Asian Journal of Control, Vol. 00, No. 0, pp. 1–11, Month 2008
Published online in Wiley InterScience (www.interscience.wiley.com) DOI: 10.1002/asjc.0000

Hierarchical Hybrid Symbolic Robot Motion Planning and Control[†]

Ali Karimoddini and Hai Lin

ABSTRACT

This paper addresses the motion planning problem using hybrid symbolic techniques. The proposed approach develops a unified hierarchical hybrid control framework using a bismulation-based abstraction technique over the partitioned motion space that can be applied to autonomous aerial robots (3-D symbolic motion planning) or ground vehicles (2-D symbolic motion planning). The bisimulation relation between the abstracted model and the original continuous system guarantees that their behaviors are the same. This allows to design a discrete supervisor for the abstracted model, and then, the designed supervisor can be applied to the original system while the closed-loop behavior does not change. To apply the discrete supervisor to the original continuous system, an interface layer is developed, which on the one hand translates discrete commands of the supervisor to a continuous form applicable to the continuous plant and on the other hand, abstracts the continuous signals of the continuous low layer to discrete symbols understandable by the supervisor. The proposed algorithm is verified through implementation of a hybrid symbolic algorithm for the formation control of unmanned aerial vehicles.

Key Words: Symbolic Control, Hybrid Control, Abstraction, Robot Motion Planning

I. INTRODUCTION

With advances in technologies it is becoming possible to develop fully autonomous vehicles that can accomplish complicated tasks. Such sophisticated capabilities require the control structure of an autonomous vehicle to have various types of sensors to recognize itself and the environment, analyze high amount of sensor readings, process the collected data and accordingly, make decision to compute the control signals. Here, for motion planning of the robots, the question is that how to process and manage excessive amount of data, and then, how to integrate the decision making unit with continuous low level control structure of the system. In fact, a typical robot has an inherent hybrid nature whose event-triggered discrete logic of the decision making unit and the corresponding discrete commands influence the continuous dynamics of the robot. This logic may require the system to satisfy several goals with a particular order, which is beyond the traditional methods for robot motion control based on optimal control techniques. To comprehensively analyze and design a control system for motion planning and control, a proper solution is to utilize the hybrid modeling and control theory and consider the discrete and continuous dynamics of the system, simultaneously and within a unified framework [1].

To develop a hybrid symbolic motion planning and control mechanism, the challenge is to design and develop a computationally effective hybrid approach for the robot motion control so that the closed-loop system satisfies the discrete logic of the decision making unit.

A. Karimoddini is with the Department of Electrical and Computer Engineering, North Carolina Agricultural and Technical State University, Greensboro, NC 27411 USA. email: akarimod@ncat.edu.
H. Lin is with the Department of Electrical Engineering, University of Notre Dame, Notre Dame, USA. email: hlin1@nd.edu.
Financial supports from NSF-CNS-1239222 and NSF-EECS-1253488 for this work are greatly acknowledged. A primary version of this paper was presented in 2013 10th IEEE International Conference on Control and Automation.

In [2], a complicated search and rescue and in [3], the motion control of a team of robots are addressed using symbolic control methods and abstraction techniques. These schemes reduce the system to a finite state transition system [4], [5], [6] for which one can design a proper discrete supervisor [7] to achieve certain properties expressed in high-level human-like languages such as linear or branching temporal logics [8], [2], [9], [10]. Most efforts in the literature have been devoted to partitioning the motion space to obtain an abstracted model [11, 12, 13, 14, 15, 16]. Here, the key is to realize the relationship between the abstract model and the original continuous system. Understanding the relationship between the abstract model and the original system allows to design a supervisor for the abstract model that is essentially a simpler model, and then, convert it back to the original continuous model. The immediate question is that whether it can be guaranteed that the control objectives, achieved in the abstract domain, can be achieved by the original system as well. To address this problem, this paper aims at developing a bisimulation-based abstraction technique by which, the equivalent behaviors of the abstract model and the original plant allows the designer to synthesis the discrete supervisor for the abstract model and then, apply it to the original plant. The preliminary result of this work was presented in [17]. Compared with [17], here, the bismulation relation between the abstract model and the original system is rigorously proved, and the results are extended to the polar partitioned spaces. Furthermore, actual implementation results are provided to verify the algorithm.

The main contribution of this paper is to develop a unified hierarchical hybrid framework for symbolic motion planning and control of robots based on a bisimulation-based abstraction technique over a rectangular or polar partitioned space. Starting from the low level continuous dynamics of the system, and using rectangular or polar partitioning techniques, the motion dynamics of the robots can be abstracted to a finite state machine over the partitioned motion space, for which we can design a discrete supervisor to achieve the desired specification. We prove that the bisimulation relation between the abstracted model and the original continuous model holds for a plant with multi-affine dynamics over the rectangular or polar partitioned space. This bisimulation relation guarantees the same behavior of the plant and its abstract model and therefore, the discrete supervisor designed for the abstracted model can be applied to the original continuous plant so that the closed-loop system's behavior does not change. To implement the idea, a hierarchical hybrid control structure is proposed whose lowest layer is a plant with continuous dynamics, and its top layer is a discrete supervisor which controls the system to satisfy the given specification. To connect the discrete supervisor to the continuous plant, an interface layer is introduced by which the discrete commands of the supervisor can be converted to a continuous form applicable to the plant. Furthermore, when the system trajectory crosses the partitioning curves, the interface layer generates detection events which inform the supervisor about the current state of the system. then, based on observing these detection events, the supervisor can issue new commands. The proposed algorithm is implemented on unmanned aerial helicopters and the flight test results are provided to evaluate the algorithm.

The rest of this paper is organized as follows. After explaining the preliminaries and notations in Section II, the symbolic motion planning and control problem is described in Section III. Then, in Section IV, the partitioning of the motion space will be described. Several controllers will be introduced to drive the system trajectory over the partitioning elements. In Section V, the partitioned system will be bisimilarly abstracted to a finite state machine and the bisimulation relation will be proven. For the resulting finite state machine one can design a discrete supervisor as explained in Section VI. Section VII describes how to implement the whole control structure through an illustrative example. Finally, the paper is concluded in Section VIII.

II. Preliminaries

To address the motion planning problem, we partition the motion space. In the literature, there are several methods that can be used for partitioning the space such as using natural invariants of the plants [18], rectangulation [12] or triangulation [11] of the motion space, or polar and spherical partitioning [16], [15], of the space. To elaborate the idea, without loss of generality, here we will use rectangular and polar partitioning of the motion space which are more convenient to work with.

