Towards Persistent Space Observations through Autonomous Multi-Agent Formations

Matthew P. Vaughan*
NASA Langley Research Center, Hampton, Virginia, 23666

Javier Puig-Navarro†
National Institute of Aerospace, Hampton, Virginia, 23666

Benjamin N. Kelley*, Walter J. Waltz*, Loc D. Tran‡, and B. Danette Allen§
NASA Langley Research Center, Hampton, Virginia, 23666

Sensing platforms must advance in scale and sophistication in order to support increasingly ambitious missions across Earth and space science; intelligence, surveillance, and reconnaissance (ISR); and planetary exploration. Distributed, persistent observation platforms have the potential to play a pivotal role in next generation missions through improved area coverage, enhanced situational awareness, and faster identification of trends and changes. The Multi-Agent Clusters for Persistent Observations from Space (MACPOS) project at NASA Langley Research Center is developing key technologies for the autonomous, heterogeneous formations that will comprise such platforms. Research thrusts include dynamic formation negotiation for self-assembling clusters of agents, distributed motion planning, and coordinated trajectory execution. Adaptive leader-follower formation negotiation allows agents to cluster and break off as necessary to adapt to both nominal and new mission objectives. Coordinated motion planning and execution maintain the formation while ensuring safe separation distances among agents and obstacles in the environment. These capabilities align MACPOS with NASA's initiative for space and surface in-situ assembly through fundamental technology development for autonomous multi-agent systems. This paper presents an overview and early progress for the MACPOS project. We describe the system architecture for both individual agents and the overall fleet. Design considerations are given for the planning, control, and metrology subsystems. Finally, we discuss planned project milestones and the expected course of development.

I. Nomenclature

LaRC	=	Langley Research Center
MACPOS	=	Multi-Agent Clusters for Persistent Observations from Space
GCS	=	Ground Control Station
IMU	=	Inertial Measurement Unit
GPS	=	Global Positioning System
CAS	=	Control Augmentation System
$\mathbb{R}^d$	=	d-dimensional real-valued vector space
$p_{d,i}(t)$	=	d-dimensional time-parameterized trajectory for agent i
AEON	=	Autonomous Entity Operations Network
DDS	=	Data Distribution Service
QoS	=	Quality of Service
DRM	=	Design Reference Mission

*Research Computer Scientist, Autonomous Integrated Systems Research Branch (AISRB)
†Research Engineer, AIAA Member
‡Computer Engineer, Flight Software Systems Branch (FSSB)
§Senior Technologist for Intelligent Flight Systems, Autonomous Integrated Systems Research Branch (AISRB), AIAA Associate Fellow

II. Introduction

As sensing technologies becomes more advanced, ubiquitous, and accessible, ambitions and expectations for that technology are growing even faster. Persistent and distributed sensing is a key enabling capability across a wide swath of applications. Large scale correlative and multi-modal sensing missions will provide the Earth science data needed to more thoroughly monitor and understand a changing climate. Networks of coordinated mobile robots can create ad hoc communication and metrology infrastructure to support space operations and exploration. Similar systems and technologies can also serve national security interests through enhanced situational awareness, more accurate trend identification, and timely notification in areas of strategic value. Regardless of the application, low-level management of the assets and data involved in persistent, distributed systems quickly becomes intractable for human operators. Autonomy will be required to scale up ubiquitous sensing missions to their full potential.

Autonomy for persistent, distributed sensing platforms is the subject of a two-year research project at NASA Langley Research Center (LaRC) called Multi-Agent Clusters for Persistent Observations from Space (MACPOS). The goal of the project is to develop and demonstrate autonomous formation flying capabilities. Autonomous, coordinated, multi-agent operations are a fundamental capability that will support NASA's space and surface in-situ servicing, assembly, and manufacturing initiative [1]. In this paper we present the MACPOS project concept and initial status. Section III offers a project overview covering goals for technology development as well as related work in multi-agent systems. Section IV discusses the problem definition in terms of algorithm design considerations and concepts that will drive the fleet. Section V describes the system architecture at both the single agent and multi-agent granularity. Section VI presents the design reference mission and other planned demonstrations, and Section VII covers the work planned for the remainder of the project along with future work. Finally, Section VIII concludes with a summary and discussion of potential project impacts.

III. Project overview

A. Objectives, Requirements, and Assumptions

The MACPOS project is developing and demonstrating algorithmic building blocks for autonomous ground, air, and space formation operations. To support long duration applications, the multi-agent system must be capable of dynamic formation reconfiguration. Mission requirements for persistent sensing applications are likely to change from the initial deployment either slowly as data reveal new insights or suddenly through the discovery of new areas of interest. Environmental conditions may also warrant reconfiguration for hazard avoidance or to improve robustness in adverse conditions, and inevitable faults and failures within the fleet can be mitigated if the formation can be restructured. The desired formation restructuring capability not only considers the physical arrangement of agents but also their command and control hierarchy. For example, a fleet may begin with a ground control station (GCS) communicating with a single leader agent driving many followers during fully centralized operations. New areas of interest may arise that require breaking out a subset of the fleet as a separate cluster to investigate. As the mission evolves over time and new tasks are dispatched and completed, clusters break off and rejoin the main fleet. In the limit, a cluster may consist of just a single agent in situations where some or all of the fleet may need to temporarily break formation to search an area or dodge a dynamic obstacle.

In addition to the primary driving characteristic of formation adaptability, MACPOS is considering several other requirements for fleets executing multi-agent autonomous sensing missions. In general, the system will consist of heterogeneous agents with varying locomotion, sensing, communication, and computation ability. While the project is geared towards persistent observations by groups of small satellites, much of the architecture and algorithms under development are platform-agnostic and can be applied to mixed teams of ground, air, and space vehicles. Regardless of the platform, the need for safe proximity operations in an uncertain environment has a major impact on the feasible sensing, planning, and control algorithms available to each agent. The complexity of the problem further increases if assumptions about reliable communication networks are relaxed to reflect more realistic hardware. Characterizing the degradation in performance of the multi-agent system as network quality deteriorates will inform mitigation strategies such as upgrading hardware or imposing constraints on formation size, shape, and composition. Lastly, predictability is paramount for the use cases targeted by the project, so the utility of algorithms relying on emergent behavior is limited.

