AFRL-VA-WP-TP-2003-333

COMMUNICATION DELAYS IN THE COOPERATIVE CONTROL OF WIDE AREA SEARCH MUNITIONS VIA ITERATIVE NETWORK FLOW

Jason W. Mitchell
Corey Schumacher
Phillip R. Chandler
Steven J. Rasmussen

OCTOBER 2003

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

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE ***(DD-MM-YY)*** October 2003	**2. REPORT TYPE** Conference Paper Preprint	**3. DATES COVERED** ***(From - To)***

4. TITLE AND SUBTITLE
COMMUNICATION DELAYS IN THE COOPERATIVE CONTROL OF WIDE AREA SEARCH MUNITIONS VIA ITERATIVE NETWORK FLOW

5a. CONTRACT NUMBER
In-house

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
N/A

6. AUTHOR(S)
Jason W. Mitchell, Corey Schumacher and Phillip R. Chandler (AFRL/VACA)
Steven J. Rasmussen (Veridian, Inc.)

5d. PROJECT NUMBER
N/A

5e. TASK NUMBER
N/A

5f. WORK UNIT NUMBER
N/A

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

Control Theory Optimization Branch (AFRL/VACA)
Control Sciences Division
Air Vehicles Directorate
Air Force Research Laboratory, Air Force Materiel Command
Wright-Patterson AFB, OH 45433-7542

Veridian, Inc.

8. PERFORMING ORGANIZATION REPORT NUMBER
AFRL-VA-WP-TP-2003-333

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/VACA

11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)
AFRL-VA-WP-TP-2003-333

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

13. SUPPLEMENTARY NOTES
To be presented at the AIAA Guidance Navigation and Control Conference, Austin, TX, August 11-14, 2003. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

14. ABSTRACT

A communication model is incorporated into the MultiUAV simulation package for wide area search munitions. This model is used to study the effect of communication delays on the performance of an iterative network flow optimization model that results in a sequence of linear programs for the optimal allocation of consecutive assignments. Fixed communication delays are considered. The resulting delayed system performance is compared to the performance of a system with a perfect communications model for a set of vehicle-target scenarios to determine the effect on attack and verify task completion.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT:**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** (Monitor)
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**	SAR	14	Andrew Sparks
Unclassified	Unclassified	Unclassified			**19b. TELEPHONE NUMBER** *(Include Area Code)* (937) 255-8682

COMMUNICATION DELAYS IN THE COOPERATIVE CONTROL OF WIDE AREA SEARCH MUNITIONS VIA ITERATIVE NETWORK FLOW

Jason W. Mitchell[*], Corey Schumacher[†], Phillip R. Chandler[‡]
Flight Control Division
Air Force Research Laboratory (AFRL/VACA)
Wright-Patterson AFB, OH 45433-7531

Steven J. Rasmussen[§]
Veridian Inc.
Wright-Patterson AFB, OH 45433-7531

Abstract

A communication model is incorporated into the MultiUAV simulation package for wide area search munitions. This model is used to study the effect of communication delays on the performance of an iterative network flow optimization model that results in a sequence of linear programs for the optimal allocation of consecutive assignments. Fixed communication delays are considered. The resulting delayed system performance is compared to the performance of a system with a perfect communications model for a set of vehicle-target scenarios to determine the effect on attack and verify task completion.

Introduction

Autonomous wide area search munitions (WASM) are small, powered, unmanned air vehicles (UAV), each with a turbojet engine and sufficient fuel to fly for a short period of time. They are deployed in groups from larger aircraft flying at higher altitudes. Individually, they are capable of searching for, recognizing, and attacking targets. Cooperation between munitions has the potential to greatly improve their effectiveness in many situations. The ability to communicate target information to one another will greatly improve the capability of future search munitions.

Several methods have been previously studied to produce optimal single task assignments.[1,2] Recently, the optimal assignment of a sequence of tasks has been investigated using an iterative network flow model.[3] A common, and often implicit, assumption used in these models is that the vehicle-to-vehicle communication model is perfect, which typically implies that communication is both instantaneous and error free. Unfortunately, any physical implementation will almost certainly violate this assumption either due to design criteria or possible adversarial activity.

In this work, we couple the iterative network flow model for task allocation with a communication simulation that is incorporated into the existing MultiUAV[4,5] simulation software. By including a communication model into simulation software, we seek to quantify the effects of network delays on the performance on the iterative task assignment.

Background

We begin with a short description of a typical MultiUAV simulation scenario and a brief review of the iterative network flow model used.