Consider that the motion space is a $[0, x_N] \times [0, y_N]$ rectangle, which is partitioned by the curves $\{x = x_i \,|\, 0 \le x_i \le x_N \textit{ such that for } i < j : x_i < x_j\,, i, j = 1, ..., N_a,\, x_1 = 0,\, x_{N_a} = x_N\}$, and $\{y = y_i \,|\, 0 \le y_i \le y_N \textit{ such that for } i < j : y_i < y_j\; i, j = 1, ..., N_b,\, y_1 = 0,\, y_{N_b} = y_N\}$ into $(N_a - 1) \times (N_b - 1)$ rectangles. In this partitioned space, the region $R_{i,j} = \{(x, y) |\, x_i \le x \le x_{i+1},\, y_j \le y \le y_{j+1}\}$ is a rectangular partitioning element which is surrounded by the curves $x = x_i$, $x = x_{i+1}$, $y = y_j$, and $y = y_{j+1}$. The interior of the region $R_{i,j}$ is denoted by $\bar{R}_{i,j}$. Each region has four vertices v_m, with $m = (a, b)$, where m_a and m_b are the binary indices which refer to the partitioning curves that have generated the vertex v_m. Hence, we have $v_0 = v_{(00)_2} = [x_i, y_j]^T$, $v_1 = v_{(01)_2} = [x_{i+1}, y_j]^T$, $v_2 = v_{(10)_2} = [x_i, y_{j+1}]^T$, and $v_3 = v_{(11)_2} = [x_{i+1}, y_{j+1}]^T$ as the vertices of the region $R_{i,j}$ as shown in Fig. 1. The set $V(*)$ stands for the vertices that belong to $*$, and $E(v_m)$ is the set of edges that touch the vertex v_m. Furthermore, the element $R_{i,j}$ has four edges $\{\mathrm{E}_x^+, \mathrm{E}_x^-, \mathrm{E}_y^+, \mathrm{E}_y^-\}$ and correspondingly, four outer normal vectors $\{\mathrm{n}_x^+ = [1, 0]^T, \mathrm{n}_x^- = [-1, 0]^T, \mathrm{n}_y^+ = [0, 1]^T, \mathrm{n}_y^- = [0, -1]^T\}$.

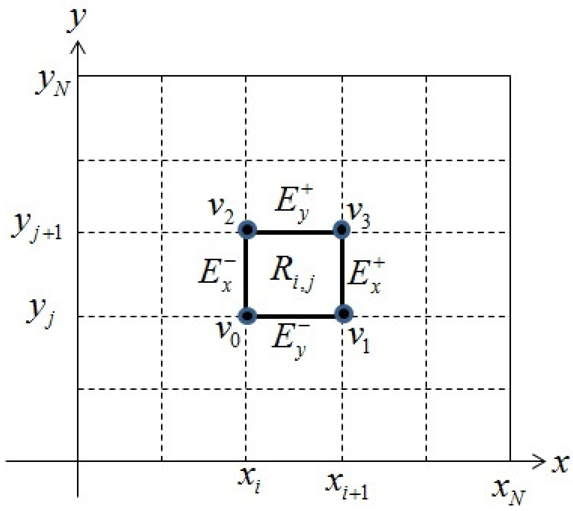

Fig. 1. Vertices and edges of the region $R_{i,j}$ in a rectangular partitioned space.

Similarly, for polar portioning of the motion space, consider the motion space as a circle with the radius of R_m. With the aid of the partitioning curves $\{r_i = \frac{R_m}{N_a-1}(i-1),\ i = 1, ..., N_a\}$ and $\{\theta_j = \frac{2\pi}{N_b-1}(j-1),\ j = 1, ..., N_b\}$, this circle can be partitioned into $(N_a - 1)(N_b - 1)$ partitioning elements. An element $R_{i,j} = \{p = (r, \theta)|\, r_i \le r \le r_{i+1},\ \theta_j \le \theta \le \theta_{j+1}\}$ has four vertices, v_0, v_1, v_2, v_3, four edges, E_r^+, E_r^-, E_θ^+, E_θ^-, and correspondingly, four outer normal vectors n_r^+, n_r^-, n_θ^+, n_θ^- (Fig. 2).

For these partitioned spaces, $\Im(\tilde{*}) = *$ relates the label $\tilde{*}$ to the set $*$. This partitioned space can be captured by the equivalence relation $Q = \{(x_1, x_2)|\exists\tilde{*}\ s.t.\ x_1, x_2 \in \Im(\tilde{*})\}$, where $*$ is one of

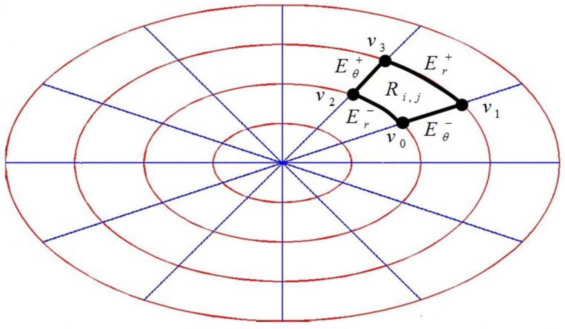

Fig. 2. Vertices and edges of the region $R_{i,j}$ in a polar partitioned space.

the above-mentioned partitioning elements. Correspondingly, $\pi_Q(x) = \tilde{*}\ s.t.\ x \in *$ and $\Im(\tilde{*}) = *$, where $\pi_Q(x)$ is a projection map.

In these partitioned spaces, let's define V_r as the set of all vertices of the partitioning elements, P as the perimeter of the motion space in which the vertices are excluded, and W as the exterior of the motion space. Also consider the *detection element* $d([i, j], [i', j']) = R_{i,j} \cap R_{i',j'} - V_r$, which is defined for two adjacent regions $R_{i,j}$ and $R_{i',j'}$ (the order is not important). With this procedure, the whole space will be partitioned into $V_r \cup R_{i,j} \cup d([i, j], [i', j']) \cup P \cup W$. Correspondingly, consider $\tilde{V}_r$, $\tilde{R}_{i,j}$, $\tilde{d}([i, j], [i', j'])$, $\tilde{P}$, and $\tilde{W}$ as the labels for these partitioning elements.

III. Problem description

Consider a robot with the dynamics $\dot{X}(t) = f(X(t), u(t))$ where X is the robot position and u is the control input. For the motion control of this robot, the motion space can be partitioned into several disjoint regions which are separated by hypersurfaces. Our objective here is to construct a hybrid controller to drive the robot through the partitioned space to satisfy a given specification. Let R_1, R_2,..., and R_n be the elements of the partitioned space, and correspondingly $\tilde{R}_1$, $\tilde{R}_2$,...,$\tilde{R}_n$ as the finite set of symbols that label these elements, where $\Im(\tilde{R}_i) = R_i$. The motion planning objective may require the robot to visit particular regions with a specific order while avoiding some other regions which can be specified either by a sequence of events in the form of an automaton or by a LTL formula [8]. A LTL formula over the set of propositions $P = \{\tilde{R}_1, \tilde{R}_2, ..., \tilde{R}_n\}$ can be constructed using the combination of traditional logical operators including negation ($\neg$), disjunction ($\bigvee$), conjunction ($\bigwedge$), and the temporal operators including next (O), until(U), eventually ($\diamond$), always ($\Box$), and release (R). For example the formula $\diamond\tilde{R}_1 \bigwedge \diamond\tilde{R}_2$ means that the robot will eventually reach region R_1 and will eventually reach region R_2. Now, the robot motion planning and control problem can be described as follows:

Problem 1 *Given the system dynamics as $\dot{X}(t) = f(X(t), u(t))$ and the desired specification, construct the hybrid controller to generate the control signal $u(t)$ such that starting from any point inside the set of initial states X_0, tehn visited regions by the robot trajectory $X(t)$ satisfy the given specification.*

To address this problem, we propose a hierarchical hybrid controller (Fig. 3) in which a discrete supervisor commands the system such that closed-loop system satisfies the formula ϕ over the partitioned space. This discrete supervisor cannot be directly connected to the plant with continuous dynamics. Hence, an interface layer is introduced which converts the discrete commands of the supervisor, u_d, to the continuous form, $u(t)$, to be applied to the plant. It also translates the continuous signals of the plant, $X(t)$, to discrete symbols, x_d, understandable by the supervisor. To construct this control hierarchy, we first need to rigorously describe the partitioning of the motion space, and then, bisimilarly abstract the system to a finite state machine to be able to design the discrete supervisor.

