On Developing User Interfaces for Piloting Unmanned Systems

James Edmondson and Gene Cahill
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213–2612
Email: (jredmondson,gmcahill)@sei.cmu.edu

Anthony Rowe
Electrical and Computer Engineering
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213–2612
Email: agr@ece.cmu.edu

***Abstract*—Current artificial intelligence (AI) control paradigms tend to be one-to-one in nature and cater to monolithic systems, e.g., between one operator and one large, multi-functional robot. However, the future of AI is likely to be smaller, more distributed, and larger scale. Along that path, major advances have been made in the commercialization of smaller unmanned autonomous systems (UAS) like quad-copters and small ground-based robots that are equipped with wireless radios for communication between these systems and human operators, control stations, or other UAS. Even for systems built with capabilities for communication between UAS, the main control paradigm for such systems largely remains the same one-to-one paradigm and is geared toward joystick control or waypoint-based systems that force operators to define the complete paths for each UAS participating in a mission environment.**

In this paper, we will discuss recent efforts in user interface (UI) design in the Group Autonomy for Mobile Systems project at Carnegie Mellon University. The goal of the UI development is to reduce the cognitive load forced on human operators, especially those wanting to use swarms of devices in mission-critical contexts like search-and-rescue. We detail the coupling of distributed AI algorithms with these map-based, minimalistic interfaces, and highlight the decreased required user actions to accomplish similar swarm-based maneuvers against waypoint-based guidance systems. We believe these types of interfaces may prove pivotal in bringing group-based UAS control into more mainstream usage.

I. INTRODUCTION

As unmanned aerial vehicles and ground robots become less expensive and more pervasive, the need for intuitive control interfaces for multiple distributed systems in environments such as search-and-rescue, resource exploration, and other situations becomes more pronounced. However, current generations of interfaces for controlling multiple unmanned systems focus on micro-manipulation of individual systems within a swarm of autonomous entities. For instance, an interface for such a grouping of systems may require an operator to select one or more entities and give them a proximate waypoint, often via GPS, to move to. Examples of existing systems that do this include the AR.Drone 2.0 Free Flight Flight Recorder Application [3]; the 3D Robotics Mission Planner, Droid Planner, APM Planner; and AFRL's ATAK mission planning environment [2]. These concepts are extended further in 3D piloting systems based on landmark navigation [5] and augmented reality [4], as these interfaces tend to require the full attention of an operator on each UAS, rather than focusing on the group operations.

This type of system is cumbersome as the group size becomes larger and the state-of-the-art trends toward control paradigms of one human operator, often with a hand held controller, to one robotic system. This has hampered the adoption of multi-robot systems by agencies and companies that could highly benefit from the usage of multi-robot systems, such as search-and-rescue crews, because a high training cost is associated with multi-robot systems, due to each robot system requiring its own pilot. To reduce the cognitive load required in multi-robot system navigation and usage, we believe that user interfaces should be combined with distributed artificial intelligence underneath the UI and within the robotic systems to all single operators to more readily control multiple unmanned systems.

In this paper, we describe work done in the Group Autonomy for Mobile Systems (GAMS) project at Carnegie Mellon University that focuses on interfaces and tightly coupled distributed artificial intelligence algorithms that enable a single operator to control multiple unmanned systems. We highlight developed interfaces that effect area coverage with thermal sensors and also network bridging to connect operators across multi-hop communication paths using a small swarm of unmanned aerial vehicles.

