# Multi-agent Cooperative Decision Making using Genetic Cascading Fuzzy Systems

**Conference Paper** · January 2015

DOI: 10.2514/6.2015-0888

Nicholas Ernest
Psibernetix Inc.
**14** PUBLICATIONS **45** CITATIONS

Kelly Cohen
University of Cincinnati
**207** PUBLICATIONS **994** CITATIONS

AIAA SciTech
5-9 January 2015, Kissimmee, Florida
AIAA Infotech @ Aerospace

AIAA 2015-0888

# Multi-agent Cooperative Decision Making using Genetic Cascading Fuzzy Systems

N. Ernest*, E. Garcia†, D. Casbeer‡, K. Cohen*, and C. Schumacher§

## I. Introduction

Missions consisting of groups of unmanned aerial vehicles (UAVs) require a high degree of coordination for successfully achieving desired goals. From the many types of applications where a group of coordinated agents is potentially able to outperform a single or a number of systems operating independently, the assignment of tasks is an important one.

Different authors have addressed multi-agent task assignment problems for UAVs. For instance, [4] presented a robust task assignment algorithm for uncertain environments. The authors of [1] provided a decentralized task consensus algorithm with asynchronous communication. Decisions to communicate are taken by each agent independently based on the outcomes of assignments using different sets of information.

In the present paper we envision a cooperative assignment of tasks in which vehicles decide which threats to attack according to collective preferences, in contrast to individual preferences which has been a common approach in the literature [3, 7–10]. The main objective behind this approach is to encourage agents to make decisions that bring greater benefit to the group. Although this objective may not necessarily suit all assignment problems, in some scenarios like the one presented in this paper the collective approach can potentially lead to an improved cooperation among agents. Additionally, the assignments need to be free of conflicts in order to use resources wisely. In the present work, agents are equipped with different types of weapons which need to be used in optimal manner and according to the specific type of threats. All these different factors need to be considered in order to make good and fast decisions. The main tool used in this paper for each local agent to generate these decisions is based on a novel method of Genetic Cascading Fuzzy Systems [6].

## II. Problem Statement

Consider a group of Unmanned Aerial Vehicles (UAVs) also called agents

$$U = \{1, 2, ..., N_u\} \tag{1}$$

and a set of targets:

$$T = \{1, 2, ..., N_t\}\,. \tag{2}$$

Let $c_{ij} \geq 0$ represent a score associated to target $j \in T$. Define $x_{ij}$ as a decision variable that is equal to 1 if UAV $i \in U$ performs a task on target $j \in T$ and it is 0 otherwise.

The mathematical formulation of the cooperative Task Assignment Problem (TAP) can be expressed as follows:

$$\max \left( J = \sum_{i=1}^{N_u} \sum_{j=1}^{N_t} c_{ij} x_{ij} \right) \tag{3}$$

*School of Aerospace Systems, University of Cincinnati, Cincinnati, OH 45221
†Infoscitex Corp. Dayton, OH 45431
‡Control Science Center of Excellence, Air Force Research Laboratory, WPAFB, OH 45433
§Air Force Research Laboratory, Wright-Patterson AFB, OH 45433

subject to

$$\sum_{j=1}^{N_t} x_{ij} \leq L_t,\ i \in U \tag{4}$$

$$\sum_{i=1}^{N_u} x_{ij} \leq 1,\ j \in T \tag{5}$$

The constraints guarantee that any given agent is awarded no more than $L_t$ tasks (4) and that a task can only be satisfied by one agent (5). Note that by satisfying these constraints we have that a total of $N_j$ tasks are assigned where $N_j \leq N_{\min} = \min\{N_t, N_u L_t\}$. A schedule of assignments is said to be free of conflicts if each task is assigned to no more than one agent.

