Collaborative LLM-Based Agents for Autonomous Multi-UAV Mission Execution

Jeremy Ludwig, Daniel Tuohy, and Forest Finnigan
Stottler Henke Associates
San Mateo, CA 94402

Abstract—**This paper investigates the use of large language model (LLM)–driven agents to coordinate multi-UAV missions, with emphasis on reliability, adaptability, and human–agent teaming. A proof-of-concept system, the Collaborative LLM-based Agents (COLA) framework, was developed to illustrate how modular language-based agents can be integrated with existing autopilot software and human supervision. The architecture uses an LLM server that uses structured prompts to convert natural language commands into JSON-formatted API calls, enabling intuitive operator control while maintaining precise execution of UAV functions. Validation and verification combined simulation, live flight trials, and automated prompt testing.**

Results showed that agents could coordinate search tasks, manage control handoffs, monitor telemetry, and reallocate routes dynamically, including in response to unplanned operator commands. Simulation enabled exploration of advanced behaviors, while live flights demonstrated feasibility under real-world constraints and revealed additional challenges not observed in simulation. The study contributes an illustrative architecture and initial insights into validation of LLM-based autonomy, establishing a foundation for scalable, safe, and reliable application of LLMs to aerospace autonomy.

TABLE OF CONTENTS

1. INTRODUCTION

Autonomous systems are playing an increasingly important role in aerospace missions, extending operational reach, reducing human workload, and enabling complex tasks in contested and dynamic environments. Unmanned aerial vehicles (UAVs) are increasingly employed across diverse missions, including surveillance, environmental monitoring, disaster response, and infrastructure inspection, in addition to various defense and security applications. Real-world developments, such as the use of AI-enabled drone swarms in Ukraine, highlight the importance of scalable and resilient autonomy capable of functioning with limited human oversight. [1]. Yet achieving collaborative autonomy that is both flexible and dependable remains challenging: current systems often rely on rigid behaviors, struggle in uncertain conditions, and offer limited means for intuitive human interaction. Recent advances in large language models (LLMs) offer new possibilities for addressing these limitations. By enabling natural language interaction and context-sensitive reasoning, LLMs may support more adaptive collaboration among UAV teams and between humans and autonomous agents. Realizing this potential requires scalable methods for creating, validating, and deploying LLM-based agents that can operate across diverse mission scenarios while maintaining safety, reliability, and operator trust.

This paper explores such methodologies through a proof-of-concept study of collaborative LLM-based agents for multi-UAV operations. The work examines how modular agent designs, lightweight validation approaches, and operator-focused interfaces can support collaborative autonomy in both simulated and live environments. While limited in scope, the study provides several contributions of interest to the community. First, it presents an illustrative architecture demonstrating how LLM-driven agents can be composed to coordinate multi-UAV tasks. Second, it reports initial observations on the challenges of validating LLM-based autonomy. Third, it offers practical lessons on human-agent interaction, identifying where natural language control proved effective and where additional operator support mechanisms are needed.

The remainder of this paper is organized as follows. Section 2 provides a brief overview of related work in LLM-enabled autonomy, UAV coordination, and human-agent teaming. Section 3 describes the system architecture and experimental setup. Section 4 presents findings from simulation and live demonstrations along with implications and lessons learned, and Section 5 concludes with directions for future work.

2. RELATED WORK

Research on multi-agent UAV coordination has produced foundational results in decentralized planning and cooperative surveillance using UAV swarms, e.g. [2]. These early studies established the principles of distributed control that continue to inform modern swarm autonomy. Academic survey efforts describe progress in communication and control strategies [3] and highlight ongoing challenges of scalability, resilience, and security in UAV swarms [4]. Complementary to these foundations, modular, field-tested swarm architectures have been demonstrated in mission-oriented designs [5].

In parallel, autonomy frameworks are rapidly evolving, shaped by both industry and government initiatives. Commercial and defense organizations, ranging from small business (e.g., [6], [7]) to major defense contractors, are advancing autonomy platforms for military, public safety, and aerospace applications. Government-led efforts, including AFRL's Autonomy Government Reference Architecture (A-GRA) and industry coordination forums such as Autonomy Industry Days, reflect the Department of War's role in guiding and standardizing emerging autonomy architectures. NASA also conducts ongoing research in UAV autonomy (e.g. [8]).

