Multi-hop Communications in a Swarm of UAVs

Rachael Purta, Saurabh Nagrecha, Gregory Madey
Department of Computer Science and Engineering
University of Notre Dame
384 Fitzpatrick Hall, Notre Dame, Indiana 46556
Email: rpurta@nd.edu, snagrech@nd.edu, gmadey@nd.edu

Keywords: UAVs, swarm, multi-hop, DDDAS

Abstract

Unmanned Aerial Vehicles (UAVs) are of increasing interest to researchers because of their diverse applications, such as military operations and search and rescue. The problem we have chosen to focus on is using a swarm of small, inexpensive UAVs to discover static targets in a search space. Though many different swarm models have been used for similar problems, our proposed model, the Icosystem Swarm Game, to our knowledge has not been evaluated for this particular problem of target search.

Further, we propose to simulate the performance of this model in a semi-realistic communications environment. The challenge here is to find the optimal multi-hop configuration for the UAVs, so that they can find the most targets, avoid collision with each other as much as possible, and still communicate efficiently. We implement this through a weighted shortest-path problem using Dijkstra's algorithm, with the weights being the transmission cost over distance. Testing has shown that our multi-hop communications perform, in terms of target-finding and collision avoidance with other UAVs, as well as an idealized communications environment.

1. INTRODUCTION

As technology advances, many devices, including Unmanned Aerial Vehicles (UAVs) become smaller and cheaper. While this makes UAVs easier to purchase and more useful for various applications, such as intelligence gathering, search and rescue, and inspections of areas too small or dangerous for humans to reach, the fact remains that each UAV requires at least one ground-based pilot [1]. Some of the larger military UAVs or UAVs with many sensors may require more pilots on the ground. A UAV that is considered "micro" or "mini," however, has less maintenance cost and uses less fuel, so they cost less to operate overall [1]. These UAVs may be less expensive, but because they are so small, they have limited use by themselves. This is the reason why swarming technology has become a major area of research; there needs to be a way to control them all without overtaxing human resources, or else their use would be impractical.

In computer simulation, a swarm is a group of simply-behaving entities that together produce significant results or emergent behavior [1]. There are a number of swarming organisms in nature, though the swarms are often referred to by different names. A group of bees or ants are called a swarm, but a group of birds is a flock. Swarms have been observed in nature, however, to be able to accomplish a variety of tasks with a surprising level of efficiency, and have the other desirable qualities of decentralization and robustness [1]. These qualities are ideal for UAVs as well, and much research has focused on being able to control UAVs as they swarm, so that only one ground pilot can control multiple UAVs. Unfortunately, control of a swarm is difficult because swarms are, by definition, emergent systems, in which global behaviors can sometimes be unpredictable when adjusted at the local level.

Most research on UAV swarms is only in the simulation stage, as reliable control has not yet been achieved. But simulation has progressed enough that realistic parameters can be added, in order to give a better approximation of how the swarm models for UAVs will behave in the real world. Our project has added semi-realistic communication parameters to an existing UAV swarm model that we have implemented in a testbed system. Furthermore, this testbed system is designed according to the principals of Dynamic Data-Driven Application Systems (DDDAS), which will in the future allow real-world data to be incorporated into our running simulations. Thus, adding more realistic communication parameters to our simulation allows us to prepare for the real-world data we will receive when the real-world sensors are fully functional in the testbed. We will further describe DDDAS and its role in our testbed later.

1.1. Existing Model

Our existing UAV swarm model is based on the Icosystem Swarm Game. More information can be found on their website, [2], however the model we use is more closely based on the MASON implementation of the swarm game, which comes with the MASON package. MASON is the multi-agent simulation toolkit we have chosen to use for our agent modeling. The toolkit was first presented in [3], and it can be freely downloaded at [4]. The rules implemented in our model can be found in Figure 1. The model is agent-based, meaning each agent, a UAV in our case, follows its own instance of the rules. A description and evaluation of some of this model's properties can be found in [5], which analyzes the proper-

