(Preprint) AAS XX-XXX

DEVELOPMENT OF A DECENTRALIZED COOPERATIVE CONTROL TECHNIQUE FOR COLLISION AVOIDANCE OF UNMANNED HELICOPTERS

Ali Karimoddini[*], Mohammad Karimadini[†], Hai Lin[‡], and Abdollah Homaifar[§]

Collision avoidance is a safety mechanism essentially to be embedded in the control structure of a team of autonomous vehicles. This is particularly important when the team is involved in cooperative tasks in which the team members should move with a relatively close distance. This paper develops a decentralized control structure for the collision avoidance of autonomous unmanned helicopters. Here, collision avoidance is considered as a safety specification which will be cooperatively satisfied by the team members. This safety specification is decomposed into local specifications, and then, local supervisors are designed to decentralizedly control each helicopter while it can be guaranteed that the collision avoidance as a global specification is achieved. The developed top-down control technique is embedded in the control structure of unmanned helicopters that are involved in a formation mission. Hardware-in-the-loop simulation results demonstrate the effectiveness of the algorithm.

INTRODUCTION

The control design of Unmanned Aerial Vehicles (UAVs) has emerged as an attractive research area, due to various military and civilian applications such as terrain and utility inspections, coordinated surveillance, search and rescue missions, disaster monitoring, rapid emergency response, aerial mapping, traffic monitoring, cinematography, and reconnaissance missions,[1],[2].[3]

The capabilities of the UAVs would be expanded when they form a team. A team of robots, taking a cooperative structure, is more robust against the failures in team members or in communication links,[4].[5] Adopting a team of cooperative UAVs with a proper coordination and tasking mechanism,,[6] results in a more powerful, flexible and efficient set of functionalities compare to a single sophisticated UAV.

Nevertheless, a cooperative team of UAVs is more complex to analyze and synthesize. Furthermore, for a safe operation, it is essential to consider a collision avoidance mechanism, particularly when the team members move with a relatively close distance. There are different approaches for collision avoidance such as geometry approaches, predictive control, probabilistic methods, and invariant sets,[7],[8],[9] and,[10] which mostly suffer from high computation cost, and and difficulty in decentralized implementation.

[*]Department of Electrical and Computer Engineering, North Carolina Agricultural and Technical State University, Greensboro, USA, akarimod@ncat.edu.
[†]Department of Electrical Engineering, Arak University of Technology, Arak, Iran, karimadini@arakut.ac.ir.
[‡]Department of Electrical Engineering, University of Notre Dame, Notre Dame, USA, hlin1@nd.edu.
[§]Department of Electrical and Computer Engineering, North Carolina Agricultural and Technical State University, Greensboro, USA, homaifar@ncat.edu.

In this paper we develop a decentralized control structure for the collision avoidance of autonomous unmanned helicopters, so that the team of UAVs as whole, can cooperatively satisfy the collision avoidance specification as a global goal. The decentralized control structure helps us to take the advantage of distributed computational load among the agents and to increase the reliability of the system against the possible failures. We consider the collision avoidance as a global safety specification to be achieved by the decision making units of the UAVs, which can be modelled as discrete event systems (DES).[11] This safety specification is decomposed into local event sets, and then, local supervisors are designed to decentralizedly control each UAV. The proposed top-down design technique can guarantee that the collision avoidance, as a global specification, will be satisfied through this decomposition-based local synthesis of the supervisors. The developed collision avoidance mechanism is then embedded in the control structure of unmanned helicopters that are involved in a formation mission. Hardware-in-the-loop simulations results demonstrate the effectiveness of the proposed algorithm.

PROBLEM FORMULATION

The decision making of autonomous systems can be effectively modeled by Discrete Event Systems where the sequence of events shows the behavior of the system. The events can be seen as commands or discrete observations which cause changes in the operation mode of the system. A capable tool to model such a discrete event system is automaton. An automaton can be formally defined as follows:

Definition 1 *(Automaton).[12] A deterministic automaton is a tuple $A := (Q, q_0, E, \delta, Q_m)$ consisting of a set of states Q; an initial state $q_0 \in Q$; a set of events E that causes transitions between the states, and a transition relation $\delta \subseteq Q \times E \times Q$ (with a partial map $\delta : Q \times E \to Q$), such that $(q, e, q') \in \delta$ if and only if state q is transited to state q' by event e, denoted by $q \xrightarrow{e} q'$ (or $\delta(q, e) = q'$). $Q_m \subseteq Q$ represents the marked states to assign a meaning of accomplishment to some states. For supervisor automaton whose all states are marked, Q_m is omitted from the tuple.*

For an automaton, a sequence of these events forms a string. We use ε to denote an empty string, and Σ^* to denote the set of all possible strings over the set Σ including ε. The language of the automaton G, denoted by $L(G)$, is the set of all strings that can be generated by G, starting from the initial states. The marked language, $L_m(G)$, is the set of strings that belong to $L(G)$ and end to the marked states.

