AFRL-VA-WP-TP-2006-339

INCREASED UAV TASK ASSIGNMENT PERFORMANCE THROUGH PARALLELIZED GENETIC ALGORITHMS (PREPRINT)

Marjorie A. Darrah, William M. Niland, Brian M. Stolarik, and Lance E. Walp

AUGUST 2006

Approved for public release; distribution is unlimited.

STINFO COPY

AIR VEHICLES DIRECTORATE
AIR FORCE MATERIEL COMMAND
AIR FORCE RESEARCH LABORATORY
WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7542

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YY)*
August 2006

2. REPORT TYPE
Conference Paper Preprint

3. DATES COVERED *(From - To)*
11/01/2005 – 08/01/2006

4. TITLE AND SUBTITLE
INCREASED UAV TASK ASSIGNMENT PERFORMANCE THROUGH PARALLELIZED GENETIC ALGORITHMS (PREPRINT)

5a. CONTRACT NUMBER
FA8650-06-2-3654

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
0602201

6. AUTHOR(S)
Marjorie A. Darrah, William M. Niland, and Brian M. Stolarik (Institute for Scientific Research, Inc.)
Lance E. Walp (West Virginia University)

5d. PROJECT NUMBER
A052

5e. TASK NUMBER

5f. WORK UNIT NUMBER
0C

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Institute for Scientific Research, Inc.
Fairmont, WV 26555-2720

West Virginia University
Industrial & Management Systems Engineering
Morgantown, WV 26506-6107

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Vehicles Directorate
Air Force Research Laboratory
Air Force Materiel Command
Wright-Patterson Air Force Base, OH 45433-7542

10. SPONSORING/MONITORING AGENCY ACRONYM(S)
AFRL-VA-WP

11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)
AFRL-VA-WP-TP-2006-339

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
This work has been submitted to the Proceedings of the AIAA Infotech at Aerospace 2007 Conference and Exhibit.
If this work is published, AIAA may assert copyright. This work was funded by Department of the Air Force Contract FA8650-06-2-3654. The U.S. Government has for itself and others acting on its behalf an unlimited, paid-up, nonexclusive, irrevocable worldwide license to use, modify, reproduce, release, perform, display, or disclose the work by or on behalf of the U.S. Government.
Report contains color. PAO Case Number: AFRL/WS 06-2008 (cleared August 21, 2006).

14. ABSTRACT
This paper explores the parallelization of a Genetic Algorithm (GA) utilized for task assignment of a team of Unmanned Air Vehicles conducting a Suppression of Enemy Air Defense mission. The GA has been developed and implemented in the MultiUAV simulation environment for testing. The algorithm has been parallelized with each UAV acting as an independent processor. Two different implementations are explored, one where each UAV independently runs a GA, and the best overall solution is selected at the end, and one where the UAVs exchange information several times during the evolution of generations. The results of these implementations are compared to the original, non-parallelized GA performance.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF: a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT:	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON (Monitor)
Unclassified	Unclassified	Unclassified	SAR	14	Corey Schumacher
					19b. TELEPHONE NUMBER *(Include Area Code)* N/A

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39-18

Increasing UAV Task Assignment Performance through Parallelized Genetic Algorithms

Marjorie A. Darrah*, William M. Niland†, and Brian M. Stolarik‡
Institute for Scientific Research, Inc., Fairmont, WV, 26555-2720

and

Lance E. Walp§
West Virginia University, Industrial & Management Systems Engineering, Morgantown, WV 26506-6107

This paper explores the parallelization of a Genetic Algorithm (GA) utilized for task assignment of a team of Unmanned Air Vehicles (UAV) conducting a Suppression of Enemy Air Defense (SEAD) mission. The GA has been developed and implemented in the MultiUAV simulation environment for testing. The original (non-parallel version) of the GA demonstrated improved performance over the Mixed Integer Linear Program (MILP) algorithm. In order to further improve on the GA performance, the algorithm has been parallelized with each UAV acting as an independent processor. Two different implementations are explored. The first employs identical algorithms on each processor seeded with a different random number, requiring an exchange of information at the end. The second utilizes a GA Island Model, necessitating information exchange several times during the evolution of generations. The results of these implementations are compared to the original GA performance.