Large language models have begun to influence UAV autonomy and human–agent teaming. For example, REAL integrates LLMs into control loops to recover from disruptions in UAV operations [9], while Talk2Drive demonstrates natural language as an effective control modality for autonomous vehicles [10]. Academic systems like Aero-LLM illustrate how language-enabled coordination can support dynamic task allocation in UAV teams [11]. Building on these targeted efforts, [12] provides a comprehensive survey of LLM architectures for UAV integration, emphasizing opportunities in communication, sensing, and decision-making. Complementary perspectives are offered in [13], which positions agentic LLM frameworks as a pathway toward adaptable low-altitude operations in real-world environments. Together, these works reinforce the view that LLM-driven can provide a foundation for UAV collaboration.

Trust, transparency, and intuitive interfaces remain central to effective human–agent teaming. Chen et al. [14] examined supervisory control of unmanned vehicles, identifying key human-performance issues and user-interface design principles that remain highly relevant for modern UAV autonomy. Their work emphasizes how interface design directly impacts operator workload, situational awareness, and trust in autonomy. More recently, [15] examined explanation mechanisms for human-on-the-loop oversight of UAV swarms, emphasizing the trade-off between sufficient detail to maintain situational awareness and avoiding operator overload. Together, these works underscore that operator-centered interaction design is essential for reliable adoption of collaborative autonomy. Importantly, natural language itself is emerging not only as a control modality but also as a key factor in shaping operator trust and usability. By enabling transparent communication of system intent, reasoning, and status, LLMs could help bridge traditional gaps in human–autonomy teaming.

While the current literature spans UAV swarm coordination, resilience, and LLM-based autonomy, there is a notable gap in scalable methods for rapidly developing safe, reliable, and operator-aligned LLM-based UAV agents. This paper advances work in this area, focusing on modular agent design, lightweight validation approaches, and operator-centric interaction as enabling strategies for scalable, collaborative autonomy in UAV operations.

3. METHODS

The Collaborative LLM-based Agents (COLA) system was developed as a proof-of-concept platform to explore the use of LLMs for coordinating multi-UAV missions. The system integrates a modular agent architecture, an operator interface, and a flight control stack to support coordination among multiple drones and interaction with a human supervisor. At its core, COLA enables communication between a Mission Supervisor agent, UAV agents, and a human operator using both natural language and structured messages. The methods described in this section outline the system architecture, prompt development process, operator interface, and validation and verification (V&V) strategy, which together establish how LLM-based agents and existing flight software were composed into a functioning collaborative autonomy system.

Architecture

The COLA architecture consists of three cooperating services:

- LLM Server (NodeJS), which handles prompt evaluation and decouples agents from specific hosted models. COLA relied on OpenAI's GPT-4o model, released in May 2024.
- Client (Angular), which provides visualization, operator chat, and sharing information across UAVs. The client also performs synthesis of prompts to send to the LLM Server for evaluation in response to commands and messages, execution of Application Programming Interface (API) calls as directed by the LLM, and synchronization of the mission with live data coming from the Mission Control Server.
- Mission Control Server (Python), which manages all vehicle interactions via a Representational State Transfer (REST) API. This service is implemented to be agnostic to the flight stack; in this work it used PyMAVLink to control UAVs through MAVProxy.

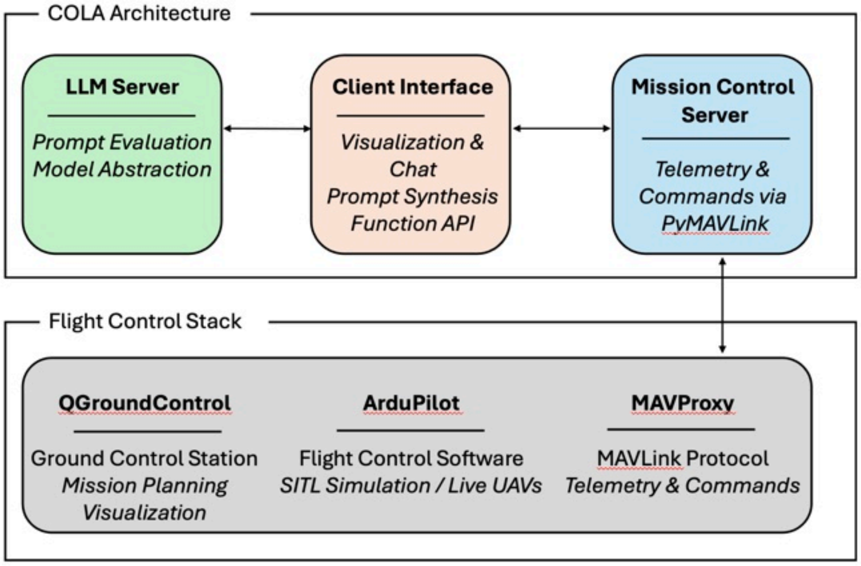


Figure 1. System overview.