Now, consider two UAVs: UAV_1 and UAV_s. For UAV_1, the discrete model of the system can be described by the automaton $A_1 = (Q_1, q_{01}, E_1, \delta_1, Q_{m1})$ whose set of discrete states is $Q_1 = \{N, C\}$, and its event set is $E_1 = \{NO_1, CA_{12}, CA_{21}, R_{12}, R_{21}, stop_1, stop_2, RCA_1\}$. Each UAV has two operation modes (states): N and C which stand for $Normal$ mode and $Collision$ mode, respectively. For the DES model of UAV_1, when there is no collision alarm, the UAV can continue its normal operation which is captured by the event NO_1. But, when UAV_2 enters the alarm zone of UAV_1, or when UAV_1 enters the alarm zone of UAV_2, the events CA_{12} and CA_{21} will appear respectively. In this case, a set of actions may need to be taken. For example, UAV_2 might need to be stopped by the command $stop_2$ so that UAV_1 can move to exit to remove the collision by the command RCA_1, or vice versa, UAV_1 may need to be stopped by the command $stop_1$, so that UAV_2 can handle the situation. After handling the collision alarm, if UAV_1 had stopped, it can be released by the command R_{21}, and if UAV_2 had stopped, it can be released by the

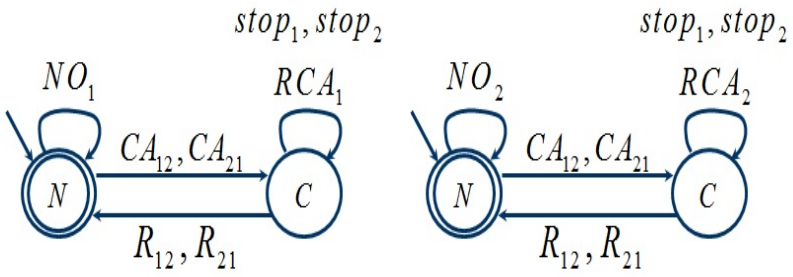


Figure 1. (a) DES model of UAV_1. (b) DES model of UAV_2.

command R_{12}. A similar model can be provided for UAV_2. For simplicity, we assume that all of the events are controllable, meaning that they can be enabled or disabled by an external supervisor. The automaton models of these two UAVs have been graphically shown in Fig. 1. In these graphs, the arrows starting from one state and ending to another state represent the transitions, labeled by the events. The entering arrows stand for the initial states. Marked states are shown by double circles.

It is clear that the discussed model of UAVs requires a tight cooperation to coordinate the UAVs to handle the collision alarms CA_1 and CA_2 in order to avoid collision. Through this paper, we will develop a decentralized collision avoidance mechanism to safely drive the UAVs to their desired position.

COORDINATION OF UAVS

Having the discrete model of the UAVs, it is possible to design the supervisor to achieve a desired order of events to drive the UAVs to their desired position. For this purpose, the supervisor, S, observes the executed strings of the plant A and disables the undesirable controllable events. Here, we assume that all of the events are observable, i.e., all are equipped with sensors. The generated language and marked language of the closed-loop system, $L(S/A)$ and $L_m(S/A)$, can be constructed as follows:
(1) $\varepsilon \in L(S/A)$
(2) $[(s \in L(S/A))\ and\ (s\sigma \in L(A))\ and\ (\sigma \in L(S))] \Leftrightarrow (s\sigma \in L(S/A))$
(3) $L_m(S/A) = L(S/A) \bigcap L_m(A)$

where s is the string that has been generated so far by the plant A, and σ is an event, which the supervisor S should decide whether keep it active or not in the supervised system S/A.