I. Introduction

FUTURE generations of Unmanned Air Vehicles (UAVs) will be able to autonomously cooperate as a team to accomplish strongly coupled tasks. Investigations into the most efficient teaming arrangements have been performed by formulating various cooperative control algorithms for the candidate mission. Algorithms have been developed for the allocation of tasks to team members with respect to timing constraints, flyable trajectories, and tasking precedence.

Several types of algorithms have been used to address the task assignment problem. Shumacher et. al.[1] used the Mixed Integer Linear Programming (MILP) approach to assign vehicles to stationary ground targets for a Wide Area Search and Destroy (WASD) mission. This MILP work was extended to a Suppression of Enemy Air Defense (SEAD) mission with survivable Unmanned Combat Air Vehicles (UCAVs) prosecuting pop-up threats by Darrah et. al.[2] Shima et. al.[3,4] applied a Genetic Algorithm (GA) to the original UAV task assignment problem and developed an encoding scheme for a feasible solution as a chromosome. Their GA work was extended for the task assignment of UCAVs prosecuting pop-up threats during SEAD missions by Darrah et. al.[5]

The research discussed in this paper applies a parallel version of a GA implementation to increase the performance for the task allocation problem in a SEAD mission. This will include a discussion of the GA approach, formulation, implementation, and integration into a high fidelity simulation. In order to test the GA approach in a rapid prototyping environment, the GA formulation was coded using a MATLAB toolbox developed by the University of Sheffield[6] and customized MATLAB scripts. Integrating the GA toolbox formulation as a new tasking resource for the MultiUAV[7,8] research software allows testing and analysis of UAV teaming with realistic trajectory constraints in a SEAD mission. In order to conduct accurate SEAD missions, an Integrated Air Defense System (IADS) model is provided by connecting MultiUAV to the FLexible Analysis Modeling and Exercise System (FLAMES).

* Principal Scientist, Computer Sciences Branch, mdarrah@isr.us, AIAA Member
† Member Research Staff, Control Systems Branch, wniland@isr.us, AIAA Member
‡ Director, Control Systems Branch, bstolarik@isr.us, AIAA Member
§ Senior, lwalp@mix.wvu.edu, under advisement of Dr. Alan McKendall, alan.mckendall@mail.wvu.edu

The parallel version of the GA is used for task assignment in a SEAD mission scenario where a number of air vehicles are searching an area for unknown IADS threats. Vehicles travel in a predefined serpentine search pattern with sensors that are capable of detecting and identifying potential threats. When a potential threat is discovered it is necessary to classify it as a target or non-target. If the threat is identified as a target, then a vehicle must attack the target. After the attack, a vehicle must verify that the target has been destroyed. There are three tasks to be executed per threat: classify, attack, and verify. The mission described in this paper has five basic assumptions: (1) the UCAVs are survivable and do not perish in an attack task; (2) the threats are not known a priori and are discovered during the mission, the tasking assignments must be completed with the discovery of each new threat; (3) the number of weapons onboard each aerial vehicle is limited; (4) the team of aerial vehicles becomes heterogeneous since weapons are depleted and mission capabilities are not consistent across the team; and (5) a vehicle depletes its weapons during a mission and can no longer be assigned the attack task once its weapons are depleted. After the depletion of all weapons the vehicle is limited to classify or kill verification.

II. Genetic Algorithm Approach

In order to try to find the optimal task assignment of UAVs to targets for the SEAD mission, a GA is employed. The GA is implemented using three different methods. The first or original method employs a GA seeded with the same random number on each UCAV, thus providing redundant centralized optimization and requiring minimum communication since all vehicles arrive at the same conclusion independently. The second, or Parallelized GA, employs the same algorithm on each UCAV, but seeds the GA on each vehicle with a different random number. This allows different sub-populations to be generated and evolve on each UCAV. They share information at the end of GA reproduction and implement the best solution. The third method, or Island Model, is similar to the parallel method but requires more communication during the evolution of generations. A small percentage of the best chromosomes are allowed to migrate during evolution to all vehicle sub-populations, helping to find comparable task allocation plans in a smaller amount of reproductive iterations. A conceptual diagram of these latter two approaches is supplied in Figure 1.

