

FLIGHT TRIALS OF DRONES CONTROLLED BY AI LARGE LANGUAGE MODELS

Simon Crase[1], Simon Pywell[2], Kent Rosser[1]

[1] Defence Science and Technology Group, Edinburgh, South Australian, Australia
[2] Defence Science and Technology Group, Fishermans Bend, Victoria, Australia

Abstract

Generative AI and Large Language Models (LLMs) such as ChatGPT have rapidly evolved to the stage where they are now influencing many industries. In the domain of drones or Uncrewed Aerial Systems (UAS), these capabilities present opportunities for more flexible and responsive drone operations compared to traditional autonomy approaches that rely on manually engineered flight plans and specialised control software. While a growing body of research has demonstrated the feasibility of LLM-based drone control in simulation environments, these have not progressed to real-world outdoor flight trials.

This paper presents the results of a multi-year investigation into the application of LLMs for autonomous drone control, culminating in successful real-world flight trials conducted in late 2025 using ArduPilot-based quadcopters. The study evaluates the ability of both online and offline LLMs to directly generate and execute Python code for drone operations without the use of dedicated middleware interfaces. The implemented approach integrates LLMs with ArduPilot Software-in-the-Loop (SITL) simulation, enabling rapid verification of generated drone-control behaviour prior to execution on physical aircraft. Experimental missions included autonomous launch, altitude control, waypoint navigation, orbital manoeuvres, and return-to-launch procedures.

The research identifies significant advances in the coding and reasoning capabilities of contemporary LLMs, demonstrating that modern frontier models can now consistently generate executable drone-control code capable of completing end-to-end autonomous flight tasks. However, the study also highlights ongoing challenges relating to reliability, repeatability, safety assurance, and mission verification. A key finding is the importance of tightly coupled simulation environments as a “simulation-in-the-loop” safety mechanism for validating LLM-generated behaviour before deployment to real aircraft.

In addition to presenting experimental results and operational insights, this paper proposes an initial framework for evaluating LLM capabilities across the full lifecycle of aerial autonomy missions, including mission planning, search execution, sensing, decision-making, and post-mission analysis. The findings demonstrate that LLM-controlled drone operations are now operationally feasible and establish a foundation for future research into adaptive, natural-language-driven aerial autonomy systems.

Keywords: Large Language Model, AI, Drone, Simulation, Flight trials

1. Introduction

Generative Artificial Intelligence (AI) and Large Language Models (LLMs) such as ChatGPT have rapidly evolved to the stage where they are now influencing many industries. While these models were initially developed for conversational and software-oriented applications, their ability to generate code and interpret high-level intent has created opportunities for their application in robotics and autonomous platforms. In the field of aerial autonomy, the ability of LLMs to interpret natural language and generate executable code could introduce new opportunities for dynamic, adaptive drone operations.

Traditional approaches to autonomous drone operations rely heavily on purpose-built software pipelines, deterministic control algorithms, and carefully engineered mission logic. Although these approaches can achieve high levels of reliability, they often require significant software development effort for each new mission type or operational environment. Dynamic tasks such as following environmental features, adapting search patterns, responding to unforeseen mission changes, or interpreting operator intent typically require either extensive manual flight planning to span all likely contingencies, or the development/training of specialised computer vision and autonomy policies that respond to visual or other sensor. Such systems remain brittle in the face of unforeseen events as they lack grounded understanding of the scene and context. As a result, conventional drone autonomy can lack flexibility and adaptability when operating in complex real-world conditions.

LLMs offer a fundamentally different approach for aerial autonomy. Rather than relying solely on pre-programmed mission logic, or pre-trained policies, LLMs can potentially translate natural language instructions into executable drone-control behaviours in real time. This capability introduces the possibility of dynamic, adaptive drone operations where human operators can dynamically task aircraft using natural language instructions instead of manually generating complex waypoint missions or developing bespoke autonomy software solutions. This approach has the potential to reduce operator workload while increasing mission agility in applications including search and rescue, environmental monitoring, infrastructure inspection, border surveillance, disaster response, and maritime patrol.

Research has demonstrated the feasibility of using LLMs to control drones in simulated environments [1-9], and more recent studies have begun exploring broader frameworks for autonomous drone reasoning, navigation and mission execution [4-12]. However, there remains limited published research demonstrating direct end-to-end control of physical drones by LLMs in real-world flight conditions. Similarly, there is limited information about how to assess the reliability, safety assurance, simulation-based verification, and the operational maturity of LLM-driven aerial autonomy systems.

