An input–output simulation approach to controlling multi-affine systems for linear temporal logic specifications

Yajuan Suna*, Hai Linb and Ben M. Chena

aDepartment of Electrical and Computer Engineering, National University of Singapore, Singapore;
bDepartment of Electrical Engineering, University of Notre Dame, Notre Dame, IN, USA

(Received 4 February 2012; final version received 26 April 2012)

This article presents an input–output simulation approach to controlling multi-affine systems for linear temporal logic (LTL) specifications, which consists of the following steps. First, the state space is partitioned into rectangles, each of which satisfies atomic LTL propositions. Then, we study the control of multi-affine systems on rectangles, including the control based on the exit sub-region to drive all trajectories starting from a rectangle to exit through a facet and the control to stabilise the multi-affine system towards a desired point. With the proposed controllers, a finitely abstracted transition system is constructed which is shown to be input–output simulated by the rectangular transition system of the multi-affine system. Since the input–output simulation preserves LTL properties, the controller synthesis of the multi-affine system for LTL specifications is achieved by designing a nonblocking supervisor for the abstracted transition system and by implementing the resulting supervisor to the original multi-affine system.

Keywords: automatic synthesis; hybrid systems; multi-affine functions; linear temporal logic

1. Introduction

Due to the integration of embedded computers and communications, high-level specifications like sequencing tasks, system synchronisation and network adaptability naturally emerge in the engineering applications, which goes beyond the traditional control tasks such as stabilisation, output regulation and so on. To address such a challenge, temporal logic, especially linear temporal logic (LTL), has been adopted from computer science to the control and robotics society (Thistle and Wonham 1986; Knight and Passino 1990; Belta et al. 2007; Ulusoy, Smith, Xu, and Belta 2012). Temporal logic can be used to form complicated specifications in a succinct and unambiguous manner. In addition, temporal logic is similar to natural languages and can be easily interpreted by human operators (Eker et al. 2002). Therefore, recent years have seen increasing activities in controller design to satisfy temporal logic specifications.

The basic idea to solve the controller design for LTL specifications is to abstract finite-state transition systems from continuous systems. The resulting finite-state transition systems preserve LTL properties, therefore enabling the controller synthesis through discrete algorithm techniques. Fainekos, Kress-Gazit, and Pappas (2005) studied the control of robots with second-order linear dynamics in a polygonal workspace to fulfil LTL specifications, where the discrete abstraction can be obtained by a triangulation of polygon and vector fields assigned in each triangles drive the produced trajectories to satisfy an LTL formula over the triangles. This work was refined in Tabuada and Pappas (2006) by approaching arbitrary-dimensional discrete-time linear system. It was shown that an equivalent discrete transition system exists for the controllable system with properly chosen observables. Specifically, it builds up the framework for generating the runs of the discrete transition system satisfying the LTL specifications. As opposed to discrete-time linear systems in Tabuada and Pappas (2006) and Kloetzler and Belta (2008) studied the control problem for the LTL specifications with respect to continuous-time linear systems. Based on the results of controlling linear systems on polytopes (Habets and van Schuppen 2004), a computational approach was provided to controller design consisting of polyhedral operator and searches on graphs. Other related work includes the control of a planar robot to achieve sensor-based LTL specifications (Kress-Gazit, Fainekos, and Pappas 2009) and robust LTL specifications (Fainekos, Girard, Kress-Gazit, and Pappas 2009). Although many of these works...
provide valuable inspiration, they are only applicable to linear systems.

In this article, we consider a particular class of nonlinear systems—multi-affine systems. This kind of continuous dynamics is widely used for system modeling in practice, such as the celebrated Ogawa (1993) and Lotka-Volterra (1926) equations, control systems for aircraft and underwater vehicles (Belta 2004) and the models of genetic regulatory networks (Sastry 1999). Formal analysis and control of such systems were investigated in the literature (Belta and Habets 2006; Habets, Kloetzer, and Bela 2006; Kloetzer and Belta 2006; Berman, Halász, and Kumar 2007). Different from their works, we propose an input–output simulation approach so that the controlled multi-affine systems fulfill the LTL specifications. It consists of the following steps. First, we partition the state space into several rectangles consistent with the coordinates. Each rectangle satisfies atomic LTL propositions. Second, we investigate the control of multi-affine systems on rectangles. A control method is provided on the exit region to drive all trajectories starting from a rectangle to exit only through a facet. In addition, we investigate the control of stabilising the system towards a desired point. Third, by using the proposed control methods, a finitely abstracted transition system of the multi-affine system is constructed. Then, we formalise the notion of input–output simulation as a behaviour inclusion between transition systems and show that the abstracted transition system is input–output simulated by the rectangular transition system of the original multi-affine system. Since input–output simulation preserves LTL properties, the controller synthesis for the original multi-affine system to enforce the linear temporal specification is achieved by designing a nonblocking supervisor for the abstracted transition system and by implementing the resulting supervisor to the original multi-affine system.

Compared with the literature, the contributions of this article mainly lie on the following aspects. First, a novel control method is proposed based on the exit region to drive the system to exit through a desired facet. It is shown that this method covers more classes of systems than those are addressed in Belta and Habets (2006) and Habets et al. (2006). Furthermore, we provide a solution for the convergence problem by stabilising the system towards a fixed point. Second, we formalise the notion of input–output simulation. Since this notion requires input equivalence as well as output equivalence, it is stronger than the conventional simulations which need either of them (Milner 1989; Tabuada and Pappas 2006). It is shown that there exists an input–output simulation between the abstracted transition system and the rectangular transition system of the multi-affine system.

Therefore, the multi-affine map of the control input, enforcing LTL specifications with respect to the abstracted transition system, is also implementable for the original multi-affine system. Third, a non-blocking supervisor is designed for the abstracted transition system in order to prevent blocking in the execution and to implement the control strategy effectively. Moreover, multiple feasible paths can be automatically chosen by using this nonblocking supervisor.

The rest of this article is organised as follows. Section 2 gives the preliminary results. Section 3 presents the control of multi-affine systems on rectangles. Section 4 investigates the finitely abstracted transition system of the multi-affine system. The controller synthesis for LTL specifications is studied in Section 5. An illustrative example is presented in Section 6. This article concludes with Section 7.

2. Preliminary results

### 2.1 Multi-affine systems on rectangles

We start by reviewing the notions of multi-affine function and multi-affine control system.

**Definition 2.1** (Belta and Habets 2006): A function $f=(f_1, f_2, \ldots, f_m): \mathbb{R}^n \rightarrow \mathbb{R}^m$ (with $n, m \in \mathbb{N}$) is said to be multi-affine, if every $f_i(x): \mathbb{R}^n \rightarrow \mathbb{R}$, where $x=(x_1, x_2, \ldots, x_n)$ and $i=1, \ldots, m$, is a polynomial in the indeterminates $x_1, x_2, \ldots, x_n$, with the property that the degree of $f_i$ in any of indeterminates $x_1, x_2, \ldots, x_n$ is less or equal to 1. That is, $f_i$ has the form

$$f_i(x) = f_i(x_1, x_2, \ldots, x_n)$$

$$= \sum_{i_1, i_2, \ldots, i_n \in [0, 1]} c_{i_1 i_2 \ldots i_n} (x_1)^{i_1} (x_2)^{i_2} \ldots (x_n)^{i_n}$$

where $c_{i_1 i_2 \ldots i_n} \in \mathbb{R}$ for all $i_1, i_2, \ldots, i_n \in \{0, 1\}$.

For example, for $n=2$ and arbitrary $m$, all multi-affine functions have the form $f(x_1, x_2) = c_0 + c_0 x_1 + c_1 x_2 + c_1 x_1 x_2$, where $c_{i,j} \in \mathbb{R}$ for $i, j \in \{0, 1\}$.