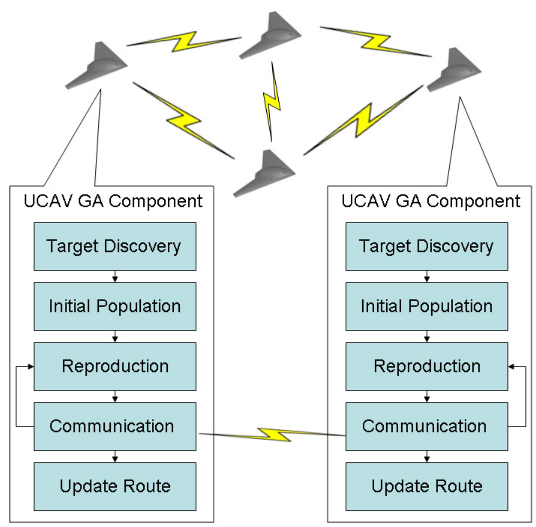

Figure 1. Distributed GA Concept. Each vehicles GA component collaborates with other team members during migration. This information can be passed during chromosome reproduction (Island Model) or at conclusion of reproduction (Parallelized).

A. Original Implementation

A GA is a search algorithm based on the mechanics of natural selection and natural genetics. GAs are in a subclass of evolutionary algorithms that are based on three main principles: reproduction, natural selection, and diversity of species. In this work, a GA is used to try to find the optimal task assignment of UAVs during a SEAD mission. For a more general approach, please refer to *Genetic Algorithms* by David Goldberg[9].

GAs require the parameter set be coded as a finite-length string over a finite alphabet; these strings are referred to as chromosomes. After the initial generation is created, reproduction, elitism, and mutation are applied to produce subsequent generations that, when implemented correctly, have improved performance.

In order to begin the problem, solutions need to be encoded as a chromosome. The following example closely follows the encoding scheme of prior GA work[3,4] related to the task assignment problem. Each solution must represent the assignment of vehicles to targets and thus there are two rows for each chromosome. The top row represents the vehicles and contains numbers from the set $V = \{1, 2, \ldots N_v\}$ and the bottom row represents the targets and tasks on each and contains numbers from the set $T = \{1, 2, \ldots N_t\}$. Table 1 shows an example of a chromosome for the scenario of three targets, each with three tasks being prosecuted by four vehicles.

Table 1. Example Chromosome for SEAD Mission Scenario.

Vehicles	4	3	1	3	2	1	4	2	1
Targets	1	2	1	3	1	2	2	3	3

Since there are three targets, the bottom row contains only the numbers 1, 2, and 3. Each target has exactly three tasks to be performed on it, so three instances of each number are found in the row. The first occurrence of a number indicates the first task on that target; the second indicates the second task; and the third occurrence the third task. The situation represented by the chromosome in Table 1 is as follows:

- Vehicle 4 performs task 1 on target 1, and then task 3 on target 2
- Vehicle 3 performs task 1 on target 2, and then task 1 on target 3
- Vehicle 1 performs task 2 on target 1, and then task 2 on target 2, followed by task 3 on target 3
- Vehicle 2 performs task 3 on target 1and task 2 on target 3

Given a feasible chromosome, the objective function is defined to determine tasking precedence and total time for mission completion using the task assignment defined by the chromosome. Fitness values are linearly assigned between 0 and 2 to the chromosomes of each generation based on a minimization objective, with 2 being the best fitness. The selection process uses the roulette wheel method that probabilistically selects individuals from a given generation for reproduction based on some measure of their performance.

The reproductive process for this problem combines both a crossover operation and an inversion operation. The first reproductive step is to use crossover on the top row (vehicle row). Using crossover on the bottom row (target row) almost always produces an infeasible solution because of the make up of this row. Therefore, the bottom row is modified using inversion. Inversion takes all numbers in the row between two random places and writes them in the reverse order. This leaves the numerical make up of the row the same but switches the ordering of the numbers.

The two other operations applied are elitism and mutation. Elitism retains a portion of the best chromosomes from each generation to go on to the next generation to ensure that new GA generations monotonically decrease. The mutation operation randomly changes one number in the vehicle row and is only applied to a small percentage of elements to insure diversity of the population.