IV. Robot Motion Control over a Partitioned Space

To address the above mentioned problem over the partitioned space, we will develop a control mechanism, by which starting from any point inside a region, the robot moves to a unique destination region on its neighbourhood. In this case, the system can be bisimilarly abstracted to a finite state machine, and the reachability problem for such a system becomes decidable [19]. The decidability property desponds on both the system dynamics and the partitioning style. For a rectangularly or polarly partitioned space, a system with multi-affine dynamics is decidable [20, 17]. A multi-affine function $f : \mathbb{R}^n \to \mathbb{R}^m$, has the property that for any $1 \le i \le n$ and any $a_1, a_2 \ge 0$ with $a_1 + a_2 = 1$, $f(x_1, ..., (a_1 x_{i_1} + a_2 x_{i_2}), x_{i+1}, ...x_n) = a_1 f(x_1, ..., x_{i_1}, x_{i+1}, ...x_n) + a_2 f(x_1, ..., x_{i_2}, x_{i+1}, ...x_n)$. In a rectangular and polar partitioned space, this property allows us to find the value of a multi-affine vector field at any point inside a partition just based on the values of the vector field at its vertices. This property has been formally described in the following proposition.

Lemma 1 *(Adopted from [16] and [21]) Given a multi - affine function $g(X)$ defined over a partitioned element $R_{i,j}$, the function g can be uniquely described based on the values of g at vertices of $R_{i,j}$ as follows:*

$$\forall X = (a, b) \in \bar{R}_{i,j} : g(X) = \sum_{m=0}^{3} \lambda_m g(v_m), \qquad (1)$$

where v_m, $m = 0, ..., 3$, are the vertices of the element $R_{i,j}$. The coefficients λ_m, can be obtained uniquely as follows:

$$\lambda_m = \lambda_a^{m_a}(1 - \lambda_a)^{1-m_a} \lambda_b^{m_b}(1 - \lambda_b)^{1-m_b}, \qquad (2)$$

where m_a, m_b, are the corresponding binary digits of the index m.

For rectangular partitioning, λ_a and λ_b can be found as follows:

$$\lambda_a = \frac{a - x_i}{x_{i+1} - x_i} \qquad \lambda_b = \frac{b - y_j}{y_{j+1} - y_j} \qquad (3)$$

and for polar partitioning,

$$\lambda_a = \frac{a - r_i}{r_{i+1} - r_i} \qquad \lambda_b = \frac{b - \theta_j}{\theta_{j+1} - \theta_j} \qquad (4)$$

In this theorem, it can be verified that $\lambda_m \ge 0$, and $\sum_m \lambda_m = 1$. Also, since the above theorem holds true for all points in $\bar{R}_{i,j}$, the theorem can be also applied to the points on the edges.

Now, using these properties, for a system with multi-affine dynamics it is possible to construct multi-affine controllers to either keep the system's trajectory inside the region (invariant region) or to push it out from the desired edge (exit edge) as it is described in the following two lemmas.

Lemma 2 *[16] (Constructing an invariant region) For a continuous multi-affine vector field $\dot{X} = h(X, u(X)) = g(X)$, the region $R_{i,j}$ is an invariant region if there exists a controller u, such that for each vertex v_m, $m = 0, 1, 2, 3$, with incident edges $E_q^s \in E(v_m)$, and corresponding outer normals n_q^s we have $U_m(Inv) = \{u | {n_q^s}^T . g(v_m) < 0,\ for\, all\, E_q^s \in E(v_m)\} \neq \emptyset$.*

Lemma 3 *[12] (Constructing an exit edge) For a continuous multi-affine vector field $\dot{X} = h(X, u(X)) = g(X)$, the edge E_q^s with the outer normal n_q^s, is an exit edge if there exists a controller u, such that for each vertex v_m, $m = 0, 1, 2, 3$, we have $U_m(Ex(F_q^s)) = \{u \in \mathbb{R}^2 | {n_q^s}^T . g(v_m) > 0,\ for\, all\, v_m\ and\ {n_{q'}^{s'}}^T . g(v_m) < 0,\ for\ all\ E_{q'}^{s'} \neq E_q^s,\ v_m \in V(E_{q'}^{s'})\} \neq \emptyset$.*

Next proposition shows that if we construct a controller based on Lemma 3, all of the points on an exit edge are reachable.

Proposition 1 *For a continuous multi-affine vector field $\dot{X} = h(X, u(X)) = g(X)$, in a region $R_{i,j}$ with the exit edge E_q^s constructed by Lemma 3, all $y \in E_q^s \setminus E$ are reachable from a point inside the region $R_{i,j}$.*

Proof: Respecting the condition of Lemma 3 for the points on the exit edge E_q^s, we will have $n_q^s(y)^T.g(y) > 0$, $\forall y \in E_q^s$. This strictly positive inequality guarantees that the trajectories that leave the region do not return back any more. In addition, it shows that the points on the exit edge are not reachable from other points on the edge. Therefore, $y \in E_q^s$ is not reachable form an adjacent region or from another point on E_q^s. Then, considering $n_q^s(y)^T.g(y) > 0$, by continuity of g, it can be concluded that there is a point inside the region $R_{i,j}$ on the neighborhood of y from which y is reachable. ∎

With these controllers defined over the partitioned space, it is possible to drive the system's trajectory to one of the adjacent regions or to keep it inside the current region. This system can be captured by a transition system $T_Q = (X_Q, X_{Q_0}, U_Q, \rightarrow_Q, Y_Q, H_Q)$, where