II. THE PROBLEM WITH THE STATE-OF-THE-ART

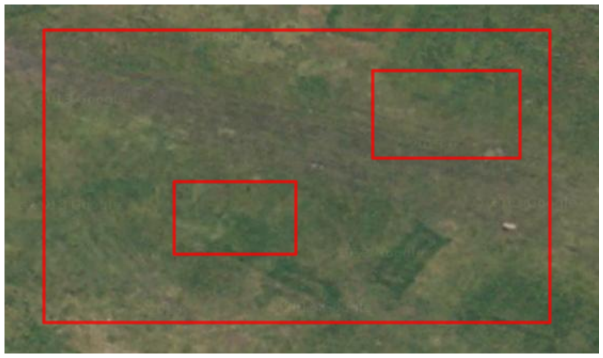

Fig. 1. Mission Scenario Provided to Three Untrained Users

To understand the benefits of the GAMS interfaces, we must first explore some of the problems with the state-of-the-art. In this section, we describe some simple, repeatable experiments with waypoint navigation systems, which as mentioned in Section **??** are used in software like AR.Drone 2.0 Free Flight Flight Recorder Application [3]; the 3D Robotics Mission Planner, Droid Planner, APM Planner; and AFRL's ATAK mission planning environment [2].

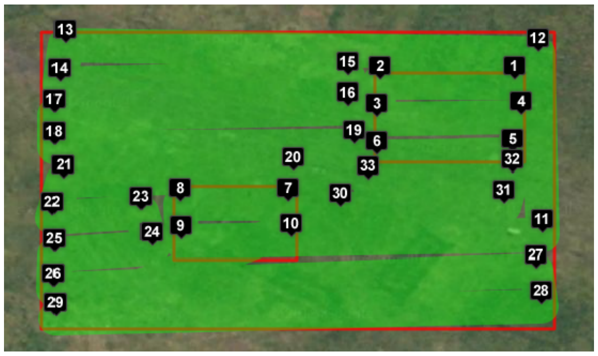

Fig. 2. User 1 First Plot

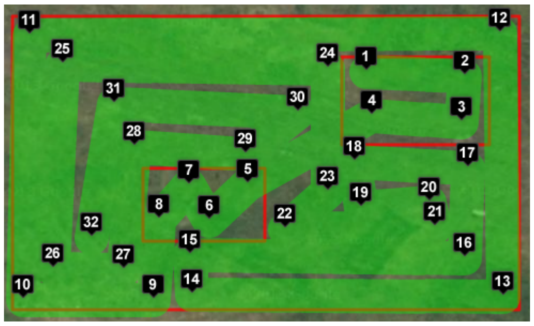

Fig. 3. User 1 Second Plot

In a waypoint navigation system, the user must select a device in an interface and then select a waypoint to traverse to. Most of these interfaces allow users to queue up waypoints (essentially plot a path of points), which allows an operator to plan out complicated missions. Figure 1 lays out the mission we asked of three volunteers, whom we allowed to replot their missions after seeing results that displayed the required clicks and the area that actually ended up being covered. The larger area is the main search area and the smaller areas are priority regions, places that are more important due to the mission and must be searched quickly (e.g., in a search-and-rescue mission where known survivors are at and triage decisions need to be made).

The following figures (Figure 2, 4, 5) are the results of handing the waypoint navigation to random people in our research lab, one of whom had experience flying waypoint systems. For space reasons, we picked only the three that showed the most interesting characters–whether showing higher user error or excessive diligence.

These are by no means meant to categorically and exhaustively explore the space of user interactions possible in the system. We show these here to highlight the differences between the mission requirements and the kinds of errors that are likely with human usage. Additionally, after these images, we breakdown the average user actions required to send one drone on a mission like this. As swarms grow larger, the waypoint system becomes even more cumbersome to use.