This algorithm was implemented and tested in the MultiUAV/FLAMES simulation environment with each UAV running an identical copy of the GA. Since each vehicles random number generator is seeded identically, each vehicle generates a common initial population and arrives at the same solution as all other team members. The team then executes the task assignment with the best objective value, given the time available to reach the solution. This approach minimizes intra-team communication requirements. The empirical results show that the GA performs better than the MILP solution for SEAD missions involving several vehicles and targets[5].

B. Parallel Version of Genetic Algorithm

In order to further explore the usefulness of the GA and to fully exploit the parallel nature of the algorithm, the GA algorithm was parallelized in order to reach a better solution in less time.

Shonkwiler[10] discusses that parallelizing a GA is simply to run the identical algorithm on each processor independently of one another. Shonkwiler called this GA approach *IIP* parallel, and several authors have shown this technique to be very effective with superlinear parallel speedup[11,12,13]. Empirical results have been supported theoretically[14].

In this research, each processor is on a separate UAV seeded with a different random number. For this first parallel implementation, the vehicles start with a different initial population and let these sub-populations evolve for a specified number of generations. The only required communication between processors (UAVs) is at the end of reproduction, which allows the best results of the vehicles sub-population to be shared among team members. Each processor (UAV) then chooses the best solution from all other team members and implements the chromosome assigned tasks.

This strategy allows for much more of the entire solution space to be explored in the same amount of time. The algorithm performance will depend on the number of members in the team. Results of the simulation of this technique and comparison with the original GA are discussed in Section IV of this paper.

C. Island Model of Genetic Algorithm

The Island Model has been shown to be an efficient way to implement a GA on a parallel machine[15,16,17]. In the Island Model each machine maintains its own sub-population and periodically exchanges a portion of their population in a process called migration.

Island Model GAs have reported to display better performance than serial single population models, both in terms of quality of the solution as well as a reduction in the effort. The improved search quality is contributed to the varied island populations maintaining some degree of independence and thus exploring different regions of the search space while at the same time sharing information via migration.

In our case, each UAV is seeded with a different random number and generates a different initial sub-population, as it does with the parallel version of the algorithm. However, in the Island Model version, each vehicles sub-population evolves for a specified number of generations and then exchanges a small percentage of its best chromosomes with the entire team. At this time, the "best of the best" are included in each sub-population so they can reproduce to yield increased diversity and ultimately better solutions. This is the migration process, which is executed several times during the evolution time interval.

The Island Model is compared in this research against the original GA implementation and the parallel implementation discussed in the last section. The comparison results are presented in Section IV.

III. Implementation

In order to test the distributed GA formulations, enhancements were needed in the MultiUAV research tool. This section will briefly describe MultiUAV and then discuss the added capabilities required for a distributed GA approach.

A. MultiUAV Overview

The U.S. Air Force Research Laboratory (AFRL) developed the MultiUAV[8] research tool to implement and evaluate cooperative control strategies for teams of UAVs. MultiUAV is based in MATLAB/Simulink, easing the effort for researchers to identify and alter simulation components required for the desired cooperative control studies. The public release version of MultiUAV can simulate up to eight vehicles coordinating concurrently. Each MultiUAV vehicle contains a set of managers responsible for duties such as planning routes, gathering sensor information, tracking target states, supervising weapon payload, and allocating tasks during target discovery[7].

All relevant manager information is passed between vehicles through the MultiUAV Virtual Communication Representation (VCR) message passing scheme[18]. The VCR protocol in MultiUAV mandates that each outgoing message is defined with a time-stamp, message layout enumeration, and data fields. This information allows each vehicle to process inbound messages appropriately. This also allows researches to study the effects of cooperative control algorithms during communication latencies, drop-outs, and data corruption.

B. MultiUAV Enhancements for Distributed GA

The GA task allocation resource from previous work did not require the vehicles to exchange information about mission plans. Therefore, it was critical that each vehicle arrive at the same solution. Since the GA formulation is based upon random number generation, there is no guarantee that each vehicle calculates the same conclusion based on common inputs. This issue was eliminated in previous work by seeding the random number generator on each vehicle identically, causing duplication of the initial chromosome generation and reproductive steps by each team member.