Scenario

We begin with a set of N vehicles, deployed simultaneously, each with a life span of 30 minutes. We index them by $i \in \mathbb{Z}[1, N]$. Targets that might be found by searching fall into known classes according to the value or score associated with destroying them. We index them with j as they are found, so that $j \in \mathbb{Z}[1, M]$ and V_j is the value of target j. We assume that there is no precise a priori information available about the number of targets and their locations. This information can only be obtained by the vehicles searching for and finding potential targets via Automatic Target Recognition (ATR) methodologies. The ATR process is modeled using a system that provides a probability that the target has been correctly classified. The probability of a successful classification is based on the viewing angle of the vehicle relative to the target. For this ex-

---

[*] AIAA member, Visiting Scientist.
[†] AIAA member, Aerospace Scientist.
[‡] AIAA member, Senior Aerospace Engineer.
[§] AIAA member, Senior Aerospace Engineer.

ASC- 03-1709

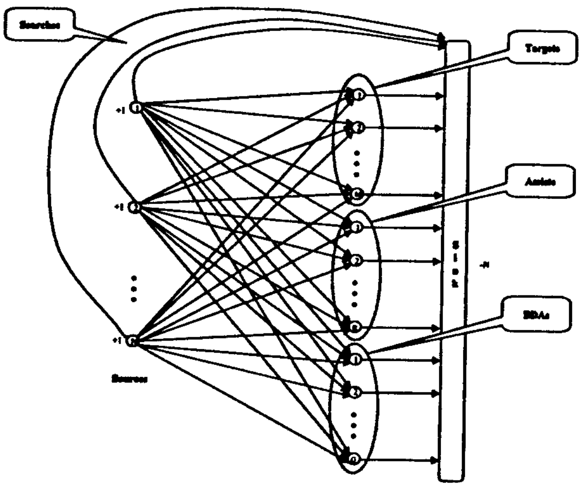

Fig. 1 Network flow model for task allocation.

ercise, the possibility of incorrect identification is not modeled, however targets are not attacked unless a 90% probability of correct identification is achieved. Further details of the ATR methodology can be found in Chandler and Pachter,[6] with a detailed discussion available in Chandler and Pachter.[7]

Network Optimization Model

Network optimization models are typically described in terms of supplies and demands for a commodity, nodes that model transfer points, and arcs that interconnect the nodes and along which flow can take place. To model weapon system allocation, we treat the individual vehicles as discrete supplies of single units, tasks being carried out as flows on arcs through the network, and ultimate disposition of the vehicles as demands. Thus, the flows are zero (0) or one (1). We assume that each vehicle operates independently, and makes decisions when new information is received. These decisions are determined by the solution of the network optimization model. The receipt of new target information triggers the formulation and solving of a fresh optimization problem that reflects current conditions, thus achieving feedback action. At any point in time, the database on-board each vehicle contains a *target* set, consisting of indexes, types and locations for targets that have been classified above the probability threshold. There is also a *speculative* set, consisting of indexes, types and locations for potential targets that have been detected, but are classified below the probability threshold and thus require an additional look before striking. Figure 1 provides an illustration of this model.

The model is demand driven, with the large rectangular node on the right exerting a demand-pull of N units (labeled with a supply of N), so that each of the nodes on the left (with supply of +1 unit each) must flow through the network to meet the demand. In the middle layer, the top M nodes represent all of the targets that have been identified with the required minimum classification probability, and thus are ready to be attacked. An arc exists from a specific vehicle node to a target node if and only if it is a feasible vehicle/target pair. At a minimum, the feasibility requirement would mean that there is enough fuel remaining to strike the target if tasked to do so. Other feasibility conditions could also enter in, if, for example, there were differences in the on-board weapons that precluded certain vehicle/target combinations, or if the available attack angles were unsuitable. The center R nodes of the middle layer represent all of the potential targets that have been identified, but do not meet the minimum classification probability. We call them *speculatives.* The minimum feasibility requirement for an arc to connect a vehicle/speculative pair is sufficient fuel for the vehicle unit to assume a position in which it can deploy its sensor to assist in elevating the classification probability beyond threshold. The lower-tier G nodes model alternatives for battle damage assessment (verification) of targets that have been struck. Finally, each node in the vehicle set on the left has a direct arc to the far right node labeled sink, modeling the option of continuing to search. The capacities on the arcs from the target and speculative sets are fixed at one (1). Due to the integrality property, the flow values are constrained to be either zero (0) or one (1). Each unit of flow along an arc has a benefit which is an expected future value. The optimal solution maximizes total value.