- $X_Q = V_r \cup R_{i,j} \cup d([i,j],[i',j']) \cup P \cup W$ is the set of system's states, where $1 \leq i, i' \leq N_a - 1$, $1 \leq j, j' \leq N_b - 1$.
- $X_{Q_0} \subseteq R_{i,j}$ is the set of initial states. Here, we assume that the system initially starts from the inside of the regions $R_{i,j}$.
- $U_Q = U_a \cup U_d$, where
  - $U_a = U_{ex} \bigcup \{C_0\}$ is the set of labels where the label C_0 corresponds to the the controller that make the region $R_{i,j}$ an invariant region. For rectangular partitioning, the set U_{ex} consists of the labels C_x^+, C_x^-, C_y^+, C_y^- correspond to the controllers that make the edges E_x^+, E_x^-, E_y^+, E_y^- the exit edges, respectively. For the polar partitioning, the set U_{ex} includes the labels C_r^+, C_r^-, C_θ^+, C_θ^- correspond to the controllers that make the edges E_r^+, E_r^-, E_θ^+, E_θ^- the exit edges, respectively.
    For these control labels, the sets of control actions that can be activated in this region are : $r(C_q^s) = \{u(X) | u(X) = \sum_m \lambda_m u(v_m)\ m = 0,1,2,3\ v_m \in V(R_{i,j}), u(v_m) \in U_m(Ex(F_q^s))\}$, and $r(C_0) = \{u(X) | u(X) = \sum \lambda_m u(v_m)\ v_m \in V(R_{i,j}), u(v_m) \in U_m(Inv)\}$, where λ_m can be obtained by (2).
  - $U_d = \{\hat{d}^+([i,j],\ [i',j'])\} \cup \{\hat{d}^-([i,j], [i',j'])\} \cup \{\hat{P}\}$ is the set of the detection events, where $1 \leq i, i' \leq N_a - 1$, and $1 \leq j, j' \leq N_b - 1$. The events $\hat{d}^+([i,j],[i',j'])$, $\hat{d}^-([i,j],[i',j'])$, and $\hat{P}$ respectively show that the detection element $d([i,j],[i',j'])$ is crossed in positive direction of x, y, r, or θ; the detection element $d([i,j],[i',j'])$ is crossed in negative direction of x, y, r, θ; and the perimeter of the partitioned motion space is crossed.
- $(X_1, X_2, v) \in \rightarrow_Q$, denoted by $X_1 \xrightarrow{v}_Q X_2$, if and only if one of the following conditions holds true:
  1. Actuation:
     - Exit edge: $v \in U_{ex}$; $\pi_Q(X_1) \neq \pi_Q(X_2)$; $\exists\, i,\, j,\, i',\, j'$ such that $\pi_Q(X_1) = \tilde{R}_{i,j}$ and $\pi_Q(X_2) = \tilde{d}([i,j],[i',j'])$, or $\pi_Q(X_2) = \tilde{P}$; Furthermore, $\exists \tau(finite)$ and $\varepsilon > 0$ such that $\psi(t) : [0, \tau + \varepsilon] \rightarrow \mathbb{R}^2$ is the solution of $\dot{X} = h(X, r(v))$, $\psi(0) = X_1$; $\psi(\tau) = X_2$, $\pi_Q(\psi(t)) = \pi_Q(X_1)$ for $t \in [0, \tau)$, and $\pi_Q(\psi(t)) \neq \pi_Q(X_1)$ for $t \in [\tau, \tau + \varepsilon]$. Here, $r(v)$ is the continuous controller corresponding to the control label v, which can be constructed as discussed above.
     - Invariant region: $v = C_0$; $\exists \tilde{R}_{i,j}$ such that $\pi_Q(X_1) = \pi_Q(X_2) = \tilde{R}_{i,j}$; $\psi(t) : \mathbb{R}^+ \rightarrow \mathbb{R}^2$ is the solution of $\dot{X} = h(X, r(v))$, $\psi(0) = X_1$, $\psi(\tau) = X_2$, and $\pi_Q(\psi(t)) = \pi_Q(X_1) = \pi_Q(X_2)\ for\ all\ t \geq 0$.
  2. Detection:
     - Crossing a detection element to enter to a new region:
       (a) $v \in \{\hat{d}^+([i,j],[i',j'])\} \subseteq U_d$; $\pi_Q(X_1) \neq \pi_Q(X_2)$; $\exists \tilde{R}_{i,j}$, $\tilde{R}_{i',j'}$, $\tilde{d}([i,j],[i',j'])$, $i' \geq i$, and $j' \geq j$ such that $\pi_Q(X_1) = \tilde{d}([i,j],[i',j'])$ and $\pi_Q(x_2) = \tilde{R}_{i',j'}$; $\exists 0 < \varepsilon < \tau$ and $\exists w \in \{C_x^+, C_y^+\}$ or $\exists w \in \{C_r^+, C_\theta^+\}$ such that $\psi(t) : [0, \tau] \rightarrow \mathbb{R}^2$ is the solution of $\dot{X} = h(X, r(w))$, $\psi(\varepsilon) = X_1$; $\psi(\tau) = X_2, \pi_Q(\psi(t)) = \tilde{R}_{i,j}$ for $t \in (0, \varepsilon)$, and $\pi_Q(\psi(t)) = \tilde{R}_{i',j'}\ for\ t \in (\varepsilon, \tau]$.

(b) $v \in \{\hat{d}^{-}([i,j],[i',j'])\} \subseteq U_d$; $\pi_Q(X_1) \neq \pi_Q(X_2)$; $\exists \tilde{R}_{i,j}$, $\tilde{R}_{i',j'}$, $\tilde{d}([i,j],[i',j'])$ $i' \leq i$, and $j' \leq j$ such that $\pi_Q(X_1) = \tilde{d}([i,j],[i',j'])$ and $\pi_Q(x_2) = \tilde{R}_{i',j'}$; $\exists 0 < \varepsilon < \tau$ and $\exists w \in \{C_x^-, C_y^-\}$ or $w \in \{C_r^-, C_\theta^-\}$ such that $\psi(t) : [0,\tau] \to \mathbb{R}^2$ is the solution of $\dot{X} = h(X, r(w))$, $\psi(\varepsilon) = X$; $\psi(\tau) = X_2, \pi_Q(\psi(t)) = \tilde{R}_{i,j}\ for\ t \in (0,\varepsilon)$, and $\pi_Q(\psi(t)) = \tilde{R}_{i',j'}\ for\ t \in (\varepsilon,\tau]$.

– Crossing the motion space's boundary: $v = \hat{P}$; $\pi_Q(X_1) = \tilde{P}$ and $\pi_Q(X_2) = \tilde{W}$; $\exists \tilde{R}_{i,j}$ and $\exists 0 < \varepsilon < \tau$ and $\exists w \in U_{ex}$ such that $\psi(t) : [0,\tau] \to \mathbb{R}^2$ is the solution of $\dot{X} = h(X, r(w))$, $\psi(\varepsilon) = X_1$; $\psi(\tau) = X_2, \pi_Q(\psi(t)) = \tilde{R}_{i,j}\ for\ t \in (0,\varepsilon)$, and $\pi_Q(\psi(t)) = \tilde{W}\ for\ t \in (\varepsilon,\tau]$.

- $Y_Q = X_Q$ is the output space.
- $H_Q : X \to Y_Q$ is the output map. Here, we have chosen $H_Q(X) = \pi_Q(X)$.

Remark 1 *The transition system T_Q captures only important transitions form one region to another region under the exit commands, or from one region to itself under the invariant controller, C_0.*

Analogous with [18], to model this partitioned system, we can define an interface layer which connects this partitioned system to a higher discrete supervision layer. The interface layer has two main blocks: Detector and Actuator. The detector converts continuous time signals to a sequence of symbols. Upon crossing partitioning hypersurfaces, plant symbols, $\hat{d}^+([i,j],[i',j'])$,$\hat{d}^-([i,j],[i',j'])$, and $\hat{P}$, will be generated, which inform the current situation of the plant to the supervisor. Based on the observed plant symbols, the supervisor decides which control signal should be injected to the plant to satisfy the desired specification. This command has a discrete nature and the control commands to the plant are continuous. The actuator, will translate these discrete commands to continuous signals. The block diagram of this control structure is shown in Fig. 3.