Within this framework one can use parallel composition to facilitate the control synthesis. Parallel composition is a binary operation between two automata which can be defined as follows:

Definition 2 *(Parallel Composition[12]) Let $A_i = \left(Q_i, q_i^0, E_i, \delta_i, Q_{mi}\right)$, $i = 1, 2$, be automata. The parallel composition (synchronous composition) of A_1 and A_2 is the automaton $A_1||A_2 = (Q = Q_1 \times Q_2, q_0 = (q_1^0, q_2^0), E = E_1 \cup E_2, \delta, Q_m = Q_{m1} \times Q_{m2})$, with δ defined as $\forall (q_1, q_2) \in Q, e \in E : \delta((q_1, q_2), e) =$*

$$\begin{cases} (\delta_1(q_1,e),\delta_2(q_2,e))\,, & \textit{if } \delta_1(q_1,e)!,\delta_2(q_2,e)!, \\ & e\in E_1\cap E_2\textit{;} \\ (\delta_1(q_1,e),q_2)\,, & \textit{if } \delta_1(q_1,e)!, e\in E_1\backslash E_2\textit{;} \\ (q_1,\delta_2(q_2,e))\,, & \textit{if } \delta_2(q_2,e)!, e\in E_2\backslash E_1\textit{;} \\ \textit{undefined}, & \textit{otherwise.} \end{cases}$$

Here, the parallel composition is used to combine the plant's discrete model and the supervisor as follows:

Lemma 1 [13] *Let $A = (Q,\, q_0,\, E,\, \alpha,\, Q_m)$, be the plant automaton and $K \subseteq E^*$ be the desired marked language. There exists a nonblocking supervisor S such that $L_m(S/A) = L_m(S\|A) = K$ if $\emptyset \neq K = \bar{K}\bigcap L_m(A)$ and K is controllable. In this case, S could be any automaton with $L(S) = L_m(S) = \bar{K}$.*

Designing the supervisor for collision avoidance

Collision avoidance requires the cooperation of the UAVs. When UAV_2 enters the alarm zone of UAV_1, first, UAV_1 asks UAV_2 to stop, and then, UAV_1 finds a way to safely get away from UAV_2. After avoiding the collision, UAV_1 releases UAV_2 and both UAVs resume their normal operations. Similar strategy is taken when UAV_1 enters the alarm zone of UAV_2.

This specification, K_C, is shown in Fig. 2 whose left side shows that after appearing CA_{12}, UAV_1 releases that UAV_2 has entered its alarm zone. Therefore, by event $Stop_2$, UAV_1 requests UAV_2 to stop for a while to safely manage the situation. After removing collision, RCA_1, UAV_1 releases UAV_2, and their normal operations can be resumed. Similarly, the right side of Fig. 2, shows the case when UAV_1 enters the alarm zone of UAV_2. If neither of collision avoidance alarm events CA_{12} and CA_{21} happen, UAV_1 and UAV_2 can do their normal operations.

Since all events are assumed to be controllable, K_C is controllable with respect to the language $L(A_1||A_2)$. Therefore, based on Lemma 1, there exists a supervisor A_c that can control the plants A_1 and A_2 to achieve this joint specification. The supervisor is the realization of the specification K_C in which all states are marked.

Decomposed local supervisors for the collision avoidance

The collision avoidance supervisor, A_C, is a centralized supervisor which manages both UAV_1 and UAV_2. To make this supervisor decentralized and to achieve local supervisors, we will utilize our proposed decomposition scheme introduced in.[6,14] Here, local supervisors can be achieved by the projection of the global supervisor to each agent's local event set. The projection of the global supervisor A_C to the event set of UAV_i, E_i, is denoted by $P_{E_i}(A_C)$, and can be obtained by replacing the events that belong to $E\backslash E_i$ by ε-moves, and then, merging the ε-related states. Once the local supervisor automata are derived through the natural projection, the decentralized supervisor is then obtained using the parallel composition of the local supervisor automata. Parallel composition captures the logical behavior of concurrent distributed systems by allowing each subsystem to evolve individually on its private events, while synchronize with its neighbors on shared events for cooperative tasks. This has been formally described in the following theorem:

Theorem 1 *(Decentralized cooperative control using supervisor decomposition) Consider a plant, represented by a parallel distributed system $A_P = A_{P_1} \parallel A_{P_2}$, with local event sets E_i, $i = 1, 2$,*