The network optimization model can be expressed as:

$$\max J = \sum_{i,j} c_{i,j}\, x_{i,j}, \qquad (1)$$

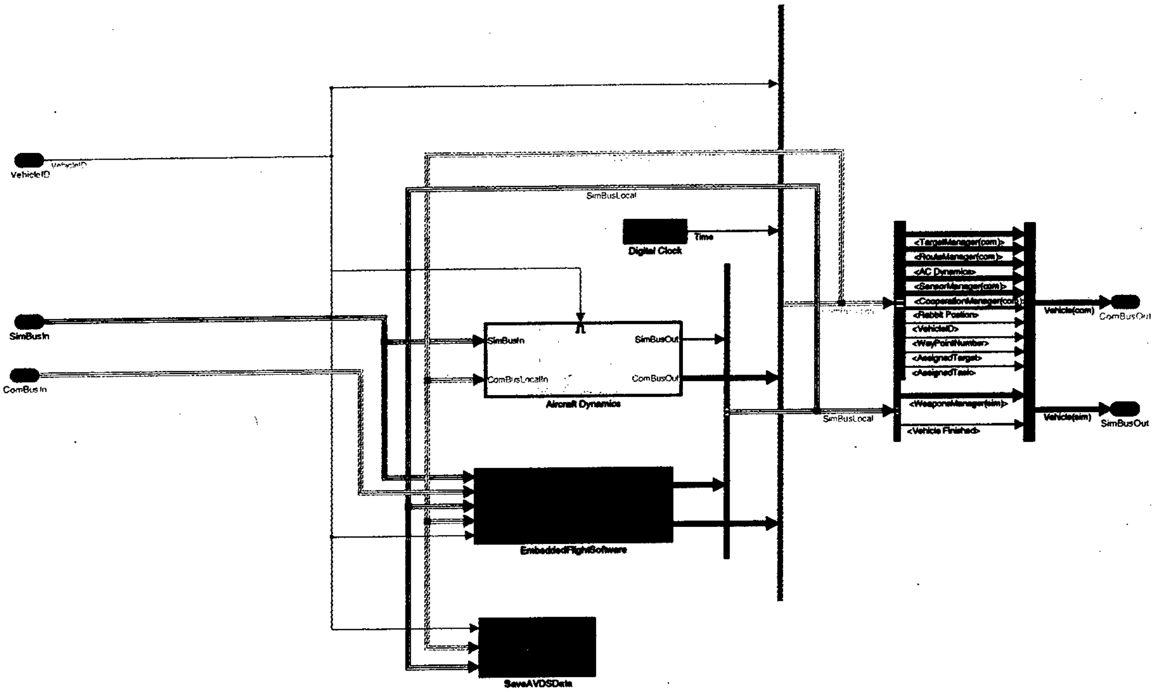

Fig. 2 Vehicle model with perfect communication.

subject to the following constraints:

$$\sum_{j,s}(x_{i,j}+x_{i,s})=1, \qquad \forall i\in\mathbb{Z}[1,n], \tag{2a}$$

$$x_{j,k}-\sum_{i}x_{i,j}=0, \qquad \forall j\in\mathbb{Z}[1,m], \tag{2b}$$

$$\sum_{i}x_{i,s}+\sum_{j}x_{j,k}=N, \tag{2c}$$

$$x\le 1, \tag{2d}$$

$$x\ge 0. \tag{2e}$$

This particular model is a capacitated transshipment problem (CTP), a special case of a linear programming problem. Due to the special structure of the problem, there will always be an optimal solution that is all integer.[2] Thus, solutions to this problem pose a small computational burden, making it feasible for implementation on the processors likely to be available on disposable wide area search munitions.

Due to the integrality property, it is not normally possible to simultaneously assign multiple vehicles to a single target, or multiple targets to a single vehicle. However, using the network assignment iteratively, *tours* of multiple assignments can be determined. This is done by solving the initial assignment problem once, and only finalizing the assignment with the shortest estimated arrival time. The assignment problem can then be updated assuming that assignment is performed, updating target and vehicle states, and running the assignment again. This iteration can be repeated until all of the vehicles have been assigned terminal attack tasks, or until all of the target assignments have been fully distributed. The target assignments are complete when classification, attack, and verification tasks have been assigned for all known targets. Assignments must be recomputed if a new target is found or a munition fails to complete an assigned task.