V. Abstraction over the partitioned space

In the partitioned system T_Q, although all important transitions have been captured, this transition

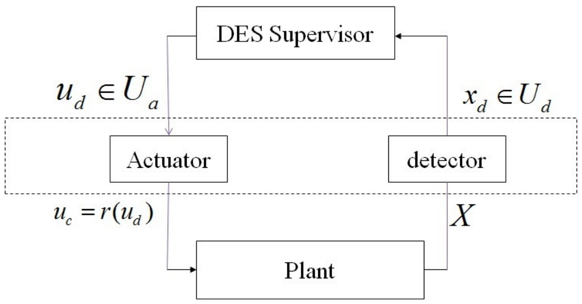

Fig. 3. The hierarchical hybrid control structure.

system still has infinite number of states which makes the control synthesis problem very difficult or even impossible. Abstraction [22] is a technique that reduces the number of states by aggregating similar states. Hence, using this strategy, and considering each partitioning element as one of the states in the abstracted model, the resulting model will be :
$T_\xi = (X_\xi, X_{\xi_0}, U_\xi, \to_\xi, Y_\xi, H_\xi)$, where

- $X_\xi = \{\tilde{R}_{i,j} | 1 \leq i \leq N_a - 1,\ 1 \leq j \leq N_b - 1\} \bigcup \{\tilde{d}([i,j],[i',j']) | 1 \leq i, i' \leq N_a - 1,\ 1 \leq j, j' \leq N_b - 1\} \bigcup \{\tilde{P}, \tilde{W}\}$. Note that since the system starts from a point inside the regions $R_{i,j}$ and due to strictly negative inequalities in Lemmas 2 and 3, the system trajectory never crosses the vertices, and hence, the set V_r does not need to be considered in the abstracted system.
- $X_{\xi_0} \subseteq \{\tilde{R}_{i,j} | 1 \leq i, i' \leq N_a - 1,\ 1 \leq j, j' \leq N_b - 1\}$.
- $U_\xi = U_a \cup U_d$ is like what we have in T_Q.
- $(r_1, r_2, v) \in \to_\xi$, denoted by $r_1 \xrightarrow{v}_\xi r_2$, if $\exists v \in U_\xi$, $X_1 \in \Im(r_1)$, $X_2 \in \Im(r_2)$ such that $X_1 \xrightarrow{v}_Q X_2$.
- $Y_\xi = X_\xi$.
- $H_\xi(r) = r$ is the output map.

With this method, the partitioned system, T_Q which previously was modelled by the regulation layer and the interface layer, now is abstracted to a finite state transition system T_ξ for which we can design a discrete supervisor [7] to achieve the desired specification. Then, with the aid of the interface layer, the designed supervisor for the abstract model can be applied to the original continuous model. To guarantee that the discrete supervisor for the abstract model can also work for the original continuous model, it is necessary that the abstract model and the original continuous model represent the same behavior which requires them to be bisimilar. A bisimulation relation between two transition systems can be formally defined as follows:

Definition 1 *[22] Given T_i=$(Q_i, Q_i^0, U_i, \rightarrow_i, Y_i, H_i)$, $(i = 1, 2)$, R is a bisimulation relation between T_1 and T_2, denoted by $T_1 \approx_R T_2$, iff:*

1. *$\forall q_1 \in Q_1^0$ then $\exists q_2 \in Q_2^0$ that $(q_1, q_2) \in R$. Also, $\forall q_2 \in Q_2^0$ then $\exists q_1 \in Q_1^0$ that $(q_1, q_2) \in R$.*
2. *$\forall q_1 \rightarrow_1 q_1'$, and $(q_1, q_2) \in R$ then $\exists q_2' \in Q_2$ such that $q_2 \rightarrow_2 q_2'$ and $(q_1', q_2') \in R$. Also, $\forall q_2 \rightarrow_2 q_2'$, and $(q_1, q_2) \in R$ then $\exists q_1' \in Q_1$ such that $q_1 \rightarrow_1 q_1'$ and $(q_1', q_2') \in R$.*

For multi-affine functions defined over a rectangular or polar partitioned space, and with the controllers which we defined to construct exit edges or to make a region invariant, the abstract model and the original partitioned system are bisimialr as proven in Theorem 1.

Theorem 1 *The original partitioned system, T_Q, and the abstract model, T_ξ, are bisimilar.*

Proof:
consider the relation $R = \{(q_Q, q_\xi) | q_Q \in X_Q,\ q_\xi \in X_\xi,$ and $q_Q \in \Im(q_\xi)\}$. We will show that this relation is a bisimulation relation between T_Q and T_ξ. To prove this bismulation relation we should verify both conditions of Definition 1.

To verify the first condition of the bisimulation relation in Definition 1, we know that for any $q_Q \in X_{Q_0}$ there exists a region $R_{i,j}$ such that $q_Q \in R_{i,j}$. For this region, there exists a label, $\tilde{R}_{i,j}$ such that $R_{i,j} = \Im(\tilde{R}_{i,j})$ and $\tilde{R}_{i,j} \in X_{\xi_0}$. Hence, $(q_Q, \tilde{R}_{i,j}) \in R$. Conversely, it can be similarly shown that for any $q_\xi \in X_{\xi_0}$, there exists a $q_Q \in X_{Q_0}$ such that $(q_\xi, q_Q) \in R$.

To verify the second condition of the bisimulation relation, following from the definition of T_ξ, we know that for any $(q_Q, q_\xi) \in R$ and $q_Q \xrightarrow{u}_Q q_Q'$, there exists a transition $q_\xi \xrightarrow{u}_\xi q_\xi'$, where $q_Q' \in \Im(q_\xi')$ or equivalently $(q_Q', q_\xi') \in R$. For the converse case, assume that $q_\xi \xrightarrow{u}_\xi q_\xi'$. According to the definition of R, all $x \in \Im(q_\xi)$ are related to q_ξ. Hence, to prove the second condition of the bisimulation relation, we should investigate it for all $x \in \Im(q_\xi)$. Based on the control construction procedure, the labels u, q_ξ, and q_ξ' can be the one of the following cases:

1. $u = C_0$ and $q_\xi = q_\xi'$. In this case, since the controller C_0 makes the region an invariant region (Proposition 2), all of the trajectories starting from any $q_Q \in \Im(q_\xi)$ will remain inside the region $\Im(q_\xi)$. Therefore, for any $q_Q \in \Im(q_\xi)$, there exists a $q_Q' \in \Im(q_\xi)$ such that $q_Q \xrightarrow{u}_Q q_Q'$ and $q_Q' = \Im(q_\xi')$.
2. $u \in U_{ex}$, $q_\xi \in \{\tilde{R}_{i,j} | 1 \le i \le N_a - 1,\ 1 \le j \le N_b - 1\}$, and $q_\xi' \in \{\tilde{d}([i,j],[i',j']) | 1 \le i, i' \le N_a - 1,\ 1 \le j, j' \le N_b - 1\}$ or $q_\xi = \tilde{P}$. In this case, based on Lemma 3 starting from any $q_Q \in \Im(q_\xi)$, the controller u drives the system trajectory towards the detection element $\Im(q_\xi')$. Therefore, for any $q_Q \in \Im(q_\xi)$, there exists a $q_Q' \in \Im(q_\xi')$ such that $q_Q \xrightarrow{u}_Q q_Q'$ and $q_Q' \in \Im(q_\xi')$.
3. $u \in \{\hat{d}^+([i,j],[i',j']) | 1 \le i, i' \le N_a - 1,\ 1 \le j, j' \le N_b - 1\} \subseteq U_d$, $q_\xi' \in \{\tilde{R}_{i',j'} | 1 \le i' \le N_a - 1,\ 1 \le j' \le N_b y - 1\}$, and $q_\xi \in \{\tilde{d}([i,j],[i',j']) | 1 \le i, i' \le N_a - 1,\ 1 \le j, j' \le N_b - 1\}$ such that $i' \ge i$ and $j' \ge j$. In this case, based on Lemma 1, for any $q_Q \in \Im(q_\xi) = d([i,j],[i',j'])$, there exists a controller $v \in \{C_x^+, C_y^+\}$ or $v \in \{C_r^+, C_\theta^+\}$ that has led the trajectory of the system from the region $R_{i,j}$ to the point q_Q on the detection element $d([i,j],[i',j'])$. Since $R_{i',j'}$ is the unique adjacent region of the element $R_{i,j}$, common in the detection element $d([i,j],[i',j'])$, based on the definition of the controller for the exit edge and Lemma 3, the controller v leads the trajectory of the system to a point inside the region $R_{i',j'}$ so that the detection event $u = \hat{d}^+([i,j],[i',j'])$ is generated. Therefore, for any $q_Q \in \Im(q_\xi)$, there exists a $q_Q' \in \Im(q_\xi')$ such that $q_Q \xrightarrow{u}_Q q_Q'$. Similar explanation can be provided for the case $u \in \{\hat{d}^-([i,j],[i',j']) | 1 \le i, i' \le N_a - 1,\ 1 \le j, j' \le N_b - 1\}$ or $u = \hat{P}$.