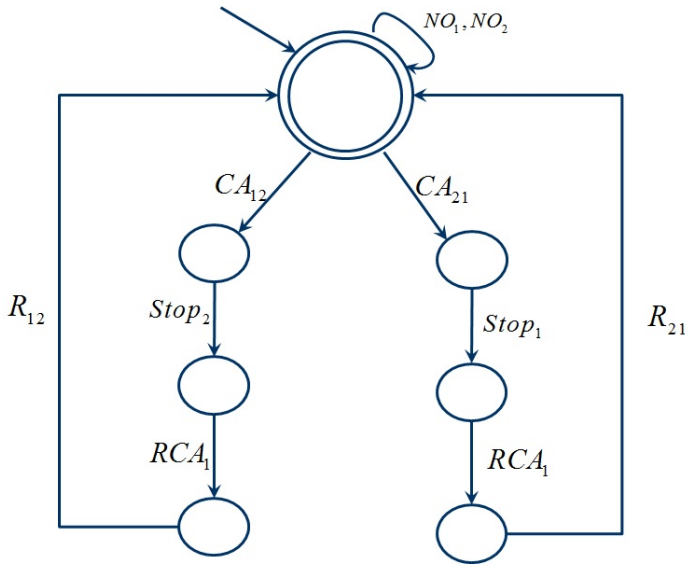


Figure 2. The specification automaton K_c for cooperative collision avoidance.

and let the global specification is given by a task automaton A_S over $E = E_1 \cup E_2$. Furthermore, suppose that there exists a deterministic global controller automaton $A_C \cong P_1(A_C) \parallel P_2(A_C)$, so that $A_C \parallel A_P \cong A_S$. Then, the entire closed loop system satisfies the global specification, in the sense of bisimilarity, i.e., $\parallel_{i=1}^{2} (A_{P_i} \parallel P_i(A_C)) \cong A_S$, provided the decomposability conditions in[15] *for A_C.*

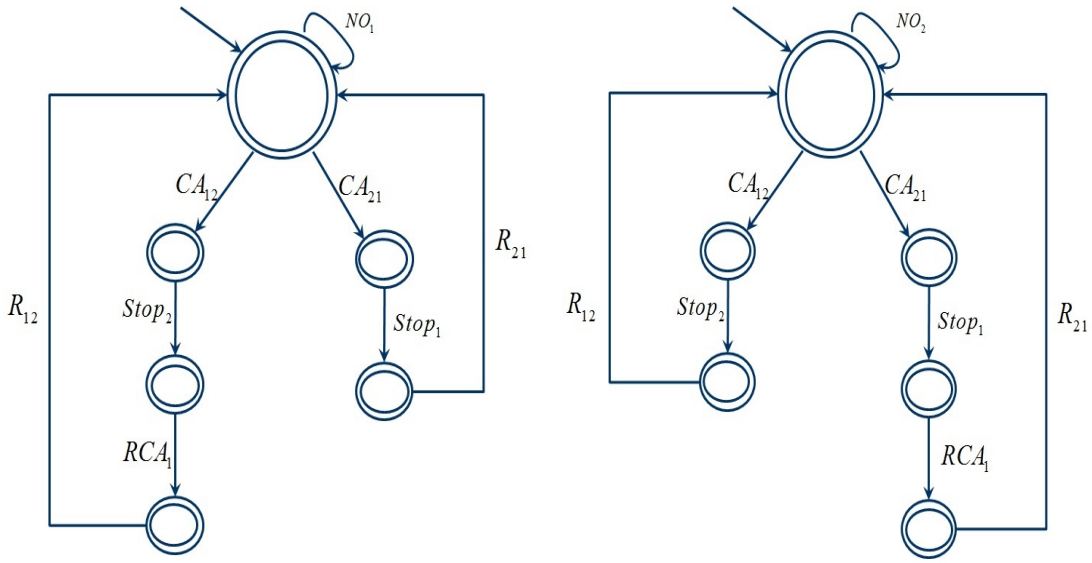
The significance of this result is the decentralized implementation of the global supervisor, A_C, given in Fig. 2, by decomposing A_C, into local supervisors. As shown in Fig. 3, the supervisor automaton A_C is decomposable into $A_{C1} = P_1(A_C)$ and $A_{C2} = P_2(A_C)$, so that $A_{C1} \parallel A_{C2} \cong A_C$.