In this work we consider a conflict-free assignment of tasks that determine decisions concerning the threats that will be attacked, by which vehicle, and the weapon to be used against the selected threat. In order to implement algorithms for assigning vehicles to destroy threats (and to select the appropriate weapon as well), it is necessary to come up with a way to determine risk scores for each threat that can capture the current environment and the current number of resources available to each vehicle. This is a challenging problem mainly because of the large space of discrete and continuous variables characterizing the number and response of threats and the current position and resources of each vehicle. A solution based on Genetic Cascaded Fuzzy Systems is proposed in the following section.

## III. Proposed Solution

### A. Genetic Cascading Fuzzy System

In this section we provide a brief description of the Learning Enhanced Tactical Handling Algorithm (LETHA). For a more detailed explanation of LETHA characteristics and functioning refer to [6]. LETHA is a novel decision making algorithm and it is designed as an intelligent high-level controller of Unmanned Combat Aerial Vehicles (UCAV). In its entirety, LETHA is the first implementation of the novel Genetic Fuzzy Tree [5], a type of Genetic Fuzzy System (GFS) with many different Fuzzy Inference Systems (FISs) with varying degrees of connectivity between them. Within the scope of this study, we examine a portion of LETHA which is classified as a Genetic Cascading Fuzzy System (GCFS) [6]. Cascading Fuzzy Systems are a recent development [2] in which a series of FISs are utilized, rather than one larger and more complex system that has many inputs and outputs and a large rule base. In this system, the outputs of the first FIS feed into the next, until the final FIS produces a crisp output. More properly emulating human decision making, each fuzzy logic system can analogously represent different sections of the brain which are fired off due to some sensory input.

LETHA is adaptable to many different scenarios with any finite number of squadrons each consisting of some finite number of aircraft. Fig. 1 displays a route taken by a single squadron in a mission containing enemy electronic warfare stations which disrupt communication and air interceptors that patrol within a defined boundary. However this paper restricts itself to a generic strategic bombing mission to properly evaluate the effectiveness of the Cooperative Task Assignment Algorithm (CTAA).

The problem approached by this portion of LETHA consists of a squad of UCAVs that are given a start and exit point from a combat area. The squadron flies in a finger four formation along a route LETHA calculates that maximizes survivability while staying within mission time constraints. Enemy critical targets in the form of unarmed structures are located within this battle space along with enemy Surface-to-Air Missile (SAM) sites. These SAM sites shoot groups of missiles at the friendly squadron which must be countered in order to complete the mission.

Each friendly UCAV is equipped with a variety of ordinances. A limited supply of Self-Defense Missiles (SDMs) is given which can both destroy enemy aircraft, and enemy air-to-air and ground-to-air missiles. A Laser Weapon System (LWS) is onboard each aircraft as well and can also destroy enemy ordinances, though not air interceptors. This LWS has a set time that it can fire before needing recharged as well as a set recharge rate. The profile, or relative azimuth angle, of the missile targeted by the laser, and the distance from the laser to the missile determines for how long a laser must fire on an ordinance before it is destroyed. In order to destroy ground targets, the UCAVs are also given a supply of air-to-ground missiles.

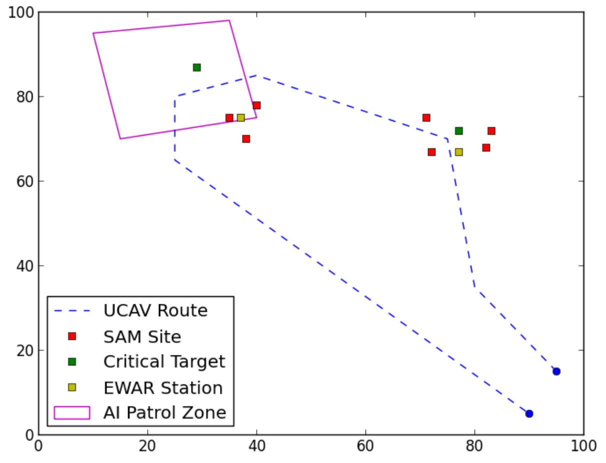

Figure 1: LETHA mission layout for one squadron

A probability of kill less than 90% is assigned to these friendly ordinances, though the enemy forces weapons are considered perfect and always result in a loss of a friendly aircraft if not countered.