This paper presents the results of a multi-year investigation into the use of LLMs for autonomous drone control, culminating in successful real-world flight trials conducted in late 2025. The study focuses on the use of LLMs to directly generate and execute drone-control code using open-source autopilot systems based on ArduPilot. Unlike prior approaches that relied on middleware abstractions, the methodology explored in this research enables LLMs to interact with drone systems in a manner analogous to human autonomy engineers writing executable drone-control scripts. The paper discusses the technical implementation, the role of simulation environments, the outcomes of simulation and live flight testing, and the lessons learned during experimentation.

In addition, this paper proposes an initial framework for evaluating LLM capabilities in aerial autonomy missions, with particular emphasis on mission planning, execution, sensing, decision-making, and post-mission analysis. By demonstrating that LLM-controlled drone flights are now operationally feasible, this research contributes toward the emerging field of adaptive natural-language-driven aerial autonomy and establishes a foundation for future investigation into trustworthy AI-enabled autonomous flight systems.

1.1 Applications for LLMs in Drone

To understand the potential value of LLMs in drone operations, consider the following scenario: The tasking of a long-range, fixed-wing drone to fly 100km following the coastline to patrol for sharks at suburban beaches. This is a task that could easily be executed by a pilot in a small crewed aircraft with the pilot dynamically controlling the aircraft to follow the coastline. However, to do this with an autonomous drone would require a human to manually generate a flight plan with hundreds of waypoints to accurately follow the coastline. Alternatively, a computer vision image classifier trained for detecting the coastline could be paired with an on-board camera, edge processing and control algorithms to control the platform to follow the coastline. This alternative requires significant software development, model training, and systems integration, and the resultant capability can only do this one specific task. The use of LLMs could potentially enable this task to be executed with minimal human effort. Regardless of whether the LLM generates hundreds of waypoints or processes live camera imagery to dynamically control the platform, these could be achieved through tasking from a human using natural language and without software development skills.

Above is one example of tasking that could be given to a drone via a LLM. Similarly, it could be following a road, the edge of a forest, a walking track, a car, searching an area for a lost hiker, identifying a bushfire outbreak, or identifying an intruder on private property. Developing each of

these capabilities using traditional approaches would require significant time, development effort, and specialised expertise. However, these could be quickly realised through the application of LLMs and could even be applied mid-flight. LLMs bring agility and dynamism for dealing with unknown circumstances and capitalising on opportunities. These capabilities align with emerging research into context-aware drone navigation, aerial intelligent agents, and multimodal drone autonomy systems [6,10,11].

In this paper, we focus on the use of LLMs for autonomous flight plan generation, execution and live drone control from a ground control station as an initial demonstration, while our ongoing research aims to progress this to live sensor processing using LLMs and drone control on-board the platform (at the edge).

2. Implementation

There are several possible architectures for integrating LLMs into drone-control systems, as well as a range of LLM types that may be employed. LLMs can be classified as online or offline models. Online models, also referred to as cloud-based models, require an internet connection to pass queries to the LLM servers located in online datacentres where the processing takes place. Due to the computing power available in these datacentres, online models such as ChatGPT, Claude and Gemini are considered frontier models with state-of-the-art capabilities. Offline models execute locally without requiring an internet connection. Local computing power is typically constrained when compared to online datacentres so offline models are normally smaller and less capable. However, the offline nature of these models means they can be run on companion computers onboard a drone (at the edge). Hence, both online and offline models are of relevance to drone applications and are explored in our study.

2.1 Approaches for LLM Control of Drones

Our research extensively utilises open-source autopilots and autonomy systems for drones, and specifically, ArduPilot autopilot which is widely used by industry, research organisations, amateurs, and in defence applications. Hence, we will consider approaches for LLMs to control ArduPilot drones. We have identified three suitable approaches for LLMs to execute drone flights: drone control via middleware, drone control via code generation and execution, and drone control via human machine interfaces.