Mission plans were created in QGroundControl, uploaded to UAVs before flight, and parsed by the Mission Supervisor agent for allocation and re-tasking. ArduPilot provided the autopilot in both Software-in-the-Loop (SITL) simulation and live quad-plane UAVs, with MAVProxy routing telemetry and commands. The COLA architecture and UAV control stack is shown in Figure 1.

Within this infrastructure, a Mission Supervisor agent coordinated the mission, and three UAV agents executed assigned waypoints, monitored status, and initiated return-to-launch (RTL) as necessary. At the core of the system's agent design is the use of structured prompts that instruct LLMs to act as natural language transpilers, converting operator commands into JSON-formatted function calls. Each agent prompt includes a detailed API specification, mapping rules, and input/output examples. When an operator issues a command like 'yellow come home, reassign 25-28 to blue,' the Mission Supervisor agent processes this through the LLM, which returns a structured JSON response containing the appropriate function calls (e.g., returnToLaunch and reallocateWaypoints) with correct parameters

The architecture is not intended as a final design but as an illustration of how language-based agents, a human operator, and existing flight software can be composed into a functioning multi-UAV system.

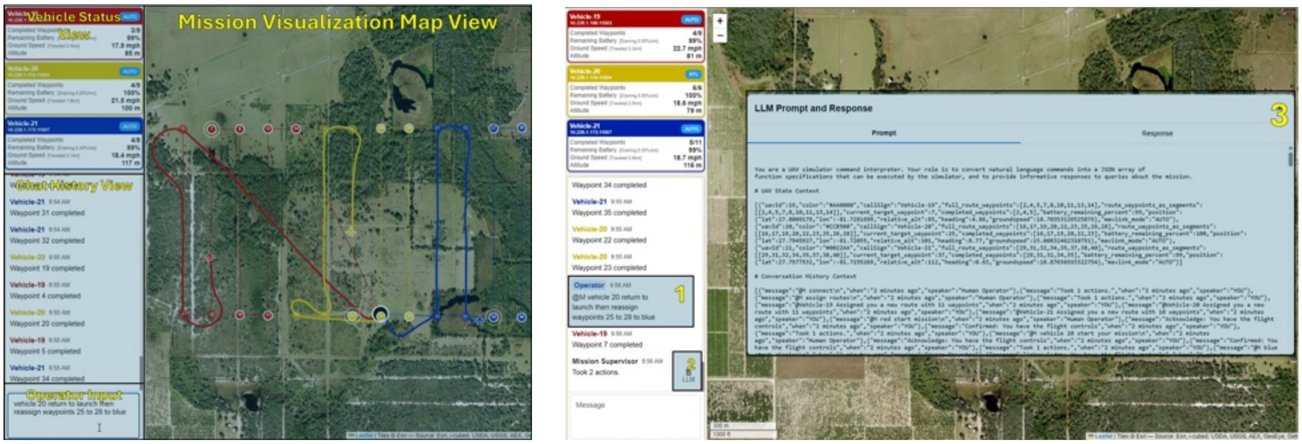
Prompt Development

Agent behaviors for the Mission Supervisor and UAVs were defined through a five-stage workflow designed to translate high-level responsibilities into prompts that combined natural language and general reasoning capabilities with precise function execution. This workflow enabled efficient creation of new agents from clearly defined responsibilities and function libraries.

First, responsibilities were described in natural language, capturing the tactics, techniques, and procedures (TTPs) expected of each agent. Second, a library of functions was developed to carry out specific tasks, either by implementing new code or integrating existing libraries. Third, these functions were consolidated into a unified software interface description, giving all agents a consistent view of callable APIs while allowing flexibility in underlying implementations. Fourth, input/output protocols were specified in natural language to define how agents communicate with each other and with the operator. Finally, prompts were generated and refined using a template that incorporated the preceding elements.

