewise linear affine* (PWA) models, i.e., considering the fixed input $(u_{k_1} = -10, u_{k_2} = 0, u_{k_3} = -10)$:

$$\begin{bmatrix} v(k+1) \\ \vartheta(k+1) \end{bmatrix} = \begin{bmatrix} 0.8187 & 0 \\ -0.3 & 1 \end{bmatrix} \begin{bmatrix} v(k) \\ \vartheta(k) \end{bmatrix} + \begin{bmatrix} 0.1813 \\ 0 \end{bmatrix} u_m(k) \; ; \; u_k = -10, \tag{16a}$$

$$\begin{bmatrix} v(k+1) \\ \vartheta(k+1) \end{bmatrix} = \begin{bmatrix} 0.8187 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v(k) \\ \vartheta(k) \end{bmatrix} + \begin{bmatrix} 0.1813 \\ 0 \end{bmatrix} u_m(k) \; ; \; u_k = 0, \tag{16b}$$

$$\begin{bmatrix} v(k+1) \\ \vartheta(k+1) \end{bmatrix} = \begin{bmatrix} 0.8187 & 0 \\ 0.3 & 1 \end{bmatrix} \begin{bmatrix} v(k) \\ \vartheta(k) \end{bmatrix} + \begin{bmatrix} 0.1813 \\ 0 \end{bmatrix} u_m(k) \; ; \; u_k = 10, \tag{16c}$$

while the discrete-time representation of the nonlinear *second part* is presented as:

$$\begin{bmatrix} x(k+1) \\ y(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x(k) \\ y(k) \end{bmatrix} + 0.2 \begin{bmatrix} v(k)\cos(\vartheta(k)) \\ -v(k)\sin(\vartheta(k)) \end{bmatrix} \tag{17}$$

using the sample time $T_s = 0.2s$. Both discrete-time models were discretized using the step-invariance method. The first part was modeled as an MLD system using the modeling language HYSDEL (see Appendix). The HYSDEL compiler [26] returned the following MLD system:

$$\begin{bmatrix} v(k+1) \\ \vartheta(k+1) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} z(k) \tag{18a}$$

$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} z(k) \leq \begin{bmatrix} -1.8187 & 0 & 0 & 0 \\ -1.4562 & 0 & 0 & 0 \\ 1.4562 & 0 & 0 & 0 \\ 1.8187 & 0 & 0 & 0 \\ 0 & -10 & 0 & 0 \\ 0 & -10.6 & 0 & 0 \\ 0 & 10.6 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & -10 & 0 \\ 0 & 0 & -10 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & -10 \\ 0 & 0 & 0 & -10.6 \\ 0 & 0 & 0 & 10.6 \\ 0 & 0 & 0 & 10 \\ 0 & 1 & 1 & 1 \\ 0 & -1 & 1 & 0 \\ 0 & -1 & 0 & 1 \\ 0 & 0 & -1 & 1 \end{bmatrix} u(k) + \begin{bmatrix} -0.8187 & 0 \\ 0.8187 & 0 \\ -0.8187 & 0 \\ 0.8187 & 0 \\ 0.3 & -1 \\ 0.3 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & -1 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ -0.3 & 1 \\ 0.3 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} x(k) + \begin{bmatrix} 1.6375 \\ 1.6375 \\ 0 \\ 0 \\ 10 \\ 10.6 \\ 0 \\ 0 \\ 10 \\ 10 \\ 0 \\ 0 \\ 10.6 \\ 10 \\ 0 \\ 0 \\ -1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \tag{18b}$$

where $z(k)$ is an auxiliary continuous variable representing v and ϑ in different operating modes. Matrices $E_{i=1,\ldots,5}$ are defined by the MLD transformation procedure (see [7]). The discrete-time representation of the motorboat example can be represented as shown in Fig. 4.

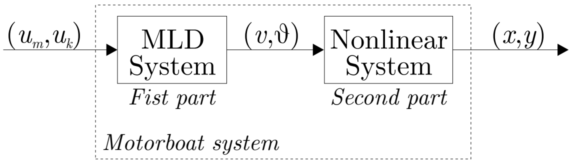

FIG. 4. Discrete time model of a motorboat

The aim of the control is to drive the motorboat as fast as possible according to the path defined in x, y plane, which means we are dealing with a time optimal control problem.

A. Complexity of the predictive control algorithm

The solution to the model predictive control is a sequence V_0^{H-1}. At each time instance, four inputs can influence the system, i.e., one for u_m and three for u_k (see Appendix).