Drone Control via Middleware

Our review revealed that many existing examples in literature use middleware or specially designed interfaces for the LLM to control the drone [1,2,3,12,15]. This approach simplifies the control problem for the LLM and enabled early experimentation in simulation environments. Recent work has also proposed modular architectures that integrate LLMs with simulation environments, middleware layers and drone command frameworks to improve reliability and safety [3,12,15]

Drone Control via Code Generation and Execution

An alternative approach is for an LLM to directly generate executable software code that, when executed, controls the drone. This approach requires minimal development and integration for a LLM to interact with the drone as python packages and commands exist to do this. However, this approach relies on the ability of the LLMs to produce accurate, executable code that can sequence the various stages of a flight, sense the current state of the drone and its environment, and take appropriate actions to complete the requested mission.

Drone Control via Human Machine Interfaces

A third approach for LLMs to control drones is through a LLM interacting with a human machine interface (HMI) to entering commands and actions throughout a flight. For ArduPilot based drones, this would be achieved by an LLM directly inputting MAVLink commands into a MAVProxy terminal window. Alternatively, the LLM could interact with software graphical user interfaces (GUIs) such as Mission Planner [13] or QGroundControl [14]. Historically, this would require significant integration to enable an LLM to interface with these control interfaces, which would effectively replicate the middleware approach of early LLM drone control research. However, modern LLMs can now interact with computer user interfaces making this a viable approach.

In determining a drone control approach for this paper, we have selected the approach of the LLM directly generating code to execute the flights. Early on in our multi-year research program (circa 2023-2024), our testing showed that LLMs were unable to directly generate successful code to

control a drone. Hence the use of middleware by other researchers [1,2]. However, we estimated that the rapid progress of LLMs would likely result in a functional capability to directly generate code in a similar timeframe to which we could generate a middleware interface for LLM control. In 2025, we identified that LLMs could successfully generate code for drone control. LLMs generating and executing code mirrors the approach of our autonomy research engineers who develop code and algorithms for drones to conduct autonomous flights. It also allows rapid testing though simulation to allow verification of expected flight behaviours before flight (see section 2.2 The Role of Simulation). Finally, it allows reuse of verified and tested code in future flights. Hence, that is the approach we have used for LLM control of drones in this study.

Note that at the time when we selected the above approach, the ability for LLMs to interact with computer interfaces (such as the MAVProxy Command Line Interface or a GUI) was not mature and would have taken significant development effort. Hence, we did not pursue this approach. We note that these computer interaction abilities of LLMs have matured in 2026 and this approach may now be feasible. This is a subject for future investigation.

2.2 The Role of Simulation

One notable insight arising from our investigation of LLM-based drone control is the valuable role that tightly coupled simulations like ArduPilot SITL[16] can play. ArduPilot SITL is a “software in the loop” simulation that allows for the safe testing of code and setting. The difference between launching a drone in SITL and a real drone can be as simple as changing the port or network address commands are sent to. Hence, SITL can be run on a Ground Control Station laptop that is used to control a real drone, allowing for rapid testing and verification of LLM-generated code outcomes in simulation before pushing it to the real drone. This could even be done mid-flight as a verification step when using an LLM to set new drone tasking.

While this approach has proven valuable in providing a safe way of small-scale flight tests of drones, we do acknowledge that there are limitations to this approach. It requires significant human observation and verification and can delay execution of LLM generated code. For widescale adoption of LLM control of drones, automated assurance approaches will be required.

2.3 Software Configuration

Our experimentation and flight trials were conducted using a Linux laptop computer. This computer also performed the roles of running the SITL simulations and as the ground control station for the real-world flights.

For our testing, we used Ollama [17] to select the LLM under test.

Open Interpreter [18] was used to enable the LLMs to directly execute code in a Linux terminal window to conduct the flights.

ArduPilot SITL (Software in the Loop) was used for simulation-based code validation, then the LLM code and commands were sent directly to the drone.

The code generated and executed by the LLM sends MAVLink commands either to the virtual drone in SITL or, via a radio modem to the physical aircraft.

Successful end-to-end flights were conducted which included the use of Open AI ChatGPT as an exemplar current state-of-the-art online LLM. Offline LLMs also tested include models from Meta, Microsoft, Mistral and Google.

2.4 Drone Configuration

The drones used for flight trials were commercially available quad-copters running ArduPilot open source autopilot firmware on a Cube Orange flight controller. Specifically, an AeroDrone MR4 manufactured by Bask Aerospace [19]. The drone was connected to a ground station laptop using an RFD900 modem manufactured by RF Design.