Each prompt template included sections for UAV state context, conversation history, mission background, available functions, governing rules, and representative examples. The prompt template instructs the LLM to return only JSON responses in a specific format, either containing a 'commands' array with function specifications, an 'error' with clarification requests, or a 'response' for conversational queries. This structured output format ensures reliable parsing and execution of commands while maintaining the flexibility of natural language input. At runtime, templates were dynamically filled with the current mission state and dialogue, ensuring up-to-date context. Prompts were iteratively refined during testing to resolve ambiguities and expand behaviors as needed.

Three distinct prompts were implemented. UAV Agent prompts supported monitoring, reporting, and command execution, including a positive exchange-of-control protocol. Mission Supervisor prompts for operator interaction translated natural language commands into functions such as route generation, mission initiation, and waypoint reallocation. Mission Supervisor prompts for UAV interaction handled UAV-to-supervisor communication, including waypoint reallocation when a vehicle reported low battery or aborted its mission.



vehicle 20 return to launch then reassign waypoints 25 to 28 to blue

Vehicle-21 9:54 AM
Waypoint 31 completed

Vehicle-21 9:54 AM
Waypoint 32 completed

Vehicle-20 9:55 AM
Waypoint 19 completed

Vehicle-19 9:55 AM
Waypoint 4 completed

Vehicle-20 9:55 AM
Waypoint 20 completed

Vehicle-19 9:55 AM
Waypoint 5 completed

Vehicle-21 9:55 AM
Waypoint 34 completed

Vehicle-21 9:55 AM
Waypoint 35 completed

Vehicle-20 9:55 AM
Waypoint 22 completed

Vehicle-20 9:55 AM
Waypoint 23 completed

Operator 9:56 AM
@M vehicle 20 return to launch then reassign waypoints 25 to 28 to blue

Vehicle-19 9:56 AM
Waypoint 7 completed

Mission Supervisor 9:56 AM
Took 2 actions.

```
You are a UAV simulator command interpreter. Your role is to convert natural language commands into a JSON array of
function specifications that can be executed by the simulator, and to provide informative responses to queries about the mission.

# UAV State Context

# Conversation History Context
```

Figure 2. The COLA Client UI with an overview of components (left) and drill down into LLM prompts and responses (right).

Operator Interface

A custom web-based interface was developed to better enable operator oversight of multiple UAVs simultaneously.

The interface comprised four components as shown in Figure 2:

- Vehicle Status View, displaying telemetry and mission progress for each UAV.
- Chat History View, logging communications among operator and agents, with drill-down access to the exact prompts and model outputs used in decision-making (See 1, 2, 3).
- Operator Input, a text entry field for issuing commands, with the Mission Supervisor as default addressee.
- Mission Visualization Map, showing planned routes, UAV locations, current waypoints, and executed paths.

This design provided traceability into LLM reasoning and supported investigation of human–agent interaction. In particular, it allowed study of when natural language control was effective (e.g., for handoffs and queries) and when additional mechanisms such as confirmations or visual cues were required to reduce operator workload.

Validation and Verification

Validation and verification activities were intended both to confirm that the COLA system could perform multi-UAV missions end-to-end and to identify potential limitations and areas for refinement in applying LLM-based autonomy. The evaluation sought to both demonstrate functionality and provide feedback on system behavior under varied operating conditions, supporting analysis of reliability and human–agent interaction.

System evaluation combined simulation, flight tests, and LLM-specific validation. In SITL simulations, the Mission Supervisor allocated a lawnmower-pattern search across three UAVs and adapted when faults were introduced, such as commanding one UAV to return early. In live flight exercises, the integrated system was connected to quad-plane UAVs (Figure 3). Ground validation first confirmed connectivity and basic command execution. Subsequent missions included waypoint assignment, multi-UAV coordination, and return-to-launch.

In parallel, LLM-specific validation was conducted using LangChain notebooks to replay prompts and queries extracted from system runs. This approach allowed targeted evaluation of decision-making and reasoning without requiring live vehicles. A secondary LLM was employed to automate evaluation of correctness and to generate additional test cases, creating a scalable pathway for testing agent logic beyond manual inspection.

Figure 3. Preparing UAVs for takeoff.