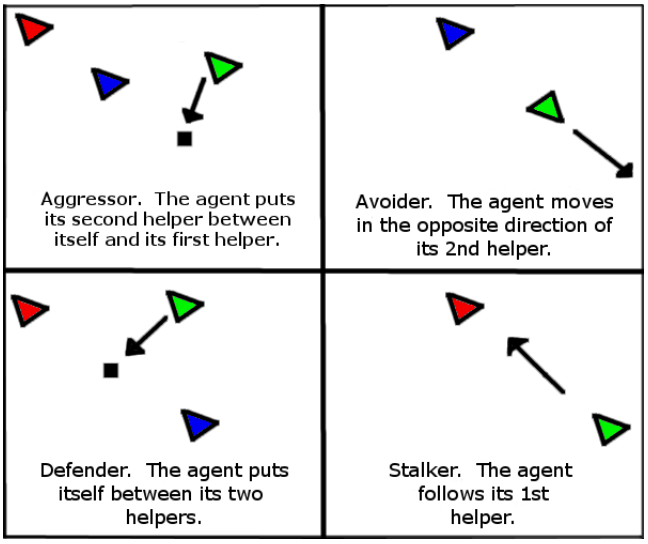

Figure 1: **Rules of Icosystem**

ties of clustering, drifting, and explosion that emerge from the model. Our version of the model has implemented all the rules shown in Figure 1, but only two, the aggressor and defender rules, are currently used. In addition, we have provided a table of the parameters used in our modified MASON simulation, which can be found in Table 1.

In order to better control the UAV swarm while still keeping the Icosystem-based model functioning, a leader-follower system has been added in the testbed implementation. Each UAV besides the leader still has two "helpers", as shown in Figure 1, but in our implementation the first helper is always the leader. We have done this because the Icosystem rules we have chosen both require the UAV agent to follow its first helper, and thus it will follow the leader. Figure 2 is an example dependency graph resulting from the addition of the leader. Note that the leader node is colored red.

2. RELATED WORK

Other swarm models have been tested in semi-realistic environments. For example, in [6], UAVs were given realistic movement, sight, and the ability to sense "pheromone signals left in the air from each other. Several strategies for covering the search space were tested, and for one of these, pheromone signals were the only form of communication the UAVs used. The signals were shown to improve the UAVs search for targets significantly compared to the other strategies. In [7], UAVs are given a cone-shaped line of sight and communicate based on this sight as well as short-range signals. In [8], UAVs can track moving targets and negotiate to

Parameter	Value Range	Explanation
Width	150	Width of the simulated world
Height	150	Height of the simulated world
GetBetweenAAndB	0.5	Defender mode of Icosystem rules
GetBehindBFromA	0.4	Aggressor mode of Icosystem rules
NumTargets	5 - 50	Number of targets in simulated world
MoveRandomly	0	Allows UAV to move randomly, instead of rules
GoTowardsA	0	Stalker mode of Icosystem rules
GoAwayFromB	0	Avoider mode of Icosystem rules
NumUAVs	6	Number of UAVs in the simulation

Table 1: Adjustable parameters of our simulation through the MASON interface. Ranges used in experiments shown with a hyphen (-).

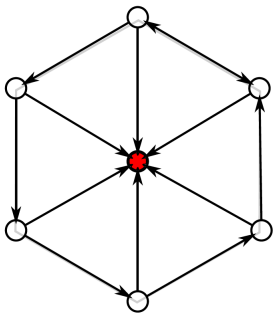

Figure 2: **Directed Graph of Dependencies.**

make sure only one UAV is investigating a particular target at a time. Targets can be covered by clouds, and UAVs can choose whether to pursue lost targets or not. Finally, [9] uses "communication swarms of UAVs as mobile nodes for communication, while their "sensor swarms perform a parallel-formation search of a space for targets. The sensor swarms have limited sight and fuel supply, are able to track a moving target, and can reorganize themselves when a UAV is shot down.

None of these studies, however, use the Icosystem Swarm Game model. The only works, to our knowledge, that have experimented with the model is [10], [11] and [5]. The first uses a genetic algorithm on the parameters of the model to

find which configurations produce interesting patterns in the swarm. The second introduces the rules of Icosystem's swarm game and shows how it and other models can benefit from agent-based modeling. The third creates mathematical models to prove whether certain model parameters will produce clustering, drifting, or explosion patterns.