A more detailed discussion that includes the issue of the *benefit calculation* necessary for the assignment algorithm may be found in Schumacher et al.[3]

Vehicle Communications

As part of recent improvements to MultiUAV, a generic message passing scheme was incorporated as the Virtual Communication Representation (VCR) for remote communication.[5] In its original form, MultiUAV[4] could simulate a maximum of eight (8) vehicles and ten (10) targets, however this recent work eases the previous burden of extending these limits. Simulated vehicles are composed of embedded flight software (EFS) managers as well as vehicle dynamics and communication subsystems. The EFS managers implement the cooperative control algorithms, includ-

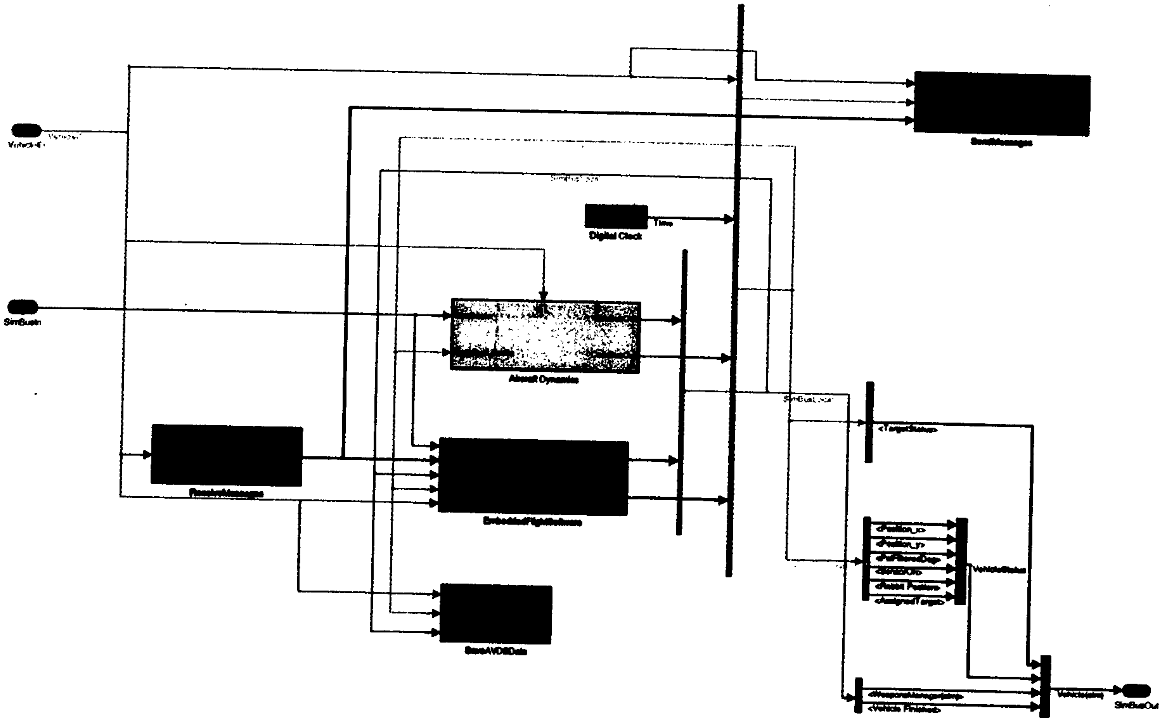

Fig. 3 New Vehicle model with communication simulation ability.

ing the iteratively applied CTP algorithm previously discussed. The vehicle dynamics are simulated with six-degree-of-freedom (SDOF) equations of motion. In addition, the vehicle model includes an autopilot that provides a waypoint navigation capability. The VCR is implemented as a combination of SIMULINK[1] and C++.

Previous versions of MultiUAV used to investigate the problem of optimal task allocation contained perfect and globally accessible information about vehicle and target states. From many perspectives, perfect information and instantaneous access is unacceptable, particularly when considering more realistic communication and processing delays. We can see from Fig. 2 that the previous version(s) provided vehicle-to-vehicle communication via a signal bus denoted by `CommBus`, while a second aggregated signal bus, labeled `SimBus`, contained the *truth* information for the simulation. The combination of these two data buses represented the complete information state of the simulation. The current vehicle model, which includes the VCR, can be seen in Fig. 3. The SIMULINK blocks that manage the communication representation are denoted `SendMessages` and `ReceiveMessages`, and are seen in Figs. 4 and 5, respectively. A detailed discussion of the messaging system design and implementation, as well as additional improvements to MultiUAV, may be found in Rasmussen et al.[5]

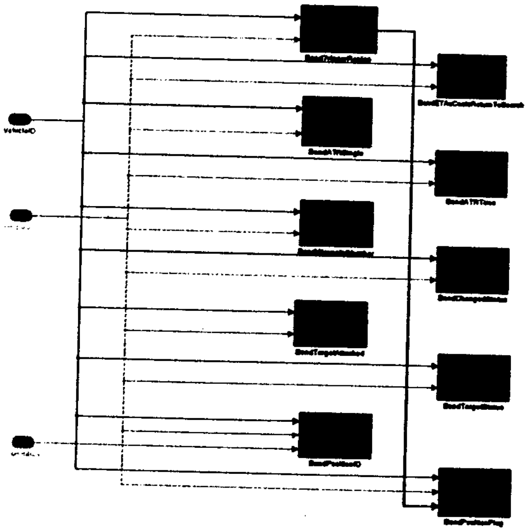

Fig. 4 SIMULINK `SendMessages` block.