4. RESULTS AND DISCUSSION

This section presents results from evaluation of the COLA proof-of-concept system in both simulation and live flight, followed by discussion of their implications. The common scenario was a collaborative search task involving a Mission Supervisor agent and three drone agents (Red, Yellow, and Blue). The Mission Supervisor read the mission plan file, assigned waypoints, coordinated handoffs of control, monitored progress, and re-tasked drones as necessary. Drone agents accepted commands, verified their feasibility, executed assigned waypoints, monitored battery levels, and initiated RTL when appropriate. The human operator remained on-the-loop, supervising mission execution and directing high-level actions through the chat interface. Natural language instructions were issued to the Mission Supervisor, which then coordinated responses with the UAV agents, while the operator UI provided real-time visualization of drone positions, planned and completed waypoints, and vehicle status. In simulation, drones performed autonomous takeoffs, while in live flights human pilots launched each vehicle and transferred control at altitude.

Simulation Results and Discussion

The simulation environment provided a flexible testbed that integrated ArduPilot, MAVProxy, PyMAVLink, and QGroundControl with the COLA system. It proved invaluable for evaluating the LLM-based agent approach, showing both the relative ease of creating and adapting agents for collaborative search tasks and the practical challenges of coordinating diverse software components.

Beyond verifying core functionality, simulations enabled testing of advanced capabilities that could not be explored within the limited duration of live flights. These included automated generation of search patterns, allocation of waypoints among multiple UAVs, and adaptation to simulated failures such as early battery depletion. Additional experiments examined heterogeneous UAV teams with different performance characteristics and negotiation protocols that allowed UAVs to report their ability to complete assigned tasks, prompting dynamic adjustments by the Mission Supervisor. In these scenarios, COLA agents frequently solved novel problems not explicitly covered in their prompts by making effective use of the available functions, suggesting flexibility beyond scripted behaviors.

Live Flight Results and Discussion

Live flight tests were conducted as part of a collaborative autonomy evaluation event and were overseen and independently verified by a third-party evaluation team. Three VTOL quad-plane UAVs participated, each operated with a designated safety pilot. Operations were staged from a local command center with wireless networking support. All flights were observed in real time by the evaluation team and recorded using onboard telemetry and command logs.

A flight was classified as a successful end-to-end autonomous execution if the following conditions were met: (1) control was transferred from the safety pilot to COLA on command (Figure 4), (2) the Mission Supervisor correctly created and distributed a mission plan among the participating UAVs, (3) each UAV autonomously executed its assigned waypoints (waypoint capture defined by ArduPilot default values), and (4) RTL was autonomously initiated without pilot intervention. Four test flights were conducted, using three UAVs and corresponding safety pilots each time.

