Hierarchical Synthesis of Hybrid Controllers from Temporal Logic Specifications

Georgios E. Fainekos[1], Antoine Girard[2], and George J. Pappas[3]

[1] Department of Computer and Information Science, Univ. of Pennsylvania, USA
fainekos@cis.upenn.edu

[2] Université Joseph Fourier, Laboratoire de Modélisation et Calcul, Grenoble, France
antoine.girard@imag.fr

[3] Department of Electrical and Systems Engineering, Univ. of Pennsylvania, USA
pappasg@ee.upenn.edu

Abstract. In this paper, the problem of synthesizing a hybrid controller for a specification expressed as a temporal logic formula ϕ is considered. We propose a hierarchical approach which consists of three steps. First, the plant to be controlled is abstracted to a fully actuated system. Using the notion of approximate simulation relation, we design a continuous interface allowing the plant to track the trajectories of its abstraction with a guaranteed precision δ. The second step, which is also the main contribution of this paper, consists in deriving a more robust specification ϕ' from the temporal logic formula ϕ such that given a trajectory satisfying ϕ', any other trajectory remaining within distance δ satisfies ϕ. Third, we design a hybrid controller for the abstraction such that all its trajectories satisfy the robust specification ϕ'. Then, the trajectories of the plant satisfy the original specification. An application to the control of a second order model of a planar robot in a polygonal environment is considered.

1 Introduction

Modern engineering challenges involve controlling complex (possibly nonlinear and/or high order) systems to achieve complicated behaviors. An automated approach to synthesize controllers that are correct by design is very desirable since it can save much of the effort required for the verification of the controlled system. For that purpose, the use of a hierarchical approach is often necessary since trying to handle at once both the complexities of the dynamics and of the specification might lead to intractable computations. A hierarchical control system consists of (at least) two layers. The first layer consists of a coarse (and simple) model of the plant. A controller is designed so that this abstraction meets the specification of the problem. The control law is then refined to the second layer which consists of a detailed model of the plant. Architectures of hierarchical controllers based on the notion of simulation relations have been proposed in [1,2]. More recently, it has been claimed that approaches based on approximate simulation relations [3] would provide more robust control laws while allowing to consider simpler discrete [4] or continuous [5] abstractions for control synthesis.

In this paper, we present such a hierarchical approach for the synthesis of hybrid controllers for specifications expressed as formulas ϕ in the propositional temporal logic over the reals which was introduced in [6]. Following [5], the system is abstracted to a fully actuated system. An interface is designed so that the system is able to track the trajectories of its abstraction with a given guaranteed precision δ. The control objective ϕ is then modified and replaced by a more robust specification ϕ'. The formula ϕ' is such that given a trajectory satisfying ϕ', any trajectory remaining within distance δ satisfies ϕ. This "robustification" procedure is the central step of our approach and constitutes also the main contribution of the paper. It then remains to design a controller for the abstraction such that all its trajectories satisfy the robust specification ϕ'. This is achieved by using one of the computational methods that have recently been developed for the synthesis of hybrid controllers from temporal logic specifications for fully actuated kinematic models [7] or for systems with affine dynamics with drift [8] or for general dynamical systems [9]. Finally, lifting the control law using the hierarchical control architecture, the controlled trajectories of the plant satisfy the original specification ϕ. Throughout the paper, an application to the control of a second order model of a planar robot in a polygonal environment is considered.

2 Problem Description

We consider a continuous time dynamical system

$$\Sigma : \begin{cases} \dot{x}(t) = f(x(t), u(t)), & x(t) \in \mathbb{R}^n,\ x(0) \in X_0 \subseteq \mathbb{R}^n,\ u(t) \in U \subseteq \mathbb{R}^p \\ y(t) = g(x(t)), & y(t) \in \mathbb{R}^k \end{cases} \tag{1}$$

where $x(t)$ is the state of the system, $u(t)$ is the control input and $y(t)$ is the observed output. The goal of this paper is to construct a hybrid controller that generates control inputs $u(t)$ for system Σ so that for the set of initial states X_0, the resulting output $y(t)$ satisfies a formula-specification ϕ in the propositional temporal logic over the positive real line $\mathbb{R}_+$ with the until connective [6]. Let us remark that we design state feedback controllers (i.e. the controller has full knowledge of the state $x(t)$). Thus, the observed output $y(t)$ is used only to specify the desired behavior of the plant.

For the high level planning problem, we consider the existence of a number of regions of interest to the user. Such regions could represent set invariants or sets that must be reached. Let $\Pi = \{\pi_0, \pi_1, \dots, \pi_n\}$ be a finite set of symbols that label these areas. The denotation $[\![\cdot]\!]$ of each symbol in Π represents a subset of $\mathbb{R}^k$, i.e. for any $\pi \in \Pi$ it is $[\![\pi]\!] \subseteq \mathbb{R}^k$. Formally, $[\![\cdot]\!] : \Pi \to \mathcal{P}(\mathbb{R}^k)$, where $\mathcal{P}(\Gamma)$ denotes the powerset of a set Γ.

In order to make apparent the use of the propositional temporal logic for the composition of temporal specifications, we first give an informal description of the traditional and temporal operators. In this paper, we refer to this logic as RTL [10]. The formal syntax and semantics of RTL are presented in Section 4.

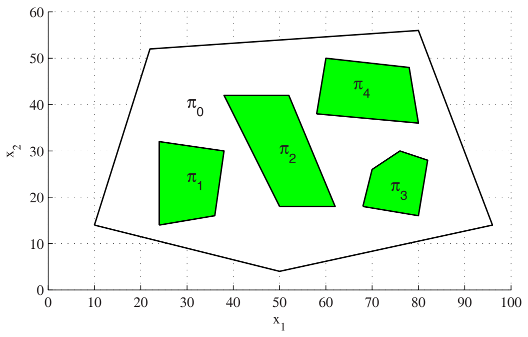

Fig. 1. The simple environment of Example 1. The four regions of interest $\pi_1, \pi_2, \pi_3, \pi_4$ appear in *gray* while the set labeled by π_0 in *white*.