Due to the decentralized and parallel nature of the MultiUAV vehicles, a better approach over the original GA formulation allows each vehicle to independently find its optimal GA solution and share results among team members. Instead of each identically seeded vehicle redundantly implementing a common solution, each vehicle solves the GA independently and passes a VCR message to all other team members containing its best chromosome combination. Once a vehicle receives chromosome information from every other vehicle, the chromosome combination with the shortest mission duration is implemented. Since each vehicle generates its own initial population, more solutions can be evaluated in this Parallelized GA approach to find the shortest mission time possible.

The addition of the VCR message for the Parallelized GA can also be used for Island Model GA studies. Instead of each vehicle solving its own GA formulation and then exchanging results, a VCR message containing the best chromosome combinations is passed among team members during reproduction. In other words, once a vehicle performs crossover and inversion on its own set of chromosomes, a small percentage of the best combinations are shared with other team members. For clarity, a timeline in provided in Figure 2.

The Island Model approach allows the differing initial populations on the vehicles to converge to a shorter mission time more quickly than either the original or Parallelized GA approaches. On the other hand, it requires an increased amount of communication and bandwidth since team members are sharing multiple chromosomes at various iteration points during the evolution of the GA solution.

IV. Results

To study the benefits of using a Parallelized or Island Model GA over the original formulation, a simulation scenario is created to compare the three approaches. The scenario involves three vehicles engaging a cluster of five threats. To fairly compare the three GA approaches, each simulation run ensures that the three vehicles are notified of the threats at the same instance in time. Once the vehicles are notified of the threats, the task allocation manager on each vehicle generates an initial chromosome population of 50 and solves the problem according to the chosen GA formulation. Both an attack and verification task must be solved for each threat, thus 10 tasks are solved in this scenario. The objective of each GA formulation is to minimize the total mission duration.

Each GA formulation is simulated 10 times with different random number seeds on the candidate scenario. During simulation the best mission time found in the initial population and each reproductive step were noted and averaged. The average values at initial population (iteration 0) and even iterations 2 through 10 are presented for each GA formulation in Figure 3.

Note, even before the reproductive iterations begin, the original GA has an average mission duration that is greater than either of the distributed approaches. Since the original GA results in an identical set of initial populations on all three vehicles, fewer combinations are explored and the average mission time is higher.

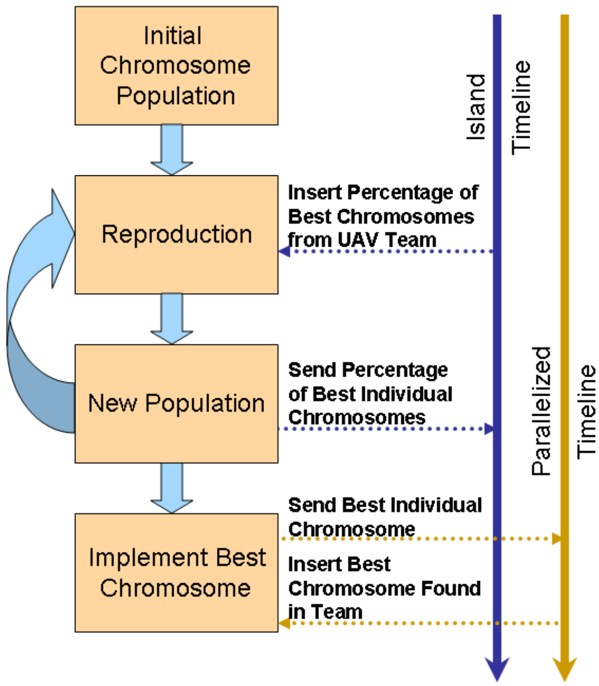

Figure 2. Comparison of GA Timelines. The amount of communication required during Island Model studies is proportional to the desired amount of iterations specified for reproduction. The Parallelized technique only requires communication once at plan implementation.

The Parallelized GA has the shortest average mission duration at the initial population. This can be contributed to the observation that the vehicle fortunate enough to have the best initial population has a head start on its team members and generally will continue to contain the shortest mission time at any iteration. The Island Model eliminates this since chromosomes are exchanged quickly after the initial population. Even though the Island Model averages a higher mission duration than the Parallelized approach in the initial population, the average mission duration between both GA approaches are very close at the second iteration.