In all of the above mentioned cases, the second condition of the bisimulation relation for the converse case holds true. Since both conditions of the bismulation relation hold, T_ξ and T_Q are bisimilar. ■

VI. Adopting the DES supervisory control to the abstracted model

For the abstracted model with finite number of states we can design a discrete supervisor using Discrete Event Systems (DES) supervisory control theory initiated by Ramadge and Wonham [7]. Formally, the finite state machine model of the abstracted system can be represented by an automaton $G = (Q, \Sigma, \alpha, Q_0, Q_m)$, where $Q = Q_\xi$ is the set of states; $Q_0 = Q_{\xi_0} \subseteq Q$ is the set of initial states; $\Sigma = U_a \cup U_d$ is the (finite) set of events; $Q_m \subseteq Q$ is the set of final (marked) states, and $\alpha : Q \times \Sigma \to Q$ is the transition function which is a partial function and determines the possible transitions in the system caused by different events. Based on the transitions in T_ξ, the function α can be defined as follows:

$$\alpha(\tilde{R}_{i,j}, \sigma) =$$

$$\begin{cases} \tilde{R}_{i,j} & if\ \sigma = C_0 \\ \tilde{d}([i,j],[i+1,j]) & if\ \sigma = C_x^+,\ C_r^+\ and\ i \neq N_a - 1 \\ \tilde{d}([i,j],[i-1,j]) & if\ \sigma = C_x^-,\ C_r^-\ and\ i \neq 1 \\ \tilde{d}([i,j],[i,j+1]) & if\ \sigma = C_y^+,\ C_\theta^+\ and\ j \neq N_b - 1 \\ \tilde{d}([i,j],[i,j-1]) & if\ \sigma = C_y^-,\ C_\theta^-\ and\ j \neq 1 \\ \tilde{d}([i,j],[i,N_b-1]) & if\ \sigma = C_\theta^-\ and\ j = 1 \\ \tilde{d}([i,j],[i,1]) & if\ \sigma = C_\theta^+\ and\ j = N_b - 1 \\ \tilde{P} & if\ \sigma = C_x^+,\ C_r^+,\ i = N_a - 1;\ \sigma = C_x^-, \\ & i = 1; \sigma = C_y^+,\ j = N_b - 1,\ or \\ & \sigma = C_y^-,\ j = 1 \\ \alpha(\tilde{d}([i,j],[i',j',]),\sigma) = \tilde{R}_{i',j'} & \\ & if\ \sigma = \hat{d}^+([i,j],[i',j']), i' \geq i,\ j' \geq j, \\ & or\ \sigma = \hat{d}^-([i,j],[i',j']),\ i' \leq i,\ j' \leq j \\ \alpha(\tilde{P},\hat{P}) = \tilde{W} & \end{cases}$$

In this automaton, the sequence of events generates a string. ε is an empty string, and Σ^* is the set of all possible strings over the set Σ including ε. The function α can be extended from acting on events to acting on the strings as $\alpha_{ext} : Q \times \Sigma^* \to Q$ in which $\alpha_{ext}(q,\varepsilon) = q$ and $\alpha_{ext}(q, s\sigma) = \alpha(\alpha_{ext}(q,s),\sigma)\ \forall\ s \in \Sigma^*$ and $\sigma \in \Sigma$. The language of the automaton is a sequence of strings that can be generated by G and can be defined as $L(G) = \{s \in \Sigma^* | \exists q_0 \in Q_0\, s.t.\ \alpha_{ext}(q_0,s)\ is\ defined.\}$. The marked language, denoted by $L_m(G)$ consists of the strings that can be generated by the automaton G and end with the marked states, which formally can be defined as $L_m(G) = \{s \in \Sigma^* | \exists q_0 \in Q_0\, s.t.\ \alpha_{ext}(q_0,s)\ is\ defined\ and\ \alpha_{ext}(q_0,s) \in Q_m\}$. The event set Σ consists of two types of events: the controllable event set $\Sigma_c = U_a$ and the uncontrollable event set $\Sigma_{uc} = U_d$. The controllable events are those that can be disabled or enabled by an external supervisor; however, the uncontrollable events cannot be affected by the supervisor. Playing with the controllable events, the supervisor can modify the plant's generable language so that $\varepsilon \in L(S/G)$ and $[(s \in L(S/G)) \wedge (s\sigma \in L(G)) \wedge (\sigma \in L(S))] \Leftrightarrow [s\sigma \in L(S/G)]$. Accordingly, the closed-loop marked language will be $L_m(S/G) = L(S/G) \bigcap L_m(G)$. This supervisor can be used to achieve a controllable language specification. A language specification K is said to be controllable with respect to the language of the plant G and set of uncontrollable events E_{uc} if $\forall s \in \overline{K},\ e \in E_{uc}, se \in L(G) \Rightarrow se \in \overline{K}$. To realize this control strategy and to combine the plant discrete model and the supervisor, we can use parallel composition [23] which is a binary operation between two automata. Next theorem shows how the parallel composition can be used to modify the plant language to achieve a desirable specification given in terms of a controllable language.

Theorem 2 *[23] Let G be the plant and $K \subseteq \Sigma^*$ be a desired language. If $\emptyset \neq K = \bar{K} \subseteq L(G)$ and K is controllable, there exist a nonblocking supervisor S such that $L(S/G) = L(S||G) = K$. In this case, S could be any automaton that satisfies $L_m(S) = L(S) = K$.*

VII. Illustrative example for symbolic motion planning and control for the formation control of unmanned helicopters

Formation control is the jointly movement of a group of agents with a relatively fixed distances, and has been addressed by different methods (see e.g. [24], [25], [26], [27], [28]). Here, we use our proposed method of symbolic motion planning for formation control of a team of unmanned aerial vehicles. The team consists of two UAV helicopters, HeLion and SheLion which are developed by our research group at the National University of Singapore. The modelling and low level control structure of the NUS UAV helicopters are explained in [29], [30], [31]. For the regulation layer of these helicopters we have proposed a two-layer control structure in which the inner-loop controller stabilizes the system using H_∞ control design techniques, and their outer-loop is used to derive the system towards the desired location. As it has been discussed in [30], in this control structure, the inner-loop is fast enough to track the given references, so that the outer-loop dynamics can be approximately described as follows:

$$\dot{x} = u,\ x \in \mathbb{R}^2,\ u \in U \subseteq \mathbb{R}^2, \tag{5}$$

where x is the position of the UAV; u is the UAV velocity reference generated by the formation algorithm, and U is the velocity constraint set, which is a convex set.