Figure 1 – Bask Aerospace AeroDrone MR4 [19].

3. Experimentation, Flight Trials, and Insights

To test the capabilities of LLMs to control a drone, we selected a mission or task that was simple enough to be achievable but included enough complexity to gain insight into the capabilities of the LLMs. The mission we chose required the LLM do the following actions in sequence:

- arm the quadcopter drone
- climb to an altitude of 60 feet
- fly 50 metres east
- fly in two circles of 10 metres radius
- then return to its launch point and land.

Of note, this requires the LLM to utilise both feet and metres, and no geolocations are supplied so the LLM must calculate the location of waypoints relative to its launch location or current location.

On the ground control station laptop where we ran the LLMs, we had a running instance of SITL (on port udp:127.0.0.1:14550) and a USB RDF900 radio modem for a radio connection to the drone. The drone was powered on but disarmed. Appropriate geofences and emergency behaviours were preloaded into the drone and a licenced pilot with a hand-controller was on standby and ready to intervene if required for safety reasons.

The prompt we used to task LLMs to conduct this flight was:

Directly execute a short single python script for a running instance of SITL that connects to udp:127.0.0.1:14550, arms the drone, sets it to guided mode and tells the drone to go to 60 ft, then fly to 50 east of its current location at 5m/s, then fly in two circles of 10m radius, then RTL.

Once this was successfully demonstrated in SITL, we immediately gave the LLM the following command to execute the flight on the real drone:

Rerun for a real aircraft connected on port "/dev/ttyUSB0" instead of "127.0.0.1:14550"

3.1 Observations of LLM Capabilities in Simulation

Since 2023, we had been testing the abilities of online and offline models in SITL using the methodology described above.

In 2023, online models could generate code that resembled functional drone control code but it would not run without errors and required significant input from a software developer. This aligns with literature [1,2] from that era where researchers use middleware or specially designed interfaces for the LLM to control the drone. This middleware approach simplifies the task for LLMs, which were previously unable to produce functional code to control drones directly. During testing, offline open

source LLMs were observed to be dysfunctional and incapable of producing code for drone control. 2024 saw capabilities of online LLMs progress but human correction of code was typically required. A tightly constrained prompt instructing the LLM to use specific python packages was required to avoid errors. 2025 was observed as when online LLMs would consistently produce code that would both execute and result in flights in SITL. Late 2025, with the release of ChatGPT5 from OpenAI and similarly capable models from Anthropic, was observed as when code could be reliably generated that resulted in successful flights in SITL. Hence, this was when our real-world flight trials were conducted.

Throughout this period, offline models were observed to perform worse than online models and seemed a model generation behind in their capabilities. In 2026, significant progress was observed in offline models, in both their ability to produce code for drone flights, and in their ability to run on low powered computers. Offline models offer potential for LLMs to operate on edge devices such as a companion computer onboard the drone. Hence, this is an area for further investigation.

In testing the LLMs ability to directly generate and execute code for drone flight, it was observed that the default output of LLMs are better suited to reading by a human that directly executable code. Outputs would include verbose text explanations of how the LLM was going to achieve its task or the task would be broken down into multiple steps that expect a human interaction or approval to produce the next section of code. This did not align with the requirements of our research so we included additional guidance in our LLM prompt to "*Directly execute a short single python script…".* This typically resulted in more precise code that would execute without human interaction.

It was observed that LLMs from different companies often utilised different approaches in the code and algorithms they would use. This included the use of different python packages for interfacing with ArduPilot, or whether calculations were conducted from first principles or if mathematical packages were called on. An example of this was when the LLMs were tasked with moving the drone 50m east. Geographic coordinates (latitude and longitude) of this location were not provided to the LLMs so they had to calculate that relative to the launch location of the drone. Some LLMs used the radius of the earth and a spherical model to calculate the destination coordinates, while others called on python packages for these calculations and transforms.

It was also observed that a given LLM does not always generate identical code when presented with the same prompt. An LLM that has produced successful code for full flight execution in SITL, was occasionally observed to produce code that failed to execute, or one component of the flight failed. The frequency of this failure was observed to reduce with more recently released LLMs, i.e. they are becoming more consistent.

When testing LLM flights in SITL, a common failure point was the transition from the initial climb to altitude, to flying on the eastward leg. SITL drones were often observed to continue to climb in altitude indefinitely and not commence moving east. When failures were identified by the LLM itself or by the operator, it was observed that the LLMs were skilled in correcting their code to produce a successful flight.