RTL formulas are built over a set of atoms, the set Π in our case, using combinations of the traditional and temporal operators. Traditional logic operators are the *conjunction* ($\wedge$), *disjunction* ($\vee$), *negation* ($\neg$). Some of the temporal operators are *eventually* ($\Diamond$), *always* ($\Box$), *until* ($\mathcal{U}$) and *release* ($\mathcal{R}$). The Temporal Logic of the Reals can describe the usual properties of interest for control problems, i.e. *reachability* ($\Diamond\pi$) and *safety:* ($\Box\pi$ or $\Box\neg\pi$). Beyond the usual properties, RTL can capture sequences of events and certain infinite behaviours. For example:

– **Reachability while avoiding regions:** The formula $\neg(\pi_1 \vee \pi_2 \vee \cdots \vee \pi_n)\mathcal{U}\pi_{n+1}$ expresses the property that eventually π_{n+1} will be true, and until $[\![\pi_{n+1}]\!]$ is reached, we must avoid all unsafe sets $[\![\pi_i]\!]$, $i = 1, \ldots, n$.
– **Sequencing:** The requirement that we must visit $[\![\pi_1]\!]$, $[\![\pi_2]\!]$ and $[\![\pi_3]\!]$ in that order is naturally captured by the formula $\Diamond(\pi_1 \wedge \Diamond(\pi_2 \wedge \Diamond\pi_3))$.
– **Coverage:** Formula $\Diamond\pi_1 \wedge \Diamond\pi_2 \wedge \cdots \wedge \Diamond\pi_m$ reads as the system will eventually reach $[\![\pi_1]\!]$ and eventually $[\![\pi_2]\!]$ and ... eventually $[\![\pi_m]\!]$, requiring the system to eventually visit all regions of interest without any particular ordering.
– **Recurrence (Liveness):** The formula $\Box(\Diamond\pi_1 \wedge \Diamond\pi_2 \wedge \cdots \wedge \Diamond\pi_m)$ requires that the trajectory does whatever the coverage does and, in addition, will force the system to repeat the desired objective infinitely often.

More complicated specifications can be composed from the basic specifications using the logic operators. In order to better explain the different steps in our framework, we consider throughout this paper the following example.

Example 1 (Robot Motion Planning). A typical example of a system such as (1) is a robot which evolves in a planar environment. The state variable $x(t)$ models the internal dynamics of the robot whereas only its position $y(t)$ is observed. In this paper, we will consider a second order model of a planar robot:

$$\Sigma : \begin{cases} \dot{x}_1(t) = x_2(t), \ x_1(t) \in \mathbb{R}^2, \ x_1(0) \in X_{1,0} \\ \dot{x}_2(t) = u(t), \quad x_2(t) \in \mathbb{R}^2, \ x_2(0) = 0, \ \|u(t)\| \le \mu \\ y(t) \ \ = x_1(t), \ y(t) \in \mathbb{R}^2 \end{cases}$$

where $\|\cdot\|$ is the Euclidean norm. The robot is moving in a convex polygonal environment π_0 with four areas of interest denoted by $\pi_1, \pi_2, \pi_3, \pi_4$ (see Fig. 1). Initially, the robot is placed somewhere in the region labeled by π_1 (i.e. $X_{1,0} = [\![\pi_1]\!]$) and its velocity is equal to zero. The robot must accomplish the following task : "Stay always in π_0 and visit area π_2, then area π_3, then area π_4 and, finally, return to and stay in region π_1 while avoiding areas π_2 and π_3," which is captured by the RTL formula:

$$\phi = \Box\pi_0 \wedge \Diamond(\pi_2 \wedge \Diamond(\pi_3 \wedge \Diamond(\pi_4 \wedge (\neg\pi_2 \wedge \neg\pi_3)\mathcal{U}\Box\pi_1))).$$

In this paper, for such spatio-temporal specifications, we provide a computational solution to the following problem.

Problem 1 (RTL Controller Synthesis). Given a system Σ, and an RTL formula ϕ, construct a hybrid controller H for Σ such that the observed trajectories of the closed-loop system satisfy the formula ϕ.

We propose a hierarchical synthesis approach which consists of three ingredients : tracking control using approximate simulation relations [5], robust satisfaction of RTL formulas and hybrid control for motion planning [7,9]. Firstly, Σ is abstracted to a first order fully actuated system:

$$\Sigma' : \ \dot{z}(t) = v(t), \ z(t) \in \mathbb{R}^k, \ z(0) \in Z_0 \subseteq \mathbb{R}^k, \ v(t) \in V \subseteq \mathbb{R}^k \tag{2}$$

where $Z_0 = g(X_0)$. Using the notion of approximate simulation relation, we evaluate the precision δ with which the system Σ is able to track the trajectories of the abstraction Σ' and design a continuous tracking controller that we call interface. Secondly, from the RTL formula ϕ and the precision δ, we derive a more robust formula ϕ' such that if a trajectory $z(t)$ satisfies ϕ', then any trajectory $y(t)$ remaining within distance δ from $z(t)$ satisfies the formula ϕ. Thirdly, we design a hybrid controller H' for the abstraction Σ', so that the trajectories of the closed loop system satisfy the formula ϕ'. Finally, by putting these three ingredients together, as shown in Fig. 2, we design a hybrid controller H solving Problem 1. In the following sections, we detail each step of our approach.

3 Tracking Control Using Approximate Simulation

In this section, we present a framework for tracking control with guaranteed error bounds. It allows to design an interface between the plant Σ and its abstraction Σ' so that Σ is able to track the trajectories of Σ' with a given precision. It is based on the notion of approximate simulation relation [3]. Whereas exact simulation relations requires the observations of two systems to be identical, approximate simulation relations allow them to be different provided their distance remains bounded by some parameter.