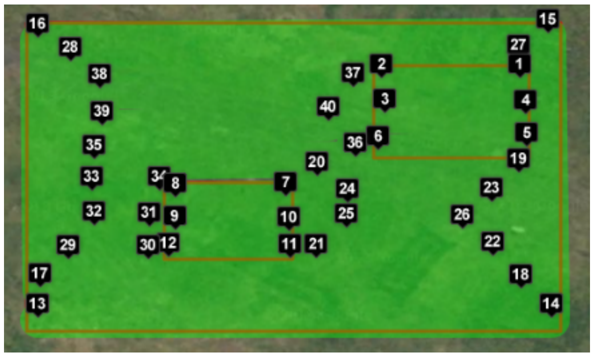

Fig. 4. User 2 Second Plot

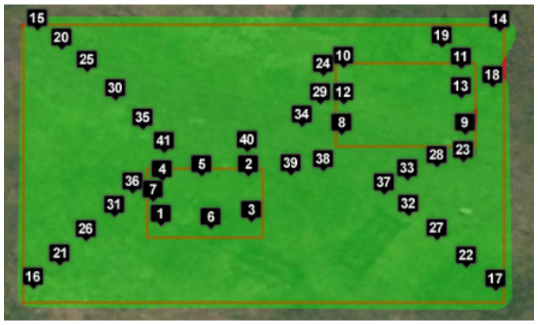

Fig. 5. User 3 Second Plot

Table I shows the marker counts, readjustments, user actions and time taken to use the waypoint navigation interface to search these three areas. The time taken and user actions varied wildly, and the accuracy of the search was also highly variable. These results highlight some of the problems in using such waypoint-based systems in mission-critical contexts like search-and-rescue but also in agriculture or any number of other scenarios where accuracy is important. A UAV attempting to spray pesticides over a field and missing large patches can be disasterous for crop growth and maintenance. A UAV that does not locate a survivor could result in unnecessary fatalities and improper triage in a disaster area.

TABLE I. RESULTS OF WAYPOINT NAVIGATION SYSTEM USAGE

Observable	Min	Max	Avg
Marker Count	32	45	37.166
Readjusts	0	31	9.166
User Actions	32	71	46.333
Time Taken	51s	123s	79.706s

In the next section, we will discuss the results of using the GAMS interface on similar problems.

III. THE GAMS INTERFACE AND SOFTWARE

As we have shown in section II, waypoint-based navigation systems are highly error prone and take a significant amount of time to task UAVs in even small mission contexts. In this section, we outline changes we have made to a map-based interface to facilitate mission-focused UAV tasking that requires less user actions and time, making it more suitable, especially, for high cognitive load situations like search-and-rescue.

The usage of the GAMS interface includes the following steps:

1) Select the UAS that will be participating with a box or by list of names
2) Select an action to perform from a list
    a) Primitive operations
    b) Network bridging between two locations
    c) Area coverage
    d) Priority area coverage

Primitive operations are functions like takeoff, land, move-to-gps, and other operations that are considered primitive AI. Network bridging is an instruction to the drones to move to locations that allow network traffic to pass between locations. Area coverage is a distributed AI that causes the drones to collaborate to cover a single rectangular area. Prioritized area coverage is a distributed AI that allows for the designation of prioritized regions within a main area coverage that the UAS will try to cover first.

The underlying artificial intelligence is implemented in the MADARA Knowledge and Reasoning Engine (KaRL) engine [1], a specialized and fast, real-time reasoning engine that was built for contextual inferencing in distributed, decentralized sensor networks. Our interface, which allows interaction with this AI, is built in Java for any device running the Android operating system. This allows us to control a swarm from a tablet or smartphone. Both the interfaces and the AI is released as open source under a BSD license at http://gams-cmu.googlecode.com.

In the next section, we will detail a small series of experiments that showcase the power of the interface and how it improves on the state-of-the-art in both required user actions and time taken to input a new swarm mission.

IV. Experimental Results

TABLE II. Comparison of GAMS and Waypoint Interface

Interface	γ	β	δ	κ	ρ
GAMS	4	20	N/A	1.0	0
GAMS	4	40	N/A	1.0	0
GAMS	4	80	N/A	1.0	0
Waypoints	12	20	0	1.0	0
Waypoints	22	40	0	1.0	0
Waypoints	42	80	0	1.0	0
Waypoints	9	20	-0.4	0.6	0
Waypoints	16	40	-0.4	0.6	0
Waypoints	31	80	-0.4	0.6	0
Waypoints	10	20	-0.2	0.8	0
Waypoints	19	40	-0.2	0.8	0
Waypoints	35	80	-0.2	0.8	0
Waypoints	15	20	0.2	1.0	0.2
Waypoints	27	40	0.2	1.0	0.2
Waypoints	52	80	0.2	1.0	0.2
Waypoints	19	20	0.4	1.0	0.4
Waypoints	35	40	0.4	1.0	0.4
Waypoints	69	80	0.4	1.0	0.4