Once LLMs were consistently producing executable code that resulted in successful flights in SITL, we progressed to real-world flight trials.

3.2 Flight Trials

In November 2025, we conducted multiple real-world flight trials at an Australian test range. Due to our integrated use of simulation, and the tight coupling between ArduPilot SITL and the ArduPilot base drones, the flight trial component of this study was relatively short and with minimal challenges or complications. Once the LLM has demonstrated that it produced code that resulted in a successful flight in SITL, we requested the LLM to rerun the same code but to connect to the USB port that connects to the aircraft instead of the udp port that connects to SITL.

The primary difference observed between SITL simulations and real-world flights was the speed at which the drone tried to attempt its manoeuvres. When first transitioning from SITL flights to real-world flights, we did not specify the speed for the LLMs to fly the drone at. This did not cause issues for the flights in SITL. However, once we ran the code on the real drone, we observed that the speeds were too high for the drone to accurately follow the circle requested by the LLM, resulting in a

distorted circle and only partial completion on the two circles. Consequently, we adjusted the prompt that we put to the LLMs to include an appropriate speed for the Bask MR4 drone to conduct the transit east and the circles. This discrepancy could likely be overcome by including accurate Bask MR4 flight models into SITL.

The arming, launch and landing component of the flight were executed correctly at the first attempt.

One key insight from our experiments is the critical role a tightly coupled simulation environment such as ArduPilot SITL can play. While LLM capabilities are advancing rapidly, performance remains inconsistent. Until LLMs can reliably execute dynamic natural language tasking in real-world missions, we advocate a "simulation-in-the-loop" approach for verification prior to execution. This strategy enables seamless transition from SITL to live flight and supports mid-flight validation of new tasking. We do acknowledge that this approach requires significant human interaction and observation of the mission execution in SITL. While well suited to experimental flights like ours, this is likely to be unsuitable for long and complex drone missions or widespread use of multiple LLM controlled drones. Here, further research is required in digital failsafes with less operator overhead. This may include transition of test, evaluation, verification and validation (TEVV) techniques from mainstream LLM usage into the aerial autonomy domain.

4. A Framework for LLM Assessment

Our initial flights of drones controlled by LLMs have shown that this is now a feasible approach for drone control and warrants further progression of these capabilities. There is a notable gap between the capabilities we have demonstrated in this study and those required to conduct an end-to-end drone mission. As LLM capabilities continue to advance, a structured framework for evaluation and comparison of competing LLMs in the context of drone missions is required. Recent surveys have highlighted the need for systematic evaluation methodologies for LLM-enabled drone systems, particularly in areas including mission planning, autonomous navigation, swarm coordination, safety assurance, human-machine interaction, and communications management [4,5,9]. While some frameworks exist for LLM evaluation [20] including some tailored for robotics applications and autonomous driving [21,22], no equivalent framework specifically targeting aerial autonomy has been identified.

In proposing a framework for assessment of LLM capabilities in the aerial autonomy domain, we suggest the use of a drone search and rescue (SAR) mission to highlight the different mission stages and requirements for assessment. The Australian Civil Aviation Safety Authority (CASA)[23] and US Federal Aviation Authority (through ASTM F3266 [24]) provide recognised documentation for operation of for small Uncrewed Aerial Systems (UAS). This doctrine provides the structure for our framework proposed below:

1. Mission Planning & Tasking

Evaluation metrics may include an LLM's abilities to:

- Produce a search plan appropriate for what is being searched for (person, vessel, debris field)
- Select and generate search patterns: e.g., expanding square, parallel track, creeping line.
- Match search patterns to platform and sensor capabilities
- Check regulatory compliance: Airspace classification, NOTAMs, permissions.
- Consider environmental aspect: Weather, wind profiles, solar angle, terrain masking.

2. Pre-Flight Preparation & System Checks

Evaluation metrics may include an LLM's abilities to:

- Configure failsafes: Geofence, Return-to-home (RTH), lost link procedures.
- Interface with sensors: Camera feed, gimbal alignment, georeferencing accuracy.
- Check platform status: Battery levels, GPS lock, compass calibration, waypoint validation.

3. Deployment & Launch

Evaluation metrics may include an LLM's abilities to:

- Conduct safe launch: Arming, Launch, Mode changes.