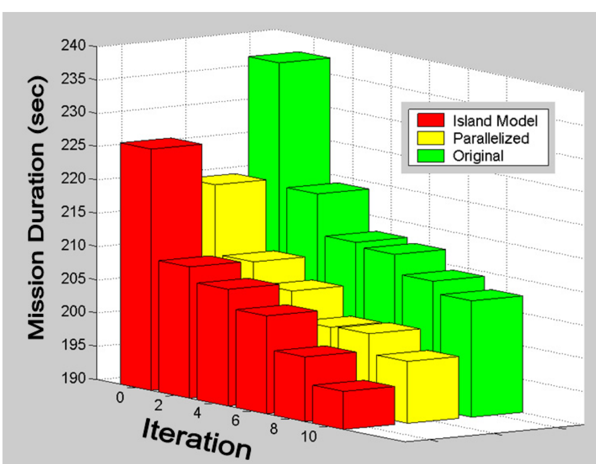

Figure 3. GA Formulation Comparison. The original GA formulation occupies the rightmost series of bars. The Parallelized GA approach is seen in the middle series of bars. The leftmost series of bars represents the Island Model.

Once the reproductive iterations begin, the benefits of using the distributed GA approaches are quickly seen. Both approaches find an average mission duration (209 seconds) in two iterations that required the original GA eight iterations to find. At four iterations the Parallelized GA can find a mission duration (206 seconds) that is less than the best results the original GA can calculate (207 seconds) at 10 iterations. As the Parallelized and Island models are allowed to iterate, they continue to find improvements in mission duration.

Because of the stochastic nature of the GA, there are times that the Parallelized approach performs as well or even better than the Island Model, but on average the Island Model should give the best performance because it diversifies the search more than the other two. Since each GA approach is only tested 10 times, it is believed a more exhaustive Monte Carlo analysis will show the Island Model performing at least as well as the Parallelized approach at each iteration.

The Island Model does begin to outperform the Parallelized GA further into the iteration process. This is due to the fact that the solution becomes harder to improve upon as the number of iterations increase. At this point the benefit of sharing chromosomes during reproduction becomes more evident. Notice each iteration of the Island Model provides a consistent mission duration improvement. At certain points in the original and Parallelized approaches, improvements in mission duration become stagnant between iterations.

V. SEAD Mission Improvement

In the original GA formulation, the task allocation algorithm performance was tested in a dynamic SEAD mission. The UAV team objective is to engage an enemy IADS and prosecute Surface-to-Air (SAM) missile sites as they are discovered. Each UAV is equipped with a Radar Warning Receiver (RWR), giving them the ability to detect SAM sites as they begin tracking vehicle positions. The FLAMES software provides the IADS capabilities. Through a previous effort[19], MultiUAV was made interoperable with FLAMES through a remote client interface and solely responsible for updating the UAV positions, monitoring RWR sensors, and planning vehicle routes according to the cooperative control task allocation algorithms. A sample snapshot of MultiUAV collaborating in the FLAMES IADS model is provided in Figure 4.

Without a priori knowledge of the threat layout, as performed in Section IV, a detailed comparison of algorithm performance is difficult. As the mission unfolds for each GA formulation, threats are discovered at different times due to different trajectory plans implemented on the previous target discovery. This section will only present the results for one simulation run with the intent to show an overall mission duration improvement when using the Parallelized and Island Model approaches. Table 2 provides a comparison of the three GA formulations during dynamic SAM discovery.

It can be seen that a significant improvement over the original GA can be found by using a Parallelized or Island formulation as the amount of SAM sites to prosecute increases. Since the chromosomes begin to increase in size, it becomes beneficial to search through a larger initial population. At the discovery of the third SAM, it is interesting to note that the original GA can find a quicker mission duration than the Parallelized approach. Due to the stochastic behavior of the GA, there are times when the original GA can find a very good solution in a small population. As seen in the previous section, on average this will not hold true.