Definition 1 (Simulation Relation). *A relation $\mathcal{W} \subseteq \mathbb{R}^k \times \mathbb{R}^n$ is an approximate simulation relation of precision δ of Σ' by Σ if for all $(z_0, x_0) \in \mathcal{W}$,*

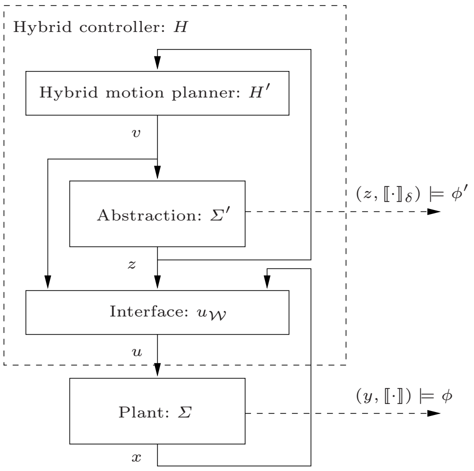
Hybrid controller: H
Hybrid motion planner: H'
v
Abstraction: Σ'
$(z, [\![\cdot]\!]_\delta) \models \phi'$
z
Interface: $u_{\mathcal{W}}$
u
Plant: Σ
$(y, [\![\cdot]\!]) \models \phi$
x

Fig. 2. Hierarchical architecture of the hybrid controller H

1. $\|z_0 - g(x_0)\| \leq \delta$
2. *For all state trajectories $z(t)$ of Σ' such that $z(0) = z_0$ there exists a state trajectory $x(t)$ of Σ such that $x(0) = x_0$ and satisfying*

$$\forall t \geq 0,\ (z(t), x(t)) \in \mathcal{W}.$$

Let us remark that for $\delta = 0$, we recover the notion of exact simulation relation as defined in [11,12]. An interface associated to the approximate simulation relation $\mathcal{W}$ allows to choose the input of Σ so that the states of Σ' and Σ remain in $\mathcal{W}$.

Definition 2 (Interface). *A continuous function $u_{\mathcal{W}} : V \times \mathcal{W} \to U$ is an interface associated to the approximate simulation relation $\mathcal{W}$, if for all $(z_0, x_0) \in \mathcal{W}$, for all trajectories $z(t)$ of Σ' associated to input $v(t)$ and such that $z(0) = z_0$, the trajectory of Σ given by*

$$\dot{x}(t) = f(x(t), u_{\mathcal{W}}(v(t), z(t), x(t))),\ x(0) = x_0$$

satisfies for all $t \geq 0$, $(z(t), x(t)) \in \mathcal{W}$.

Thus, by interconnecting Σ and Σ' through the interface $u_{\mathcal{W}}$ as shown on Fig. 2, Σ tracks the trajectories of the abstraction Σ' with precision δ.

Proposition 1 (Proof in [13]). *Let $x_0 \in X_0$, $z_0 = g(x_0) \in Z_0$ such that $(z_0, x_0) \in \mathcal{W}$, then for all trajectories $z(t)$ of Σ' associated to input $v(t)$ and initial state z_0, the observed trajectory $y(t)$ of Σ given by*

$$\begin{cases} \dot{x}(t) = f(x(t), u_{\mathcal{W}}(v(t), z(t), x(t))),\ x(0) = x_0 \\ y(t) = g(x(t)) \end{cases}$$

satisfies for all $t \geq 0$, $\|y(t) - z(t)\| \leq \delta$.

Let us remark that the choice of the initial state z_0 of the abstraction Σ' is not independent of the initial state x_0 of the system Σ ($z_0 = g(x_0)$).

Remark 1. Usual hierarchical control approaches assume that the plant Σ is simulated by its abstraction Σ'. In this paper, the contrary is assumed. The abstraction Σ' is (approximately) simulated by the plant Σ: the approximate simulation relation is used as a tool for tracking controller design.

The construction of approximate simulation relations can be done effectively using a simulation function [3], that is a positive function bounding the distance between the observations and non-increasing under the parallel evolution of the systems.

Definition 3 (Simulation Function). *Let $\mathcal{V} : \mathbb{R}^k \times \mathbb{R}^n \to \mathbb{R}_+$ be a continuous and piecewise differentiable function. Let $u_{\mathcal{V}} : V \times \mathbb{R}^k \times \mathbb{R}^n \to \mathbb{R}^p$ be a continuous function. $\mathcal{V}$ is a simulation function of Σ' by Σ, and $u_{\mathcal{V}}$ is an associated interface if for all $(z, x) \in \mathbb{R}^k \times \mathbb{R}^n$,*

$$\mathcal{V}(z,x) \geq \|z - g(x)\|^2, \tag{3}$$

$$\sup_{v \in V} \left(\frac{\partial \mathcal{V}(z,x)}{\partial z} v + \frac{\partial \mathcal{V}(z,x)}{\partial x} f(x, u_{\mathcal{V}}(v,z,x)) \right) \leq 0 \tag{4}$$

Then, approximate simulation relations can be defined as level sets of the simulation function.

Theorem 1 (Proof in [13]). *Let the relation $\mathcal{W} \subseteq \mathbb{R}^k \times \mathbb{R}^n$ be given by*

$$\mathcal{W} = \left\{ (z,x) \mid \mathcal{V}(z,x) \leq \delta^2 \right\}.$$

If for all $v \in V$, for all $(z,x) \in \mathcal{W}$, $u_{\mathcal{V}}(v,z,x) \in U$, then $\mathcal{W}$ is an approximate simulation relation of precision δ of Σ' by Σ and $u_{\mathcal{W}} : V \times \mathcal{W} \to U$ given by $u_{\mathcal{W}}(v,z,x) = u_{\mathcal{V}}(v,z,x)$ is an associated interface.