4. Transit to Search Area

Evaluation metrics may include an LLM's abilities to:

- Navigation execution: Autonomous waypoint flight.
- Collision Avoidance: Maintain separation from other aircraft/obstacles.
- Formation control: For teams of drones conducting missions.

5. Search Execution (Core Mission Phase)

Evaluation metrics may include an LLM's abilities in:

- Pattern adherence: Fly predefined search geometry precisely.
- Sensor operation: Continuous EO/IR scanning.
- Altitude/speed optimization: Trade-off between coverage rate and resolution.
- Real-time adjustments: Modify pattern based on findings or updated information.

6. Detection, Identification & Tracking

Evaluation metrics may include an LLM's abilities in:

- Initial detection: Visual, thermal, or algorithmic cue.
- Classification: Confirm whether the object matches search criteria.
- Geolocalisation: Fuse detection data with autopilot location data to calculate geolocation.
- Persistent tracking: Orbit or loiter to maintain visual contact if required.
- Evidence capture: Still imagery, video, metadata logging.

7. Data Transmission & Decision Support

Evaluation metrics may include an LLM's abilities in:

- Context sensing: Generate additional context from the detection image for decision making.
- Data transmission: Information transmitted is appropriate for communications bandwidth
- Decision-making: Recommended courses of action appropriate for context.

8. Return, Recovery & Landing

Evaluation metrics may include an LLM's abilities in:

- Landing skills: Automated landing, terrain knowledge and avoidance.
- Post-flight procedures: Disarming, data transmission

9. Post-Mission Analysis & Reporting

Evaluation metrics may include an LLM's abilities in:

- Data processing: Image review, stitching, enhancement, AI-assisted analysis.
- Coverage assessment: Verify search completeness (coverage maps).
- Log Analysis: Battery usage, airframe status, performance, deviations.
- Reporting: Mission logs, findings, recommendations.
- Lessons learned: Automated correction or enhancement to capabilities.

From our flights of LLM controlled drones, we have demonstrated aspects of "Deployment & Launch", "Transit to Search Area", and "Return, Recovery & Landing". Hence, there is significantly more scope for demonstration of LLM capabilities in the execution of drone missions. The above framework provides a structure against which the capabilities of LLMs can be assessed for execution of drone missions.

5. Conclusion and Future Work

This research has demonstrated that Large Language Models (LLMs) are now capable of directly controlling autonomous drone flights through natural language tasking and dynamic code generation. Unlike earlier approaches that relied upon middleware, custom interfaces, or manually engineered autonomy pipelines, modern LLMs can now generate executable drone-control code capable of performing real-world flights with minimal human intervention. Through the successful real-world flight trials using ArduPilot-based quadcopters, our study confirms that LLM-enabled aerial autonomy has progressed beyond simulation and into operational feasibility.

A key insight from this work is the importance of tightly coupled simulation environments such as ArduPilot SITL for verification and safety assurance. While LLM-generated code has become significantly more reliable, inconsistencies and unexpected behaviours still occur. The “simulation-in-the-loop” methodology demonstrated in this paper provides a practical mechanism for validating generated flight behaviour before execution on physical aircraft, hence, reducing operational risk while retaining the flexibility and adaptability offered by LLM-driven autonomy.

This research also highlighted several important limitations and challenges that warrant future investigation. The missions demonstrated in this study represent only a subset of the requirements for fully autonomous aerial operations. Higher-level mission execution tasks such as dynamic search planning, onboard sensor interpretation, collaborative multi-agent operations, collision avoidance, adaptive decision-making, and autonomous target identification remain areas requiring substantial further investigation. In addition, robust approaches for trust, testing, evaluation, verification, and validation (TEVV) of LLM-enabled autonomy systems must be developed before these capabilities can be safely deployed at scale in operational or safety-critical environments.

Future work will therefore focus on extending LLM capabilities from ground-station-assisted autonomy toward onboard edge-based autonomy using local offline models running directly on companion computers integrated with the drone. This direction aligns with recent research investigating aerial intelligent agents, agentic drone architectures, multimodal aerial reasoning systems, and autonomous swarm coordination enabled by LLMs [6,8,11]. This extension to edge implementation of LLMs enables the use of multimodal LLMs for real-time sensor processing, adaptive mission replanning, and autonomous interpretation of imagery and environmental context. The framework proposed in this paper for evaluating LLM performance across the full lifecycle of aerial missions provides an initial foundation for systematic assessment and comparison of future systems.