The defensive weapon systems portion of LETHA, which is the GCFS examined in this work, utilizes three different FISs as its decision-making engine. The first analyzes the remaining known number of threats and remaining mission time to determine how confident LETHA should be with its ordinances. Due to the suboptimal probability of kill, if resources allow, sending more than one counter to an incoming threat pre-emptively increases survival odds compared to waiting to send another after the first fails which may be too late. The next FIS then determines, for each enemy missile in the air, whether each UCAV would utilize a SDM or the LWS to counter the threat.

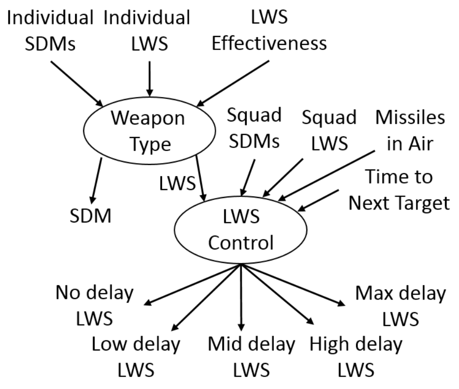

Figure 2: Weapon Selection FISs [5]

The SDMs are considered fire-and-forget, but there is an infinite number of methods to firing the LWS based on the delay between detection and firing. Again, the closer the target is to the UCAVs the more effective the laser is. The cost to waiting is that if the lase fails, there is less time to react and potentially salvage the situation. The third and final FIS determines what delay will be implemented if the LWS is selected. These controllers are run on each individual UCAV iteratively until all the desired actions are determined for an encounter. This process will be referred to as the Iterative method. Fig. 2 depicts this process and all the states LETHA analyzes at these steps.

Originally, any UCAV loss was considered a failure for the mission [6]. This meant that the risks each UCAV would perceive would be identical, regardless of a missile's target within the squad. This constraint has been alleviated and LETHA can now continue a mission until all friendly UCAVs have been destroyed. However, even with knowing the targets of each enemy missile, the iterative process used prior [6] does not optimally protect UCAVs with many resources remaining or intelligently sacrifice doomed aircraft without wasting resources in an attempt to save it.

In missions where LETHA is given enough SDMs to survive with optimal usage of the LWS, near 100% mission completion rates are obtained. The missions investigated in this study represent near-worst-case scenarios, where aircraft were sent on a mission under-equipped. That is, given far fewer SDMs than required to complete the mission without any losses. Since the CTAA relies on some amount of communication between the aircraft, the no-communications missions with enemy electronic warfare will not be considered. While losses will be encountered, both the original, henceforth referred to as Iterative, method and the CTAA will be tested in multiple missions with varying levels of under-equipped squadrons. Their performance will be evaluated in terms of average mission score, a user-defined point total corresponding to a sum of all negative and positive actions in a mission, as well as mission completion percentage over 100 runs.

## B. Cooperative Task Assignments

Let $N_t(t)$ represent the number of threats that are detected by the agents at the current time $t \in \mathbb{R}_{\geq 0}$. These threats represent the current targets that need to be accounted for or performed a task on. Each threat represents a different level of risk for each agent depending on relative distance and the characteristics of the threat. Each agent evaluates the risk that each threat represents using the FIS's described in Section A. Individual risk scores are obtained from LETHA, which was described in Section A. Since the given threat can potentially be a risk for more than one vehicle, then if agent $i$ eliminates a threat $j$, the associated individual risk $r_{ij} \geq 0$ that the threat $j$ represents to agent $i$ is eliminated. Additionally, since the threat has been eliminated, it does not represent a risk to any other agent in the group and the collective risk (reward) defined by

$$s_j = \sum_{i=1}^{N_u} r_{ij} \tag{6}$$

is gained by the group.