Example 2. Let us go back to our example. The system Σ modelling the dynamics of the robot is abstracted to a system Σ' such as (2) where the set of inputs is $V = \{v \in \mathbb{R}^2 | \ \|v\| \leq \nu\}$ and the set of initial states is $Z_0 = X_{1,0}$. Let $\alpha > 0$, we define the following functions

$$\mathcal{V}(z,x) = \max\left(Q(z,x), 4\nu^2\right) \text{ where } Q(z,x) = \|x_1 - z\|^2 + \alpha\|x_1 - z + 2x_2\|^2,$$
$$u_{\mathcal{V}}(v,z,x) = \tfrac{v}{2} + \tfrac{-1-\alpha}{4\alpha}(x_1 - z) - x_2.$$

First, let us remark that equation (3) clearly holds. If $Q(z,x) \leq 4\nu^2$, then it is clear that equation (4) holds. If $Q(z,x) \geq 4\nu^2$, then

$$\frac{\partial \mathcal{V}}{\partial z} v + \frac{\partial \mathcal{V}}{\partial x_1} x_2 + \frac{\partial \mathcal{V}}{\partial x_2} u_{\mathcal{V}} = 2(x_1 - z) \cdot (x_2 - v)$$
$$+2\alpha(x_1 - z + 2x_2) \cdot (x_2 - v + 2u_{\mathcal{V}}).$$

After the substitution of the expression of $u_{\mathcal{V}}$ and simplification, we arrive to

$$\frac{\partial \mathcal{V}}{\partial z}v + \frac{\partial \mathcal{V}}{\partial x_1}x_2 + \frac{\partial \mathcal{V}}{\partial x_2}u_{\mathcal{V}} = -Q(z,x) - 2(x_1 - z)\cdot v \leq -Q(z,x) + 2\nu\|x_1 - z\|$$

because $\|v\| \leq \nu$. Since $\|x_1 - z\|^2 \leq Q(z,x)$, we have

$$\frac{\partial \mathcal{V}}{\partial z}v + \frac{\partial \mathcal{V}}{\partial x_1}x_2 + \frac{\partial \mathcal{V}}{\partial x_2}u_{\mathcal{V}} \leq -Q(z,x) + 2\nu\sqrt{Q(z,x)} \leq \sqrt{Q(z,x)}(2\nu - \sqrt{Q(z,x)}).$$

Since $Q(z,x) \geq 4\nu^2$, equation (4) holds and $\mathcal{V}$ is a simulation function of Σ' by Σ, and $u_{\mathcal{V}}$ is an associated interface.

Let us define $\mathcal{W} = \{(z,x) |\ \mathcal{V}(z,x) \leq 4\nu^2\}$. Let $v \in V$, $(z,x) \in \mathcal{W}$, let us remark that

$$\begin{aligned}\|u_{\mathcal{V}}(v,z,x)\| &= \left\| \frac{v}{2} + \frac{-1+\alpha}{4\alpha}(x_1 - z) - \frac{1}{2}(x_1 - z + 2x_2)\right\| \\ &\leq \frac{\nu}{2} + \frac{|-1+\alpha|}{4\alpha}\sqrt{\mathcal{V}(z,x)} + \frac{1}{2}\sqrt{\frac{\mathcal{V}(z,x)}{\alpha}} \\ &\leq \frac{\nu}{2}\left(1 + |1 - 1/\alpha| + 2/\sqrt{\alpha}\right).\end{aligned}$$

We assume that the velocity bound ν of the abstraction Σ' has been chosen small enough so that $\frac{\nu}{2}\left(1 + |1 - 1/\alpha| + 2/\sqrt{\alpha}\right) \leq \mu$. Then, Theorem 1 applies and $\mathcal{W}$ is an approximate simulation relation of precision 2ν of Σ' by Σ and an associated interface is given by

$$u_{\mathcal{W}}(v,z,x) = \frac{v}{2} + \frac{-1-\alpha}{4\alpha}(x_1 - z) - x_2.$$

Let the initial state of the abstraction Σ' be chosen so that $z(0) = x_1(0)$, then from Proposition 1, by interconnecting Σ' and Σ through the interface $u_{\mathcal{W}}$, the observed trajectories of system Σ tracks the trajectories of Σ' with precision 2ν.

4 RTL as a Controller Specification Language

Temporal logics are useful for reasoning about the occurrence of events with respect to some time model. In this paper, we advocate the applicability of the propositional temporal logic over the reals (RTL) [6,10] as a natural formalism for a controller specification language. RTL has the same temporal connectives as the Linear Temporal Logic (LTL) [14], but now the underlying time line is the positive real line instead of the natural numbers. In this section, after a brief presentation of RTL, we show how we can derive from an RTL specification ϕ a more robust specification ϕ' such that given a trajectory satisfying ϕ', any other trajectory remaining within distance δ satisfies ϕ. The goal of this "robustification" is then to design a controller for the abstraction Σ' so that it satisfies ϕ'. Then, refining this control law to Σ using the interface presented in the previous section, we can guarantee that the plant satisfies the original specification ϕ.

We first introduce the syntax of RTL formulas in Negation Normal Form (NNF). In NNF, we push the negations inside the subformulas such that the only allowed negation operators appear in front of atoms.

Definition 4 (RTL Syntax in NNF). *For $\pi \in \Pi$, the set Φ_Π of all well formed RTL formulas over Π in NNF is constructed using the grammar*

$$\phi \ ::= \ \pi \mid \neg\pi \mid \phi \vee \phi \mid \phi \wedge \phi \mid \phi \mathcal{U} \phi \mid \phi \mathcal{R} \phi$$

As usual, the boolean constants $\top$ (true) and $\bot$ (false) are defined as $\top = \pi \vee \neg\pi$ and $\top = \pi \wedge \neg\pi$ respectively.