---

[1] See The Mathworks website at `http://www.mathworks.com/`.

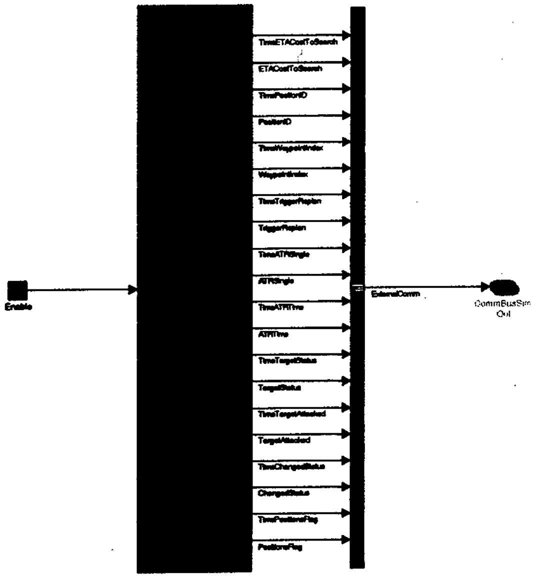

Fig. 5 **SIMULINK** **ReceiveMessages block.**

Simulation

To compare the performance of the CTP assignment algorithm with and without communication delays, we chose four particular cases of fixed delays and applied a MonteCarlo approach consisting of 50 simulations each. These simulations, for same initial seed, are compared using attack and verification task completion as metrics. Additionally, we directly compare two (2) same seed simulations for each of the delay cases.

Cases

The MultiUAV scenario chosen was three (3) vehicles and two (2) targets. The targets are placed randomly in a search-box, while the vehicles are aligned and tasked to search in a *lawn-mowing* fashion for a maximum mission time of 200 s. This scenario was then run for each of four communication delay cases shown in Table 1.

Table 1 **Communication delay cases.**

Case	Comm Delay
0	0 s
1	1 s
2	2 s
3[a]	2 s

[a] Includes an approximate 0.5 s self-processing delay on originating vehicle.

The specific delay values above were chosen largely out of convenience, however they are intended to represent a significant delay as compared to the lifetime of the vehicle in question. In this way, we hope to clearly quantify the effect such delays have on the cooperative control algorithms.

Table 2 **Percentage of successful attack**

Case (#)	None (%)	1-K (%)	2-K (%)
0	0.0	100.0	72.0
1	0.0	100.0	6.0
2	0.0	100.0	14.0
3	4.0	96.0	36.0

Table 3 **Percentage of successful verify**

Case (#)	None (%)	1-V (%)	2-V (%)
0	0.0	100.0	54.0
1	64.0	36.0	4.0
2	48.0	52.0	2.0
3	36.0	64.0	8.0

The self-processing delay of case #3 indicates that when a vehicle sends new information to remote vehicles, it may require additional processing time before the vehicle arrives at a decision. This, of course, is an initial step in moving all internal vehicle communication from a signal to a message base.

For all simulation runs, replanning is triggered only by the change of a target state. No timed or other replans are injected into the system.

Results

The summary data saved in the `MonteCarloMetrics` field of vehicle memory, shown in Tables 2 and 3, provides some interesting results. For no delay, we see that both targets are attacked approximately 72% of the time before exceeding the mission time. For verify, there is at least one verify 100% of the time and both targets are verified approximately 54% of the time prior to maximum mission time. For the delay cases, the percentage of both targets successfully being attacked is considerably reduced. This is largely due to a *target sink effect* induced by the delayed information because the vehicles no longer maintain a central information repository, thus the same task may be assigned to multiple vehicles. Since the vehicles keep their current task list until a target state changes, it is possible for the information to arrive too