The horizon applied was set to $H = 3$. Considering the number of discrete inputs and the length of the horizon, there exist $2^{4\cdot 3=12} = 4096$ possible combinations of the input vector $V_0^{H-1} \in \{0,1\}^{12}$. Of course not all input combinations are possible, i.e., considering six possible inputs (see Table I) and the horizon $H = 3$, $6^3 = 216$ feasible input combinations exist, and many of them lead to non-optimal solutions. By increasing the length of the horizon the quality of the control does not increase significantly. The reason is the limited range of the values through which the discrete inputs can influence the dynamics of the system. Therefore, it is reasonable to choose shorter horizons, as the longer ones increase the complexity of the control problem exponentially without improving the quality of the control.

B. Cost-function selection

The aim is to drive the motorboat as fast as possible along a given path. Based on the cost function and the node selection criteria for the time optimal control problem introduced earlier, we use the following cost function:

$$J_i = k - K \cdot s(k+h|k) + D_1|x_r - x(k+h|k)| + D_2|y_r - y(k+h|k)|, \tag{19}$$

where h represents the current step, K is a parameter whose properties will be explained later, s is the sailed path, D_1 and D_2 are parameters weighting the deviations from the given path. According to (14) the cost function (19) has to be monotonically increasing

$$\begin{aligned} &J\big(x(k{+}h{+}1|k), y(k{+}h{+}1|k), h{+}1\big) - J\big(x(k{+}h|k), y(k{+}h|k), h\big) = \\ &= \Big(h{+}1{-}K{\cdot}s(k{+}h{+}1|k){+}D_1|x_r{-}x(k{+}h{+}1|k)|{+}D_2|y_r{-}y(k{+}h{+}1|k)|\Big) - \\ &\quad - \Big(h - K \cdot s(k{+}h|k){+}D_1|x_r{-}x(k{+}h|k)|{+}D_2|y_r{-}y(k{+}h|k)|\Big) \geq 0. \end{aligned} \tag{20}$$

For the given path, which can be seen in Fig. 6, the parameters D_1 and D_2 were set to $D_1 = 1$ and $D_2 = 0$ or $D_1 = 0$ and $D_2 = 1$ depending on the relative position of the motorboat with respect to the given path. This way we forced the motorboat to sail as close as possible to the given path. Furthermore, the following relation for parameter K is obtained:

$$\begin{aligned}
&J(x(k{+}h{+}1|k), y(k{+}h{+}1|k), h{+}1) - J(x(k{+}h|k), y(k{+}h|k), h) = \\
&\quad = 1 - K\big(s(k+h+1|k) - s(k+h|k)\big) + \\
&\qquad + D_1\big(|x_r - x(k+h+1|k)| - |x_r - x(k+h|k)|\big) \geq 0 \\
&K \leq \frac{1 + D_1\big(|x_r - x(k+h+1|k)| - |x_r - x(k+h|k)|\big)}{s(k+h+1|k) - s(k+h|k)}
\end{aligned} \quad . \tag{21}$$

By considering the worst-case scenario and $Dx \triangleq |x_r - x(k+h+1|k)| - |x_r - x(k+h|k)| = |x_r - x(k+h|k) - \Delta x| - |x_r - x(k+h|k)| \Rightarrow Dx \in [-\Delta x, \Delta x]$, where $\Delta x = T_s \cdot \max(v\cos(\vartheta)) = 0.2 \cdot 1 = 0.2$, and $\Delta s \triangleq s(k+h+1|k) - s(k+h|k) = T_s \cdot \max(v\cos(\vartheta)) = 0.2 \cdot 1 = 0.2$, the parameter K is estimated to be:

$$K \leq \frac{1 - 1 \cdot 0.2}{0.2} = 4. \tag{22}$$

Regarding the node selection criterion, it is reasonable to chose the node that leads to the best current optimal solution, i.e., the node with the smallest associated cost-function value J_i at step h (the influence of step instance h is the same, but the influence of s is greater).

C. Results

During each time sample the tree of evolution was calculated, resulting in an optimal path from the initial position as fast as possible and as close as possible to the given path. Only the first input vector was applied to the system. The average tree size for the horizon $H = 3$ was 52 (compared to the maximum possible tree size of 216). The results of path-tracking are shown in Fig. 5, while the states and inputs are presented in Fig. 6. The simulation shows that the motorboat would sail the given path in 83.6 seconds. The overall solution was computed in 46.5 seconds using MATLAB 6.1 on a Pentium 4, 2.4 GHz machine. The average time needed to calculate the solution during each sampling period was 0.11 seconds, and the maximum time did not exceed the sampling period of $T_s = 0.2$ seconds.