B. Related Work

Multi-agent systems and networking are well-studied topics with more recent efforts attempting to encourage more pragmatic, robust, distributed systems. Inclusion of agent and network characteristics add complexity to multi-agent optimization problems that combine system goal criteria, dynamics, and maintenance of network communication [2]. Leader/follower configurations are common for multi-agent systems where a leader guides follower agents to specified goal conditions or objectives. Leaders are expected to coordinate with their followers as well as other leaders to convey goals, commands, and progress toward global objectives. The leader must also solve consensus problems among its followers and other leaders to determine goals and sub-goals. The follower agents in the group use techniques such as solving potential fields to maintain proximity to the leader.

Realistic implementations of multi-agent systems must consider network communication characteristics, available agent power [3], computational resources, algorithm convergence and optimality, and incorporation of vehicle dynamics [4]. Multi-agent systems that employ consensus-based algorithms or otherwise depend on information sharing among agents will observe a strong coupling between system performance and network bandwidth and reliability [5]. As increasingly complex functionality is expected of each agent, leaders need to evaluate agents' fitness to perform tasks or meet objectives due to lack of power or insufficient computation. Wireless mesh networks supporting these systems need to be robust, versatile, scalable, and have sufficient range [6].

C. Research and Development Efforts

Research and development (R&D) thrusts for the project fall into five categories. First, adaptive coordinated clusters of autonomous agents must explicitly consider network communication constraints during planning and execution. Information about formation structure, command hierarchy, and mission objectives must propagate through a dynamic network topology where connectivity may be intermittent. Second, given a formation structure and mission objective, agents must collectively generate feasible trajectories to establish and maintain the formation, achieve the objective, and ensure safe separation distances among each other and any environmental hazards. Fast multi-agent trajectory generation will require a novel integration of proximity queries that provide additional geometric information to prune the search space and reduce the computational burden. Third, safe and accurate formation flying in the presence of real world disturbances and faults requires agents to continually coordinate their trajectory execution progress. Agents in a formation may need to speed up or slow down to ensure that both the safe separation distance and the nominal formation structure are maintained within the tolerance required by a particular mission's risk profile.

The last two categories of R&D work for the MACPOS project are predominantly engineering efforts rather than research. First, any robotic system must implement a metrology strategy to sense and interpret the world to the extent required by its operation. Depending on the vehicle platform, the specific sensors available to each agent will vary. However, for the purpose of this project, each is assumed to have at least an inertial measurement unit (IMU), a camera, and some source of external positioning information such as GPS during outdoor operation or a motion capture system indoors. These sensing modalities represent a baseline of external sensing, inertial sensing, and positioning that small satellite platforms typically have with one important caveat. The sub-millimeter precision of the indoor motion capture system must be filtered to emulate the noise, fault, and data rate characteristics of GPS modules that would be available to a realistic vehicle fleet. The final category is the software framework that ties the multi-agent system together as described in more detail in Section V.

IV. Problem Definition

The research efforts in MACPOS focus on the development of collaborative algorithms for groups of autonomous agents that are tasked to solve distributed motion planning and motion control problems over lossy communication networks. The algorithms in this research are tailored for ground robots, autonomous aircraft, and small satellites, but several components are vehicle-agnostic and could be extended to manipulators and other autonomous systems.

The motion planning problem aims to generate a set of smooth, piecewise, time-parameterized trajectories $\boldsymbol{p_{d,i}}(t)$ for a set of n agents, that satisfy the kinodynamic constraints of these vehicles under certain simplifying assumptions with $i \in \{1, \ldots, n\}$. To this end, the fleet members are organized as leaders and followers. Leaders can either be real or virtual vehicles and are assigned a group of follower agents, as shown in Fig. 1. Each follower is assigned to a single leader and must maintain a formation around its leader in coordination with the other followers. For the purposes of this work, flight formations can change over time, and are specified by either an operator or a higher-level algorithm. Note that in Fig. 1, one of the teams reorganizes on the fly, and hence the vehicles must collaborate to carefully design the timing of their trajectories and avoid collisions. These transition paths are represented by dotted lines. The next step in

the problem formulation is to separate the kinodynamic constraints into kinematic and dynamic constraints to address how these are tackled by the trajectory generation algorithm.

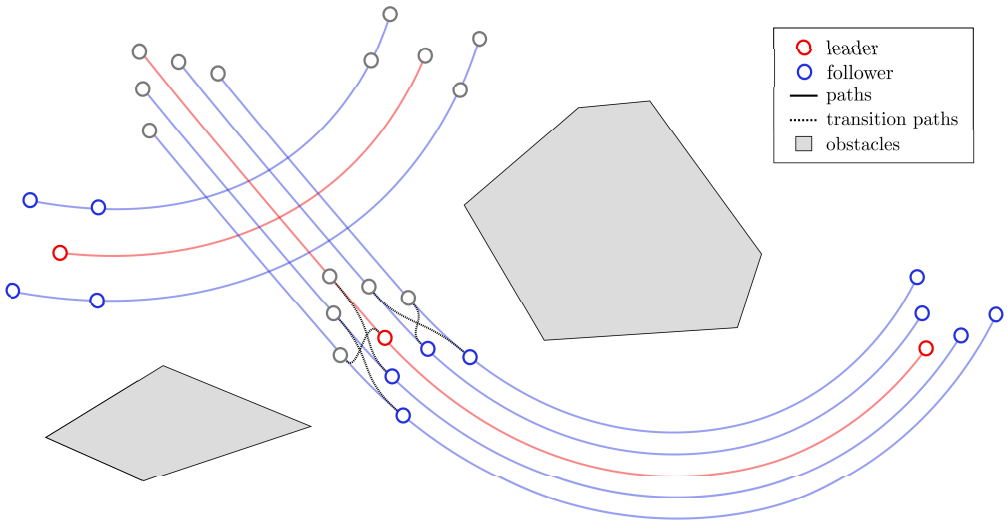

Fig. 1 Two teams with a leader and a group of followers in a cooperative mission.