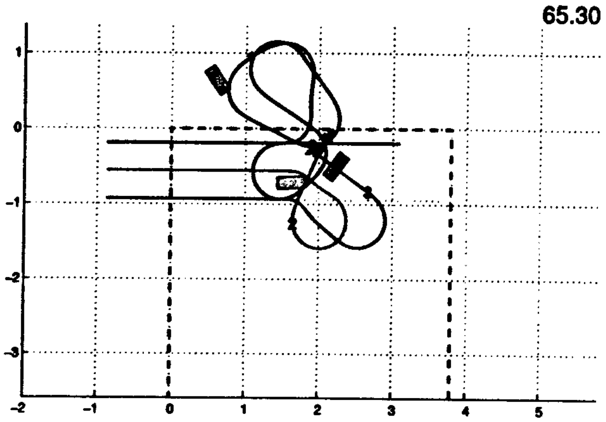

Fig. 6 Delay case #0 for seed value #1.

late to be of use, unintentionally resulting in multiple classifies, attacks, and verifies on a target. This becomes a significant problem in areas of high target density and with tight vehicle spacing. Surprisingly, as the delay increases, the percentage of two (2) successful attacks increases. For high target density and close vehicle spacing, the added communication delay results in slightly longer paths to the target and provides a buffer for new information to arrive before the vehicle acts on its replanned decision. The trend is similar for verification tasks, except that the number of two (2) target verifies is greatly reduced. This was due to a combination of the target sink effect, and incomplete tasks at maximum mission time.

It is more instructive to compare specific simulations individually. Qualitatively, this provides less ambiguous information regarding the effect of the communication delay. For each of the delay cases, simulation results for the first and last seed values were plotted.

For seed #1, we see the CTP selected paths in Figs. 6–9, where the graph scale is in miles. With the introduction of delay #1 [Fig. 7], we see the previously mentioned sink behaviour as vehicles two (V2) and three (V3) both strike target two (T2), while vehicle one (V1) has a sufficient travel distance to receive the attack information and replan to verify T2, then attack target one (T1). For delay #2 [Fig. 8], there is little difference from Fig. 7. Finally, with delay #3, seen in Fig. 9, we notice that V2 attacks T1, V3 attacks T2, and V1 verifies the attack on T2 and is scheduled to verify the attack on T1. Unfortunately, the maximum time is reached prior making the final verification.

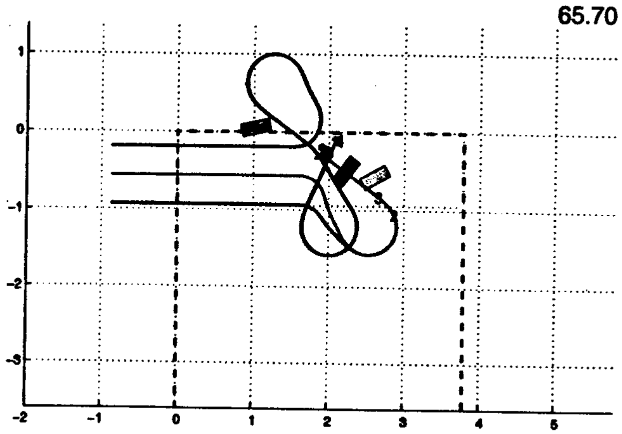

Fig. 7 Delay case #1 for seed value #1.

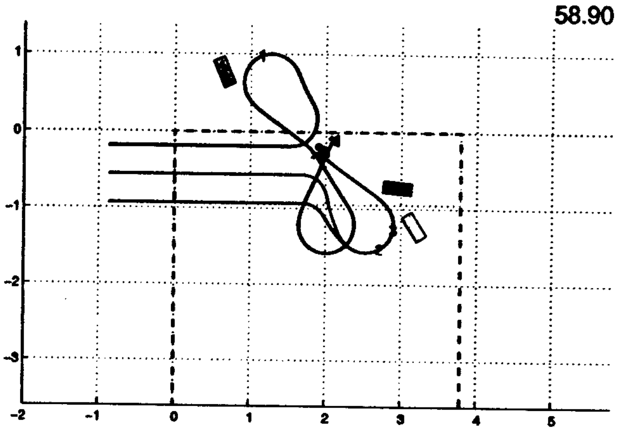

Fig. 8 Delay case #2 for seed value #1.