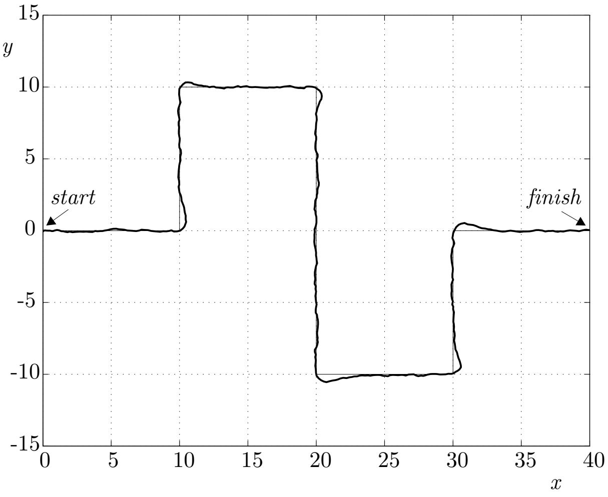

FIG. 5. Motorboat trajectory using model predictive controller

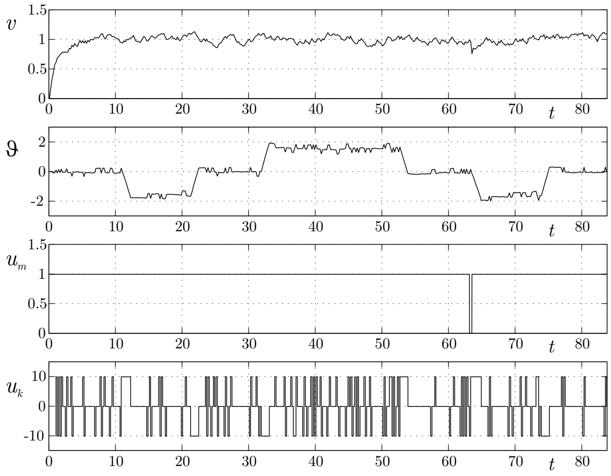

FIG. 6. The states and the inputs of the motorboat system

D. Sub-optimal approach

The speed of the optimization algorithm used for the predictive control is crucial when applying the predictive control in real time. This means that the optimization algorithm has to finish within the sample time T_s. When dealing with discrete (binary) inputs the problem is even more explicit, as the problem's complexity grows exponentially with the number of discrete inputs m_b and the horizon H.

The presented approach builds the tree of evolution, proceeding according to a depth-first strategy. This means that the first sub-optimal solution is obtained as soon as the horizon H is reached ($J_{\text{opt}} = J_i$, initially $J_{\text{opt}} = \infty$). Basically, the algorithm is improving the first sub-optimal solution until the whole tree is explored and the last sub-optimal solution found is declared as the optimal one. If the optimization algorithm does not finish the exploration of the tree within the sample time T_s, the optimization is interrupted and the last sub-optimal solution is applied to the system. The obtained solution could be the optimal one, but we cannot guarantee that.

The sub-optimal approach was applied to the motorboat system with the horizon increased to $H = 4$. Since the number of inputs is $m_b = 4$, then theoretically $2^{4 \cdot 4} = 65536$ possible input sequences $V_0^{H-1} \in \{0,1\}^{16}$ exist. In each step only six different inputs can be applied to the system, and therefore the number of input sequences is reduced to only $6^4 = 1296$.

The average tree size for the horizon $H = 4$ was 102 (compared to the maximum possible tree size of 1296). The results of the path-tracking are shown in Fig. 7, while the states and inputs are presented in Fig. 8. The simulation shows that the motorboat finishes the path in 85.4 seconds. This is also the time needed to complete the simulation. The time needed to calculate the solution during each sampling period was equal to the sampling period $T_s = 0.2$, which is, of course, in compliance with the sub-optimal approach described earlier.