Table II details the results of comparing the optimal usage of a waypoint system and the GAMS interface to perform area coverage with two UAS ($n = 2$) with a 4 meter sensor radius (θ). Due to space limitations and symmetry between number of drones used and minimum actions required, we only show the usage of two drones. The table reflects the following mathematical equation for determining user actions from a waypoint system.

$$\gamma = 2\left(\frac{\beta}{\theta(1-\delta)}\right) + n$$

Where β is the length of the shortest side of a rectangular search area. This is the key metric for determining the minimum number of waypoints required by a waypoint interface because of how it interacts with θ, the sensor radius of the UAS. γ is the minimum number of user actions required and is what we attempt to minimize in this research. δ is the user error as an underestimate or overestimate of the sensor radius by the human operator. κ represents the percentage of area covered as a result of the user actions. This becomes more important as results are shown when the user has deviations in δ (the user error in approximating sensor radius). ρ represents the percentage of overlap, which happens when the user underestimates θ and is indicated as a positive δ. Finally, n is the number of UAS, which is set to 2 due to space reasons and not shown in table.

The major takeaways from this data are that the waypoint system interface, even with just two drones, requires more user actions than the GAMS interface. Additionally, as shown in Section II, waypoint systems are prone to error, and the more error inserted into guessing the sensor radius, the more actions required and the more the area covered has overlaps or gaps. In the expressed data, we do not require additional user actions (γ) to fix gaps in the area covered by waypoint navigation systems.

V. Future Work and Conclusion

In this paper, we have discussed insights into research conducted in the Group Autonomy for Mobile Systems project at Carnegie Mellon University in regards to user interfaces for swarms of unmanned autonomous systems. The interfaces and coupled artificial intelligence discussed here are available as open source at http://gams-cmu.googlecode.com. Future work on this project is directed toward using these interfaces to direct dozens of UAS to perform complex mission tasks for search-and-rescue, agriculture, border patrol, and other tasks.

In addition to the group autonomy features described in this paper, the GAMS middleware includes support for directed autonomy, per UAS, via the user interace. We are also involved in research, via the Model Checking for Distributed Applications (MCDA) project at http://mcda.googlecode.com, to verify distributed applications for certain properties such as safety before deployment in real world scenarios.

VI. Acknowledgements

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. This material has been approved for public release and unlimited distribution. Carnegie Mellon is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University. DM-0001018

References

[1] J. Edmondson and A. Gokhale. Design of a scalable reasoning engine for distributed, real-time and embedded systems. In *Proceedings of the 5th conference on Knowledge Science, Engineering and Management, KSEM 2011*, Lecture Notes in Artificial Intelligence (LNAI). Springer, 2011.

[2] M. Gillen, J. Loyall, K. Usbeck, K. Hanlon, A. Scally, J. Sterling, R. Newkirk, and R. Kohler. Beyond line-of-sight information dissemination for force protection. In *MILITARY COMMUNICATIONS CONFERENCE, 2012-MILCOM 2012*, pages 1–6. IEEE, 2012.

[3] J. Irizarry, M. Gheisari, and B. N. Walker. Usability assessment of drone technology as safety inspection tools. *Journal of Information Technology in Construction (ITcon)*, 17:194–212, 2012.

[4] V. Koch, S. Ritterbusch, A. Kopmann, M. Müller, T. Habel, and P. von Both. Flying augmented reality.

[5] B. Ranft, J.-L. Dugelay, and L. Apvrille. 3d perception for autonomous navigation of a low-cost mav using minimal landmarks. In *International Micro Air Vehicle Conference and Flight Competition (IMAV2013), Toulouse, France*, 2013.