The cooperative decision making problem is to design a decentralized protocol to obtain conflict-free assignment of tasks that maximize the eliminated collective risk, that is, maximize the following collective reward metric

$$J = \sum_{i=1}^{N_u} \sum_{j=1}^{N_t} s_j x_{ij}. \tag{7}$$

We consider a receding horizon type approach where, at every iteration, the agents are assigned and execute only one task each of them. This means that $L_t = 1$ in (4). After execution of the current assignment, if there are remaining threats, the agents start the process again by evaluating new individual risks and making new assignments based on collective rewards. This approach represents a simple and practical way to deal with pop-up threats such as new enemy missiles being fired and also to deal with unsuccessful attacks that need a second counter. Note that the number of threats could be smaller than the number of agents. In this case some agents will make a no-attack/conserve-resources decision which is, of course, acceptable and in accordance to the problem (3)-(5).

In the case that all agents are able to communicate with each other (complete communication graph), a simple algorithm can be implemented by all agents in order to arrive to the same assignment, the one that maximizes the collective reward.

Each agent transmits the vector $b_i \in \mathbb{R}_{\geq 0}^{N_t(t)}$ that contains the individual scores $r_{ij}$ for $j = 1, ..., N_t(t)$ and receives the vectors $b_{i'}$ from its teammates $i' = 1, ..., N_u$, $i' \neq i$. The individual scores $r_{ij}$ are provided by LETHA$_i$, that is, the local algorithm embedded within agent $i$ that evaluates the risk corresponding to every detected opponent. After transmitting and receiving these vectors the agents assign the tasks. The procedure to obtain receding-horizon optimal conflict-free assignments is given in Algorithm 1. Each agent computes the collective rewards $s_j$, $j = 1, ..., N_t(t)$. The $N_{\min}$ threats with highest collective rewards are selected and assigned each to the best available agent based on the scores $r_{ij}$ (lines 5-6). Conflicts are automatically avoided by assigning the threats with highest $s_j$ in simple iterations (lines 4-10) and by resetting the collective reward for the current selected threat (line 8) and also resetting the individual scores corresponding to the current winning agent (line 9) after every iteration. The assignments are stored in the

**Algorithm 1** *assign_tasks*$(b_i, b_{i'})$

```
1: for $j \in T(t)$ do
2:     $s_j = \sum_{i \in U} r_{ij}$
3: end for
4: for $k = 1$ to $N_{\min}$ do
5:     $j^* = \text{argmax}_j(s_j)$
6:     $\omega_i = \text{argmax}_i(r_{i,j^*})$
7:     $\phi_{i,\omega_i} = j^*$
8:     $s_{j^*} = 0$
9:     $r_{\omega_i,j} = 0 \; \forall \; j \in N_t(t)$
10: end for
```

winning list $\phi_i$. Note that $\phi_i = \phi_{i'}$ for $i, i' \in U$ since all agents use the same procedure on the same set of information (individual scores); then they obtain the same assignments.

## IV. Implementation

This section will cover specifically what LETHA outputs for every event, and how this is fed to the cooperative task assignment algorithm. Each mission analyzed in this study utilizes the default values of four UCAVs in a squadron, and six enemy missiles, each with a slight delay, from a single SAM site firing.

LETHA seeks to optimize its score throughout the mission, where every action has an arbitrarily user-defined score (or penalty) assigned to it. In the Iterative method, simple post-processing takes the results from the GCFS and determines which method each UCAV should utilize on each incoming missile. While LETHA is assumed to be aware of which UCAV every enemy missile is heading, this information is not utilized to its fullest in this method. The combined conservation of resources is then the main deciding factor in the squadron's actions.

The CTAA has been implemented as an intermediate step between the output of the GCFS and its post-processing. Just as in the Iterative method, LETHA determines what type of action to carry out on every incoming missile first. The possible outputs at this stage are:

- 0 : Unable to take any action
- 1 : Fire SDM at red missile
- 2 : Utilize LWS (delay calculated later) to lase red missile

Label this value as $\alpha_i$ if $> 0$, where $I \leq N_u$ is the total number of UCAVs which are able to take an action on any of the incoming $J$ enemy missiles. Note that the confidence output from LETHA artificially inflates $J$ by some factor, $C$, with $C \geq 1$. However, LETHA ensures that every incoming missile has at least one counter assigned to it before doubling up on any particular missile.