Formally, the semantics of RTL formulas is defined over continuous time boolean signals. Here, we instantiate the definitions of the semantics over abstractions of the output trajectories of the system Σ with respect to Π. Let $(y, [\![\cdot]\!]) \models \phi$ to denote the satisfaction of the RTL formula ϕ over the output trajectory $y(t)$ starting at time $t = 0$ with respect to the atom mapping $[\![\cdot]\!]$. If all the output trajectories $y(t)$ of system Σ driven by a controller H and associated to an initial state in X_0 are such that $(y, [\![\cdot]\!]) \models \phi$, then we write $([\Sigma, H], [\![\cdot]\!]) \models \phi$ and we say that $[\Sigma, H]$ satisfies ϕ. In the following, given any function f from $\mathbb{R}_+$ to some normed space A, we define $f|_t$ for $t \in \mathbb{R}_+$ to be the t time shift of f with definition $f|_t(s) = f(t + s)$ for $s \in \mathbb{R}_+$.

Definition 5 (RTL Semantics). *Let $y(t)$ be a function from $\mathbb{R}_+$ to $\mathbb{R}^k$ and Π be the set of atoms. For $t, s \in \mathbb{R}_+$, the semantics of any formula $\phi \in \Phi_\Pi$ can be recursively defined as*

– $(y, [\![\cdot]\!]) \models \pi$ *iff* $y(0) \in [\![\pi]\!]$.
– $(y, [\![\cdot]\!]) \models \neg\pi$ *iff* $y(0) \notin [\![\pi]\!]$.
– $(y, [\![\cdot]\!]) \models \phi_1 \vee \phi_2$ *if* $(y, [\![\cdot]\!]) \models \phi_1$ *or* $(y, [\![\cdot]\!]) \models \phi_2$.
– $(y, [\![\cdot]\!]) \models \phi_1 \wedge \phi_2$ *if* $(y, [\![\cdot]\!]) \models \phi_1$ *and* $(y, [\![\cdot]\!]) \models \phi_2$.
– $(y, [\![\cdot]\!]) \models \phi_1 \mathcal{U} \phi_2$ *if there exists* $t \geq 0$ *such that* $(y|_t, [\![\cdot]\!]) \models \phi_2$ *and for all* s *with* $0 \leq s \leq t$ *we have* $(y|_s, [\![\cdot]\!]) \models \phi_1$.
– $(y, [\![\cdot]\!]) \models \phi_1 \mathcal{R} \phi_2$ *if for all* $t \geq 0$ *it is* $(y|_t, [\![\cdot]\!]) \models \phi_2$ *or there exists some* s *such that* $0 \leq s \leq t$ *and* $(y|_s, [\![\cdot]\!]) \models \phi_1$.

Note that due to the definition of the negation operator, the duality property of the logic holds and, thus, by using RTL in NNF we do not loose in expressive power. The path formula $\phi_1 \mathcal{U} \phi_2$ intuitively expresses the property that over the trajectory $y(t)$, ϕ_1 is true until ϕ_2 becomes true. Note that here the semantics of until require that ϕ_1 holds when ϕ_2 becomes true. Thus, this is a less expressive version of until than the strict until [10]. Intuitively, the release operator $\phi_1 \mathcal{R} \phi_2$ states that ϕ_2 should always hold, a requirement which is released when ϕ_1 becomes true. Furthermore, we can also derive additional temporal operators such as *eventually* $\Diamond\phi = \top \mathcal{U} \phi$ and *always* $\Box\phi = \bot \mathcal{R} \phi$. The formula $\Diamond\phi$ indicates that over the trajectory $y(t)$ the subformula ϕ becomes eventually true, whereas $\Box\phi$ indicates that ϕ is always true over $y(t)$.

Initially, an RTL formula ϕ is provided as a controller specification for the concrete system Σ. Since we design hybrid control laws for the subsystem Σ', we need to translate the initial RTL specification for Σ to a modified RTL specification ϕ' for Σ'. For this purpose, we introduce a new set of atoms and a new atom's mapping. First, similar to [15], we introduce the notion of δ-contraction for sets in order to define our notion of robustness.

Definition 6 (Contraction, Expansion). *Given a radius $\delta \in \mathbb{R}_+ \cup \{+\infty\}$ and a point α in a normed space A, the δ-ball centered at α is defined as $B_\delta(\alpha) = \{\beta \in A \mid \|\alpha - \beta\| \leq \delta\}$. If $\Gamma \subseteq A$, then $C_\delta(\Gamma) = \{\alpha \in A \mid B_\delta(\alpha) \subseteq \Gamma\}$ is the δ-contraction*[1] *and $B_\delta(\Gamma) = \{a \in A \mid B_\delta(a) \cap \Gamma \neq \emptyset\}$ is the δ-expansion.*

Consider now a new set of atomic propositions Ξ_Π such that $\Xi_\Pi = \{\xi_\alpha \mid \alpha = \pi \text{ or } \alpha = \neg\pi \text{ for } \pi \in \Pi\}$. For a given $\delta \in \mathbb{R}_+$, we define a new map $[\![\cdot]\!]_\delta : \Xi_\Pi \to \mathcal{P}(\mathbb{R}^k)$ based on the map $[\![\cdot]\!]$ as follows:

$$\forall \xi \in \Xi_\Pi, \quad [\![\xi]\!]_\delta =: \begin{cases} C_\delta(\overline{[\![\pi]\!]}) & \text{if } \xi = \xi_{\neg\pi} \\ C_\delta([\![\pi]\!]) & \text{if } \xi = \xi_\pi \end{cases}$$

Here, $\overline{\Gamma}$ denotes the complement of a set Γ. For clarity of the presentation, we define a translation algorithm $\mathbf{rob} : \Phi_\Pi \to \Phi_{\Xi_\Pi}$ which takes as input an RTL formula ϕ in NNF and it returns a formula $\mathbf{rob}(\phi)$ where the occurrences of atomic propositions of π and $\neg\pi$ have been replaced by the members ξ_π and $\xi_{\neg\pi}$ of Ξ_Π respectively.