Now, in a leader follower formation scenario, consider the follower velocity in the following form:

$$V_{follower} = V_{leader} + V_{rel}. \tag{6}$$

In this relative framework, consider a circle with the radius of R_m that is centered at the desired position

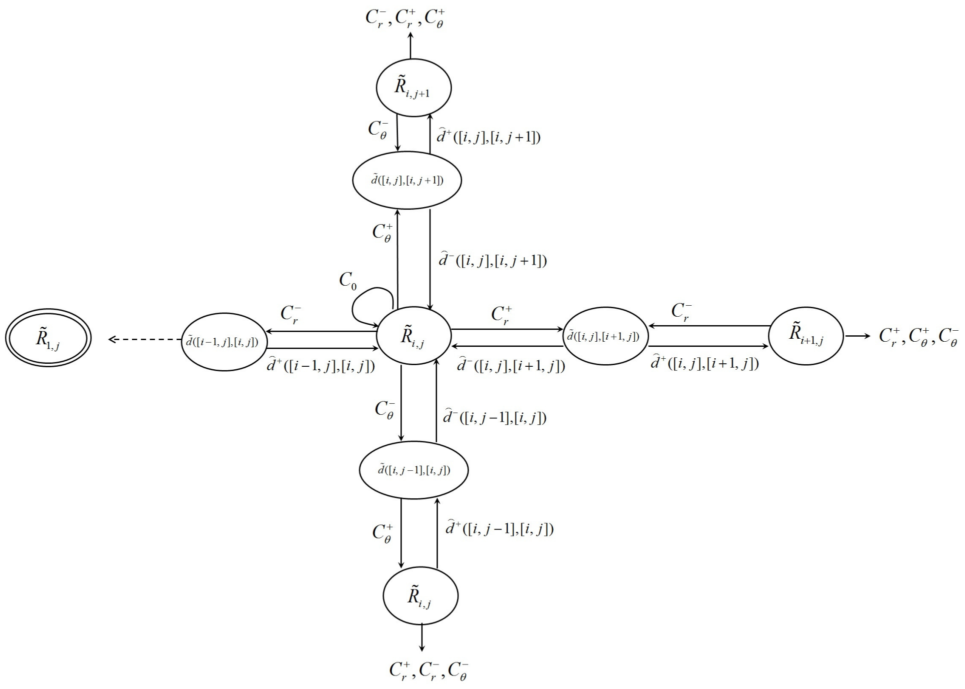
C_r^-, C_r^+, C_θ^+
$\tilde{R}_{i,j+1}$
C_θ^-
$\tilde{d}^+([i,j],[i,j+1])$
$\tilde{d}([i,j],[i,j+1])$
C_θ^+
$\tilde{d}^-([i,j],[i,j+1])$
C_0
$\tilde{R}_{1,j}$
$\tilde{d}([i-1,j],[i,j])$
C_r^-
$\tilde{d}^+([i-1,j],[i,j])$
$\tilde{R}_{i,j}$
C_r^+
$\tilde{d}^-([i,j],[i+1,j])$
$\tilde{d}([i,j],[i+1,j])$
C_r^-
$\tilde{d}^+([i,j],[i+1,j])$
$\tilde{R}_{i+1,j}$
$C_r^+, C_\theta^+, C_\theta^-$
C_θ^-
$\tilde{d}^-([i,j-1],[i,j])$
$\tilde{d}([i,j-1],[i,j])$
C_θ^+
$\tilde{d}^+([i,j-1],[i,j])$
$\tilde{R}_{i,j}$
C_r^+, C_r^-, C_θ^-

Fig. 4. An abstract model for the UAV motion dynamics over a partitioned space.

of the follower, and is partitioned as discussed in Section II. A part of the discrete abstracted model of the follower motion dynamics is shown in Fig. 4.

For these helicopters, our aim is to design the formation controller to generate the relative velocity of the follower, V_{rel}, such that starting from any initial point inside the control horizon, it eventually reaches the desired relative distance with respect to the leader. Moreover, after reaching the formation, the follower UAV should remain at the desired position. Using the proposed polar partitioning approach, the formation can be achieved if the controller drives the system directly to the regions $R_{1,j}$, $1 \leq j \leq n_\theta - 1$. After reaching the formation by this scenario, the controller should maintain the state of the system at the final state to keep the formation by activating the command C_0. This specification can be achieved by the supervisor, S_F, as shown in Fig. 5.

To verify the algorithm and to monitor how the follower can reach the formation and maintain the achieved formation, a flight test has been conducted in which the leader tracks a line path, and the follower should reach and keep the formation. In this flight test, the control horizon R_m is 50 meter, $N_a = 10$, and $N_b = 20$. The follower is initially located at a point which has a relative distance of $(dx, dy) = (-17.8m, 11.4m)$ with respect to the desired position and the distance between the desired position and the leader is $(dx, dy) = (-5m, -15m)$.

The position of the UAVs in x-y plane is shown in Fig. 6. The relative distance of the follower UAV from the desired position is shown in Fig. 7. As it can be seen the follower UAV has finally reached the first circle and then, it has been able to maintain the formation.

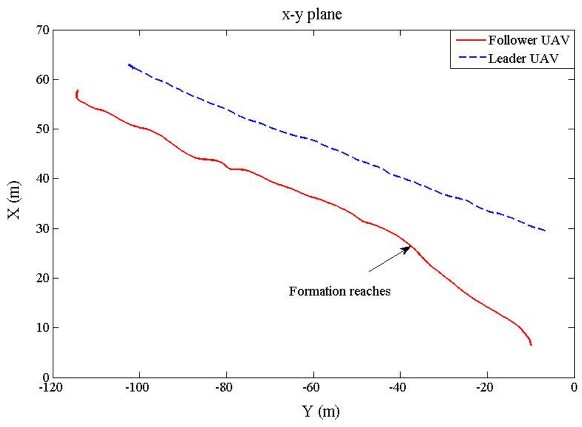

Fig. 6. The position of the UAVs in the x-y plane.

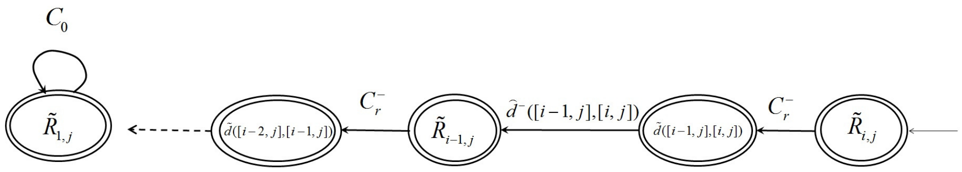

Fig. 5. The supervisor, S_F, for the motion control of the follower involved in a formation mission.