Rather than proceeding with the final LWS FIS as in the iterative method, the CTAA intervenes here. The risk scores $R$ utilized by the CTAA are calculated for combination of active UCAV, $I$, and desired number of counters, $J \geq N_t$.

$$r_{i,j} = (\alpha_i) * (1 + w_1 * f_{LWS_{t_j}}) * (1 + w_2 * f_{SDM_{t_j}}) * (\beta_j) + (w_3 * b_{t_j}) - (w_4 * m_{t_j})$$

Where:

- $W$ : Array of positive weights $w_1, w_2, w_3, w_4$ which correspond to user-defined scores
- $f_{LWS_{t_j}}$ : Fuzzified capabilities of the LWS of the UCAV targeted by missile $j$, (0:1)
- $f_{SDM_{t_j}}$ : Fuzzified capabilities of the remaining SDMs of the UCAV targeted by missile $j$, (0:1)
- $\beta_j$ : 1 if missile $j$ has not been countered by any UCAV yet, 0.2 otherwise
- $b_{t_j}$ : Air to ground weapons remaining onboard UCAV targeted by missile $j$

- $m_{t_j}$ : Number of enemy missiles sharing the same target as missile $j$

The goal of this function is to assign greater mission failure risk to the loss of UCAVs with higher quantities of resources remaining. This evaluation of the missile's target is present in the first two terms in the equation above. The first term is a positive factor increasing with the number of defensive resources remaining on the missile's target. The second term, $(w_3 * b_{t_j})$ is a non-negative value corresponding to the number of offensive weapons remaining on the same UCAV.

At the same time, this function ensures that any UCAV doomed to death in the event that many missiles are targetting it and the squadron as a whole can only counter a few missiles currently, will not cause the squadron to squander resources in a futile attempt to save it. On the contrary, the doomed aircraft will instead sacrifice itself and help defend any of its squad mates who are in danger, even if those missiles have already received one counter from another UCAV. This is caused by the third and final term, $(w_4 * m_{t_j})$.

The output of the CTAA with this risk score function intelligently targets each friendly counter sent by a UCAV and delays actions by UCAVs that should wait for squadmates. The LWS result then follows the normal post-processing to calculate the exact amount of delay before the LWS fires, if any. This method is still somewhat iterative, as the maximum number of actions taken by one calculation is $I$. However, the output control is significantly different. This is especially true in cases where the states of each UCAV are different from each other, which typically occurs immediately after the first encounter of an enemy as the UCAVs all start off with maximum LWS charge and the same number of SDMs.

## V. Results

Four missions were utilized that LETHA had previously been tested on and achieved near-perfect performance with an appropriate amount of SDMs supplied, depicted below in Fig. 3.