**Definition 2.2:** A control system $\Sigma: \dot{x} = f(x, u) = g(x) + Bu$ with $B \in \mathbb{R}^{n \times m}$ is said to be multi-affine if $g: \mathbb{R}^n \rightarrow \mathbb{R}^m$ is a multi-affine function.

For a multi-affine control system, we write $x_{0, u}(t)$ to denote the point reached at time $t$ under the control input $u$ from initial condition $x_0$. In this article, the state space of the multi-affine system is assumed to be bounded and rectangular, which holds in lots of engineering applications (Belta and Habets 2006; Berman et al. 2007). Given such a state space, we would like to rectangularly partition it with respect to...
the coordinates. Then, the following concepts are provided.

An \( n \)-rectangle is described by \( E = \prod_{i=1}^{n} (a_i, b_i) \), where \( a_i, b_i \in \mathbb{R} \) satisfy \( a_i < b_i \) for \( i = 1, 2, \ldots, n \). The closure of \( E \) is defined as \( \overline{E} = \prod_{i=1}^{n} [a_i, b_i] \). A facet of \( E \) is the intersection of \( \overline{E} \) with one of its supporting hyperplanes. The set of facets of \( E \) is denoted by \( F(E) \). The set of vertices of \( E \), denoted by \( V(E) \), is \( V(E) = \{(x_1, x_2, \ldots, x_n) | x_i \in [a_i, b_i], \quad i = 1, 2, \ldots, n \} \).

Given \( v \in V(E) \), we denote \( F(v) \) the set of all facets containing \( v \).

The state space can be partitioned into \( \prod_{i=1}^{n} n_i \) rectangles as follows. Let \( x_i \in \bigcup_{i=1}^{n} (a_i', b_i') \), where \( a_i' < b_i' \) and \( a_i'^{+1} = b_i' \). Then, \( R_{k_{1},k_{2}, \ldots, k_{n}} = \prod_{i=1}^{n} (a_i^{+}, b_i^{k}) \) is a rectangle in the partitioned state space, where \( 1 \leq k_i \leq n_i \). The facet of \( R_{k_{1},k_{2}, \ldots, k_{n}} \) is described by

\[
F_{k_{1},k_{2}, \ldots, k_{n}} = \begin{cases} \{x \in \mathbb{R}^n | x_j = b_j^{k_j}\} & \text{if } d = + \\ \{x \in \mathbb{R}^n | x_j = a_j^{d_j}\} & \text{if } d = - 
\end{cases}
\]

where \( d \in \{+, -\} \) and \( j = 1, \ldots, n \).

The outer normal of \( F_{k_{1},k_{2}, \ldots, k_{n}} \) is given by

\[
n^{\text{out}}_{d} = e_j^T \quad \text{if } d = + \\
- e_j^T \quad \text{if } d = -
\]

where \( d \in \{+, -\} \), \( j = 1, \ldots, n \) and \( e_j \) is the Euclidian basis of \( \mathbb{R}^n \).

Given \( w = (w_1, w_2, \ldots, w_n) \in V(R_{k_{1},k_{2}, \ldots, k_{n}}) \), the vertex membership function \( S: \{w_1, \ldots, w_n\} \rightarrow \{0, 1\} \) is defined as

\[
S(w_j) = \begin{cases} 1 & \text{if } w_j = b_j^{k_j} \\ 0 & \text{if } w_j = a_j^{d_j} 
\end{cases}
\]

Denote \( \xi \) as the set of rectangles generated by rectangularly partitioning the state space. The rectangular projection map \( \pi_{Q}: \mathbb{R}^n \rightarrow \xi \) is defined as

\[
\pi_{Q}(x) = \{R_{k_{1},k_{2}, \ldots, k_{n}} | x \in R_{k_{1},k_{2}, \ldots, k_{n}} \}.
\]

Subsequently, the property of the multi-affine function on rectangles is presented as follows.

**Lemma 2.3** (Belta and Habets 2006): Consider a multi-affine function \( f \) and a rectangle \( R_{k_{1},k_{2}, \ldots, k_{n}} \). In every point \( x \in R_{k_{1},k_{2}, \ldots, k_{n}} \), the function \( f(x) \) is uniquely determined by the values of \( f \) at vertices of \( R_{k_{1},k_{2}, \ldots, k_{n}} \).

\[
f(x) = \sum_{w \in V(R_{k_{1},k_{2}, \ldots, k_{n}})} \lambda_\xi(x)(w) f(w)
\]

where for any \( w = (w_1, \ldots, w_n) \in V(R_{k_{1},k_{2}, \ldots, k_{n}}) \) and \( x = (x_1, x_2, \ldots, x_n) \in R_{k_{1},k_{2}, \ldots, k_{n}} \), the coefficient \( \lambda_\xi(x)(w) \) is defined as

\[
\lambda_\xi(x)(w) = n \prod_{j=1}^{n} \left( \frac{x_j - a_j^{d_j}}{b_j^{k_j} - a_j^{d_j}} \right)^{1-S(w_j)} \left( \frac{b_j^{k_j} - x_j}{b_j^{k_j} - a_j^{d_j}} \right)^{S(w_j)}
\]

By using this property, we review the results on the existence of a multi-affine feedback controller for a multi-affine system to keep the system in a rectangular invariant (Lemma 2.4) and to drive all initial states in a rectangle through a desired fact in finite time (Lemma 2.5).

**Lemma 2.4** (Belta and Habets 2006): Given a multi-affine control system \( \Sigma: \dot{x} = g(x) + Bu \) and a rectangle \( R_{k_{1},k_{2}, \ldots, k_{n}} \), there exists a multi-affine feedback controller \( K(x) \) such that \( u = K(x) \) and all trajectories of the closed-loop system that start from \( R_{k_{1},k_{2}, \ldots, k_{n}} \) remain in \( R_{k_{1},k_{2}, \ldots, k_{n}} \) for all times if and only if for any \( w \in V(R_{k_{1},k_{2}, \ldots, k_{n}}) \), the following set is nonempty:

\[
U_{\text{f}}(w) = \bigcap_{F_{k_{1},k_{2}, \ldots, k_{n}} \in F(w)} \{v \in \mathbb{R}^m | n^{\text{out}}_{d}(g(w) + Bv) \leq 0 \}. \tag{3}
\]

**Lemma 2.5** (Belta and Habets 2006): Given a multi-affine control system \( \Sigma: \dot{x} = g(x) + Bu \) and a rectangle \( R_{k_{1},k_{2}, \ldots, k_{n}} \), there exists a multi-affine feedback controller \( K(x) \) such that \( u = K(x) \) and all trajectories of the closed-loop system that start from \( R_{k_{1},k_{2}, \ldots, k_{n}} \) are driven only through \( F_{k_{1},k_{2}, \ldots, k_{n}} \) in finite time if for any \( w \in V(R_{k_{1},k_{2}, \ldots, k_{n}}) \), the following set is nonempty:

\[
U_{\text{f}}(w) = \bigcap_{F_{k_{1},k_{2}, \ldots, k_{n}} \in F(w)} \{v \in \mathbb{R}^m | n^{\text{out}}_{d}(g(w) + Bv) > 0 \} \wedge n^{\text{out}}_{d}(g(w) + Bv) \leq 0 \}. \tag{4}
\]