The following theorem is the connecting link between the specifications satisfied by the abstraction Σ' and its concrete system Σ. Informally, it states that given $\delta \geq 0$ if we δ-expand the sets that must be avoided and δ-contract the sets that must be reached, then we will obtain a δ-robust specification. The latter implies that if a trajectory satisfies the δ-robust specification, then any other trajectory that remains δ-close to the initial one will also satisfy the same non-robust initial specification.

Theorem 2 (Proof in [13]). *Consider a formula $\phi \in \Phi_\Pi$ which is built on a set of atoms Π, a map $[\![\cdot]\!] : \Pi \to \mathcal{P}(\mathbb{R}^k)$, and a number $\delta \in \mathbb{R}_+$, then for all functions $y(t)$ and $z(t)$ from $\mathbb{R}_+$ to $\mathbb{R}^k$ such that for all $t \geq 0$, $\|z(t) - y(t)\| \leq \delta$, the following holds $(z, [\![\cdot]\!]_\delta) \models \mathbf{rob}(\phi) \implies (y, [\![\cdot]\!]) \models \phi$.*

Two remarks are in order here.

Remark 2. When $(z, [\![\cdot]\!]_\delta) \not\models \mathbf{rob}(\phi)$ we cannot conclude that $(y, [\![\cdot]\!]) \not\models \phi$. The only conclusion we can make in this case is that $z(t)$ is not a δ-robust trajectory in the sense of [16]. Potentially, we can gain more information if we define 3-valued semantics [17] for the satisfaction relation of $(z, [\![\cdot]\!]_\delta) \models \mathbf{rob}(\phi)$, but for the scope of this paper Theorem 2 is sufficient.

Remark 3. Theorem 2 does not particularly refer to the output trajectories of systems Σ and Σ'. If we consider both functions z and y to be trajectories of Σ', then Theorem 2 classifies which trajectories of Σ' are δ-robust.

[1] In cases when the δ-contraction of a set must be a polyhedral set, we under-approximate the δ-contraction by a δ-offset (see Sect. 6).

5 From RTL to Hybrid Controllers

It then remains to design a hybrid controller H' for the simpler system Σ' such that its trajectories satisfy an RTL specification ϕ', i.e. $([\Sigma', H'], [\![\cdot]\!]_\delta) \models \phi'$. In previous work, we have proposed two different solutions to this problem [7,9].

The first methodology [7] comprises the following steps. We first create an observation preserving partition of the state space of the system. By observation preserving we mean that all the states that belong to a set in the partition satisfy the same set of labels $\xi \in \Xi_\Pi$. The sets in the partition do not have to be convex and a particular set of labels can be mapped to more than one set in the partition. Then, we can abstract the continuous state space to a Finite State Machine (FSM) where each state corresponds to one set in the partition. Under the assumption of finite variability [18], i.e. within a finite interval of time there can exist only a finite number of changes in the atomic propositions, we can solve the RTL planning problem using Linear Temporal Logic (LTL) planning techniques [19]. The later consists of converting the LTL specification into a Büchi automaton, taking its product with the FSM and, then, finding a path on the product automaton that would satisfy the LTL formula [7]. The other important ingredient of the approach is the existence of simple feedback controllers [20,21] which are defined over polygons. These controllers satisfy the property that if a trajectory in their domain reaches a facet of the polygon, then every other trajectory in the domain of operation does the same. Using the discrete path on the FSM, we can guide the composition of the local feedback controllers and derive a hybrid automaton that generates trajectories that satisfy the initial RTL specification by construction. Note though that in our final construction, we have to make sure that no Zeno behavior occurs in order to satisfy the finite variability assumption.

On the other hand, in [9], we present some preliminary results on the design of hybrid controllers from RTL specifications by directly operating on RTL formulas. The method consists of 3 main steps. First, the RTL formula is converted to an abstract hybrid automaton where the dynamics in each discrete location remain undefined. Then, from each discrete location we extract controller constraints in the form of triplets $c = \{Init, Inv, Goal\}$, where $Init$ is a set of initial conditions, Inv is an invariant set and $Goal$ is a goal set. In the final step, we design controllers that satisfy the constraints c and derive a hybrid automaton whose trajectories satisfy the RTL specification. The main advantages of the new approach involve the possibility of using different control design methodologies for each discrete location of the hybrid automaton and the lack of mandatory partitioning of the environment. Currently, the theory solves the problem for a fragment of RTL, but we conjecture that the recent results by Maler et al. [22] can provide the basis for a solution to the complete RTL.

Both approaches [7,9] constitute valid solutions to the controller design problem that we consider in this paper. Nonetheless in the next section, we present some implementation results for our simple example using the framework presented in [9]. The choice of [9] over the approach in [7] was made for the following reasons. First, because the only modification required to the computational framework of [9] is just an implementation of the C_δ operator. Second, because approaches like [7,8] require the finest partition with respect to the set of atoms Ξ_Π. This implies that for each π in Π, we must create in advance two sets : a δ-contraction $C_\delta([\![\pi]\!])$ and a 2δ-annulus $B_\delta([\![\pi]\!]) \setminus C_\delta([\![\pi]\!])$, and then, take all the appropriate intersections with the contraction and annulus sets of the rest of the atoms. Finally, with the methodology proposed in [9] we can potentially design just one local feedback controller for each part of the specification. On the contrary, the methods in [7,8] would require a larger number of controllers even for the simple cases.

Given the hybrid controller H', the following theorem, which is immediate from Proposition 1 and Theorem 2, states the main result of the paper.