IMPLEMENTATION AND INTEGRATION

We have integrated the proposed collision avoidance mechanism with the formation control of the UAVs. Consider two follower UAVs, UAV_k, $k = 1, 2$, following a leader as follows:

$$V_{follower_k} = V_{leader} + V_{relk}, \tag{1}$$

where the k'th follower should reach the desired position with respect to the leader and maintain it by controlling the relative velocity, V_{relk}. Alternatively, one can consider a relatively fixed frame for each follower UAV, in which the follower moves with the relative velocity, V_{relk}. Then, the formation problem is to find V_{relk} to drive the follower UAVs toward their desired position to form the chosen formation. Sometimes the formation needs to be changed. In this case, the formation reconfiguration problem is to drive the follower UAVs toward their new positions. This process, reaching the formation and formation reconfiguration, may cause collision, and hence, a collision avoidance mechanism is an essential part of the procedure. For this purpose, when one UAV enters the alarm



(a) (b)

Figure 3. (a) The local supervisor for collision avoidance for UAV_1. (b) The local supervisor for collision avoidance for UAV_2.

zone of the other one, it will be asked to stop moving in the relative frame, so that the other UAV can manage the situation. Once the collision alarm removed, then the UAV can resume its normal operation. This algorithm has been implemented through the proposed decentralized supervisory control approach and has been verified through a hardware-in-the-loop simulation platform.[16] In this platform, the nonlinear dynamics of the UAVs have been replaced with their nonlinear models, and all software and hardware components that are involved in a real flight test remain active during the simulation so that the simulation results achieved from this simulator are very close to the actual flight tests.

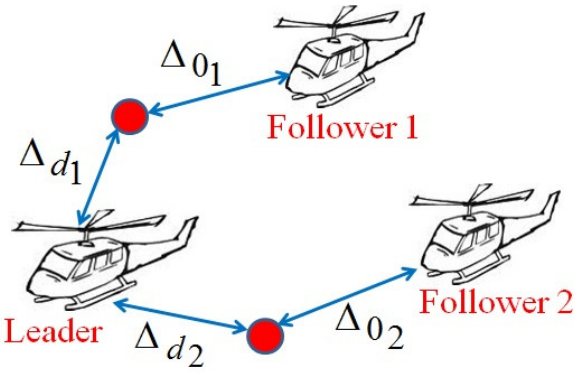


Figure 4. The schematic of a formation scenario with two followers and one leader

Consider two followers that should track a leader UAV with a desired distance, as shown in Fig. 4. The distance between the desired position of the $Follower_1$ and $Follower_2$ and the leader UAV

are $\triangle_{d1} = (12, 10)$ and $\triangle_{d2} = (-12, -10)$, respectively. The follower UAVs initially are not at the desired position. It takes 34.8 Sec for $Follower_1$ and 14.3 Sec for $Follower_2$ to form the desired formation. After 50 Sec, the formation switches. For the new formation, the desired distances of the followers from the leader are $\triangle_{d1} = (-30, -10)$ and $\triangle_{d2} = (0, 10)$. When the followers are trying to reach the desired formation, at $t = 55.8\ Sec$, $Follower_2$ enters the alarm zone of $Follower_1$. To avoid collision, $Follower_1$ asks $Follower_2$ to stop in the relative frame, and then it turns to handle the situation. After removing the collision alarm, both followers have resumed their normal operations to reach and keep the formation. The position of the UAVs in x-y plane is shown in Fig. 5. Further details of the implementation of this algorithm can be found in.[14,17]

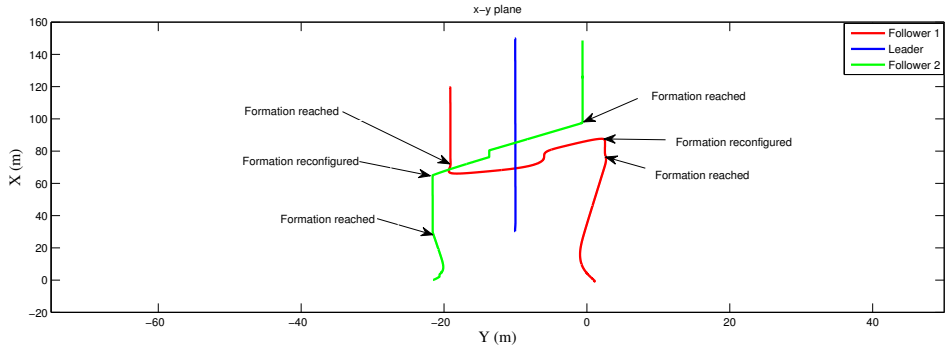


Figure 5. The position of the UAVs in the x-y plane.

CONCLUSION

In this paper, a collision avoidance mechanism was proposed. The algorithm was developed for the decision making unit of the UAVs where a discrete model of the UAVs are available. The proposed algorithm employed the supervisory control of discrete event systems and provided decentralized local supervisors for the UAVs, so that local (decomposed) supervisors can treat the distributed agents (UAVs) to achieve a globally safe and collision-free environment. The collision avoidance algorithm was integrated with a formation mission involving two follower UAVs, the efficiency of the algorithm was verified through hardware-in-the-loop simulations.