To formulate the dynamic constraints of this problem, each agent is first given a simplified dynamic model [7–9]. This dynamic model is used to derive a nominal path-following algorithm, and a nominal Control Augmentation System (CAS), as depicted in Fig. 2. The simplified dynamic model reflects the effects of the different forces and torques on the states to be controlled. Then, the path-following algorithm uses the trajectory assigned to the vehicle and the dynamic model to compute the desired torques and forces that guarantee the trajectory is tracked under nominal conditions as in [10]. Finally, these desired torques and forces are transformed into actuator commands by the CAS [11]. This cascaded system is used to compute the nominal actuator commands for a given trajectory. Then, the trajectory generation algorithm must check whether the resulting actuator commands remain within a safety envelope. In this work, the safety envelope is assumed to be a convex polytope [12], a conservative region within the actual limits of the vehicle. Hence, to discard or accept a trajectory candidate, proximity queries will be developed to check whether a collection of time-parameterized curves in $\mathbb{R}^d$ remain within a polytope, where d represents the number of actuators on each vehicle. An initial approach to developing some of those queries could leverage tools formulated in [13].

To formulate the kinematic constraints of this problem, the environmental obstacles are modeled as convex polyhedra. The desired position for the center of mass of each vehicle is given by a time-parameterized curve, and each vehicle is assigned a safety distance for collision avoidance that defines the radius of a sphere around the center or mass. All pairs of trajectories and safety distances define a set of tubes through which each vehicle will move if the mission evolves as planned. To ensure the designed trajectories do not violate the safety distance between obstacles, tolerance verification queries are used [14]. Tolerance verification queries allow the system to consider not only the bounding sphere around a vehicle, but also any additional uncertainty that may not be captured in the geometric modeling of the environment or the expected path-following errors. In addition, to avoid collisions among cooperating peers, tolerance verification queries assist in temporally deconflicting each pair of time-parameterized tubes that could potentially lead to vehicle collisions [15]. As highlighted in Fig. 2, a candidate trajectory must pass both the dynamic and kinematic checks to be approved as a valid trajectory.

Trajectory generation and proximity query algorithms for collision detection, tolerance verification, and distance computation are rarely developed in a cohesive manner. However, proximity queries are often the backbone of trajectory generation protocols since they are the fundamental tool to determine whether a candidate trajectory maintains a safe separation with obstacles and cooperating vehicles. Normally some research groups develop proximity query algorithms, and separate groups leverage them as state validation black boxes in their trajectory generation algorithms. As a result, the proximity query algorithms spend computational resources to reconstruct relevant geometric information that could be cached and leveraged to speed up trajectory generation. However, this information is rarely exposed through the proximity query Application Programming Interface (API). For this reason, this effort also aims to integrate

proximity queries and trajectory generation in a more synergistic manner than existing approaches. For a fleet of n cooperating agents in close proximity, the number of proximity queries required to ensure safe separation is $O(n^2)$. A synergistic approach holds the potential to discard a portion of these proximity queries and speed up trajectory generation. Other recent approaches augment the proximity queries algorithms with a neural network to quickly discard certain queries [16].

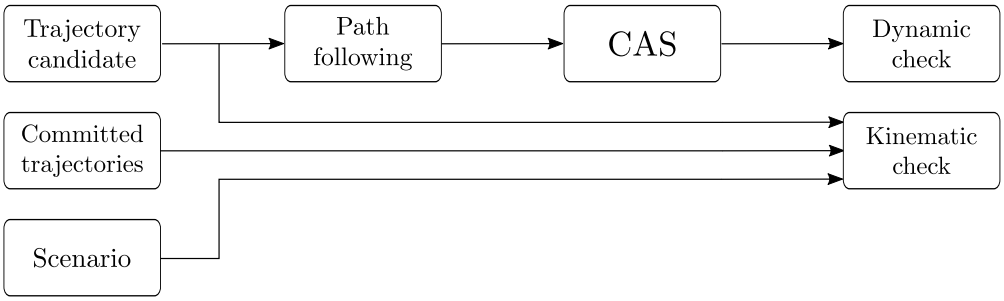

Fig. 2 Flow of information in the trajectory validity checker.

If the algorithm finds a solution for all the vehicles, the result is a set of n smooth, piecewise, time-parameterized trajectories that guarantee safe separation among all vehicles and obstacles, while maintaining all inner control commands within a specified safety envelope. However, the resulting paths assigned to multiple agents may intersect in space. This is especially likely when working with ground vehicles constrained to move in two dimensions. In the context of this research, there are two main reasons to ensure that all agents execute their trajectories in a synchronized fashion: *i*) if certain fleet members fall behind or are ahead their planned schedule, the cooperating team may not maintain the prescribed flight formation, and *ii*) intersecting paths that are temporally deconflicted [17] may lead to vehicle collisions unless the mission execution is synchronized online. For these reasons, a distributed time-critical coordination algorithm must be implemented to allow the cooperating team to synchronize the trajectory execution online [15]. To this end, the vehicles exchange a set of states, referred to as the coordination variables, over a lossy wireless communication network. Consequently, the time-critical coordination algorithm equips the fleet with an online tool to negotiate the progress of the mission as the different assets are subject to unexpected or unpredictable events such as winds or partial vehicle failures. For example, if a vehicle falls behind with respect to its peers, then the peers may slow down to give the delayed vehicle the opportunity to catch up. A correct motion planning solution, combined with online coordination, enforces relative spatiotemporal constraints among fleet members in real time, which ensures the flight formation is maintained and guarantees safe separation among agents. In addition, the coordination algorithms also enforce absolute spatiotemporal constraints such as a window of arrival or a precise time of arrival at a particular location. These types of constraints are handled through the introduction of a reference agent that does not negotiate with other agents [18]. The motion of the reference agent encodes the actual mission schedule and allows absolute timing information to propagate across the cooperating fleet. The reference agent is often directly controlled by the ground station, so operators can modify the mission schedule on the fly if necessary. Connectivity to the reference agent introduces a new vehicle classification depending on the relative location of the agents in the wireless communication network, as shown in Fig. 3. Link peers can communicate with the reference agent, whereas end peers have no direct communication with the reference. As depicted in Fig. 3, both link and end peers can communicate with other link and end peers.