### 2.2 Transition system and LTL

A transition system is a tuple \( S = (E, E_0, U, \rightarrow, \overline{E}, Y, H) \), where \( E \) is a set of states, \( E_0 \subseteq E \) is a set of initial states, \( U \) is a set of control inputs, \( \overline{E} \subseteq E \times U \times E \) is a transition relation, \( \overline{E} \) is a set of marked states, \( Y \) is a set of outputs and \( H: E \rightarrow \{0, 1\} \) is an output function. The evolution of a system is captured by the transition relation. A transition \((e_1, u, e') \rightarrow \) is denoted as \( e \rightarrow e' \). Let \( U^n \) be a set of all finite strings over \( U \), including the empty string \( e \). The transition relation \( \rightarrow \subseteq E \times U \times E \) can be extended to \( \rightarrow \subseteq E \times U^n \times E \) in a natural way: \( e \rightarrow^n e' \) if there exists an \( e'' \) such that \( e \rightarrow e'' \) and \( e'' \rightarrow^n e' \), where \( s \in U^n \) and \( u \in U \). For \( E_1 \subseteq E \), the notation \( \rightarrow^1_{(E_1 \times U \times E_1)} \) means \( \rightarrow \) is restricted to a smaller domain \( E_1 \). Consider a set of propositions \( \Pi \), the label function \( L: Y \rightarrow 2^{\Pi} \) assigns each output a set of atomic propositions satisfied by this output. Consider \( e_1 \rightarrow e_2 \rightarrow \cdots \rightarrow e_n \rightarrow e_{n+1} \). A finite path generated from \( e_1 \), denoted as \( P_{e_1} \), is a finite alternating sequence of outputs and inputs: \( P_{e_1} = H(e_1)u_1H(e_2)u_2 \cdots H(e_n)u_nH(e_{n+1}) \). A finite run generated from \( e_1 \), denoted as \( R_{e_1} \), is a finite sequence of outputs: \( R_{e_1} = H(e_1)H(e_2) \cdots H(e_n) \). If the lengths of the above
sequences are infinite, they are called to be an infinite path and an infinite run, respectively. Denote $P(S)$, $P^*(S)$, $R(S)$ and $R^*(S)$ as the set of all finite paths generated by $S$, the set of all infinite paths generated by $S$, the set of all infinite runs generated by $S$ and the set of all infinite runs generated by $S$, respectively. Given $B \subseteq R^*(S)$, the prefix of $B$ is defined as $\overline{B} = \{ s \in R(S) \mid \exists r \in R^*(S) : st \in B \}$.

A transition system defines different languages. The finite language of $S$ is defined as $L(S) = \{ r \in R(S) \mid e \in E_0 \}$. The infinite language of $S$ is defined as $L^\omega(S) = \{ r \in R^*(S) \mid e \in E_0 \}$. Let $Y_m = \{ y : y = H(e), e \in E_m \}$. The accepted language of $S$ is defined as $L^a(S) = \{ r \in R^*(S) \mid \inf(r) \cap Y_m \neq \emptyset \}$, where $\inf(r)$ denotes the set of outputs appearing infinitely often in run $r$. The finite path language of $S$ is defined as $L_\pi(S) = \{ P_r \in P(S) \mid e \in E_0 \}$. The infinite path language of $S$ is defined as $L^\omega_\pi(S) = \{ P_r \in P^*(S) \mid e \in E_0 \}$. Given a label function $L : Y \rightarrow 2^\mathbb{P}$, an infinite run $r = R(1) \times R(2)R(3) \cdots$ defines a word $W = W(1)W(2)W(3) \cdots$, where $W(i) = L(R(i))$ for $i = 1, 2, 3, \ldots$.

The syntax and semantics of LTL formulas over the words of the transition system are introduced (Kloetzer and Belta 2008).

**Definition 2.6** (Syntax of LTL formulas): An LTL formula over $\Pi$ is recursively defined as:

- Every proposition $\pi \in \Pi$ is a formula.
- If $\varphi_1$ and $\varphi_2$ are formulas, then $\varphi_1 \wedge \varphi_2$, $\neg \varphi_1$, $\varphi_1 \supset \varphi_2$ and $\varphi_1 \lor \varphi_2$ are also formulas.

**Definition 2.7** (Semantics of LTL formulas): The satisfaction of an LTL formula $\varphi$ at position $i = 1, 2, 3, \ldots$ of the word $W$, denoted by $W(i) \models \varphi$, is recursively defined as:

- $W(i) \models \pi$, if $\pi \in W(i)$;
- $W(i) \not\models \neg \varphi$, if $W(i) \not\models \varphi$, where $\not\models$ denotes the negation of $\models$;
- $W(i) \models \varphi$, if $W(i+1) \models \varphi$;
- $W(i) \models \varphi_1 \wedge \varphi_2$, if $W(i) \models \varphi_1$ and $W(i) \models \varphi_2$;
- $W(i) \models \varphi_1 \lor \varphi_2$, if there exists a $j > i$ such that $W(j) \models \varphi_2$ and for all $i \leq k < j$ we have $W(k) \models \varphi_1$.

If $W(1) \not\models \varphi$, we say that the word $W$ satisfies $\varphi$, written as $W \models \varphi$. The symbols $\wedge$ and $\neg$ stand for conjunction and negation, respectively. The other Boolean connectors $\lor$ (disjunction), $\supset$ (implication), and $\equiv$ (equivalence) are defined in the usual way. The temporal operator $-$ is called the next operator. Formula $- \varphi$ specifies that $\varphi$ will be true in the next step. The temporal operator $\U$ is called the until operator. Formula $\varphi_1 \U \varphi_2$ means that $\varphi_1$ must hold until $\varphi_2$ holds. Two additional operators, ‘eventually’ and ‘always’ are defined as $\varphi_1 = \mathsf{true} \U \varphi$ and $\Box \varphi = \neg \varphi$.

Formula $\varphi$ means that $\varphi$ becomes eventually true whereas $\Box \varphi$ indicates that $\varphi$ is true at all positions of $W$. This set of operators can be employed to express many interesting specifications such as system synchronisation (Tabuada and Pappas 2006) and obstacle avoidance (Example 1).

3. Control of multi-affine systems on rectangles

In the previous section, several rectangles have been produced by a rectangular partition of the state space. Now, we investigate the control of multi-affine systems on rectangles. First, the notion of state-based switch multi-affine function is introduced.

**Definition 3.1:** Given multi-affine functions $U : \mathbb{R}^n \rightarrow \mathbb{R}^m$ and $U' : \mathbb{R}^n \rightarrow \mathbb{R}^m$, $x_f \in \mathbb{R}^n$ and $s \in \mathbb{R}^+$, a function $U \circ U' : \mathbb{R}^n \rightarrow \mathbb{R}^m$ is said to be a state-based switch multi-affine function from $U$ to $U'$ with respect to $x_f$ and $s$ if

$$U \circ U'(x) = \begin{cases} U(x) & \text{if } x \notin B_s(x_f) \\ U'(x) & \text{if } x \in B_s(x_f) \end{cases}$$

where $B_s(x_f) = \{ x \mid \| x - x_f \| \leq \epsilon \}$ with $\| \|$ denoting the Euclidean norm.

In this article, the control input for a multi-affine system $\dot{x} = g(x) + Bu$ is in terms of $u = K(x)$, where $K$ is multi-affine function or a state-based switch multi-affine function. Therefore, the feedback law is automatically bounded on $R_{k1,k2-k3}$. In the rest of this section, we propose a control method based on the exit sub-region to drive all trajectories of the closed-loop system starting from $R_{k1,k2-k3}$ to exit through a desired facet of $R_{k1,k2-k3}$, where the exit sub-region is defined as follows.

**Definition 3.2:** Let $\Sigma : \dot{x} = g(x) + Bu$ be a multi-affine control system, $K(x)$ be a multi-affine feedback controller, $R_{k1,k2-k3}$ be a rectangle and $F_{k1,k2-k3}$ be a facet of $R_{k1,k2-k3}$. A sub-region of $R_{k1,k2-k3}$ is called to exit sub-region with respect to $F_{k1,k2-k3}$ and $K(x)$, denoted as $[K]_{k1,k2-k3}$, if for any $x_0 \in [K]_{k1,k2-k3}$, there exists a $\tau \in \mathbb{R}^+$ such that

1. $\exists_{x_0,K(x)}(t_1) \in R_{k1,k2-k3}$ for $t_1 \in [0, \tau)$;
2. $\exists_{x_0,K(x)}(t_2) \in F_{k1,k2-k3}$ for $t_2 = \tau$;
3. $\exists_{x_0,K(x)}(t_3) \notin R_{k1,k2-k3} \cup F_{k1,k2-k3}$ for $t_3 \in (\tau, \tau + \epsilon)$ and $\epsilon \in \mathbb{R}^+$.