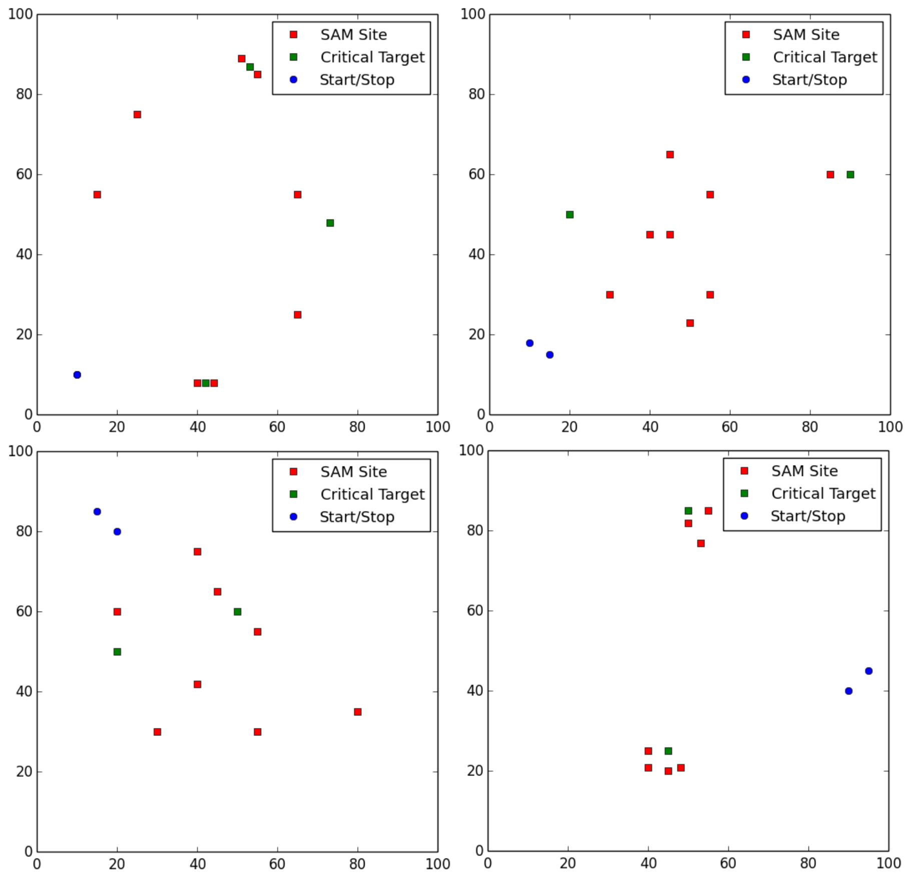

Figure 3: Top Left: Mission #1, Top Right: Mission #2, Lower Left: Mission #3, Lower Right: Mission #4

Again, due to the randomness and uncertainties in the problem, no controller can boast a 100% success rate in any mission, with success being classified as all targets and SAM sites destroyed. However, in this study these four missions were run with extremely low SDMs supplied at first, and then SDMs were increased to measure both average score and completion percentage over 100 runs at each SDM point.

Fig. 4 and 5 below show LETHA running through Mission #1 with two SDMs supplied to each UCAV using the Iterative and CTAA method respectively. In these figures, the blue stars in formation represent the UCAVs, black dashes are SDMs fired by a UCAV, green lines are LWS firings by a UCAV, teal lines are air-to-ground weapons launched by a UCAV, and red lines are SAMs fired by SAM sites.

As can be seen, the Iterative method is able to destroy two of the three critical targets before the squadron is entirely destroyed in this attempt. While the CTAA method does not complete the mission in this attempt, it does allow LETHA to destroy all three critical targets and all but one SAM site before the squadron is eliminated. Towards the end of this attempt, UCAV #'s 1 and 4 were destroyed, but #'s 2 and 3 were still remaining. In the final encounter, UCAV # 3 was out of air-to-ground munitions and thus, despite having more defensive capabilities remaining to survive, sacrificed its remaining resources to defend UCAV # 2 such that the final critical target and one last SAM site could be destroyed.

Both methods had their first death occur from the same SAM site in the third encounter. This reinforces the notion that during nominal conditions, when resources are sufficient for mission completion, the CTAA adds little value to the overall system. However, the benefits during near-worst-case scenarios is visibly noticable. Additional, on average the CTAA adds 0.0062 seconds of computation time to the running of the GCFS. This time is when implemented in Cython, running on a desktop with ample RAM and a 3.6 GHz processor.

Graphics for the remainder of the runs will not be presented for the sake of brevity, but Table 1 below shows the average score and mission completion percentage over 100 runs of each mission at each quantity of SDMs provided per UCAV. The mission score again is a unitless summation of user-defined points assigned to each positive and negative action inside the simulation. Thus the exact numbers have little true value, but the trends and differences between scenarios shows more detail as to how well LETHA was able to perform in each case.