In addition, we have applied the results of [12] to the communications network we have built for our UAV agents. The paper presents a method of efficiently determining the path a communication should take among multiple nodes, in order to extend the communication range of the system. An edge weight for the communication network, as suggested by [13], is used in order to model the attenuation of wireless signals. Mathematically, a signal transmitted with power P_t over a link with distance D fades and is received with power

$$P_r \propto \frac{P_t}{D_K} \quad K \geq 2,$$

3. IMPLEMENTATION

We have built a discrete event simulator to model semi-realistic UAV wireless communications as they search for static targets. With this system, we hope to simulate semi-realistic communications using our Icosystem-based swarm model, while still obtaining good search performance. Namely, we wish to optimize the number of discovered targets, while preventing as much collision between agents as possible.

Note that, because our system is aimed at small, inexpensive UAVs that can potentially have hundreds of agents in a swarm, collisions are viewed as possibly harmful but not destructive for our individual agents. The reasons for this are: first, small UAVs cannot move very fast in the first place and thus will not have enough impact force to destroy both UAVs; second, the UAVs are inexpensive and thus easily replaced; and third, full collision avoidance calculations may be too computationally expensive for a processor that fits on a small UAV. Thus, our simulation model uses simplistic collision avoidance calculations and we wish to minimize collisions through other methods.

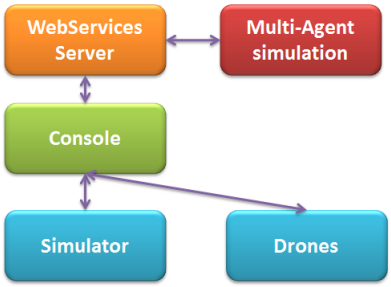

Figure 3: **Components of the UAV Target-Search Program**

The testbed on which we have built the discrete event simulator is implemented in Java and consists of five parts, shown in Figure 3. We will only be using four of the five parts, as the actual drones and their interface are not needed to run the simulation portion of the project. In fact, the system can run either the simulator or the drones, not both. The testbed uses a local Apache Tomcat server with RESTful web services installed in order to communicate between the simulator and the multi-agent simulation. The multi-agent simulation has been implemented using the MASON multi-agent simulation toolkit, developed at George Mason University and freely available for download at [4].

The use of each part of the testbed is as follows: the console displays the information log and the interaction of the UAVs with their environment, whether real or simulated; the multi-agent simulation displays and allows the setting of the parameters for the UAV swarm model; web services allows the multi-agent simulation to communicate and retrieve data from the real or simulated environment; and the simulator serves as the simulated environment. Our testbed system is designed this way in order to use the principles of Dynamic Data-Driven Application Systems (DDDAS), defined as a distributed system that has "the ability to incorporate dynamically data into an executing application simulation, and in reverse, the ability of [the system] to dynamically steer measurement processes" [14]. As applied in our system, DDDAS allows our swarm simulation (MASON) to receive location and other information from either real-world UAVs or simulated UAVs, and in return allows our swarm simulation to steer the UAVs.

Because this testbed is a distributed system, it can be challenging to make changes to the various modules, depending on the type of modification. The pieces of the UAV target-search project communicate heavily with one another, and any new data that needs to be passed between the MASON simulator and the console will require changes in both modules as well as, perhaps, the web services module. Such adjustments may demand significant time just to locate what areas of code to modify. In order to avoid this, our implementation of the discrete event simulator does not make any visual changes to the console, as we discovered doing so would result in the data passing mentioned. Instead, variable adjustment is within the code and hop networks are printed out to the screen using Java's standard output.

3.1. Multi-Hop Communication

Wireless communications are restricted by range. In a bid to attain maximum sweep of the target area, the agents of the UAV formation cannot always be in direct communication with their respective helpers. One way to achieve this would be to have a tight formation where each helper is within range. This reduces the sweep area of the formation and increases