(a)

(b)

(c)

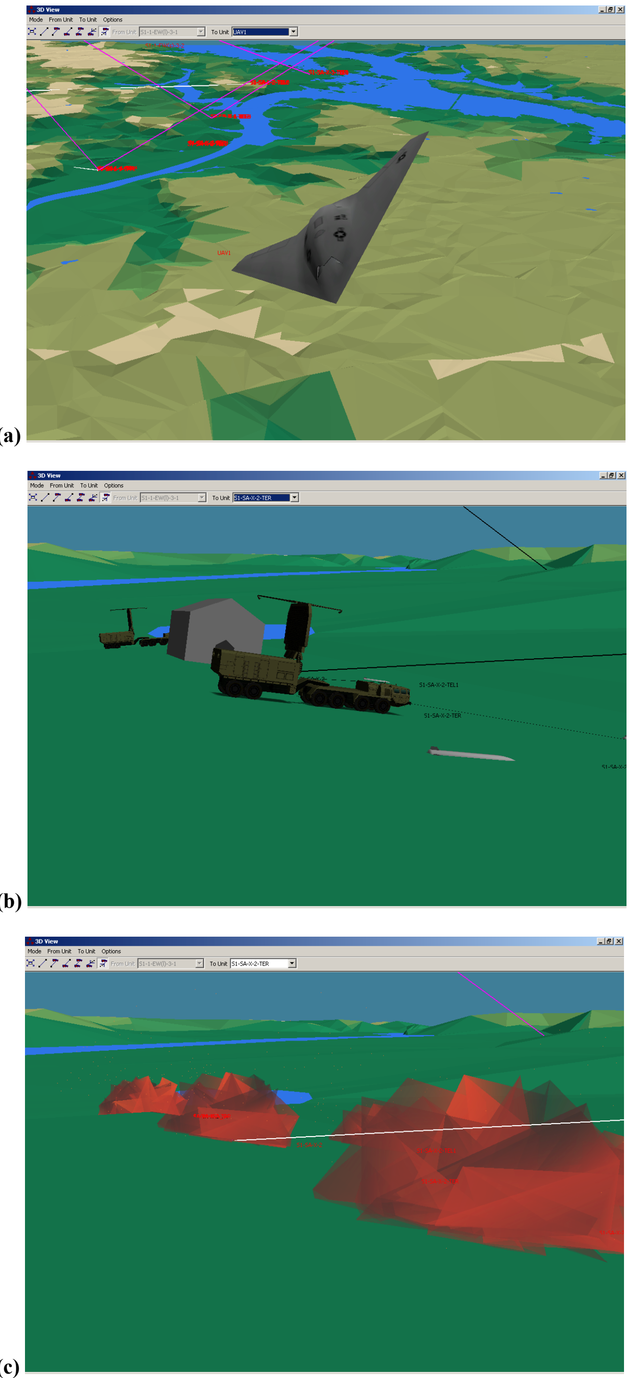

Figure 4. MultiUAV/FLAMES collaborative environment. MultiUAV controls vehicle position and orientation along with sensor processing (a). SAM sites detect presence of vehicles and emit radar emissions (b). UCAV successfully attacks SAM (c).

Table 2. IADS Mission Duration Comparison

SAM	Original GA	Parallelized GA	Island GA
1	1:56.70	1:56.70	1:56.70
2	2:06.82	2:05.26	2:03.82
3	3:02.96	3:05.92	2:41.35
4	4:41.07	3:34.7	4:17.55
5	6:41.94	4:21.83	4:35.28

Table 2 shows that in this simulation the distributed GA improves the effectiveness of the UCAVs in a SEAD mission. It does not provide much insight into whether the Island Model will provide a greater benefit than the Parallelized approach. In fact the Parallelized model performs better in this example after the fourth SAM discovery. It is important to remember that this is only one simulation run and the scenario unfolds very differently when running both distributed approaches. More investigation is needed to draw clearer conclusions and weigh the benefits of each approach.

VI. Conclusion

The Parallelized and Island Model versions of the GA outperform both the original GA formulation and the MILP. However, it is important to mention certain drawbacks. The Parallelized GA comes to a better solution faster, but requires an extra communication step at the end. This can be a problem if intra-team communication must be limited or if communication failures occur. On average, the Island Model shows improved performance over the Parallelized GA, but requires increased communication, thus leaving it more vulnerable to communication errors. The performance increase of the Island Model may not outweigh the demand in communication requirements compared to the Parallelized approach. These problems were not addressed in this research by assuming perfect communication between vehicles. Future work will explore possible techniques to ensure good solutions even when faced with imperfect communication.