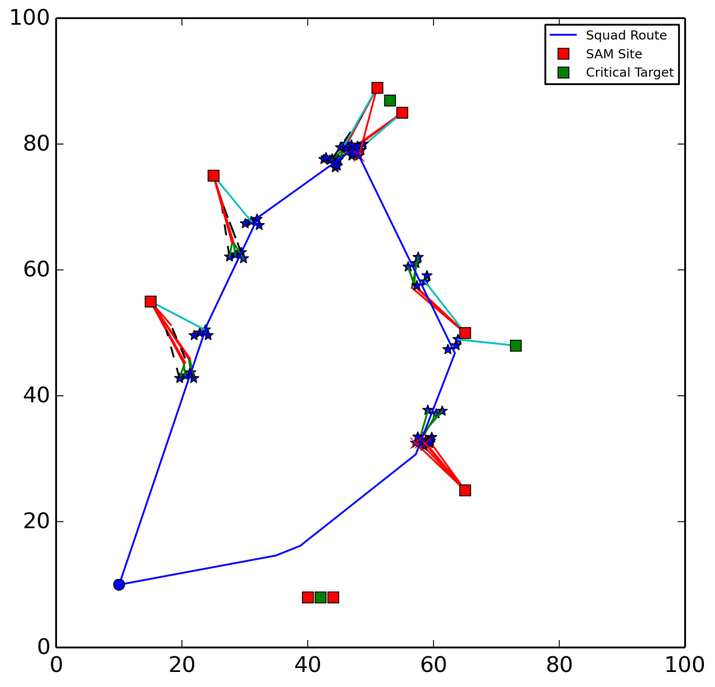

Figure 4: Iterative Method Mission #1 with 2 SDMs per UCAV

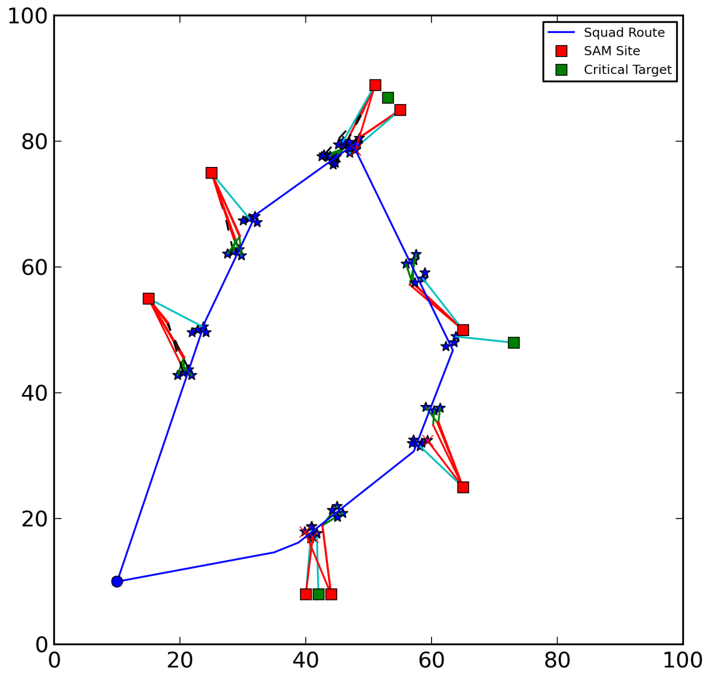

Figure 5: CTAA Method Mission #1 with 2 SDMs per UCAV

Table 1: Results Over 100 Runs

| Mission | SDMs | Iter. Avg. Score | Iter. Completion % | CTAA Avg. Score | CTAA Completion % |
|---|---|---|---|---|---|
| # 1 | 2 | -17.78 | 2% | -6.28 | 8% |
| | 3 | -7.60 | 7% | 54.56 | 25% |
| | 4 | 146.00 | 63% | 169.08 | 71% |
| | 5 | 246.50 | 98% | 247.46 | 97% |
| # 2 | 2 | -32.94 | 0% | -36.60 | 0% |
| | 3 | -9.84 | 1% | -6.84 | 5% |
| | 4 | 58.06 | 23% | 73.34 | 33% |
| | 5 | 252.04 | 84% | 256.00 | 98% |
| # 3 | 3 | -13.58 | 0% | -14.88 | 0% |
| | 4 | -3.60 | 0% | 3.56 | 4% |
| | 5 | 40.4 | 12% | 85.48 | 39% |
| | 6 | 255.44 | 94% | 258.46 | 97% |
| # 4 | 2 | -28.07 | 0% | -19.33 | 5% |
| | 3 | 56.12 | 40% | 64.4 | 46% |
| | 4 | 267.96 | 98% | 274.96 | 100% |
| | 5 | 290.01 | 100% | 290.92 | 100% |

Table 1 further reinforces the usefulness of applying the CTAA inside LETHA's GCFS. Both completion percentage and average score were all significantly higher for the CTAA method when near-worst-case scenarios and losses were present. Any differences between the two methods at nominal conditions was negligible, which combined with the low computation cost, presents no strong argument as to a weakness inherited by including the CTAA.

## VI. Conclusion

The genetic cascading fuzzy system presented here can utilize the described cooperative task assignment algorithm to great effect. While involved in a near-worst-case scenario, the inclusion of this algorithm provides a great deal of robustness and gives LETHA the ability to push ever so farther into an inhospitable environment. This inclusion would prove vital in application as a system to increase safety margin, or even as a planner to missions where maximum damage to the enemy is desired even if destruction of the squadron is all but certain.