We can see that all trajectories of the closed-loop system $\dot{x} = g(x) + BK(x)$ originating in the sub-region $[K]_{k1,k2-k3}$ will leave $R_{k1,k2-k3}$ only through $F_{k1,k2-k3}$. It implies that if we can find a controller $K(x)$ such that all trajectories of the closed-loop system $\dot{x} = g(x) + BK(x)$ starting from $R_{k1,k2-k3}$ can reach
the exit sub-region \([K]^{vd}_{k_1 k_2 \ldots k_6}\) in finite time, then the
control of multi-affine systems with respect to the exit
facet \(F^{vd}_{k_1 k_2 \ldots k_6}\) can be realised by using \(K(x)\) together
with \(K(x)\). That is, we can first apply the controller
\(K'(x)\) to the multi-affine system and then update the
controller to \(K(x)\) once the trajectories arrive in
\([K]^{vd}_{k_1 k_2 \ldots k_6}\). To implement this idea, the following
problems should be addressed. Problem 1: how to find a
controller \(K(x)\) to guarantee the existence of an
exit sub-region \([K]^{vd}_{k_1 k_2 \ldots k_6}\)? Problem 2: if there exists an
exit sub-region \([K]^{vd}_{k_1 k_2 \ldots k_6}\), how to compute it? Problem
3: how to design a controller \(K(x)\) to drive all
trajectories of the closed-loop system starting from
\(R_{k_1 k_2 \ldots k_6}\) towards \([K]^{vd}_{k_1 k_2 \ldots k_6}\)? For Problem 1, we provide the following proposition.

**Proposition 3.3:** Given a multi-affine control system
\(\Sigma: \dot{x} = g(x) + Bu\), a multi-affine feedback controller
\(K(x)\), a rectangle \(R_{k_1 k_2 \ldots k_6}\), and a facet \(F^{vd}_{k_1 k_2 \ldots k_6}\) of
\(R_{k_1 k_2 \ldots k_6}\), there exists an exit sub-region \([K]^{vd}_{k_1 k_2 \ldots k_6}\) with respect to
\(F^{vd}_{k_1 k_2 \ldots k_6}\) and \(K(x)\) if

1. \(\exists w \in V(F^{vd}_{k_1 k_2 \ldots k_6});
   \quad n^{vd} [g(w) + BK(w)] > 0;\)
2. \(\forall v \in V(R_{k_1 k_2 \ldots k_6}) \setminus V(F^{vd}_{k_1 k_2 \ldots k_6}), \forall F^{vd'}_{k_1 k_2 \ldots k_6} \in F(v);
   \quad n^{vd'} [g(v) + BK(v)] \leq 0;\)
3. \(\forall x \in R_{k_1 k_2 \ldots k_6},
   g(x) + BK(x) \neq 0.\)

**Proof:** We have \(n^{vd} [g(w) + BK(w)] > 0\) at the vertex
\(w \in V(F^{vd}_{k_1 k_2 \ldots k_6}).\) Because the vector field is continuous, there exist some points at the neighbourhood of \(w\) that have strictly positive vector field outwards \(R_{k_1 k_2 \ldots k_6}\) through \(F^{vd}_{k_1 k_2 \ldots k_6}.\) Moreover, \(6)\) implies that the trajectories of the closed-loop system cannot leave through the facets whose vertices all satisfy the condition \((6)\) and \((7)\) implies there does not exist an equilibrium point inside \(R_{k_1 k_2 \ldots k_6}.\) We conclude that some trajectories of the closed-loop system starting from \(R_{k_1 k_2 \ldots k_6}\) will leave through \(F^{vd}_{k_1 k_2 \ldots k_6}.\) That is, there
is an exit sub-region \([K]^{vd}_{k_1 k_2 \ldots k_6}\) of \(R_{k_1 k_2 \ldots k_6}\) with respect to
\(F^{vd}_{k_1 k_2 \ldots k_6}\) and \(K(x).\)

It intuitively states that there exists an exit
sub-region \([K]^{vd}_{k_1 k_2 \ldots k_6}\) with respect to \(F^{vd}_{k_1 k_2 \ldots k_6}\) and \(K(x)\)
if the multi-affine feedback controller \(K(x)\) is such that:
1. there exists a vertex \(w\) on the exit facet such that the
velocity of the closed-loop system \(g(w) + BK(w)\) at \(w\)
has a strictly positive projection along the outer
normal of the exit facet; 2. for any vertex \(v\) which is
not on the exit facet, the velocity of the closed-loop
system \(g(v) + BK(v)\) at \(v\) has a negative projection
along the outer normal of the facet containing \(v;\) 3. there
does not exist an equilibrium point inside \(R_{k_1 k_2 \ldots k_6}.\)

Thus, Problem 1 is solved. Then, we consider
Problem 2, i.e. the computation of the exit sub-
region. Before presenting the calculation algorithm,
we need the concept of time-elapse cone.

**Definition 3.4** (Berman et al. 2007): Given a multi-
affine control system \(\Sigma: \dot{x} = g(x) + Bu\), a multi-affine
feedback controller \(K(x)\) and a rectangle \(R_{k_1 k_2 \ldots k_6},\) the
time-elapse cone for \(R_{k_1 k_2 \ldots k_6}\) with respect to \(K(x),\)
denoted by \(C_{R_{k_1 k_2 \ldots k_6}, K(x)}\), is defined as

\[
C_{R_{k_1 k_2 \ldots k_6}, K(x)} = \left\{ \sum_{w \in V(R_{k_1 k_2 \ldots k_6})} \mu_w \left[ g(w) + BK(w) \right] \mid \mu_w \geq 0 \right\}. \tag{8}
\]

The following lemma shows that the reachability of
multi-affine systems can be estimated by the
time-elapse cone.

**Lemma 3.5** (Berman et al. 2007): Given a multi-affine
control system \(\Sigma: \dot{x} = g(x) + Bu\), a multi-affine feedback
controller \(K(x)\), a rectangle \(R_{k_1 k_2 \ldots k_6},\) a state set
\(B \subset R_{k_1 k_2 \ldots k_6}\) and a reachable set of trajectories
\(X_{R_{k_1 k_2 \ldots k_6} \times K(x)}(B) = \left\{ x_{0} \times K(x) \mid x_{0} \in B \land t \in [0, T] \right\}\) for
\(t \in \mathbb{R^+}\) with respect to \(K(x),\) then \(X_{R_{k_1 k_2 \ldots k_6} \times K(x)}(B) \subset \left\{ B \oplus C_{R_{k_1 k_2 \ldots k_6}, K(x)} \right\},\) where \(\oplus\) is the Minkowski sum.

Similarly, the exit sub-region can be calculated, as it is
illustrated in Algorithm 3.6.

**Algorithm 3.6** (Computation of exit sub-regions)

**Input:** a multi-affine control system \(\Sigma: \dot{x} = g(x) + Bu\), a multi-affine feedback controller \(K(x)\), a rectangle \(R_{k_1 k_2 \ldots k_6}\), a facet \(F^{vd}_{k_1 k_2 \ldots k_6}\) of \(R_{k_1 k_2 \ldots k_6}\) and an accuracy
limitation \(\varepsilon.\)

**Output:** an exit sub-region \([K]^{vd}_{k_1 k_2 \ldots k_6}\) with respect to
\(F^{vd}_{k_1 k_2 \ldots k_6}\) and \(K(x)\).