Theorem 3. *Let $\mathcal{W}$ be an approximate simulation relation of precision δ between Σ' and Σ and $u_{\mathcal{W}}$ be the associated interface. Consider a formula $\phi \in \Phi_\Pi$ and define $\phi' = \mathbf{rob}(\phi)$. Let H' be a controller for Σ' and H the associated controller for Σ obtained by interconnection of the elements as shown on Fig. 2. Then, $([\Sigma', H'], [\![\cdot]\!]_\delta) \models \phi'$ implies $([\Sigma, H], [\![\cdot]\!]) \models \phi$.*

6 Implementation and Simulations

In this section, we demonstrate the applicability of our framework by presenting some numerical results. We have implemented the algorithms presented in [9] in MATLAB using the polytope library of the Multi-Parametric Toolbox (MPT) [23]. In the following, we just provide an informal high-level description of our toolbox. The user inputs to the toolbox are : an RTL formula ϕ and the associated map $[\![\cdot]\!]$, the initial conditions x_0 and the maximum acceleration μ of the system Σ as well as the parameter α as described in Example 2 and an integration step ds. Currently, the set valued function $[\![\cdot]\!]$ is restricted to map only to convex or concave polyhedral sets. This is not a fundamental restriction and it was made only for simplifying the implementation of the operator C_δ.

Using the algorithms in [9], we first derive the controller specifications for the design of a hybrid controller H' such that $([\Sigma', H'], [\![\cdot]\!]_\delta) \models \mathbf{rob}(\phi)$. Note that the δ-contraction of a polyhedral set is not always a polyhedral set. In order to maintain a polyhedral description for all the sets, we under-approximate the δ-contraction by the inward δ-offset. Informally, the δ-offset of a polyhedral set is the inward δ-displacement of its facets along the corresponding normal directions. Since the δ-offset is an under-approximation of the δ-contraction, Theorem 2 still holds.

For the generation of the local feedback controllers such that they satisfy a controller specification triplet c (see Sect. 5), we use again a hierarchical approach. If the set Inv is convex, then we can use just one potential field controller [21] that converges inside $Goal$. If on the other hand the set invariant Inv is not convex, then we perform a convex decomposition on the set Inv and we apply the path planning methodology of Conner et al. [21]. By hierarchically composing the resulting controllers, in the sense that a controller that satisfies a triplet c can itself be a hybrid automaton, we obtain the hybrid controller H' for Σ'.

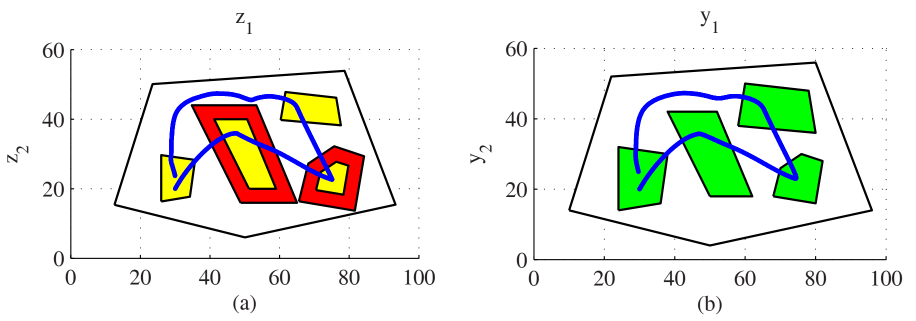

Fig. 3. Simulation results for $\nu = 1$. (a) A trajectory of the kinematic model - The *light gray* polygons denote the 2-contraction of the regions while the *dark gray* polygons the 2-expansion. (b) The corresponding trajectory of the dynamic model.

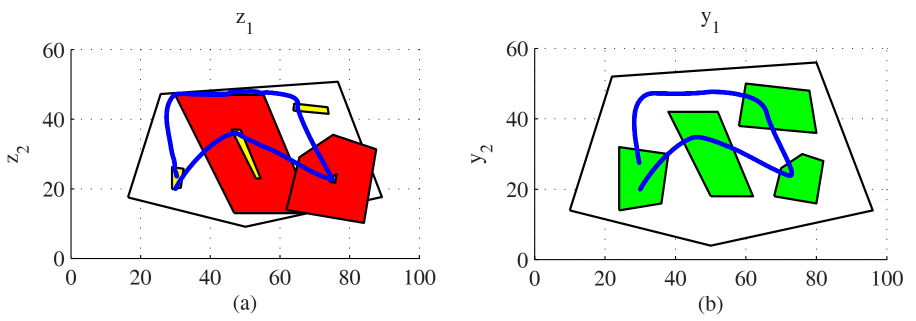

Fig. 4. Simulation results for $\nu = 2.5$. (a) Kinematic model. (b) Dynamic model.

Finally, the hybrid controller H for Σ is obtained from H' by composing with the interface $u_{\mathcal{W}}$ (see Fig. 2).

In the following, we present some numerical results for the robot motion planning example that we consider in this paper. In all the simulations, the parameter α is set to 100 and the initial conditions to $x_0 = [30\ 20\ 0\ 0]^T$. The first simulation (Fig. 3) shows the resulting trajectories for Σ' and Σ for $\nu = 1$ (i.e. the velocity bound for the kinematic model Σ'). It is easy to see that the resulting trajectory of Σ (Fig. 3.b) satisfies the RTL formula ϕ. The total running time for this example on a Pentium 4 at 2.4GHz with 768MB of RAM is 9 sec (including the simulation and the plotting of the graphs in Fig. 3).

Next, in Fig. 4 we present simulation results for $\nu = 2.5$. Note that there barely exists a 2ν-robust trajectory for the kinematic model with respect to the RTL specification. Nevertheless, the trajectory of the dynamic model does satisfy the formula. Also, in Fig. 5.a you can observe that the distance between the two trajectories $z(t)$ and $y(t)$ is always bounded by the precision $2\nu = 5$ of the approximate simulation relation and that this bound is tight. Finally, when we increase ν to 3, then there does not exist a 6-robust trajectory