V. CONCLUSIONS

In this paper a model predictive control approach for hybrid systems with discrete inputs is proposed. The optimization problem associated with the prediction is solved by abstracting the behavior of the hybrid system into a "tree of evolution". The main advantage of

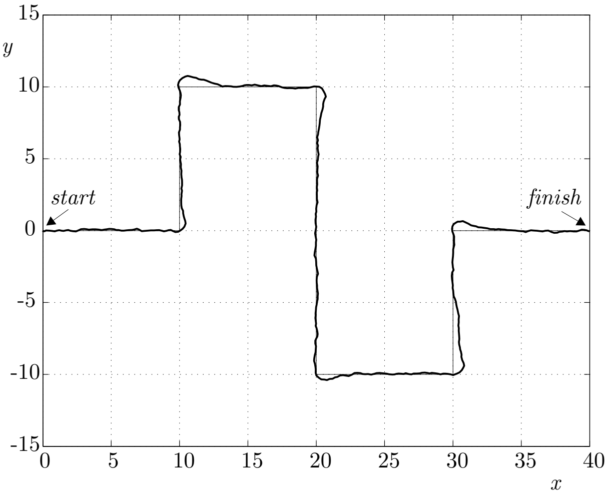

FIG. 7. Motorboat trajectory using sub-optimal model predictive controller

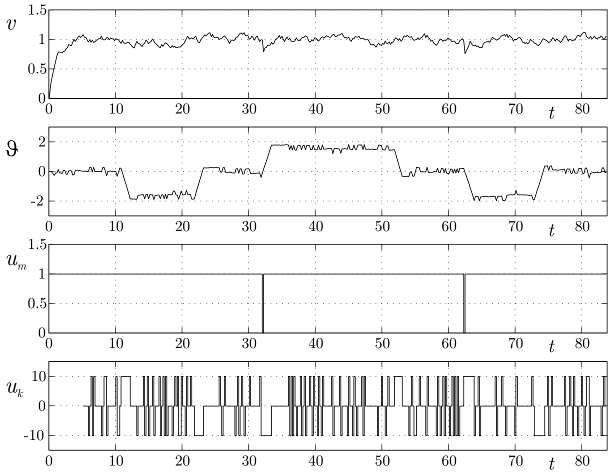

FIG. 8. The states and the inputs of the motorboat system using sub-optimal approach

this approach is that the tree is cut from both sides, top and bottom, which results in a considerable reduction in the size of the tree. The efficiency of the proposed algorithm was demonstrated on a motorboat system. Future research will be devoted to exploring the possibilities of using the presented approach for multi-parametric programming purposes.

APPENDIX A: HYSDEL CODE OF A POWERBOAT

HYSDEL code of a powerboat, where $[um = 1] \leftrightarrow [u_m = 1]$, $[uk1 = 1] \leftrightarrow [u_k = -10]$, $[uk2=1] \leftrightarrow [u_k=0]$, $[uk3=1] \leftrightarrow [u_k=10]$ and consequently $v_b = [um, uk1, uk2, uk3]^T$.

```
SYSTEM motorboat {
 INTERFACE {
  STATE {
   REAL v [0,2];
   REAL th [-10,10];}
  INPUT {
   BOOL um,uk1,uk2,uk3;}
  PARAMETER {
   REAL v1=0.1813;
   REAL v2=0.8187;}}
 IMPLEMENTATION {
  AUX {
   REAL vp;
   REAL thp1,thp2,thp3;    }
  DA {
   vp = {IF um THEN v2*v+v1 ELSE v2*v };
   thp1={IF uk1 THEN th-0.3*v};
   thp2={IF uk2 THEN th+0*v};
   thp3={IF uk3 THEN th+0.3*v};}
  CONTINUOUS {
   v=vp;
   th=thp1+thp2+thp3;}
  MUST {
   (uk1&~uk2&~uk3)|(~uk1&uk2&~uk3)|(~uk1&~uk2&uk3);}}
}
```

REFERENCES

[1] P.J. Antsaklis, editor. Special issue on hybrid systems: Theory and applications, *Proceedings of the IEEE*, volume 88, 2000.

[2] M. Baotić, F.J. Christophersen, and M. Morari. A new algorithm for constrained finite time optimal control of hybrid systems with a linear performance index. Technical report, Automatic Control Laboratory, Swiss Federal Institute of Technology (ETH), Zrich, 2002.

[3] P.I. Barton, J.R. Banga, and S. Galán. Optimization of hybrid discrete/continuous dynamic systems. *Computers and Chemical Engineering*, 24:2171–2182, 2000.