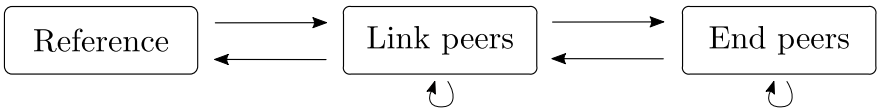

Fig. 3 Network structure for the cooperative motion control algorithm.

V. System architecture

A. Software Framework

The dynamic and distributed nature of the operations targeted by MACPOS requires a robust, yet flexible architecture to enable rapid prototyping, integration, and deployment of software across one or more agents. The Autonomous Entity Operations Network (AEON) [19] is a lightweight software framework for autonomous systems development built around the Data Distribution Service (DDS) communication standard as implemented by Real-Time Innovations' (RTI) Connext® DDS [20]. DDS is a publish/subscribe messaging protocol that enables modular, data-driven system architectures where consumers are dependent not on the implementation details of their paired producers but rather on the data types flowing between them. In contrast to other middleware software, DDS uses peer-to-peer dynamic endpoint discovery, which improves the robustness of the system through the lack of a centralized message broker, increases scalability, and enables flexible deployment across computational resources. AEON applications utilize the DDS middleware to build up autonomous systems as collections of standalone modules called nodes that perform narrowly-scoped functions such as interfacing with hardware, computing waypoint-following controls, or generating obstacle-avoiding trajectories. Figure 4 shows how nodes in the AEON framework can be layered to build up autonomous behavior with each layer generally only interfacing with the adjacent layers. Highly decoupled nodes can be easily swapped out with compatible upgrades or replacements as long as the DDS data interfaces are obeyed. For example, a navigation subsystem can be written as independent nodes that provide interfaces to specific pieces of hardware and publish data to a consumer node that implements sensor fusion algorithms. Because the sensor fusion node is agnostic to the hardware, various sensor interfaces can be swapped out as the system requirements evolve or different hardware becomes available.

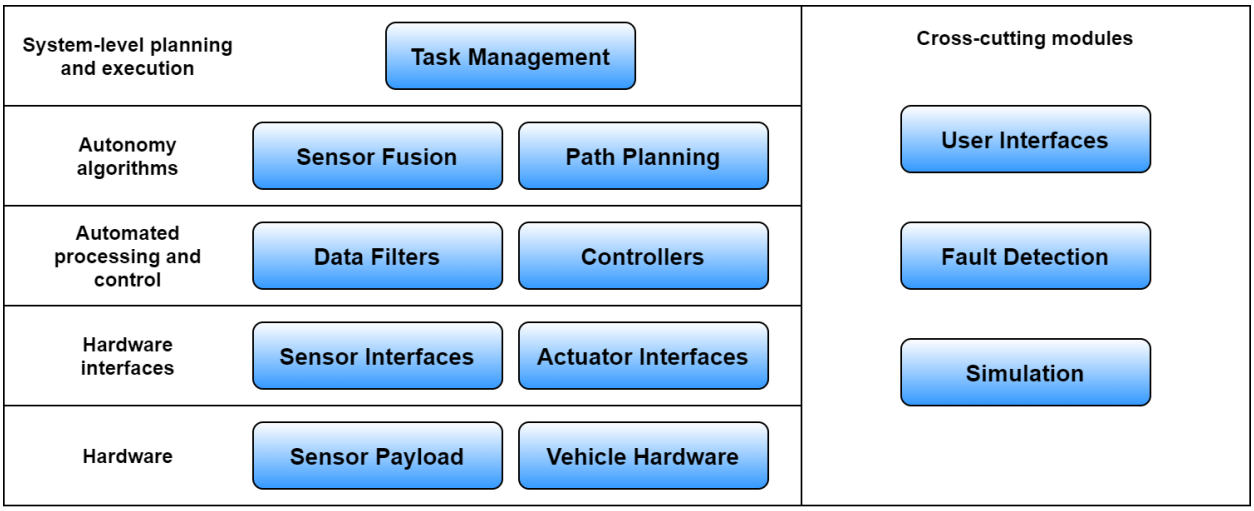

Fig. 4 Autonomy capabilities in the AEON framework built up as layers of modular software.

B. Single Agent Autonomy

The MACPOS system architecture can be decomposed in a hierarchical manner first with the software nodes comprising a single agent, and then with the deployment among agents and the ground station. As shown in Fig. 5, the software structure on individual agents looks similar to a typical AEON use case with layers of increasing autonomy from bottom to top. The figure highlights the major DDS nodes under development along with the communications among them as labeled arrows.

The mission manager node receives a mission description over the DDS network consisting of one or more sub-tasks to execute. Each sub-task is represented as a specification for a formation and a motion goal. The formation specification contains the list of agents in the formation, the relative positions of each agent, and the roles of each agent as either leader or follower. The motion goal may impose either locomotion constraints, pointing requirements to foveate a sensor payload at an observation target, or both. With the formation goal as input, the trajectory generator computes dynamically feasible, hazard-avoiding trajectories for each agent to achieve the desired formation and motion. Each agent's trajectory

generator node communicates over the network with other members of the formation to share information needed to arrive at a consensus. Both centralized and distributed generation schemes are being considered for comparison, so the shared information may either be intermediate data such as optimization costs or auction-based bids, or fully computed trajectories ready to execute. Trajectories are sent to the execution node that implements time-critical coordination. The trajectory execution node iteratively updates the coordination states that keep track of the progress of each agent along their respective trajectories, and computes coordination-aware controls to converge on and follow the target path. The controls feed into black box motor controllers or autopilot modules through vehicle hardware interface nodes.