References

[1]C. J. Schumacher, P. R. Chandler, and S. J. Rasmussen, " Task Allocation for Wide Area Search Munitions," in *Proc. 2002 American Control Conference*, Anchorage, Alaska, 2002.

[2]M. Darrah, W. Niland, and B. Stolarik, "Multiple UAV Dynamic Task Allocation using Mixed Integer Linear Programming in a SEAD Mission," in *Proc. 2005 AIAA Infotech@Aerospace Conference*, Alexandria, VA, AIAA 2005-7164, September 2005.

[3]T. Shima, S. J. Rasmussen, and A. G. Sparks, "UAV Cooperative Control Multiple Task Assignments using Genetic Algorithms," in *Proc. 2005 American Control Conference*, Portland, OR, 2005.

[4]T. S. Shima and C. Schumacher, "Assignment of Cooperating UAVs to Simultaneous Tasks using Genetic Algorithms," in *Proc. 2005 AIAA Guidance, Navigation, and Control Conference*, San Francisco, California, 2005.

[5]M. Darrah, W. Niland, B. Stolarik, and L. Walp, "UAV Cooperative Task Assignments for a SEAD Mission using Genetic Algorithms," 2006 Guidance, Navigation and Control Conference, submitted for publication.

[6]A. J. Chipperfield, P. J. Fleming, H. Pohlheim, and C. M. Fonseca, "Genetic Algorithm Toolbox User's Guide," ACSE Research Report No. 512, University of Sheffield, 1994.

[7]S. Rasmussen, J. Mitchell, C. Schulz, C. Schumacher, and P. Chandler, "A Multiple UAV Simulation For Researchers," in Proc. 2003 AIAA *Guidance, Navigation and Control Conference*, Austin, TX, AIAA 2003-5684, Aug 2003.

[8]Available to public: http://www.isr.us/research_sim_muav.asp

[9]D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, 1989.

[10]R. Shonkwiler, "Parallel genetic algorithms," in *Proc. of the Fifth International Conference on Genetic Algorithms (S. Forrest, Ed.)*, Morgan Kaufmann, Los Altos, CA, 1993, pp. 199–205.

[11]K. Miller, and R. Shonkwiler, "Genetic Algorithm/Neural Network Synergy For Nonlinearly Constrained Optimization Problems," in *Proc. of the 1992 International Joint Conf. on Neural Networks,* Baltimore, Maryland, 1992.

[12]E. Omiecinskii, and R. Shonkwiler, "Parallel Join Processing Using Nonclusterded Indexes for a Shared Memory Multiprocessor," in *Proc. of the IEEE Symposium on Parallel and Distributed Processing,* 1990.

[13]R. Shonkwiler, F. Mendivil, and A. Deliu, "Genetic Algorithms for the 1-D Fractal Inverse Problem", in *Proc. of the Fourth International Conference on Genetic Algorithms*, edited by R. Belew, and L. Booker, Morgan Kaufmann, San Mateo, CA, 1991, pp. 495-501.

[14]R. Shonkwiler, and E. Van Vleck, "Parallel speed-up of Monte Carlo methods for global optimization", *Journal of Complexity,* 1993.

[15]D. Whitley, S. Rana and R. B. Heckendorn, "Island Model Genetic Algorithms and Linearly Separable Problems," in *Proc. AISB Workshop on Evolutionary Computation,* 1997.

[16]A. L. Corcoran, and R. L. Wainwright, "A Parallel Island Model Genetic Algorithm for the Multiprocessor Scheduling Problem," in *Proc. 1994 ACM/SIGAPP Symposium on Applied Computing*, ACM Press, March 6-8, 1994.

[17]T. Hiroyasu, M. Miki, M. Hamasaki, and Y. Tanimura, "A New Model of Distributed Genetic Algorithm for Cluster Systems: Dual Individual DGA," in *Proc. International Conference on Parallel and Distributed Processing Techniques and Applications,* Volume 1, 2000.

[18]Air Force Research Laboratory. (2004, May 5). MultiUAV2 Simulation Users Manual. [Online]. Version 2.0, pp. 25-34. Available: http://www.isr.us/research_sim_muav.asp

[19]Niland, W., "The Migration of a Collaborative UAV Testbed into the FLAMES Simulation Environment," *Winter Simulation Conference*, Monterey, CA, 2006 (to be published).