ACKNOWLEDGMENT

The authors would like to thank the NUS UAV team particularly Professor Ben. M. Chen, Dr. Xiangxu Dong, Dr. Guowei Cai, and Dr. Feng Lin for their technical support during the flight simulations. The first and last authors would like to acknowledge the support from Air Force Research Laboratory and OSD for sponsoring this research under agreement number FA8750-15-2-0116, and the third author appreciates the financial support from NSF-CNS-1239222, NSF-EECS-1253488 and NSF-CNS-1446288. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Research Laboratory, OSD, NSF or the U.S. Government.

REFERENCES

[1] N. Metni and T. Hamel, "A UAV for bridge inspection: Visual servoing control law with orientation limits," *Automation in Construction*, Vol. 17, November 2007, pp. 3–10.

[2] S. V. and S. T., "UAV navigation by an expert system for contaminant mapping with a genetic algorithm," *Guidance, Control, and Dynamics*, Vol. 31, No. 5, 2008, pp. 1360–1371.

[3] M. E. Campbell and M. Wheeler, "Vision-Based Geolocation Tracking System for Uninhabited Aerial Vehicles," *Guidance, Control, and Dynamics*, Vol. 33, No. 2, 2010, pp. 521– 532.

[4] R. Fierro, P. Song, A. Das, and V. Kumar, "Cooperative Control of Robot Formations," *Cooperative Control and Optimization* (R. Murphey, P. M. Pardalos, P. M. Pardalos, and D. W. Hearn, eds.), Vol. 66, pp. 73–93, Springer US, 2002.

[5] M. Karimadini and H. Lin, "Fault-tolerant cooperative tasking for multi-agent systems," *International Journal of Control*, Vol. 84, No. 12, 2011, pp. 2092–2107.

[6] M. Karimadini and H. Lin, "Guaranteed global performance through local coordinations," *Automatica*, Vol. 47, No. 5, 2011, pp. 890–898.

[7] J. W. Park, H. D. Oh, and M. J. Tahk, "UAV collision avoidance based on geometric approach," *SICE Annual Conference*, 2008, pp. 2122 –2126.

[8] E. Boivin, A. Desbiens, and E. Gagnon, "UAV collision avoidance using cooperative predictive control," *Control and Automation, 2008 16th Mediterranean Conference on*, 2008, pp. 682–688.

[9] K. Y. Kim, J. W. Park, and M. J. Tahk, "UAV collision avoidance using probabilistic method in 3-D," *Control, Automation and Systems, 2007. ICCAS '07. International Conference on*, 2007, pp. 826 –829, 10.1109/ICCAS.2007.4407015.

[10] F. Borrelli, T. Keviczky, and G. Balas, "Collision-free UAV formation flight using decentralized optimization and invariant sets," *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, Vol. 1, 2004, pp. 1099–1104.

[11] P. Ramadge and W. Wonham, "The control of discrete event systems," *Proceedings of the IEEE*, Vol. 77, Jan. 1989, pp. 8–98.

[12] C. G. Cassandras and S. Lafortune, *Introduction to discrete event systems*. USA: Springer, 2008.

[13] R. Kumar and 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.

[14] A. Karimoddini, M. Karimadini, and H. Lin, "Decentralized hybrid formation control of Unmanned Aerial Vehicles," *American Control Conference (ACC), 2014*, June 2014, pp. 3887–3892, 10.1109/ACC.2014.6858770.

[15] M. Karimadini and H. Lin, "Guaranteed global performance through local coordinations," *Automatica*, Vol. 47, No. 5, 2011, pp. 890 – 898, DOI: 10.1016/j.automatica.2011.01.078.

[16] G. Cai, B. M. Chen, T. H. Lee, and M. Dong, "Design and implementation of a hardware-in-the-loop simulation system for small-scale UAV helicopters," *Mechatronics*, Vol. 19, No. 7, 2009, pp. 1057 – 1066.

[17] A. Karimoddini, M. Karimadini, and H. Lin, *Decentralized Hybrid Formation Control of Unmanned Aerial Vehicles*. Technical Report: NCAT-ACCESS-14-001, North Carolina Agricultural and Technical State University, Autonomous Cooperative Control of Emergent Systems of Systems (ACCESS) Lab, [Online]. Available at: http://arxiv-web3.library.cornell.edu/abs/1403.0258.