At the bottom left of Fig. 5 are the collection of sensor interface nodes represented with one box, and MACPOS will take advantage of existing AEON interfaces to cameras, IMUs, and other sensors to provide data to the sensor fusion node. The specifics of the sensor fusion node will vary as the project progresses, but at a minimum the node must publish the pose of the agent and the locations of hazards including other agents.

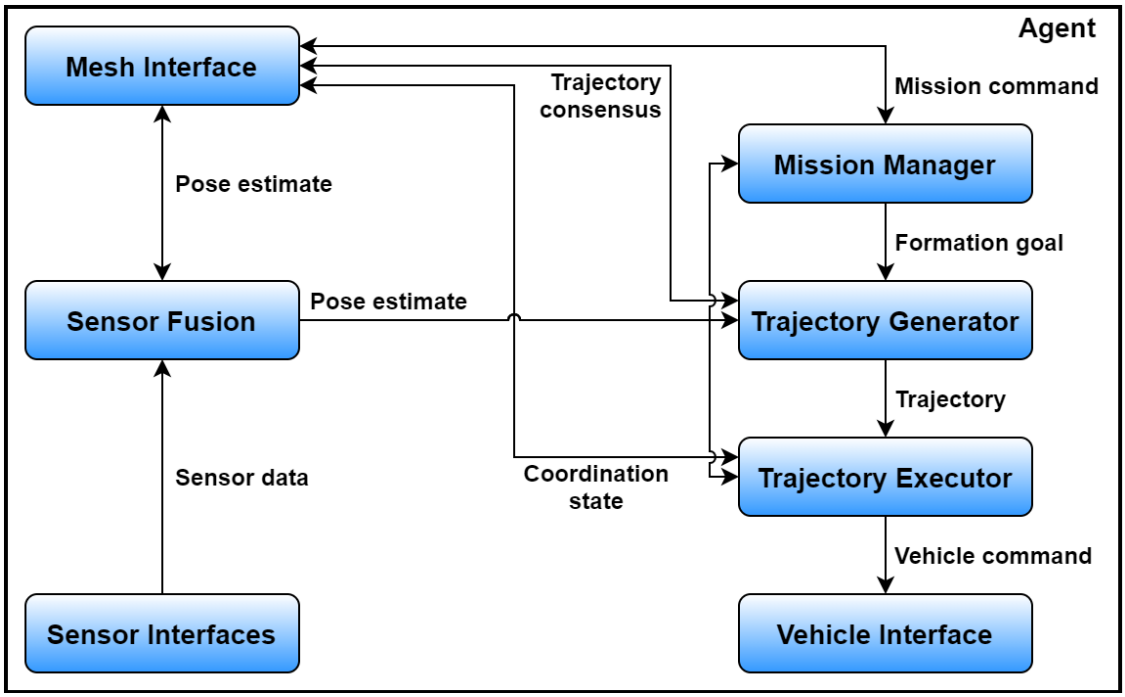

Fig. 5 Single agent MACPOS software diagram illustrating modules and the messages passed among them.

C. Multi-Agent System

The mesh interface at the top left of Fig. 5 is the most significant departure from previous AEON use cases [21]. Typical AEON nodes have no communication restrictions, and matched DDS publishers and subscribers can connect anywhere on the same network across threads, processes, or computers. The mesh interface node is responsible for emulating mesh network dynamics in a manner that is controllable but as transparent as possible to other nodes. A simple DDS quality of service (QoS) setting can be used to prevent the other nodes from directly connecting to another agent or the ground station. The mesh interface listens to messages intended for other agents and relays them according to a configurable pipeline of filters that specify inter-agent network connectivity over time. Other mesh interface nodes receive those messages and either forward them towards the intended target in the fleet or publish the messages to internal consumer nodes. The mesh interface node emulates not only the network dynamics but also the software architecture for a multi-agent system utilizing mesh radios. By concentrating the mesh-related code in one place, the mesh emulation interface can be easily replaced with a mesh hardware interface with the same inputs and outputs.

The role of the mesh interface node in controlling the connectivity across the multi-agent system is illustrated in Fig. 6. The underlying network hardware and QoS settings allow the collection of mesh interface nodes to talk to one another freely and reliably, ensuring that messages are only dropped or restricted due to the intentional mesh topology emulation or fault injection. The diagram also shows the graphical user interface (GUI) that runs on the GCS for commanding and

introspecting the fleet. The GUI allows a human operator to manually construct and command formations, visualize the mesh network topology, and monitor sensor data streams. While the diagram indicates that the command GUI also uses the mesh interface to route messages to and from the ground station, dynamic discovery in the DDS protocol allows the GUI to listen to additional network traffic for testing and debugging purposes. The modular architecture and the ability to dynamically plug into message channels facilitate the implementation of a sliding scale of autonomy. The nominal control modality for the GUI is a mid-level, manually-generated formation specification, but DDS hooks exist at multiple layers all the way down to direct teleoperation of individual vehicles. The extensible architecture allows for the eventual usage of high-level task objectives and automated planners that can construct the same formation messages as the GUI.