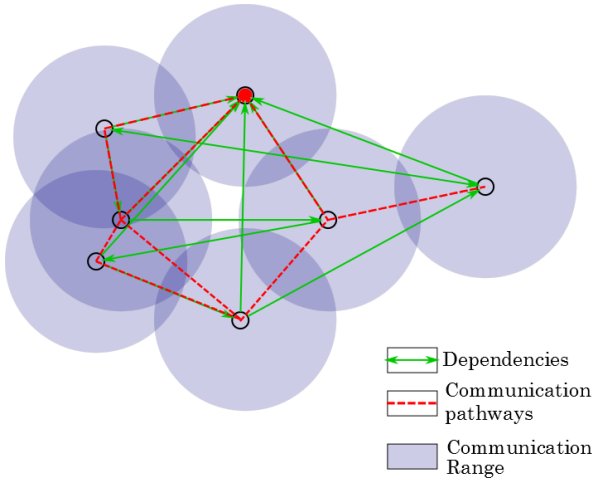

Figure 4: **Multilayer structure of the topologies involved.**

risk of UAV collision due to increased proximity. Multi-hop communications would not encounter such a problem as the UAVs use multi-hop communication in order to reach their respective helpers without compromising on formation as long as the network is one connected unit. Thus, we have chosen to implement multi-hop communication.

We can consider the communications network as a graph which straddles the graph of helper dependencies. Due to the limited range of communication of individual agents, they can only form communication links with agents within a certain distance. The dependencies stipulated by the latter can be realized by using paths formed in the communication graph. This setup is summarized in Figure 4. The solid arrows indicate dependencies between helpers as displayed in Figure 2. The circles indicate the communication range of individual UAVs. The dotted lines indicate communication pathways established due to proximity. Notice that these can be formed between UAVs which do not have any direct dependencies. Hence the graphs of dependencies and that of communication pathways are not necessarily isomorphic.

Multi-hop communication can be achieved in several ways. A variety of routing metrics can be considered to implement a multi-hop system, as shown in [15]. The more pertinent choice, however, was to consider the paths which involved a minimum expenditure of energy, as in [13]. In terms of algorithmic implementation, these paths are similar to that followed by Extremely Opportunistic Routing (ExOR) [16]. The idea is that splitting high power hops into smaller, low power hops, while keeping bit error rates in mind, might be advantageous and a more reliable means of communication. We use an inverse square decay model to characterize attenuation of signal. These scores are considered to be edge-weights in the communication network. The edge weights belong to the set of positive reals and the minimum energy path can be directly calculated using Djikstra's algorithm on these scores. These paths are used to communicate with helpers efficiently.

4. RESULTS

4.1. Evaluation Metric

The system has been implemented as a discrete event simulator on a desktop computer running on a Windows 7 (64-bit) operating system. The processor is an Intel i7 2760QM, which has a speed of up to 2.4GHz, and the computer itself contains 8 GB of memory. The testbed setup consists of the console window, which runs on the local Tomcat server, and the MASON window. The console contains static targets randomly distributed over a search space, a base station, and various UAVs, which are considered as point objects able to move in two dimensions. Modifiable parameters are present in the MASON window, which allows adjustment of the number of targets, the number of UAVs, and the ratio of Icosystem rules. As mentioned, though the communication range is adjustable, it is done in the code.

Our UAV model was verified using face validity - that is, it was verified with the function of small UAVs in mind, especially the commercially-available Parrot AR.Drone 2.0 that our earlier testbed experiments had worked with. We tested the system to make sure that it followed the leader when communication conditions were ideal, as well as verified that targets were detected when expected. We also, as mentioned above, made the collision-detection calculations simplistic so that a processor likely to be on such small UAVs would be able to handle them.

Our experiments test the discovery of these randomly-placed targets for the basic configuration of the UAVs versus the multi-hop network within a limited span of time. This time is the amount of time-steps needed to complete, to the best of the UAVs' knowledge, the mission. Note that the UAVs are not aware of how many targets are able to be found in the world, and so their definition of complete means that, first, all "discovered" (found but not investigated yet) targets have been investigated by the leader and second, all pre-set waypoints have been achieved. Pre-set waypoints are needed to make sure the UAVs cover all areas of the world, while still being allowed to move as freely as possible. We hypothesized that the multi-hop communications setup would out-perform the basic one in terms of the number of targets discovered, as well as lower the number of UAV collisions. As will be discussed, our hypothesis was only partially correct, though with more model adjustments could be improved.