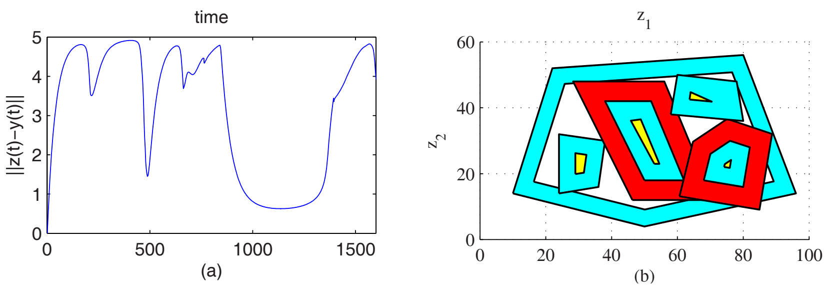

Fig. 5. (a) Distance between trajectories $z(t)$ and $y(t)$ when $\nu = 2.5$. (b) The modified environment when $\nu = 3$. *Medium gray* : original regions, *dark gray* : 6-expanded regions, *light gray* : 6-contracted regions, *white* : 6-contracted workspace π_0.

for the system Σ' (Fig. 5.b). Notice that the modified environment is partitioned into two disconnected components (the *white* workspace is separated by the *dark gray* regions).

7 Conclusions

We have presented a new approach to the hybrid controller synthesis problem from temporal logic specifications. Our proposed hierarchical framework is based on the notion of approximate simulation relations and a new definition of robustness for temporal logic formulas. The hierarchical synthesis approach comprises three basic steps : (i) tracking control using approximate simulation relations [5], (ii) robust satisfaction of RTL formulas and (iii) hybrid control for motion planning [7,9]. We strongly believe that a hierarchical approach can provide a viable solution to a large class of control problems. Future work will concentrate on developing interfaces for other types of systems (i.e. for the unicycle) and applications of the framework to control problems.

Acknowledgments. The authors would like to thank David Conner for providing them with his implementation of the potential field controllers and the reviewers for many useful remarks. This research is partially supported by NSF EHS 0311123, NSF ITR 0121431, and the Army Research Office MURI Grant DAAD 19-02-01-0383.

References

1. Pappas, G.J., Lafferriere, G., Sastry, S.: Hierarchically consistent control systems. IEEE Trans. on Auto. Cont. **45**(6) (2000) 1144–1160
2. Tabuada, P., Pappas, G.J.: Hierarchical trajectory generation for a class of nonlinear systems. Automatica **41**(4) (2005) 701–708
3. Girard, A., Pappas, G.J.: Approximation metrics for discrete and continuous systems. IEEE Trans. Auto. Cont. (2005) Accepted.

4. Tabuada, P.: An approximate simulation approach to symbolic control. (2006) Submitted.
5. Girard, A., Pappas, G.J.: Hierarchical control using approximate simulation relations. In: 45th IEEE Conference on Decision and Control. (2006)
6. Kamp, H.: Tense Logic and the Theory of Linear Order. PhD thesis, University of California (1968)
7. Fainekos, G.E., Kress-Gazit, H., Pappas, G.J.: Hybrid controllers for path planning: A temporal logic approach. In: 44th IEEE CDC and ECC. (2005)
8. Kloetzer, M., Belta, C.: A fully automated framework for control of linear systems from LTL specifications. In: Hybrid Systems: Computation and Control. Volume 3927 of LNCS. Springer (2006) 333–347
9. Fainekos, G.E., Loizou, S.G., Pappas, G.J.: Translating temporal logic to controller specifications. In: 45th IEEE Conference on Decision and Control. (2006)
10. Reynolds, M.: Continuous temporal models. In: the 14th Australian Joint Conference on Artificial Intelligence. Volume 2256 of LNCS., Springer (2001) 414–425
11. Pappas, G.J.: Bisimilar linear systems. Automatica **39**(12) (2003) 2035–2047
12. van der Schaft, A.: Equivalence of dynamical systems by bisimulation. IEEE Trans. Auto. Cont. **49** (2004) 2160–2172
13. Fainekos, G.E., Girard, A., Pappas, G.J.: Hierarchical synthesis of hybrid controllers from temporal logic specifications. Technical report, Dept. of CIS, Univ. of Pennsylvania (2006)
14. Emerson, E.A.: Temporal and modal logic. In van Leeuwen, J., ed.: Handbook of Theoretical Computer Science: Formal Models and Semantics. Volume B., North-Holland Pub. Co./MIT Press (1990) 995–1072
15. Moor, T., Davoren, J.M.: Robust controller synthesis for hybrid systems using modal logic. In: Proceedings of the 4th HSCC. Volume 2034 of LNCS., Springer (2001) 433–446
16. Fainekos, G.E., Pappas, G.J.: Robustness of temporal logic specifications. In: Proceedings of FATES/RV. Volume 4262 of LNCS., Springer (2006) 178–192
17. Fainekos, G.E.: An introduction to multi-valued model checking. Technical Report MS-CIS-05-16, Dept. of CIS, Univ. of Pennsylvania (2005)
18. Barringer, H., Kuiper, R., Pnueli, A.: A really abstract concurrent model and its temporal logic. In: POPL '86: Proceedings of the 13th ACM SIGACT-SIGPLAN symposium on Principles of programming languages, New York, NY, USA, ACM Press (1986) 173–183
19. Giacomo, G.D., Vardi, M.Y.: Automata-theoretic approach to planning for temporally extended goals. In: Proceedings of the 5th ECP. Volume 1809 of LNCS., Springer (1999) 226–238
20. Belta, C., Isler, V., Pappas, G.J.: Discrete abstractions for robot motion planning and control in polygonal environments. IEEE Trans. on Robotics **21**(5) (2005) 864–874
21. Conner, D.C., Rizzi, A., Choset, H.: Composition of local potential functions for global robot control and navigation. In: Proceedings of 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems. (2003) 3546–3551
22. Maler, O., Nickovic, D., Pnueli, A.: From MITL to timed automata. In: Proceedings of FORMATS. Volume 4202 of LNCS., Springer (2006) 274–289
23. Kvasnica, M., Grieder, P., Baotić, M.: Multi-Parametric Toolbox (MPT) (2004) http://control.ee.ethz.ch/ mpt/.