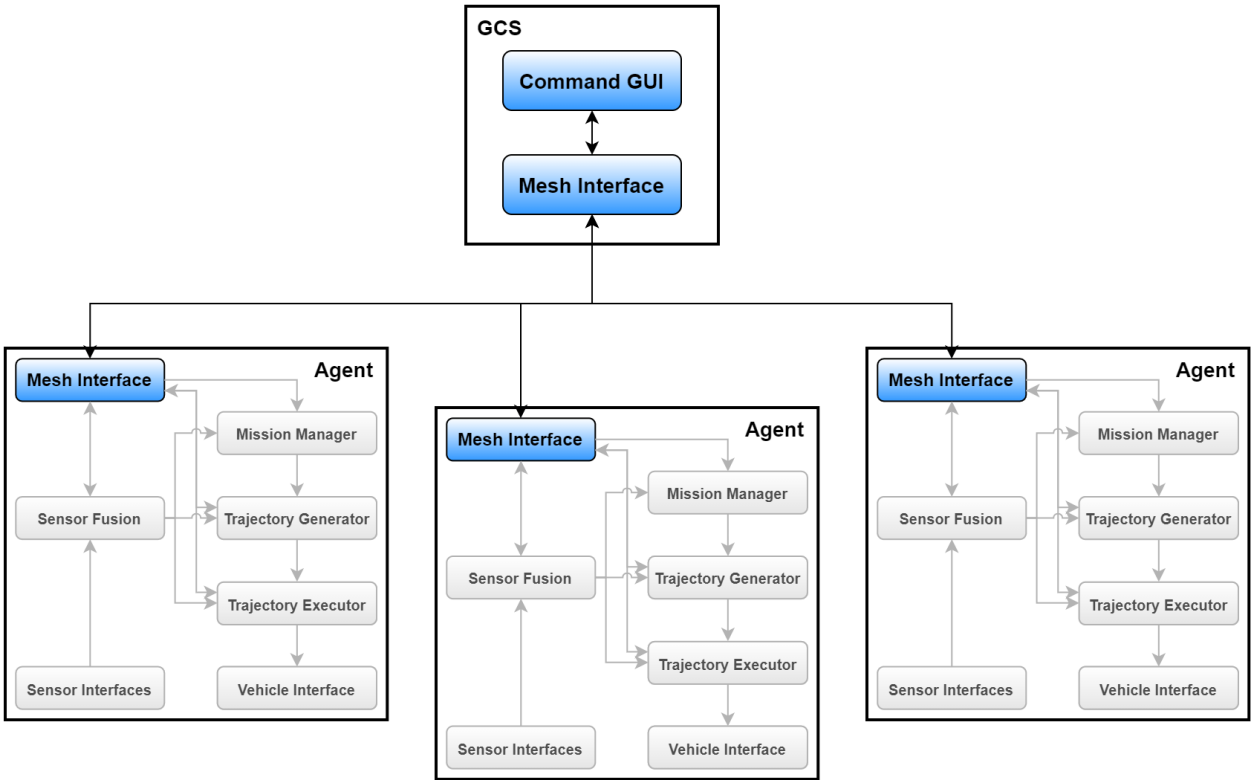

Fig. 6 Inter-agent communication filtered through mesh interface nodes.

VI. Design Reference Mission

Software development for MACPOS is proceeding along a roadmap of target demonstrations that highlight key functionality and build up to a design reference mission (DRM) that showcases the major elements of the project. Fig. 7 illustrates the notional sequence of events for the DRM with a fleet of quadrotors. First, the operator selects the agents that will comprise the formation, specifies the shape and leader, and sets an observation target. The agents generate trajectories to organize into the formation from their starting locations and proceed to the target. Next, the agents discover a new area of interest while at the original target, and a sub-group of vehicles is chosen to explore further. The two leader/follower clusters carry out their tasks and then regroup into the original formation, dynamically negotiating which agents are leaders and followers. Finally, when a fault renders one vehicle inoperable, the fleet must reconfigure to account for the loss and carry on with the mission.

A functional decomposition of the DRM results in intermediate demonstrations highlighting subsystem development and integration milestones. Subsystem demonstrations include using the command GUI to visualize mesh connectivity and construct formation specifications, generating a set of obstacle-avoiding trajectories to move in formation to a predetermined goal point, performing time-critical coordination to execute a precomputed set of trajectories, and detecting when a member of the fleet is no longer capable of coordination due to a fault. While Fig. 7 depicts the DRM using a set of quadrotors, project demonstrations at various stages of development will utilize rovers, unmanned aerial

vehicles (UAVs), and simulated ground, air, and space assets. The rovers, shown in Fig. 8 performing a preliminary proof of concept of leader/follower transitions, have been used in previous work as a low-risk platform for early testing on the path to flight as well as demonstrations of mixed reality and heterogeneous vehicle operations [22]. Integration milestones include running one or more subsystems on each of the aforementioned platforms and building up system complexity towards the DRM.