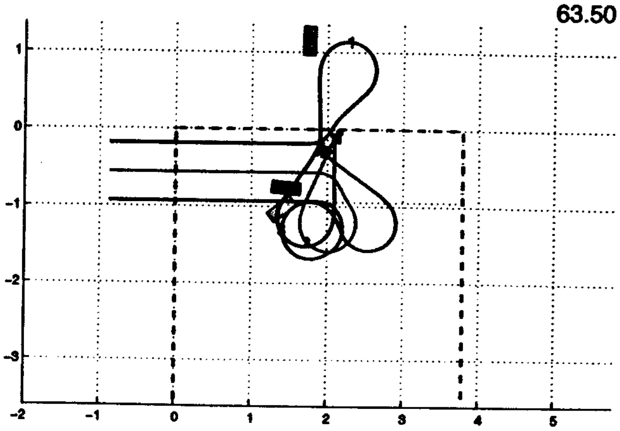

Fig. 9 Delay case #3 for seed value #1.

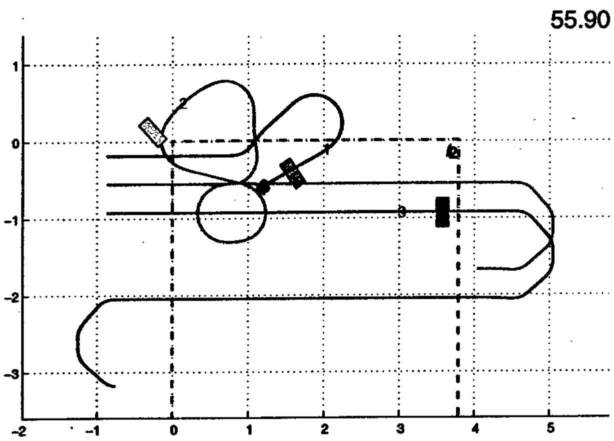

Fig. 10 Delay case #0 for seed value #50.

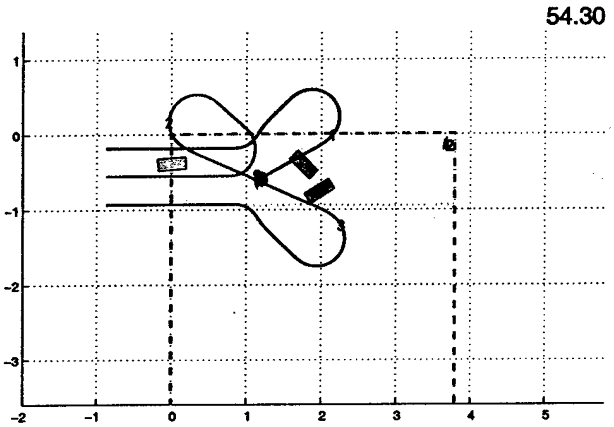

Fig. 11 Delay case #1 for seed value #50.

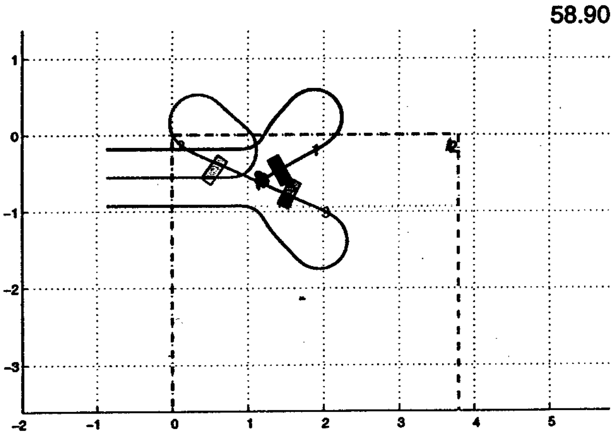

Fig. 12 Delay case #2 for seed value #50.

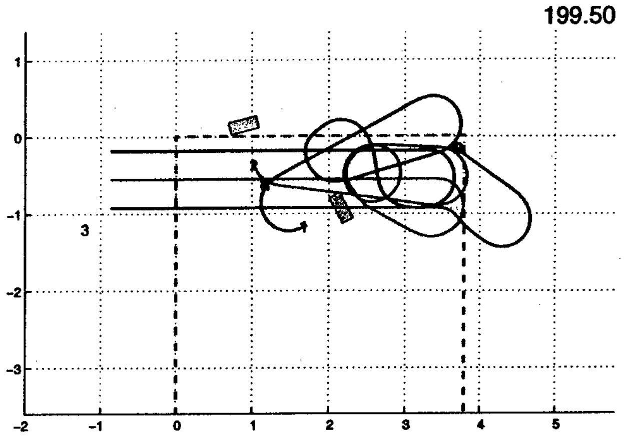

Fig. 13 Delay case #3 for seed value #50.

Interestingly, V3 displays a *churning* behaviour as a result of missing the initial replanned waypoint, i.e. V3 had already pasted the waypoint before it was received, resulting in a minimum turn radius circle to reacquire the planned path. This raises the possibility that a vehicle could churn forever if a missed waypoint fell sufficiently far inside the minimum turn radius.