Since the goal of the UAV mission is to find targets, how many targets are discovered seems to be an appropriate choice of metric. Collision avoidance also seems appropriate, as it is a desirable attribute for our system. As mentioned, we are aware that the collisions present in the system are due to the

UAVs' simplistic way of calculating potential collisions between themselves. In order to make the calculations as simple as possible for use on small UAVs, currently our algorithm does not take deceleration of surrounding UAVs, its own deceleration, and the time the individual UAV needs to stop and turn, into account. We felt that these calculations, while highly accurate, were unnecessary for good performance. Instead, collisions are predicted from the vector of all UAVs within an adjustable radius, and the vector of the UAV avoiding collision is adjusted accordingly.

4.2. Experiments

Baseline tests were performed in order to establish a comparison for our completed system. These tests consisted of two sets, one without any modifications to the original program, the other with initialization changes in order to mitigate the amount of random factors. Originally, the program placed each UAV in a random starting point somewhere in the world. Our second set of tests removed this random factor and instead started all UAVs in an evenly distributed circle around the base station, which is in the middle of the world. In addition, since we noticed in our first set of tests that some targets would not be found if they happened to be placed near the edges of the world, we also modified the second set of tests to have a wider range of waypoints for the leader to follow, thus forcing the UAVs to cover the edges of the world. After the implementation and testing of the multi-hop system was completed, we performed the same set of tests that we did for the second set of baseline. The multi-hop system also has these waypoints and non-random starting point changes.

Each set of tests consisted of 20 runs per target amount tested, with 10 of these recording the number of collisions between UAVs and all runs recording the number of targets discovered. A comparison of the UAV collisions recorded in the first and second set of baseline, as well as those recorded in the multi-hop system, can be found in Figure 5, while the comparison of the targets discovered in the three sets can be found in Figure 6. Each set displayed in the graphs was run with 10 targets generated in the simulated world, as well as 6 UAVs. Graphs show how many times the amount on the x axis (times collided or targets found) occurred. Note that in general, though the number of targets discovered in the second set improved, the number of UAV collisions worsened. This could be due to our change in the UAVs' original location, as it forces them to start off much closer together than the random location version, which tends to space the UAVs so far apart that it is difficult for them to cluster, and thus collide. The close initial positioning of the second set allows this clustering, but it also allows the UAVs to ensure that they are close enough to be in another's communication range. Perhaps this is why, in the third set shown in Figure 5, the amount of UAV collisions have settled at an average between the first and second sets. The number of targets discovered in the multi-hop system seems to follow this averaging pattern as well.

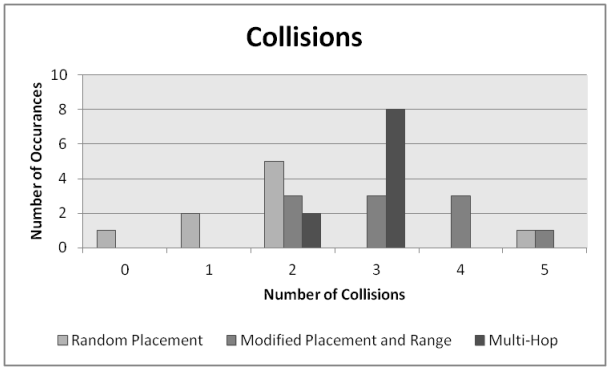

Figure 5: **Comparison of the number of UAV collisions**. Three sets are compared: random placement, modified placement and range, and multi-hop. Random placement has a random initial placement of UAVs, while modified placement and multi-hop do not. Modified placement has all UAVs start around the base station in the center and expands the range of waypoints. Multi-hop has multi-hop communication added.

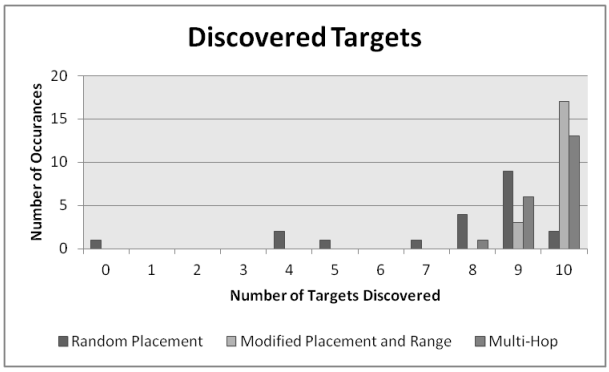

Figure 6: **Comparison of the number of targets discovered**. Three sets are compared: random placement, modified placement and range, and multi-hop. Random placement has a random initial placement of UAVs, while modified placement and multi-hop do not. Modified placement has all UAVs start around the base station in the center and expands the range of waypoints. Multi-hop has multi-hop communication added.