[4] A. Bemporad, F. Borrelli, and M. Morari. On the optimal control law for linear discrete time hybrid systems. In *Hybrid Systems: Computation and Control*, volume 2289 of *Lecture Notes in Computer Science*, pages 105–119. Springer Verlag, 2002.

[5] A. Bemporad, G. Ferrari-Trecate, D. Mignone, M. Morari, and F.D. Torrisi. Model predictive control - ideas for the next generation. In *European Control Conference*, Karlsruhe, Germany, 1999.

[6] A. Bemporad, L. Giovanardi, and F.D. Torrisi. Performance driven reachability analysis for optimal scheduling and control of hybrid systems. In *Proceedings of the 39th IEEE Conference on Decision and Control*, pages 969–974, Sydney, Australia, 2000.

[7] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35(3):407–427, 1999.

[8] A. Bemporad, M. Morari, V. Dua, and E.N. Pistikopoulos. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002.

[9] 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, volume 1790 of *Lecture Notes in Computer Science*, pages 45–58. Springer Verlag, 2000.

[10] F. Borrelli. *Discrete Time Constrained Optimal Control.* PhD thesis, Automatic Control Laboratory, Swiss Federal Institute of Technology (ETH), Zrich, 2002.

[11] F. Borrelli. *Constrained Optimal Control of Linear and Hybrid Systems*, volume 290 of *Lecture Notes in Control and Information Sciences.* Springer Verlag, 2003.

[12] M.S. Branicky, V.S. Borkar, and S.K. Mitter. A unified framework for hybrid control: Model and optimal control theory. *IEEE Transactions on Automatic Control*, 43(1):31–45, 1998.

[13] C.G. Cassandras, D.L. Pepyne, and Y. Wardi. Optimal control of a class of hybrid systems. *IEEE Transactions on Automatic Control*, 46(3):398–415, 2001.

[14] V. Dua, N.A. Bozinis, and E.N. Pistikopoulos. A multiparametric programming approach for mixed-integer quadratic engineering problems. *Computers & Chemical Engineering*, 26(4–5):715–733, 2002.

[15] G. Ferrari-Trecate, F.A. Cuzzola, D. Mignone, and M. Morari. Analysis of discrete-time piecewise affine and hybrid systems. *Automatica*, 38(12):2139–2146, 2002.

[16] K. Gokbayrak and C.G. Cassandras. A hierarchical decomposition method for optimal control of hybrid systems. In *Proceedings of the 38th IEEE Conference on Decision and Control*, pages 1816–1821, Phoenix, Arizona, USA, 1999.

[17] S. Hedlund and A. Rantzer. Optimal control of hybrid systems. In *Proceedings of the 38th IEEE Conference on Decision and Control*, pages 3972–3976, Phoenix, Arizona, USA, 1999.

[18] W.P.M.H. Heemels, B. De Schutter, and A. Bemporad. Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091, 2001.

[19] M. Šikovc. Control of hybrid systems (in Slovene). Master's thesis, Faculty of Electrical Engineering, University of Ljubljana, 1998.

[20] B. Lincoln and A. Rantzer. Optimizing linear system switching. In *Proceedings of the 40th IEEE Conference on Decision and Control*, pages 2063–2068, Orlando, Florida, USA, 2001.

[21] M. Morari, M. Baotic, and F. Borrelli. Hybrid systems modeling and control. *European Journal of Control*, 9(2–3):177–189, 2003.

[22] B. Potočnik, A. Bemporad, F.D. Torrisi, G. Mušič, and B. Zupančič. Hybrid modelling and optimal control of a Multiproduct Batch Plant. *Control Engineering Practice*, 12(9):1127-1137, 2004.

[23] A. Rantzer and M. Johansson. Piecewise linear quadratic optimal control. *IEEE Transactions on Automatic Control*, 45(5):629–637, 2000.

[24] E.D. Sontag. Nonlinear regulation: The piecewise linear approach. *IEEE Transactions on Automatic Control*, 26(2):346–358, 1981.

[25] J.A. Stiver, P. J. Antsaklis, and M.D. Lemmon. Hybrid control system design based on neural invariants. In *Proceedings of the 34th IEEE Conference on Decision and Control*, pages 1455–1460, New Orleans, LA, 1995.

[26] F.D. Torrisi and A. Bemporad. HYSDEL – a tool for generating computational hybrid models. IEEE Transactions on Control Systems Technology – Special Issue on Computer Automated Multi-Paradigm Modeling, in press, 2002.