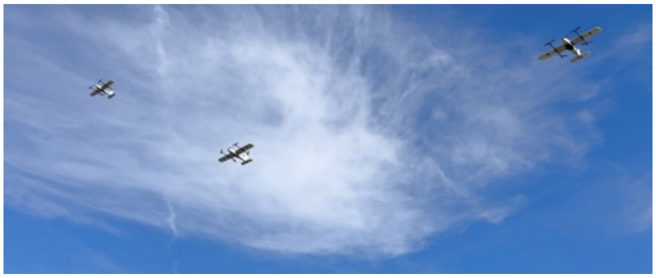


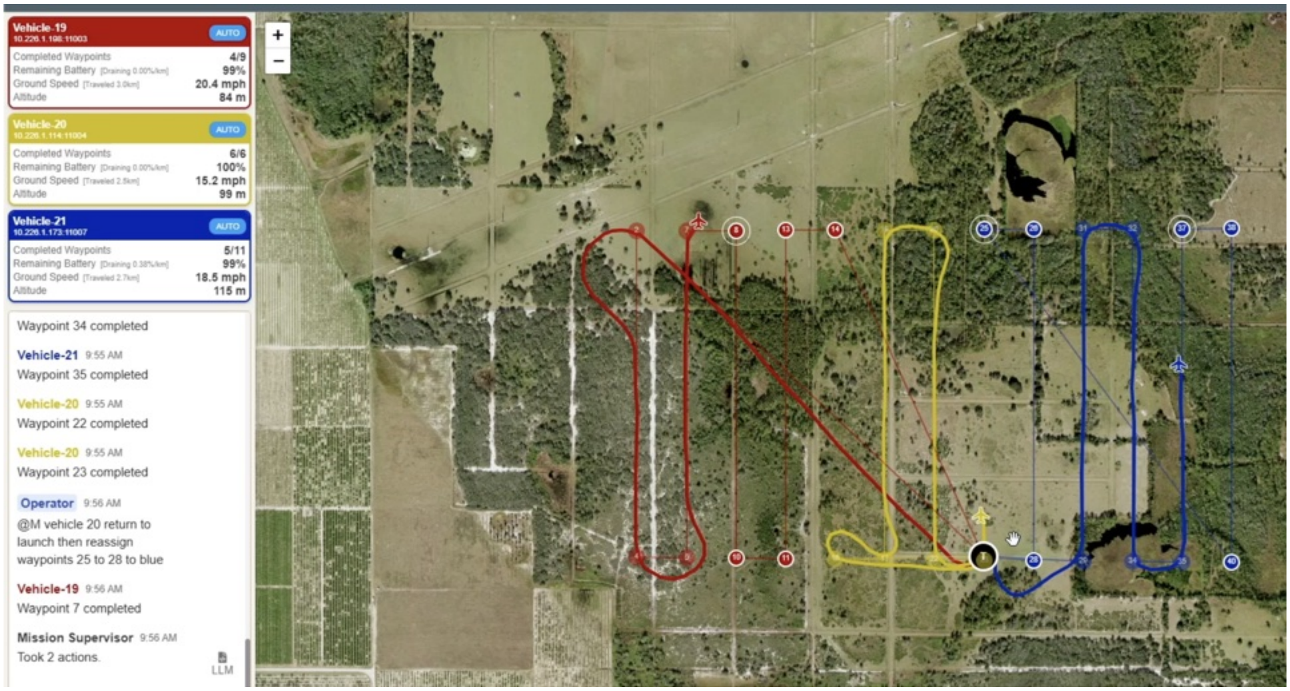
Figure 4. Hovering during the handoff from pilot to agent control.

Flights 1 and 4 demonstrated successful end-to-end operation with a single RTL exception in Flight 1. The Mission Supervisor coordinated handoff from the human pilots, divided the mission plan among the drones, and executed the positive exchange-of-control sequence. UAVs flew their assigned routes and initiated RTL as expected with one exception. Windy conditions caused missed waypoints occasionally as measured by the underlying flight control software. This did not affect flying the assigned waypoints in order, but it did lead to one vehicle missing its final waypoint and therefore failing to trigger RTL automatically. This vehicle required a manual override, though the remaining vehicles completed autonomous RTL as expected.

Flight 2 revealed an inconsistency in the handoff process: the Mission Supervisor declined to assume control after takeoff, reporting that safety conditions were unmet. Post-flight analysis suggested a possible altitude reporting discrepancy, and code adjustments were performed to attempt to resolve the issue. The same handoff failure persisted in Flight 3.

Flight 4, following safety-related refinements to the prompt, proceeded as intended. During this flight, the operator issued an unplanned instruction requesting one UAV to return early and reassign its waypoints to another vehicle; the agents successfully carried out the request, illustrating adaptability similar to that observed in simulation (Figure 5).

These live flight results illustrate both the potential and the limitations of LLM-based autonomy. On one hand, the architecture allowed emergent handling of unplanned scenarios using existing prompts and APIs. On the other, issues encountered with safety checks, telemetry accuracy, and waypoint completion underscore the importance of layered validation, deterministic safety logic, and robustness to real-world environmental factors.



Vehicle-19
10.226.1.198:11003
Completed Waypoints 4/9
Remaining Battery [Draining 0.00%/km] 99%
Ground Speed [Traveled 3.0km] 20.4 mph
Altitude 84 m

Vehicle-20
10.226.1.114:11004
Completed Waypoints 6/6
Remaining Battery [Draining 0.00%/km] 100%
Ground Speed [Traveled 2.5km] 15.2 mph
Altitude 99 m

Vehicle-21
10.226.1.173:11007
Completed Waypoints 5/11
Remaining Battery [Draining 0.38%/km] 99%
Ground Speed [Traveled 2.7km] 18.5 mph
Altitude 115 m

Waypoint 34 completed

Vehicle-21 9:55 AM
Waypoint 35 completed

Vehicle-20 9:55 AM
Waypoint 22 completed

Vehicle-20 9:55 AM
Waypoint 23 completed

Operator 9:56 AM
@M vehicle 20 return to launch then reassign waypoints 25 to 28 to blue

Vehicle-19 9:56 AM
Waypoint 7 completed

Mission Supervisor 9:56 AM
Took 2 actions.