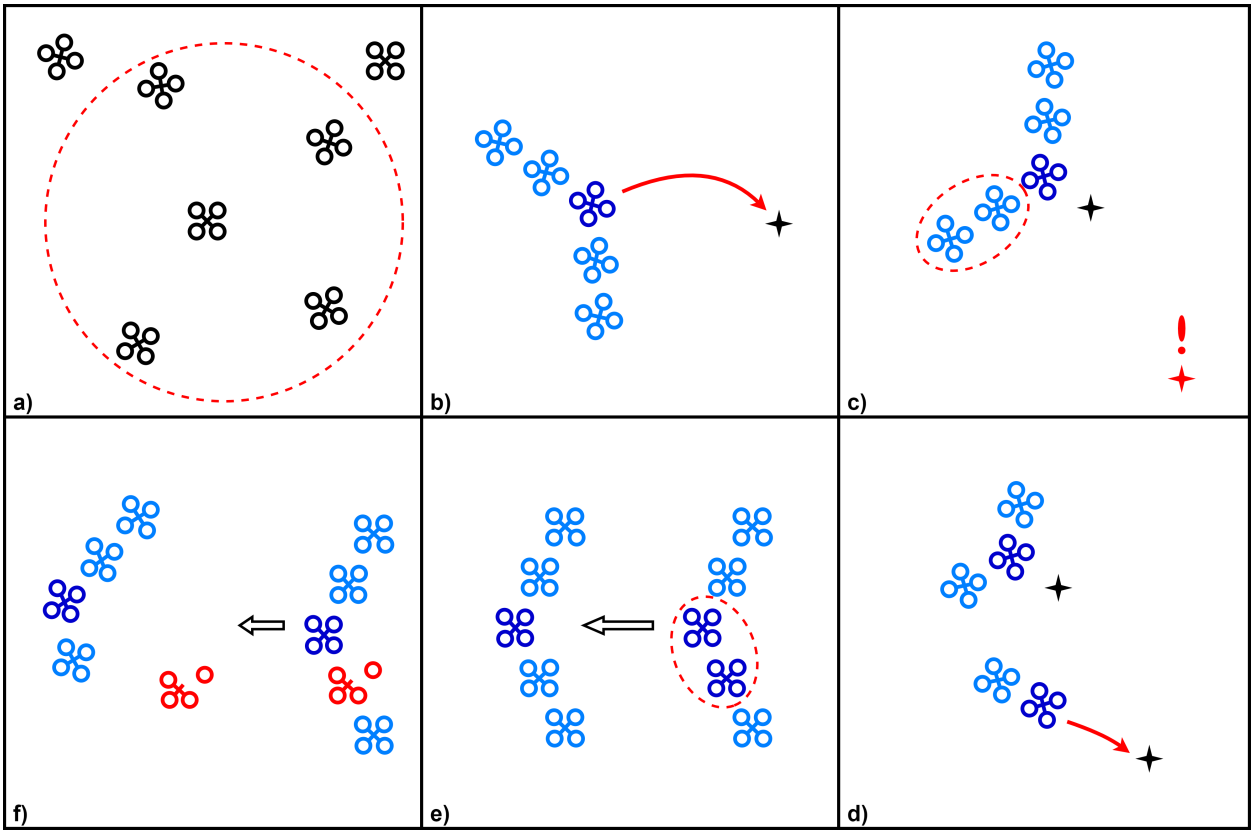

Fig. 7 Design reference mission. a) Formation selection from a group of unorganized agents. b) Formation movement to an area of interest. c) New area of interest and subgroup selection. d) Subgroup break-off to visit new area. e) Rejoining and renegotiation of leader role. f) Formation reconfiguration after a vehicle failure.

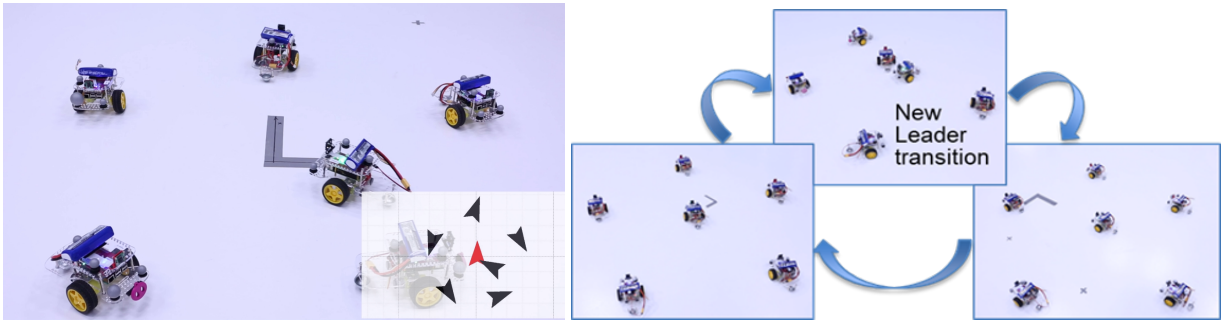

Fig. 8 Rover fleet executing a formation leader transition.

VII. Future Work

The first six months of the project were dedicated to the research and design work discussed in this paper. Subsystem development and hardware selection are currently underway with the individual component tests described above occurring through the end of the first year of project execution. Integration of the software pipeline, simulation environment, and hardware platforms will take place during the second project year. Integration tests will initially focus on real ground vehicles and simulated agents performing simple formation maneuvers and transitions. To highlight that the algorithms are fundamental capabilities rather than a point solution, intermediate demonstrations will include use cases not explicitly called out in the DRM such as dynamic obstacle avoidance and target following in formation. The minimum criterion for project success is performing the DRM using both the rovers shown in Fig. 8 and simulated air vehicles. Nevertheless, barring unforeseen circumstances, limited flight demonstrations using real UAVs are expected to take place towards the end of the project. Indoor testing of a single UAV will ensure that the low level control and trajectory following modules are stable and accurate. The previous simple formation exercises will then be repeated using a combined group of rovers and the UAV. Schedule and labor permitting, the stretch goal is to execute the DRM using multiple UAVs at the outdoor flight range at NASA LaRC.

Beyond the DRM, the next priority will be to integrate planning algorithms and decision aids into the system to allow the human operator to command a fleet with high level sensing objectives. These objectives may include maximizing area coverage, positioning a set of sensor payloads for simultaneous correlative data collection, or minimizing information gaps when tracking a target. These capabilities would be easily accessible to the command GUI through the modular DDS-backed architecture to provide an additional layer of autonomy and a more complete solution to managing the multi-agent system. Outside the immediate scope of the project, avenues of particular interest for complementary future work include incorporating more comprehensive simultaneous localization and mapping (SLAM) solutions, implementing additional sensor modalities for relative navigation in the fleet, and exploring distributed sensor fusion algorithms.

VIII. Conclusion

In this paper we have presented an overview of the ongoing MACPOS project along with early research and design progress. Implementation of the architecture described above is underway as we work towards subsystem demonstrations, integration milestones, and the execution of the DRM. Successful development of the algorithms and other software discussed in this paper will establish a foundation for robust, dynamic, and predictable multi-agent formation operations. The modular, extensible architecture opens a straightforward path to augment the system with higher level autonomy such as automated mission planning and complementary functionality such as image classification and anomaly detection for the dynamic discovery of areas of interest.

The capabilities being developed by this project have the potential to be transformative for missions that require persistent and distributed sensing. In particular, the usage of collaborative mobile robots to provide enhanced situational awareness and communication infrastructure in extreme environments may possibly become a key enabling technology in NASA's push for space and surface in-situ assembly.

Acknowledgments

We would like to acknowledge Meghan Chandarana for her role in leading the preliminary work and proposal development for the MACPOS project. We also thank the Analytical Mechanics Associates software team, especially Ralph Williams, Kyle McQuarry, and Thea Avila, for their engineering support. Finally, we thank the members of the Autonomous Integrated Systems Research Branch for program support as well as Erica Rodgers and the Office of Chief Technologist (now the Office of Technology, Policy, and Strategy) for the external funding opportunity.