For any \(R_{k_1 k_2 \ldots k_6} = \prod_{i=1}^{n} (a_{k_i}, b_{k_i})\), we define the following functions:

\[
\mathcal{L}(R_{k_1 k_2 \ldots k_6}) := \max_{i \in \{1, 2, \ldots, n\}} \left( b_{k_i}^{vd'} - a_{k_i}^{vd'} \right);
\]

\[
\mathcal{T}(R_{k_1 k_2 \ldots k_6}) := \bigcup_{m.p. \in \{1, 2\}} \left( a_{k_i'}^{vd''} - a_{k_i''}^{vd''} \right) \times \left( a_{k_j}^{vd''} - a_{k_j}^{vd''} \right) \times \cdots \times \left( a_{k_i}^{vd''} - a_{k_i}^{vd''} \right),
\]

where \(a_{k_i}^{vd'} = a_{k_i}, a_{k_i}^{vd''} = \frac{a_{k_i}^{vd'} + b_{k_i}^{vd'}}{2},\) and \(a_{k_i}^{vd''} = b_{k_i}.\)

Let \(p_{R_{k_1 k_2 \ldots k_6}} = \phi;\)

if \(\exists w \in V(F^{vd}_{k_1 k_2 \ldots k_6}); n^{vd} [g(w) + BK(w)] > 0\) and
\(\forall v \in V(R_{k_1 k_2 \ldots k_6}) \setminus V(F^{vd}_{k_1 k_2 \ldots k_6}), \forall F^{vd'}_{k_1 k_2 \ldots k_6} \in F(v);
   \quad n^{vd'} [g(v) + BK(v)] \leq 0\)

1468

Y. Sun et al.
Proposition 3.8: Algorithm 3.6 is correct.

Proof: Since (5)–(7) are satisfied, there exists an exit sub-region with respect to $F_{k_1:k_2-\kappa}$ and $K(x)$. Let $F_{x_0}^{\text{exit}} = \{x_0\}$. Choose $R_r \in \text{Rect}$ and such that $P_{\text{exit}} = \{x_0\}$. By putting $\lambda(x) = \text{dist}(x, \bar{F})$, we obtain $x_n(t) = \text{exit}$. It can be seen that $x_n(t) \to x_0$. Therefore, $\lim_{t \to \infty} x_n(t) = x_0$.

Next, we present the result for Problem 3.

Proposition 3.9 (Control to an exit facet): Given a multi-affine control system $\Sigma: \dot{x} = g(x) + Bu$, a rectangle $R_{k_1:k_2-\kappa}$ and a facet $F_{k_1:k_2-\kappa}$ of $R_{k_1:k_2-\kappa}$, there exists a feedback controller such that all trajectories of the closed-loop system starting from $F_{k_1:k_2-\kappa}$ remain in $R_{k_1:k_2-\kappa}$ for all times by Lemma 2.4. Let $u(w) \in U(w)$ be the control input at $w$ such that $x_f$ is a unique point in $R_{k_1:k_2-\kappa}$ satisfying (9). Then, we design $K(x) = \sum_{w \in V(R_{k_2:k_2-\kappa})} \alpha(x)u(w)$. For all rectangle $\alpha R_{k_1:k_2-\kappa}$, where $\alpha \in [0,1]$, the vertex set $V(\alpha R_{k_1:k_2-\kappa}) = \{w + (1 - \alpha)x\}$. It can be seen that $\alpha R_{k_1:k_2-\kappa}$ is just a shrunken version of $R_{k_1:k_2-\kappa}$ by multiplying $R_{k_1:k_2-\kappa}$ from $x_f$ by the factor $\alpha$. Thus, the velocity vector of the closed-loop system at the vertex of $\alpha R_{k_1:k_2-\kappa}$ is just $\alpha$-multiple the velocity vector at the corresponding vertex of $R_{k_1:k_2-\kappa}$. Since the vector field of the closed-loop system in all vertices of $\alpha R_{k_1:k_2-\kappa}$ is pointing inside to $\alpha R_{k_1:k_2-\kappa}$, there exist $\varepsilon > 0$ such that $\chi_{\alpha R_{k_1:k_2-\kappa}}(t) \in U(R_{k_1:k_2-\kappa})$. Then, $x_n(t) \to x_f$ for all $x_0 \in R_{k_1:k_2-\kappa}$ and $t \to \infty$. Similarly, we obtain $x_n(t) \to x_f$ for $t \to \infty$.

It indicates that if we can construct a controller of the form $u = K(x) = \sum_{w \in V(R_{k_2:k_2-\kappa})} \alpha(x)u(w)$, where $u(w) \in U(w)$ is a fixed point controller with respect to $x_f$. By putting $x_f$ inside the exit sub-region $[K]_{k_1:k_2-\kappa}$, the fixed point controller yields a solution for Problem 3.

Finally, we present the result on the control with respect to a desired exit facet.

Proposition 3.10: Given a multi-affine control system $\Sigma: \dot{x} = g(x) + Bu$, a rectangle $R_{k_1:k_2-\kappa}$ and a facet $F_{k_1:k_2-\kappa}$ of $R_{k_1:k_2-\kappa}$, there exists a feedback controller such that all trajectories of the closed-loop system starting from $R_{k_1:k_2-\kappa}$ are driven towards $x_f$. This kind of multi-affine function $K$ is called a fixed point controller with respect to $x_f$. By putting $x_f$ inside the exit sub-region $[K]_{k_1:k_2-\kappa}$, the fixed point controller yields a solution for Problem 3.
from $R_{k_1 k_2 - k_3}$ will exit only through $F_{k_1 k_2 - k_3}^{lid}$ in finite time.

Proposition 3.9 provides two different ways to drive the trajectories of the corresponding closed-loop system starting from $R_{k_1 k_2 - k_3}$ to exit only through a desired facet. One (condition (1)) is based on the result of Lemma 2.5 and the other (condition (2)) is based on the exit sub-region. Thus, the proposed control method for an exit facet covers more classes of systems than those and addressed in Belta and Habets (2006) and Habets et al. (2006). We call the multi-affine function or the state-based switch multi-affine function $U$, which drives all trajectories of the closed-loop system starting from $R_{k_1 k_2 - k_3}$ to exit only through $F_{k_1 k_2 - k_3}^{lid}$ as an exit controller with respect to $F_{k_1 k_2 - k_3}^{lid}$. Such an exit controller can be obtained by the following algorithm.

Algorithm 3.10 (Synthesis of exit controllers)

**Input:** a multi-affine control system $\Sigma: x = g(x) + Bu$, a rectangle $R_{k_1 k_2 - k_3}$, a facet $F_{k_1 k_2 - k_3}$ of $R_{k_1 k_2 - k_3}$ and $|u| \leq \eta$

**Output:** an exit controller with respect to $F_{k_1 k_2 - k_3}^{lid}$.

1. If $\{U_j(w_j)\} \neq \emptyset$ for any $w_j \in V(R_{k_1 k_2 - k_3}) \setminus U_k(w)$
   
   - Let $V_1 := \{j \in [1, 2, \ldots, 2^n] \mid U_k(w_j) \neq \emptyset, w_j \in U_k(w)\}$
   
   - If $U_1 \neq \emptyset$
     
     - $U_{R_{k_1 k_2 - k_3}}(x) = \sum_{j=1,2,\ldots,2^n} \lambda_{wj}(x) U_{R_{k_1 k_2 - k_3}}^{lj}(w_j)$