Our second set of tests had additional experiments not run in the first set, and we repeated these tests for the multi-hop system once it had been implemented. These tests varied the number of targets generated in the simulated world to 5 and 50, but still consisted of 20 target discovery runs and 10 collision counting runs each. The results can be found in Table 2 and Table 3. Note that the average number of timesteps needed to complete the search increases as the number of targets in the world increases, though the waypoints the leader

needs to visit are the same no matter the number of targets. The increasing time steps are perhaps due to the fact that the leader must verify each target it and its followers find, thus spending more time verifying as the number of targets found increases. In addition, the order in which the leader verifies each target has not been optimized, meaning that sometimes the leader may not visit the next closest discovered target and thus will waste time traveling. Another disadvantage of this extra traveling, as can be seen in Table 2, is the slight increase in the average number of UAV collisions as the number of targets existing in the simulated world increases. Not only does the extra time for travel allow more time for collisions between UAVs, but it was observed that the inefficient ordering of target verification would sometimes cause the leader to change direction sharply, giving UAVs around it less time to avoid collision. As also shown in the tables, these collisions were lessened with the implementation of the multi-hop system, however the average targets found per mission did not increase.

No of Targets	Avg No of Collisions for Baseline	Avg No of Collisions for Multi-hop Communications
5	2.4	1.4
10	3.2	2.8
50	5.6	4.5

Table 2: Average Number of UAV Collisions for Baseline and Multi-Hop Tests

	Baseline Tests		*Multi-Hop Communication*	
No of Targets	Avg Found	Avg Time Steps to Find	Avg Found	Avg. Time Steps to Find
5	4.85	903.00	4.8	871.9
10	9.85	1093.95	9.6	1095.6
50	49.85	2730.95	49.95	2838.55

Table 3: Average Number of Targets Discovered and Time Steps for Baseline and Multi-Hop Tests

Because we were curious as to how much communication range the UAVs needed before their communications broke down, we performed another set of tests with the multi-hop system. The results are shown in Table 4. Note that the pixel range given is the radius of the communication circle, centered at the UAV's location. The experiments at the range of 75 pixels were gathered from the other data sets, as it was our default range. We lowered the range, as shown in the table, until we began running into functional issues between the UAVs at a range of 40 pixels. Many runs at this range could not be fully completed, as the UAVs often lost track of the leader. In our current implementation, when this happens the UAVs travel to the top left-hand corner of the world. We found that this often prevents the UAVs from finding the leader again, so in the future we will change this to improve target-finding performance even when communication is breaking down. Note that the lower collision average and lower targets found average shown in Table 4 is a sign of communication breakdown, as when the UAVs become separated from the leader, they neither collide nor find targets. Still, the UAVs were mostly able to communicate with a range as low as 50 pixels, as shown in the table. We hope to be able to improve the communication to work at even lower ranges in the future.

Comm range (px)	Avg No of Targets Found	Avg No of Collisions
75	9.6	2.8
60	9.6	2.9
50	9.2	1.8
40*	8.6	1.1

Table 4: Effect of Communication Range in Multi-Hop Communication. Please note that "Targets Found" is out of 10. Not all runs were completed because UAV communications broke down. Thus, no experiments with smaller ranges were attempted.

5. CONCLUSIONS

Clustering the initial positions of UAVs in order to bring them within communication range is shown to increase the chances of collisions between UAVs during operation, but multi-hop communication seems to slightly mitigate these chances. The number of targets seems to affect the number of UAV collisions as well, where we see that a larger number of targets results in more collisions. Still, these results show that our hypothesis, in which we believed the number of UAV collisions would decrease with the implementation of multi-hop communications, was partially correct.

Where our hypothesis was incorrect was in the increase in the number of targets discovered. As shown in our results, the number of targets discovered with multi-hop was often (except for the tests with 50 targets) below that of the modified placement and range baseline. Perhaps this is because the communication range naturally clusters the swarm during

the entire simulation, and thus the UAVs do not spread out as much to find targets. However, this can be easily fixed with the addition of extra waypoints, to make sure that the UAVs cover the area of the simulated world. The addition of waypoints, however, will add to the time of the simulation, which has already been shown in our results increase as the number of targets increases. We wish, in the future, to optimize total mission time to be as low as possible. We shall discuss this in the next section.