Figure 5. Operator interacting with agents during live flight exercise.

Lessons Learned

Several lessons emerged across simulation and flight testing that inform future design:

Architectural implications. The modular composition of LLM-based agents with existing autopilot infrastructure proved workable and adaptable. However, integration across ArduPilot, MAVProxy, and UAV-specific telemetry exposed deep dependencies within the technology stack not evident in simulation. For example, there are undocumented inconsistencies in how different drone manufacturers implement the ArduPilot interface. Additionally, real-world sensor data (GPS, battery sensors, etc.) are often noisy and can be subject to errors or limitations. This underscores the need for architecture that tolerates heterogeneous hardware and inconsistent sensor data.

Validation challenges. Even with extensive simulation and notebook-based testing, live flights revealed unanticipated behaviors. Prompt sensitivity, telemetry inconsistencies, and environmental effects all contributed to unexpected outcomes. These observations reinforce the importance of V&V approaches that combine scalable automated testing with repeated field trials to capture conditions that cannot be simulated fully.

Human–agent interaction. The operator interface provided useful visualization and communication tools, but operator responses under real-time pressure differed from simulation. For example, when the Mission Supervisor declined handoff, the operator attempted repeated commands, revealing the need for clearer confirmation mechanisms and fallback control paths. The ability to inspect underlying prompts and responses was valuable for transparency, but simplifying operator workload in stressful conditions remains an open challenge.

5. CONCLUSION AND FUTURE WORK

This paper presented the COLA system, a proof-of-concept demonstration of collaborative autonomy using LLM-driven agents for multi-UAV mission execution. The system integrated a modular architecture, structured prompt development, a specialized operator interface, and a layered validation and verification process. Evaluation in both simulation and live flight showed that LLM-based agents can coordinate UAV teams, adapt to changing mission conditions, and accept natural language input from a human operator. At the same time, the results highlighted limitations in reliability, integration, and operator interaction that must be addressed before operational deployment.

Three main contributions were identified. First, the work illustrated how multiple LLM-based agents can be composed with existing autopilot infrastructure to coordinate multi-UAV missions. Second, observations from a limited number of live flights emphasized the challenges of validating LLM-based autonomy, particularly through the emergence of multiple distinct behaviors not encountered during prior simulation testing, including those arising from model variability, telemetry inconsistencies, and differences between simulation and field conditions. The challenges arose despite extensive simulation-based testing of similar scenarios. Third, practical lessons on human–agent interaction were identified, including the utility of natural language for task handoffs and the importance of fallback mechanisms and transparent interfaces to maintain operator trust.

Future research will focus on scaling the agentic workflow and developing a fully automated validation and verification framework to ensure the creation of safe, secure, and reliable agent teams. These advances will be coupled with more complex mission demonstrations, emphasizing generalization and natural language control, across both defense and commercial applications. The long-term objective is a robust system capable of supporting a broad range of unmanned vehicle operations and enabling transition from research prototypes to practical, operational systems.

In summary, the COLA system demonstrates the potential of LLM-driven agents for collaborative autonomy, while also clarifying the steps required to move from early demonstrations toward reliable aerospace applications.

ACKNOWLEDGEMENTS

This material is based in part upon work supported by the United States Air Force under Contract No. FA8730-24-P-B027. The views, opinions, and/or findings contained in this article/presentation are those of the authors/presenters and should not be interpreted as representing the official views or policies, either expressed or implied, of the United States Air Force.

We gratefully acknowledge Overhead Intelligence (OI), a geospatial mapping company that employs UAS platforms for large-scale surveying and mineral exploration, for planning and executing the live flight exercise.

REFERENCES

[1] A. M. | P. by J. M. for WSJ, “Exclusive | AI-Powered Drone Swarms Have Now Entered the Battlefield,” The Wall Street Journal. Accessed: Sep. 05, 2025. [Online]. Available: https://www.wsj.com/world/ai-powered-drone-swarms-have-now-entered-the-battlefield-2cab0f05

[2] R. W. Beard, T. W. McLain, D. B. Nelson, D. Kingston, and D. Johanson, “Decentralized Cooperative Aerial Surveillance Using Fixed-Wing Miniature UAVs,” *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1306–1324, Jul. 2006, doi: 10.1109/JPROC.2006.876930.