2. Else if ($V_1 \subset \{1, 2, \ldots, 2^n\}$)
   
   - If $U_k(w) \neq \emptyset$ for any $w_j \in V(R_{k_1 k_2 - k_3})$
     
     - $U_3 := \{U_{R_{k_1 k_2 - k_3}}^{lj}(w_j) \mid j = 1, 2, \ldots, 2^n\}$

   - Obtain the exit sub-region $\{U_{R_{k_1 k_2 - k_3}}^{lj} \}$ w.r.t. $F_{k_1 k_2 - k_3}^{lid}$ and $U_{R_{k_1 k_2 - k_3}}^{lj}(x)$

   - For all $U_{R_{k_1 k_2 - k_3}}^{lj}(w_j) \mid j = 1, 2, \ldots, 2^n \in U_3$
     
     - $U_{R_{k_1 k_2 - k_3}}^{lj}(x) = \sum_{j=1,2,\ldots,2^n} \lambda_{wj}(x) U_{R_{k_1 k_2 - k_3}}^{lj}(w_j)$

   - If ($\exists \ v \in \mathbb{R}^n$ and a unique point $x' \in R_{k_1 k_2 - k_3}$ s.t. $g(x') + Bu^3_{R_{k_1 k_2 - k_3}}(x') = 0$ and $R_{k_1 k_2 - k_3}$ is an exit controller for $F_{k_1 k_2 - k_3}^{lid}$)

     - $\text{end if}$

3. End if

Proposition 3.11: Algorithm 3.10 is correct.

**Proof:** The proof is obvious according to Proposition 3.9.

4. Finely abstracted transition systems of multi-affine systems

The control of multi-affine systems on rectangles enables the construction of a finely abstracted transition system for the multi-affine system, as illustrated in Definition 4.1. Here we assume that any initial state of the multi-affine system is inside the rectangles and the duration of the trajectories staying on the boundary of the rectangle is ignored.
These assumptions result in no loss of generality since they always hold in the implementation.

**Definition 4.1:** Given a multi-affine control system \( \Sigma : \dot{x} = g(x) + Bu \) and a rectangle set \( \xi \) generated by rectangularly partitioning the state space, the abstracted transition system of \( \Sigma \) associated with \( \xi \), denoted as \( S_{\Sigma,\xi} \), is a tuple
\[
S_{\Sigma,\xi} = (X_0, X_{00}, U_0, \rightarrow_{\xi}, X_{m0}, Y_0, H_0)
\]
- \( X_0 = \xi = X_{m0} \);
- \( X_{00} = \{ R_{k_1,k_2,k_3} | k_3 \in \xi \} \), \( R_{k_1,k_2,k_3} \) contains an initial state of the multi-affine control system;
- \( U_0 = \{ U_{R_{k_1,k_2,k_3}} | U_{R_{k_1,k_2,k_3}} \) is a multi-affine function or a state-based switch multi-affine function, \( R_{k_1,k_2,k_3} \in \xi \};
- \( R_{k_1,k_2,k_3} \rightarrow_{\xi} R_{k_1,k_2,k_3} \) if any of the following two conditions is satisfied:
  \begin{enumerate}
  \item \( R_{k_1,k_2,k_3} = R_{k_1,k_2,k_3} \) holds and for any \( w \in F(R_{k_1,k_2,k_3}) \), \( U(w) \neq \emptyset \) and \( U_{R_{k_1,k_2,k_3}}(w) \in U_{R_{k_1,k_2,k_3}} \).
  \item \( R_{k_1,k_2,k_3} \neq R_{k_1,k_2,k_3} \) with \( \frac{R_{k_1,k_2,k_3}}{R_{k_1,k_2,k_3}} \rightarrow_{\xi} \) holds and \( U_{R_{k_1,k_2,k_3}} \) is an exit controller with respect to \( F_{R_{k_1,k_2,k_3}} \).
\end{enumerate}
- \( Y_0 = \xi \);
- \( H_0(R_{k_1,k_2,k_3}) = R_{k_1,k_2,k_3} \).

An abstracted transition system is a finite-state system, therefore it facilitates the synthesis of the controller for finite-state requirements while accommodating to infinite-state dynamics. Next, a rectangular transition system of the multi-affine control system is established, and it can be understood as a transition system form of the multi-affine control system over a rectangularly partitioned state space.

**Definition 4.2:** Given a multi-affine control system \( \Sigma : \dot{x} = g(x) + Bu \), a rectangle set \( \xi \) generated by rectangularly partitioning the state space and a rectangular project map \( \pi_\Omega \) defined by \( \xi \), the rectangular transition system of \( \Sigma \) associated with \( \xi \), denoted as \( S_{\Sigma,\xi,\Omega} \), is a tuple
\[
S_{\Sigma,\xi,\Omega} = (X_0, X_{00}, U_0, \rightarrow_{\xi \rightarrow_{\Omega}}, X_{m0}, Y_0, H_0)
\]
- \( X_0 = \mathbb{R}^n = X_{m0} \);
- \( X_{00} = \{ x | x \) is an initial state of the multi-affine control system\};
- \( U_0 = \{ k | k(x) \) is a feedback control law\};
- \( x \rightarrow_{\Omega} x' \) if any of the following two conditions is satisfied:
  \begin{enumerate}
  \item \( \pi_\Omega(x) = \pi_\Omega(x') \) holds and there exists \( \tau \in \mathbb{R}^+ \) such that \( x_{0,k_0}(\tau) = x' \) and \( \pi_\Omega(x_{0,k_0}(\tau)) = \pi_\Omega(x') \), where \( t \in [0, \infty) \).
  \item \( \pi_\Omega(x) \neq \pi_\Omega(x') \) holds and there exist \( \tau, \epsilon \in \mathbb{R}^+ \) such that \( x_{0,k_0}(\tau) = x' \) and \( \pi_\Omega(x_{0,k_0}(\tau)) \).
\end{enumerate}
Theorem 4.5: Given a multi-affine control system \( \Sigma : x = g(x) + Bu \), a rectangle set \( \xi \) generated by rectangularly partitioning the state space and a rectangular project map \( \pi_Q \) defined by \( \xi \), the relation \( \phi \) defined as
\[
\phi = \{(R_{k_1k_2\ldots k_r}, x) \in \xi \times \mathbb{R}^n | x \in R_{k_1k_2\ldots k_r}\}
\]
is an input–output simulation relation from \( S_{\Sigma,\delta} \) to \( S_{\Sigma,Q} \).

Proof: For any \( (R_{k_1k_2\ldots k_r}, x) \in \phi \), we have \( H_\delta(R_{k_1k_2\ldots k_r}) = R_{k_1k_2\ldots k_r} = H_\delta(x) = \pi_Q(x) \). Further, if there is a transition \( R_{k_1k_2\ldots k_r} \xrightarrow{u} R_{k_1'k_2'\ldots k_r'} \), we have the following two cases: (a) \( R_{k_1k_2\ldots k_r} \neq R_{k_1'k_2'\ldots k_r'} \) with \( F_{k_1k_2\ldots k_r}^d = R_{k_2k_3\ldots k_r} \cap R_{k_1'k_2'\ldots k_r'} \). According to the construction of \( S_{\Sigma,\delta} \), there exists a controller \( U_{k_1k_2\ldots k_r} \) such that all trajectories of the closed-loop system \( \dot{x} = g(x) + BU_{k_1k_2\ldots k_r}(x) \) starting from \( R_{k_1k_2\ldots k_r} \) are driven only through \( F_{k_1k_2\ldots k_r}^d \). Then, for any \( x \in R_{k_1k_2\ldots k_r} \), there is \( x' \in R_{k_1'k_2'\ldots k_r'} \) such that \( x \xrightarrow{u} q' \) and \( (R_{k_1'k_2'\ldots k_r'}, x') \in \phi \). (b) \( R_{k_1k_2\ldots k_r} = R_{k_1'k_2'\ldots k_r'} \) and \( F_{k_1k_2\ldots k_r}^d = R_{k_2k_3\ldots k_r} \). The controller \( U_{k_1k_2\ldots k_r} \) satisfying \( U_{k_1k_2\ldots k_r}(w) \equiv U_{k_1'k_2'\ldots k_r'}(w) \) for any \( w \in F(R_{k_1k_2\ldots k_r}) \) drives all trajectories of the closed-loop system \( \dot{x} = g(x) + BU_{k_1k_2\ldots k_r}(x) \) starting from \( R_{k_1k_2\ldots k_r} \) to remain in \( R_{k_1k_2\ldots k_r} \) for all times (Belta and Habets 2006). Therefore, there exists an \( x' \in R_{k_1'k_2'\ldots k_r'} \) such that \( x \xrightarrow{u} q' \) and \( (R_{k_1'k_2'\ldots k_r'}, x') \in \phi \). Moreover, the definition of \( \phi \) indicates that for any \( R_{k_1k_2\ldots k_r} \in X_{\emptyset} \), there exists an \( x \in X_Q \) such that \( (R_{k_1k_2\ldots k_r}, x) \in \phi \). As a result, \( S_{\Sigma,\delta} \prec_{\text{IO}_S} S_{\Sigma,Q} \). \( \square \)