With the implementation of the multi-hop system, the communication between UAVs is now more realistic, with small problems in communication affecting the whole system. Currently, a range of 50 pixels seems to be the point at which communication begins to break down, though the system is still functional. At 40 pixels, however, even the functioning of the swarm begins to fall apart. As mentioned, this is partially due to the current instructions the UAVs are given when they are unable to communicate with the leader. We hope to improve the range at which these problems occur in the future, by changing these instructions. With large (about 60 pixels and up for our current world size of 150 by 150) ranges, however, multi-hop communication performs reasonably well.

One advantage of our multi-hop communications system, however, is the minimization of energy utilization, which guides which UAV is chosen for the next hop. Since each UAV's score in our Dijkstra-based hopping algorithm is calculated from this energy use, we are guaranteed the least amount of energy possible in the path chosen. In a real system, especially with small UAVs that have very limited battery life, such conservation of energy use is invaluable. However, to be able to use our simulation with a real system, which is the ultimate goal of our DDDAS testbed, we still need to improve other parts of the simulation.

6. DISCUSSION AND FUTURE WORK

Parts of the simulation that still need improvement, for the sake of better performance, include the optimization of mission time, the addition of waypoints in order to cover the entire search space, and more effective handling of situations where one or more UAVs are unable to communicate with the leader. The first problem, that of mission time, we believe can be done by implementing a sorting algorithm for all targets found, in which they are ordered by distance, closest to farthest. We may have to implement other optimizations as well, since long missions are unfavorable for small UAVs with limited battery. The second, that of waypoints, we think requires a full circle of the world, which is only one or two additional waypoints. As mentioned, additional waypoints can add to mission time and thus having too many is not advantageous.

The third problem relates to communication, and is not as easy to solve. One possible solution would be to allow the UAVs to wander, obeying the rules of Icosystem's swarm model that we have chosen. As the rules we have chosen tend to keep the UAVs in one area, this does not seem to be a good solution. Another would be to have all UAVs go to the last point they can remember seeing the leader, and then allow them to wander there until the leader comes within range of one of them. Again, this will tend to keep the UAVs in one area, but at least that area is more likely to be near to the leader. Currently we believe this second option to be the better to improve communication with smaller ranges.

We also hope to incorporate further cosmetic changes in the future, which will allow visualization of the communication paths taken between UAVs. As mentioned, the paths taken are currently printed out to the standard Java output because such visual changes require modification to the console, web services, and the MASON multi-agent simulation, making it more difficult. With these visual changes added, it will be much easier to run our experiments with more than the 6 UAVs we used, as the printing will not overwhelm the computation needed for the program. We think that using more UAVs would be interesting, as it may help improve our communication as modeled.

All changes we make to the system, ultimately, will aid in improving the DDDAS-based testbed. The more realistic the simulation, the easier it will be to incorporate real-world data into it. Cosmetic changes also improve the system, as it helps the user understand, in a visual way, how the data is being represented. Once the changes listed above have been added, we hope to continue making improvements to the testbed in other areas besides communication, in order to make it more robust to all types of real-world data.

ACKNOWLEDGMENTS

This research was supported in part under grants from the Air Force Office of Scientific Research, award No. FA9550-11-1-0351, and the National Science Foundation, award No. 1063084.

The authors would like to thank Mikolaj Dobski, Artur Jaworski, Yi "David" Wei, and Ryan McCune for their contributions to the testbed used in this project. We would also like to thank Aaron Striegel, Hongsheng Lu, and Christian Poellabauer for their contributions to the project and paper.

REFERENCES

[1] R. L. Lidowski, "A Novel Communications Protocol Using Geographic Routing for Swarming UAVs Performing a Search Mission," Master's thesis, Air Force Institute of Technology, 2008.

[2] Icosystem Corporation, "The game." Online. url: http://www.icosystem.com/labsdemos/the-game/, accessed 12/20/12.

[3] S. Luke, C. Cioffi-Revilla, L. Panait, K. Sullivan, and G. Balan, "Mason: A multiagent simulation environment," *Simulation*, vol. 81, no. 7, pp. 517–527, 2005.