[3] S. Javaid *et al.*, “Communication and Control in Collaborative UAVs: Recent Advances and Future Trends,” *IEEE Trans. Intell. Transport. Syst.*, vol. 24, no. 6, pp. 5719–5739, Jun. 2023, doi: 10.1109/TITS.2023.3248841.

[4] Y. Alqudsi and M. Makaraci, “UAV swarms: research, challenges, and future directions,” *J. Eng. Appl. Sci.*, vol. 72, no. 1, p. 12, Jan. 2025, doi: 10.1186/s44147-025-00582-3.

[5] Z. Liu *et al.*, “Mission Oriented Miniature Fixed-wing UAV Swarms: A Multi-layered and Distributed Architecture,” *IEEE Trans. Syst. Man Cybern, Syst.*, vol. 52, no. 3, pp. 1588–1602, Mar. 2022, doi: 10.1109/TSMC.2020.3033935.

[6] “Shield AI,” Shield AI. Accessed: Sep. 08, 2025. [Online]. Available: https://shield.ai/hivemind-solutions/

[7] “Skydio Autonomy: AI-driven expert pilot skills in aerial robotics | Skydio.” Accessed: Sep. 08, 2025. [Online]. Available: https://www.skydio.com/skydio-autonomy

[8] “NASA Flies Drones Autonomously for Air Taxi Research - NASA.” Accessed: Sep. 08, 2025. [Online]. Available: https://www.nasa.gov/aeronautics/nasa-flies-autonomous-drones/

[9] A. Tagliabue, K. Kondo, T. Zhao, M. Peterson, C. T. Tewari, and J. P. How, “REAL: Resilience and Adaptation using Large Language Models on Autonomous Aerial Robots,” in *2024 IEEE 63rd Conference on Decision and Control (CDC)*, Dec. 2024, pp. 1539–1546. doi: 10.1109/CDC56724.2024.10885890.

[10] C. Cui *et al.*, “Personalized Autonomous Driving with Large Language Models: Field Experiments,” in *2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC)*, Sep. 2024, pp. 20–27. doi: 10.1109/ITSC58415.2024.10919978.

[11] B. Dharmalingam, R. Mukherjee, B. Piggott, G. Feng, and A. Liu, “Aero-LLM: A Distributed Framework for Secure UAV Communication and Intelligent Decision-Making,” Feb. 05, 2025, *arXiv*: arXiv:2502.05220. doi: 10.48550/arXiv.2502.05220.

[12] S. Javaid, H. Fahim, B. He, and N. Saeed, “Large Language Models for UAVs: Current State and Pathways to the Future,” *IEEE Open Journal of Vehicular Technology*, vol. 5, pp. 1166–1192, 2024, doi: 10.1109/OJVT.2024.3446799.

[13] Y. Tian *et al.*, “UAVs meet LLMs: Overviews and perspectives towards agentic low-altitude mobility,” *Information Fusion*, vol. 122, p. 103158, Oct. 2025, doi: 10.1016/j.inffus.2025.103158.

[14] J. Y. C. Chen, M. J. Barnes, and M. Harper-Sciarini, “Supervisory Control of Multiple Robots: Human-Performance Issues and User-Interface Design,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 41, no. 4, pp. 435–454, Jul. 2011, doi: 10.1109/TSMCC.2010.2056682.

[15] A. Agrawal and J. Cleland-Huang, “Explaining Autonomous Decisions in Swarms of Human-on-the-Loop Small Unmanned Aerial Systems,” *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, vol. 9, pp. 15–26, Oct. 2021, doi: 10.1609/hcomp.v9i1.18936.

BIOGRAPHY

Jeremy Ludwig *is a Director at Stottler Henke, where he leads teams applying AI and ML to create solutions for defense, government, and industry. His work spans intelligent training systems, decision-support tools, and autonomy and collaborative autonomous systems, He earned a Ph.D. in Computer Science from the University of Oregon.*

Daniel Tuohy *is a Lead Software Engineer and Architect at Stottler Henke, with extensive experience in AI research and development. His expertise includes applying large language models to a wide range of challenges and designing and implementing collaborative mission planning systems. He holds an M.S. in Computer Science from the University of Georgia.*

Forest Finnigan *is an AI Software Engineer at Stottler Henke, where he has contributed to defense and aerospace projects focused on autonomy, simulation, and automated scheduling. He received his M.S. in EECS from UC Berkeley.*