5. Controller synthesis for LTL specifications

This section studies the controller synthesis for LTL specifications. It is well known that an LTL formula \( \varphi \) over a proposition set \( \Pi \) can be effectively converted into a Büchi automaton which accepts every infinite string over \( \Pi \) satisfying \( \varphi \) (Wolper, Vardi, and Sistla 1983). This kind of Büchi automaton is described as follows.

Definition 5.1: Given an LTL formula \( \varphi \) over a proposition set \( \Pi \), the Büchi automaton with respect to \( \varphi \), denoted as \( B_\varphi \), is a tuple
\[
B_\varphi = (B, B_0, \Sigma^1, \rightarrow_B, B_m)
\]
- \( B, B_0 \subseteq B \) and \( B_m \subseteq B \) are finite sets of states, initial states and marked states, respectively;
- \( \Sigma^1 \) is an input alphabet;
- \( \rightarrow_B \subseteq B \times \Sigma^1 \times 2^B \) is a transition relation.

Since the abstracted transition system \( S_{\Sigma,\delta} \) is input–output simulated by the rectangular transition system \( S_{\Sigma,Q} \), if there exists a supervisor (discrete controller) \( S_c \) for \( S_{\Sigma,\delta} \) enforcing the LTL specifications, then such a supervisor also works for \( S_{\Sigma,Q} \). i.e. the implementation of \( S_c \) drives the multi-affine system to fulfill the LTL specifications. Thus, we first focus on the synthesis of \( S_c \). Here a supervisor conducts the control through restricting the behaviours of the transition system, which is captured by the following notion.

Definition 5.2: Given transition systems \( S_a = (X_a, X_{\emptyset}, U_a, \rightarrow_a, X_{\text{mark}}, Y_a, H_a) \) and \( S_b = (X_b, X_{\emptyset}, U_b, \rightarrow_b, X_{\text{mark}}, Y_b, H_b) \), the input–output parallel composition of \( S_a \) and \( S_b \), denoted as \( S_a \parallel \parallel S_b \), is a transition system
\[
S_a \parallel S_b = (X_{ab}, X_{\emptyset}, U_{ab}, \rightarrow_{ab}, X_{\text{mark}}, Y_{ab}, H_{ab})
\]
where
\[
X_{ab} = \{(x_a, x_b) \in X_a \times X_b | H_a(x_a) = H_b(x_b)\};
X_{\emptyset,ab} = (X_a \times X_b) \cap X_{\emptyset};
U_{ab} = U_a \cup U_b;
Y_{ab} = Y_a \cup Y_b;
H_{ab}(x_a, x_b) = H_a(x_a).
\]

The presented input–output parallel composition is different from the usual synchronisation operator in the supervisory control literature, as besides a same control symbol \( \rightarrow \) between the synchronised transitions \( \rightarrow_a \) and \( \rightarrow_b \), it also requires identical output values \( H_a(x_a) = H_b(x_b) \) between the state pairs. Thus, the behaviours (finite/infinite language, accepted language and finite/infinite path language) of \( S_a \parallel \parallel S_b \) are contained in those of \( S_a \). It follows that the supervisor \( S_c \) can restrict the behaviours of \( S_{\Sigma,\delta} \) which do not satisfy the LTL specifications. This observation motivates us to construct the supervisor \( S_c \) by working with \( S_{\Sigma,\delta} \) and \( B_\varphi \). Hence, we introduce the notion of product automaton.

Definition 5.3: Given an abstracted transition system \( S_{\Sigma,\delta} = (X_\delta, X_{\emptyset}, U_\delta, \rightarrow_\delta, X_{\text{mark}}, Y_\delta, H_\delta) \), a Büchi automaton \( B_\varphi = (B, B_0, \Sigma^1, \rightarrow_B, B_m) \) and a label function \( L : \rightarrow_\delta \rightarrow \Sigma^1 \), the product automaton of \( S_{\Sigma,\delta} \) and \( B_\varphi \), denoted as \( S_{\Sigma,\delta} \times A B_\varphi \), is a transition system
\[
S_{\Sigma,\delta} \times A B_\varphi = (A, A_0, U_A, \rightarrow_A, A_m, Y_A, H_A)
\]
where
\[
A = X_\delta \times B;
A_0 = \{(x_\delta, b) \in X_\delta \times B | \exists b_0 \in B_0; b_0 \rightarrow L(H_\delta(x_\delta)) b\};
U_A = U_\delta \cup U_B;
(x_\delta, b) \rightarrow_A (x'_\delta, b') \text{ iff } x_\delta \rightarrow_\delta x'_\delta \text{ and } b \rightarrow b';
A_m = X_{\text{mark}} \times B_m;
Y_A = Y_\delta;
H_A(x_\delta, b) = H_\delta(x_\delta).
\]

The result provided by de Giacomo and Vardi (2000) indicates that a string \( r \) satisfies the LTL
Since any string in \( L^\|_h(S_t) \subseteq L^\|_s(A) \) satisfies the LTL formula \( \psi \). Thus, when \( \text{CoAc}(S_t \times A B_u) \) is chosen to be the supervisor \( S_c \), it guarantees the accepted language equivalence while preventing the blocking, as stated in the following theorem.

**Theorem 5.5:** Given a rectangular transition system \( S_{\Sigma, Q} \) and a product automaton \( S_{\Sigma, Q} \times A B_u \), there exists a supervisor \( S_c \) for \( S_{\Sigma, Q} \) such that \( L^\|_s(A) = L^\|_s(S_t) \), and \( L(S_t ||_h S_{Q}) = L^\|_s(S_{\Sigma, Q} \times A B_u) \) if \( L^\|_s(S_{\Sigma, Q} \times A B_u) \neq \emptyset \).

**Proof:** Since \( L^\|_s(S_{\Sigma, Q} \times A B_u) \neq \emptyset \), let \( S_c = \text{CoAc}(S_{\Sigma, Q} \times A B_u) \). We use the facts: (1) \( L^\|_s(S_{\Sigma, Q} \times A B_u) \subseteq L^\|_s(S_t) \) and \( L^\|_s(S_{\Sigma, Q} \times A B_u) \subseteq L^\|_s(S_{\Sigma, Q}) \), and (2) \( S_{\Sigma, Q} \) is a product of \( S_t \) and \( S_{Q} \), and (3) \( L^\|_s(S_{\Sigma, Q} \times A B_u) \). Thus, \( L^\|_s(S_t ||_h S_{Q}) \subseteq L^\|_s(S_{\Sigma, Q} \times A B_u) \). Moreover, we have \( L(S_{\Sigma, Q} \times A B_u) = L^\|_s(S_{\Sigma, Q} \times A B_u) \), and \( L^\|_s(S_{\Sigma, Q} \times A B_u) \subseteq L^\|_s(A) \). Therefore, the implementation of \( S_c \) drives the multi-affine system to satisfy the LTL formula \( \psi \).