Distance to the threat, profile of the threat, and a variety of weapon systems' states are all considered here, with future work potentially bringing many more states that must be analyzed. Once additional communication constraints are implemented other than communications either being on or off, such as latency, packet loss, etc., a further level of realism to this work will be present. This step will provide an even greater importance of the cooperative task assignment algorithm in properly controlling the squadron.

## References

[1]M. Alighanbari and J. P. How. A robust approach to the UAV task assignment problem. *International Journal of Robust and Nonlinear Control*, 18(2):118–134, 2008.

[2]S. Barker, C. Sabo, and K. Cohen. Intelligent algorithms for MAZE exploration and exploitation. In *AIAA InfotechAerospace Conference*, 2011.

[3]H. L. Choi, L. Brunet, and J. P. How. Consensus-based decentralized auctions for robust task allocation. *IEEE Transactions on Robotics*, 25(4):912–926, 2009.

[4]D. Dionne and C. A. Rabbath. Multi-UAV decentralized task allocation with intermittent communications: The DTC algorithm. In *American Control Conference*, pages 5406–5411, 2007.

[5]N. Ernest, K. Cohen, E. Kivelevitch, C. Schumacher, and D. Casbeer. Genetic fuzzy trees and their application towards autonomous training and control of a squadron of unmanned combat aerial vehicles. *Unmanned Systems*, 2015. Under Review.

[6]N. Ernest, K. Cohen, C. Schumacher, and D. Casbeer. Learning of intelligent controllers for autonomous unmanned combat aerial vehicles by genetic cascading fuzzy methods. In *SAE Aerospace Systems and Technology Conference*, 2014.

[7]S. S. Ponda, J. Redding, H. L. Choi, J. P. How, M. Vavrina, and J. Vian. Decentralized planning for complex missions with dynamic communication constraints. In *American Control Conference*, pages 3998–4003, 2010.

[8]C. Schumacher, P. Chandler, S. J. Rasmussen, and D. Walker. Task allocation for wide area search munitions with variable path length. In *American Control Conference*, pages 3472–3477, 2003.

[9]T. Shima, S. J. Rasmussen, and P. Chandler. UAV team decision and control using efficient collaborative estimation. *ASME Journal of Dynamic Systems, Measurement, and Control*, 129:609–619, 2007.

[10]A. K. Whitten, H. L. Choi, L. B. Johnson, and J. P. How. Decentralized task allocation with coupled constraints in complex missions. In *American Control Conference*, pages 1642–1649, 2011.