References

[1] Allen, B. D., "Autonomy for Space and Surface In-Situ Assembly: NASA's Vision and Progress," *AIAA Scitech Forum and Exposition*, 2022.

[2] Reily, B., Reardon, C., and Zhang, H., "Leading Multi-Agent Teams to Multiple Goals While Maintaining Communication," *Robotics: Science and Systems*, 2020.

[3] Kim, H.-M., and Lim, Y., "A Communication Framework in Multiagent System for Islanded Microgrid," *International Journal of Distributed Sensor Networks*, 2012.

[4] Arabneydi, J., Baharloo, M., and Aghdam, A., "Optimal Distributed Control for Leader-Follower Networks: A Scalable Design," *IEEE Canadian Conf. on Elec. and Comp.*, 2018.

[5] Hasanyan, J., Zino, L., Alberto, D., Lombana, B., Rizzo, A., and Porfiri, M., "Leader-follower consensus on activity-driven networks," *The Royal Society*, 2019.

[6] Aijaz, A., "Infrastructure-less Wireless Connectivity for Mobile Robotic Systems in Logistics: Why Bluetooth Mesh Networking is Important?" *Wireless Communication in Internet of Things*, 2020.

[7] Ghabcheloo, R., Pascoal, A., Silvestre, C., and Kaminer, I., "Non-linear co-ordinated path following control of multiple wheeled robots with bidirectional communication constraints," *International Journal of Adaptive Control and Signal Processing*, Vol. 21, No. 2-3, 2007, pp. 133–157. https://doi.org/https://doi.org/10.1002/acs.923, URL https://onlinelibrary.wiley.com/doi/abs/10.1002/acs.923.

[8] Stepanyan, V., Krishnakumar, K. S., and Bencomo, A., "Identification and reconfigurable Control of Impaired Multi-Rotor Drones," *AIAA Guidance, Navigation and Control Conference*, San Diego, CA, USA, 2016, pp. 1–20. AIAA 2016-1384.

[9] Vijayan, R., Bilal, M., and Schilling, K., "Nonlinear dynamic modeling of satellite relative motion with differential <tex>$\boldsymbol{J}_2$</tex> and drag," *2020 IEEE Aerospace Conference*, 2020, pp. 1–8. https://doi.org/10.1109/AERO47225.2020.9172627.

[10] Ackerman, K. A., Gregory, I. M., and Hovakimyan, N., "Flight Control Methods for Multirotor UAS," *International Conference on Unmanned Aircraft Systems (ICUAS)*, Washington, DC, USA, 2019, pp. 353–361.

[11] Stepanyan, V., and Krishnakumar, K. S., "Estimation, Navigation and Control of Multi-Rotor Drones in an Urban Wind Field," *AIAA Information Systems*, Grapevine, TX, USA, 2017, pp. 1–26. AIAA 2017-0670.

[12] Goyal, V., and Ierapetritou, M. G., "Determination of operability limits using simplicial approximation," *AIChE Journal*, Vol. 48, No. 12, 2002, pp. 2902–2909. https://doi.org/https://doi.org/10.1002/aic.690481217, URL https://aiche.onlinelibrary.wiley.com/doi/abs/10.1002/aic.690481217.

[13] Lakshmanan, A., Patterson, A., Cichella, V., and Hovakimyan, N., "Proximity Queries for Absolutely Continuous Parametric Curves," *Proceedings of Robotics: Science and Systems*, Freiburgim-Breisgau, Germany, 2019.

[14] Puig-Navarro, J., Hovakimyan, N., Alexandrov, N., and Allen, B. D., "Silhouette-Informed Trajectory Generation through a Wire Maze for UAS," *AIAA Aviation Technology, Integration, and Operations Conference*, Atlanta, GA, USA, 2018.

[15] Puig-Navarro, J., "Distributed Time-Critical Coordination Strategies for Unmanned Aerial Systems in Cluttered Environments," Ph.D. thesis, University of Illinois Urbana-Champaign, Urbana, IL, USA, 2021.

[16] Yu, C., and Gao, S., "Reducing Collision Checking for Sampling-Based Motion Planning Using Graph Neural Networks," *Conference on Neural Information Processing Systems*, Sydney, Australia, 2021.

[17] Choe, R., "Distributed Cooperative Trajectory Generation for Multiple Autonomous Vehicles Using Pythagorean Hodograph Bézier Curves," Ph.D. thesis, University of Illinois at Urbana-Champaign, Urbana, IL, USA, 2017.

[18] Puig-Navarro, J., Xargay, E., Choe, R., and Hovakimyan, N., "Time-Critical Coordination of Multiple UAVs with Absolute Temporal Constraints," *AIAA Guidance, Navigation and Control Conference*, Kissimmee, FL, USA, 2015. AIAA 2015-0595.

[19] Kelley, B. N., and Vaughan, M., "A Distributed Simulation-to-Flight Framework to Support Investigating Trust/Trustworthiness in Multi-Agent Systems," *AIAA Scitech 2021 Forum*, 2021, p. 1772.

[20] Pardo-Castellote, G., "Omg data-distribution service: Architectural overview," *23rd International Conference on Distributed Computing Systems Workshops, 2003. Proceedings.*, IEEE, 2003, pp. 200–206.

[21] Allen, B. D., Tran, L., Neilan, J. H., Trujillo, A., Kelley, B., McQuarry, A. K., Vaughan, M., Williams, R., and Crisp, V., *An Autonomous Unmanned Science Mission*, ???? https://doi.org/10.2514/6.2017-3988, URL https://arc.aiaa.org/doi/abs/10.2514/6.2017-3988.

[22] Vaughan, M., and Kelley, B. N., "Bootstrapping Multi-Agent Unmanned Aerial Vehicle (UAV) System Integration Using Ground-Based Assets: Lessons Learned," *AIAA Scitech 2021 Forum*, 2021, p. 1884.