### Remark 5.6
The proof of Theorem 5.5 is constructive as if \( L^\|_s(S_{\Sigma, Q} \times A B_u) \neq \emptyset \), \( S_c = \text{CoAc}(S_{\Sigma, Q} \times A B_u) \) provides a supervisor to achieve the LTL formula \( \psi \) in a non-blocking manner (\( L(S_t ||_h S_{Q}) = L^\|_s(S_{\Sigma, Q} \times A B_u) \)).

In this article, we call the supervisor obtained in Theorem 5.5 as a nonblocking supervisor.

### 5.1 Implementation of discrete controllers to multi-affine systems
We have already outlined how the nonblocking supervisor \( S_c \), where \( S_c = \text{CoAc}(S_{\Sigma, Q} \times A B_u) \), enforces the satisfaction of LTL specifications with respect to \( S_{\Sigma, Q} \). Then, we discuss the implementation of \( S_c \) to the multi-affine system. Since any string in \( L^\|_s(S_t ||_h S_{Q}) \) satisfies the LTL formula \( \psi \), let \( R_{k_{1}-k_{2}, k_{1}'-k_{2}'} \) be a string in \( L^\|_s(S_t ||_h S_{Q}) \) and \( R_{k_{1}-k_{2}, k_{1}'-k_{2}'} U_{R_{k_{1}-k_{2}, k_{1}'-k_{2}'}} \) be the corresponding infinite path. To realise \( R_{k_{1}-k_{2}, k_{1}'-k_{2}'} \), we can apply the controller \( U(R_{k_{1}-k_{2}, k_{1}'-k_{2}'}) \) to the multi-affine system as long as \( x \in R_{k_{1}-k_{2}, k_{1}'-k_{2}'} \). When and if \( x \notin R_{k_{1}-k_{2}, k_{1}'-k_{2}'} \), the string is updated to \( R_{k_{1}'+k_{2}'-k_{2}} \), then the process continues.

Therefore, the implementation of \( S_c \) drives the multi-affine system to satisfy the LTL formula \( \psi \).

### 6. Example
Consider a path-planning example adopted from Belta and Habets (2006), where a robot with detection and positioning capabilities moves inside a rectangular region \([0, 3] \times [1, 4]\). In particular, the robot system takes the form of the following differential equation:

\[
\frac{dx}{dt} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 4 \end{bmatrix} u
\]

where \( x \) is the position of the robot and \( u \) is the control input. The rectangular region is partitioned into nine small rectangular sub-regions with respect to the coordinates (Figure 1 (left)). Let \( R_{23} \) be a dangerous sub-region and \( R_{33} \) be a goal sub-region. Thus, for each sub-region we define the label function \( L \): \( L(R_{23}) = \text{Danger} \cup \text{Goal} \), \( L(R_{13}) = \text{Danger} \cup \text{Goal} \), and \( L(R) = \text{Danger} \cup \text{Goal} \) for each sub-region (\( \text{Goal} \)), where \( \text{Danger} \) represents the dangerous sub-region and \( \text{Goal} \) represents the goal sub-region. In this example, the specification is to eventually go to the goal sub-region (\( \text{Goal} \)) while avoiding the dangerous sub-region (\( \text{Danger} \)). Such an obstacle avoidance specification can be naturally expressed by the LTL formula \( \neg \text{Danger} \land \neg \text{Goal} \).

To achieve the specification, we first explore the control of the robot on sub-regions. Take \( R_{12} \) as an example. If we would like to control the robot to exit from \( R_{12} \) to \( R_{33} \) through the face \( F_{13} \), then \( U(1, 3) = \{ \} \), \( U_0 = \{ 1, 0 \} \), \( 0 \leq u \leq 4 \). Obviously, such a controller does not exist
according to Lemma 2.5 (Belta and Habets 2006; Habets et al. 2006). However, by using the proposed method in this article, we can obtain a controller for the exit problem. Here we assume the accuracy limitation \( \varepsilon = 10^{-4} \) and the control limitation \( |u| \leq 10^7 \). By Algorithm 3.10, we can design a state-based switch multi-affine controller in terms of

\[
I_{R_{12}} \circ U_{R_{12}}(x) = \begin{cases} 
-30x_1 - 12x_2 + 10x_1x_2 + 34 & \text{if } x \notin B_{0.01}(0.767, 2.494) \\
-11x_1 + x_1x_2 + 10 & \text{if } x \in B_{0.01}(0.767, 2.494)
\end{cases}
\]

to drive the robot to exit only through \( F_{R_{12}} \). Similarly, for each sub-region \( R_{mn}(m, n = 1, 2, 3) \) we can establish the controllers that steer the robot from \( R_{mn} \) to its neighbourhood sub-region (Algorithm 3.10) or to be invariant (Lemma 2.4) in \( R_{mn} \), respectively. Thus, an abstracted transition system \( S_{\Sigma, \delta} \) can be constructed (Figure 1 (right)).

On the other side, we convert the LTL formula \( \varphi \) to a Büchi automaton (Figure 2 (left)) and then establish the product automaton \( S_{\Sigma, \delta} \times_A B_{\varphi} \) (Figure 2 (right)). According to Theorem 5.5, we design CoAc\( (S_{\Sigma, \delta} \times_A B_{\varphi}) \) (Figure 3 (left)) to be the nonblocking supervisor for \( S_{\Sigma, \delta} \). After the implementation of CoAc\( (S_{\Sigma, \delta} \times_A B_{\varphi}) \) to the robot system, the controlled system achieves the LTL formula \( \varphi \). Moreover, the simulation results of two feasible paths initialising from \( R_{31} \) and satisfying \( \varphi \) are shown in Figure 3 (right).

7. Conclusion

This article provided an input–output simulation approach to controlling the multi-affine system for LTL specifications in a rectangularly partitioned state space. Two novel methods were derived to control the multi-affine system on rectangles. One is based on the exit sub-region to drive all trajectories starting from a rectangle to exit only through a facet, which enlarges the classes of control systems in the context of existing literature (Belta and Habets 2006). The other provides a solution for the convergence problem by stabilising the multi-affine system towards a desired point. With the proposed control methods, a finitely abstracted transition system was constructed and it was shown to be input–output simulated by the rectangular transition system of the multi-affine system. Therefore, the controller synthesis for the multi-affine system to enforce the LTL specification can be achieved by

\[
\text{Figure 1. Rectangularly partitioned state space (left) and abstracted transition system } S_{\Sigma, \delta} \text{ (right).}
\]

\[
\text{Figure 2. Büchi automaton } B_{\varphi} \text{ (left) and the product automaton } S_{\Sigma, \delta} \times_A B_{\varphi} \text{ (right).}
\]

\[
\text{Figure 3. Nonblocking supervisor for } S_{\Sigma, \delta} \text{ (left) and the controlled system achieving } \varphi \text{ (right).}
\]
designing a nonblocking supervisor for the abstracted transition system and then mapped into continuous control signals. From the application point of view, this input–output simulation approach not only enables automatic and effective implementation, but also prevents blocking in the execution.

However, the result on the existence of a non-blocking supervisor enforcing LTL, i.e. Theorem 5.5, is sufficient only in the sense that if the condition of Theorem 5.5 does not satisfy, there is no conclusion on the existence of a controller for the original multi-affine system. To address this issue, our future work will investigate the necessary and sufficient condition by strengthening the input–output simulation to an input–output bisimulation. Other interesting directions are extensions of this approach to branching time logical specifications, such as computation tree logic specifications (Clarke 1997), and to more complicated dynamics, such as polynomial dynamics (Benedetto 2002).

References