[4] S. Luke. Online. url: http://cs.gmu.edu/ ẽclab/projects/mason/, accessed 12/20/12.

[5] I. A. Gravagne and R. J. Marks, "Emergent Behaviors of Protector, Refugee, and Aggressor Swarms," in *IEEE Transactions on Systems, Man, and Cybernetics. Part B: Cybernetics*, vol. 37, pp. 471 – 476, IEEE, April 2007.

[6] Paolo Gaudiano, et al, "Swarm Intelligence: a New C2 Paradigm with an Application to Control Swarms of UAVs," tech. rep., Air Force Research Laboratory, 2003.

[7] J. Schlecht, K. Altenburg, B. M. Ahmed, and K. E. Nygard, "Decentralized Search by Unmanned Air Vehicles Using Local Communication," in *Proceedings of the International Conference on Artificial Intelligence, Las Vegas, NV*, pp. 757–762, 2003.

[8] H. Hexmoor, B. McLaughlan, and M. Baker, "Swarm Control in Unmanned Aerial Vehicles," in *Proceedings of International Conference on Artificial Intelligence (IC-AI), CSREA Press*, 2005.

[9] P. J. Vincent and I. Rubin, "A Swarm-Assisted Integrated Communication and Sensing Network," pp. 48–60, 2004.

[10] E. Bonabeau, P. Funes, and B. Orme, "Exploratory Design Of Swarms," in *2nd International Workshop on the Mathematics and Algorithms of Social Insects* (C. Anderson and T. Balch, eds.), pp. 17 – 24, December 2003. Online. Available at http://www.insects.gatech.edu /MASI2003_published_proceedings.pdf.

[11] E. Bonabeau, C. W. Hunt, and P. Gaudiano, "Agent-Based Modeling for Testing and Designing Novel Decentralized Command and Control System Paradigms," in *Proceedings of the 8th International Command and Control Research and Technology Symposium*, June 2003. Online. Available at http://www.dodccrp.org/events/8th_ICCRTS/pdf/037.pdf.

[12] M.Kodialam and T. Nandagopal, "Characterizing Achievable Rates in Multi-hop Wireless Networks:The Joint Routing and Scheduling Problem," in *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking*, pp. 42–54, ACM, 2003.

[13] S. Banerjee and A. Misra, "Minimum Energy Paths for Reliable Communication in Multi-hop Wireless Networks," in *Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pp. 146–156, ACM, 2002.

[14] F. Darema, "Dynamic Data Driven Applications Systems: New Capabilities for Application Simulations and Measurements," in *ICCS 2005* (V. S. Sunderam, ed.), pp. 610–615, 2005. Online. Available from http://www.dddas.org/iccs2005/papers/darema.pdf.

[15] R. Draves, J. Padhye, and B. Zill, "Comparison of Routing Metrics for Static Multi-hop Wireless Networks," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 4, pp. 133–144, 2004.

[16] S. Biswas and R. Morris, "Opportunistic Routing in Multi-hop Wireless Networks," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 1, pp. 69–74, 2004.

[17] Y. Wei, G. R. Madey, and M. B. Blake, "Agent-based Simulation for UAV Swarm Mission Planning and Execution," in *Proceedings of the 2013 Symposium on Agent Directed Simulation (ADS '13/SpringSim 2013)*, Society for Computer Simulation International, 2013. San Diego, CA.

[18] R. R. McCune and G. R. Madey, "Agent-Based Simulation of Cooperative Hunting with UAVs," in *Proceedings of the 2013 Symposium on Agent Directed Simulation (ADS '13/SpringSim 2013)*, Society for Computer Simulation International, 2013. San Diego, CA.

[19] A. G. Madey and G. R. Madey, "Design and Evaluation of UAV Swarm Command and Control Strategies," in *Proceedings of the 2013 Symposium on Agent Directed Simulation (ADS '13/SpringSim 2013)*, Society for Computer Simulation International, 2013. San Diego, CA.

[20] R. Purta, M. Dobski, A. Jaworski, and G. Madey, "A Testbed for Investigating the UAV Swarm Command and Control Problem Using DDDAS," in *ICCS 2013*, June 2013.