Considering seed #50, the selected paths are seen in Figs. 10–13, again with a graph scale in miles. With no delay, T1 is successfully prosecuted, however T2 remains undetected since it resides in the search-lane of V1, who attacked T1. For delay #1 seen in Fig. 11, we again see the sink effect but for a single target, leaving vehicle spacing as the culprit. Again, there is essentially no change between Figs. 11 and 12. The paths for delay #3 become quite tangled as demonstrated in Fig. 13. In this case, the self-delay results in an unintentionally non-communicated message. As V2 detects T1, it does not trigger a position update. Thus, the vehicles find no need to replan. Upon discovering T2, a position update is triggered and a successful replan occurs. While T2 is prosecuted quickly, the verification is much later because T1 requires multiple passes to classify for attack, i.e. low ATR values. As before, we see a churning loop for V2 resulting from a delayed replan waypoint.

Improvements

With this first attempt to incorporate communication delay into the MultiUAV software, several possible improvements became apparent. The first possible improvement should aid in preventing churning motion.

When a vehicle receives position information, to perform a replan, the positions are simply extracted from the message an used without regard to the time-stamp indicating the time it was sent. Since the vehicle knows when the message was sent and received, it could assume the vehicles continue on their last reported heading and filter the positions to the current time. This should prevent the generation of a waypoint that the vehicle is near, but has passed. Of course, we could expect this to cause difficulty if a target status update occurred between the time the filtered-replan too place and new messages were checked.

A second potential improvement would be a move to message based internal vehicle communication. Passing messages internally would provide sufficient granularity to prioritize information and permit more realistic component modeling of vehicles.

For assigned tasks derived from delayed information, further improvement of may be made by transmitting each vehicles task list after a replan. This would provide a hand-shaking mechanism for verifying a successful replan and an opportunity to disambiguate same-task assignments that could result in sink behaviour. However, considerable care would need to be exercised in implementing such a mechanism so as to avoid circular replanning cycles.

Conclusion

In this paper, a communication model, which was incorporated into the MultiUAV simulation package for wide area search munitions, was used to study the effect of fixed-time communication delays on the performance of an iterative network flow optimization model. This network flow model results in a sequence of linear programs for the optimal allocation of consecutive assignments. The resulting delayed system performance was compared to the performance of a system with a perfect communications model for a set of three vehicle, two target scenarios. The effect of the delayed communication was seen as a significant decrease in successful attack and verify task completion. Most notably, we saw a target sink effect that resulted in a task being performed more than once on a target due to the absence of synchronized information. The typical sink behaviour resulted in a target receiving two or more classify-attack task combinations resulting from shorter delays and close target or vehicle proximity. This result is not particularly surprising since a lack of information implies a lack of cooperation. Further delay of the information improved the performance slightly by giving the vehicles a larger decision window before acting on a particular task assignment. In addition, vehicle churning was observed as a result of replan waypoints arriving too late.

While the observed cooperation showed significantly degraded performance, several improvements to address these issues became apparent. Of the potential improvements, projecting vehicle positions from the received data prior to performing a replan seems the most promising.

References

[1] Schumacher, C., Chandler, P., and Rasmussen, S., "Task Allocation for Wide Area Search Munitions Via Network Flow Optimization," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2001.

[2] Nygard, K., Chandler, P., and Pachter, M., "Dynamic Network Flow Optimization Models for Air Vehicle Resource Allocation," *Proceedings of the Automatic Control Conference*, 2001.

[3] Schumacher, C., Chandler, P., and Rasmussen, S., "Task Allocation for Wide Area Search Munitions Via Iterative Network Flow Optimization," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2002.

[4] Schumacher, C., Chandler, P., and Rasmussen, S., "MultiUAV: A Multiple UAV Simulation for Investigation of Cooperative Control," *Proceedings of the Winter Simulation Conference*, 2002.

[5] Rasmussen, S., Schulz, C., Schumacher, C., and Mitchell, J. W., "A Multiple UAV Simulation for Researchers," *Proceedings of the AIAA Modeling and Simulation Technologies Conference*, 2003.

[6] Chandler, P. R. and Pachter, M., "UAV Cooperative Classification," *Workshop on Cooperative Control and Optimization*, Klewer Academic Publishers, 2001 (to appear).

[7] Chandler, P. R. and Pachter, M., "Hierarchical Control of Autonomous Teams," *Proceedings of the AIAA Guidance, Navigation, and Control Conference*, 2001.