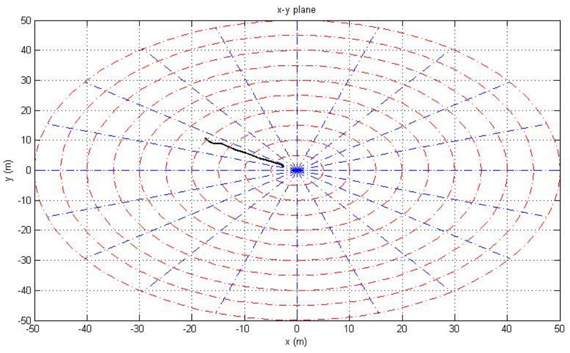

Fig. 7. The distance of the follower from the desired position.

VIII. Conclusion

In this paper, a hybrid framework was proposed for the symbolic motion planning and control of robots. The approach was based on rectangular and polar partitioning of the motion space and then, abstracting the original continuous system with infinite number of states to a finite state machine. To implement the idea, a multi-layer control structure was proposed in which the discrete supervisor was connected to the plant via an interface layer. The continuous plant and the interface layer together were shown to be bisimilar with the abstract model. This bismilarity let us apply the discrete supervisor which was designed for the abstract model to the continuous plant while the closed-loop behavior does not change. The algorithm was successfully applied to the formation control of unmanned helicopters.

REFERENCES

1. P. Antsaklis, A. Nerode, Hybrid control systems: An introductory discussion to the special issue, IEEE Transactions on Automatic Control, 43 (4) (1998) 457–460.
2. H. Kress-Gazit, G. Fainekos, G. Pappas, Where's waldo? sensor-based temporal logic motion planning, in: IEEE International Conference on Robotics and Automation, 2007, pp. 3116–3121.
3. M. Kloetzer, C. Belta, Temporal logic planning and control of robotic swarms by hierarchical abstractions, Robotics, IEEE Transactions on 23 (2) (2007) 320–330.
4. C. Belta, A. Bicchi, M. Egerstedt, E. Frazzoli, E. Klavins, G. Pappas, Symbolic planning and control of robot motion, Robotics and Automation Magazine, 14 (1) (2007) 61–70.
5. P. Tabuada, An approximate simulation approach to symbolic control, IEEE Transactions on Automatic Control, 53 (6) (2008) 1406–1418.
6. G. Pola, P. Tabuada, Symbolic models for linear control systems with disturbances, in: 46th IEEE Conference on Decision and Control, 2007, pp. 432–437.
7. P. Ramadge, W. Wonham, The control of discrete event systems, Proceedings of the IEEE 77 (1) (1989) 81–98.
8. E. Emerson, Temporal and modal logic, Handbook of theoretical computer science, 2 (1990) 995–1072.
9. P. Tabuada, G. Pappas, Linear time logic control of discrete-time linear systems, IEEE Transactions on Automatic Control, 51 (12) (2006) 1862–1877.
10. P. Tabuada, G. Pappas, From discrete specifications to hybrid control, in: 42nd IEEE Conference on Decision and Control, Vol. 4, 2003, pp. 3366–3371.
11. G. Fainekos, H. Kress-Gazit, G. Pappas, Temporal logic motion planning for mobile robots, in: IEEE International Conference on Robotics and Automation, 2005, pp. 2020–2025.
12. C. Belta, L. Habets, Controlling a class of nonlinear systems on rectangles, IEEE Transactions on Automatic Control, 51 (11) (2006) 1749–1759.
13. C. Belta, V. Isler, G. Pappas, Discrete abstractions for robot motion planning and control in polygonal environments, IEEE Transactions on Robotics, 21 (5) (2005) 864–874.
14. G. Fainekos, H. Kress-Gazit, G. Pappas, Hybrid controllers for path planning: A temporal logic approach, in: 44th IEEE Conference on Decision and Control, 2005, pp. 4885–4890.

15. A. Karimoddini, H. Lin, B. M. Chen, T. H. Lee, Hybrid three-dimensional formation control for unmanned helicopters, Automatica 49 (2) (2013) 424–433.
16. A. Karimoddini, H. Lin, B. Chen, T. Heng Lee, Hybrid formation control of the unmanned aerial vehicles, Mechatronics, 21 (5) (2011) 886–898.
17. A. Karimoddini, H. Lin, Hybrid symbolic control for robot motion planning, in: Control and Automation (ICCA), 2013 10th IEEE International Conference on, 2013, pp. 1650–1655. `doi:10.1109/ICCA.2013.6565125`.
18. X. Koutsoukos, P. Antsaklis, J. Stiver, M. Lemmon, Supervisory control of hybrid systems, Proceedings of the IEEE, 88 (7) (2000) 1026–1049.
19. T. Henzinger, P. Kopke, A. Puri, P. Varaiya, What's decidable about hybrid automata?, Journal of Computer and System Sciences, 57 (1) (1998) 94–124.
20. G. Lafferriere, G. Pappas, S. Yovine, A new class of decidable hybrid systems, Hybrid Systems: Computation and Control, (1999) 137–151.
21. L. Habets, M. Kloetzer, C. Belta, Control of rectangular multi-affine hybrid systems, in: 45th IEEE Conference on Decision and Control, 2006, pp. 2619–2624.
22. R. Alur, T. Henzinger, G. Lafferriere, G. Pappas, Discrete abstractions of hybrid systems, Proceedings of the IEEE, 88 (7) (2000) 971 –984.
23. R. Kumar, V. K. Garg, Modeling and Control of Logical Discrete Event Systems, Vol. 300 of The Springer International Series in Engineering and Computer Science, Springer, 1995.
24. Y. Zhao, Z. Duan, G. Wen, Y. Zhang, Distributed finite-time tracking control for multi-agent systems: An observer-based approach, Systems and Control Letters 62 (1) (2013) 22 – 28.
25. F. Giulietti, M. Innocenti, M. Napolitano, L. Pollini, Dynamic and control issues of formation flight, Aerospace Science and Technology 9 (1) (2005) 65–71.
26. Y. Zhao, G. Wen, Z. Duan, X. Xu, G. Chen, A new observer-type consensus protocol for linear multi-agent dynamical systems, Asian Journal of Control 15 (2) (2013) 571–582.
27. D. M. Stipanovic, G. Inalhan, R. Teo, C. J. Tomlin, Decentralized overlapping control of a formation of unmanned aerial vehicles, Automatica 40 (8) (2004) 1285 – 1296.
28. G. Hassan, K. Yahya, I. ul Haq, Leader-follower approach using full-state linearization via dynamic feedback, in: Emerging Technologies, ICET '06. International Conference on, 2006, pp. 297 –305.
29. G. Cai, B. M. Chen, K. Peng, M. Dong, T. H. Lee, Modeling and control system design for a uav helicopter, in: 14th IEEE Mediterranean Conference on Control and Automation, 2006, pp. 1–6.
30. A. Karimoddini, G. Cai, B. M. Chen, H. Lin, T. H. Lee, Hierarchical Control Design of a UAV Helicopter," in Advances in Flight Control Systems, INTECH, Vienna, Austria, 2011.
31. A. Karimoddini, G. Cai, B. M. Chen, H. Lin, T. H. Lee, Multi-layer flight control synthesis and analysis of a small-scale uav helicopter, in: IEEE Conference on Robotics Automation and Mechatronics, 2010, pp. 321–326.