Overall, this work demonstrates that LLM-controlled drone operations have progressed beyond proof-of-concept demonstrations and are now operationally feasible. The successful real-world flight trials presented in this study establish a practical baseline for future research into adaptive, natural-language-driven aerial autonomy and represent an important step toward more flexible, intelligent, and responsive autonomous drone operations.

Copyright Statement

The authors confirm that they, and/or their company or organization, hold copyright on all of the original material included in this paper. The authors also confirm that they have obtained permission, from the copyright holder of any third party material included in this paper, to publish it as part of their paper. The authors confirm that they give permission, or have obtained permission from the copyright holder of this paper, for the publication and distribution of this paper as part of the ICAS proceedings or as individual off-prints from the proceedings.

References

[1] Chen G, Yu X, Ling N and Zhong L. TypeFly: Flying Drones with Large Language Model. arXiv preprint arXiv:2312.14950, 2023.

[2] Tazir M L, Mancas M and Dutoit T. From Words to Flight: Integrating OpenAI ChatGPT with PX4/Gazebo for Natural Language-Based Drone Control. Proc International Workshop on Computer Science and Engineering 2023, 2023.

[3] Phadke A, Hadimlioglu A, Chu T and Sekharan C N. Integrating Large Language Models for UAV Control in Simulated Environments: A Modular Interaction Approach. arXiv preprint arXiv:2410.17602, 2024.

[4] Javaid S, Fahim H, He B and Saeed N. Large Language Models for UAVs: Current State and Pathways to the Future. IEEE Open Journal of Vehicular Technology, 2024.

[5] Cidjeu D D, Fendji J L K E, Kamla V C and Tchappi I. UAV Leveraging GenAI/LLMs, a Brief Survey.

Procedia Computer Science, Vol. 265, pp. 382–389, 2025.

[6] Zhao J and Lin X. General-Purpose Aerial Intelligent Agents Empowered by Large Language Models. arXiv preprint arXiv:2503.08302, 2025.

[7] Choutri K, Fadloun S, Khettabi A, Lagha M, Meshoul S and Fareh R. Leveraging Large Language Models for Real-Time UAV Control. Electronics, Vol. 14, No. 21, Article 4312, 2025.

[8] Koubaa A and Gabr K. Agentic UAVs: LLM-Driven Autonomy with Integrated Tool-Calling and Cognitive Reasoning. arXiv preprint arXiv:2509.13352, 2025.

[9] Emami Y, Zhou H, Reddy R, Arani A H, Wang B, Li K, Almeida L and Han Z. Large Language Model-Assisted UAV Operations and Communications: A Multifaceted Survey and Tutorial. arXiv preprint arXiv:2602.19534, 2026.

[10] Khan A, Rehman I U, Saeed N, Sobnath D, Khan F and Khattak M A K. Context-Aware Autonomous Drone Navigation Using Large Language Models. Proc AAAI Symposium Series, Vol. 6, pp. 102–107, 2025.

[11] Ping, Y., Liang, T., Ding, H., Lei, G., Wu, J., Zou, X., Shi, K., Shao, R., Zhang, C., Zhang, W. and Yuan, W., Multimodal large language models-enabled UAV swarm: Towards efficient and intelligent autonomous aerial systems. *IEEE Wireless Communications*, 2025.

[12] Nunes, D., Amorim, R., Ribeiro, P., Coelho, A. and Campos, R., 2025, July. A Framework Leveraging Large Language Models for Autonomous UAV Control in Flying Networks. *IEEE International Mediterranean Conference on Communications and Networking (MeditCom)* (pp. 12-17). IEEE., 2025.

[13] ArduPilot Development Team. Mission Planner Overview. ArduPilot Documentation, 2025. Available: https://ardupilot.org/planner/docs/mission-planner-overview.html

[14] QGroundControl Development Team. QGroundControl: Drone Control and Ground Control Station for MAVLink Vehicles. QGroundControl Project Documentation, 2025. Available: https://qgroundcontrol.com

[15] Lim S K, Chong M J Y, Khor J H and Ling T Y. Taking Flight with Dialogue: Enabling Natural Language Control for PX4-Based Drone Agent. arXiv preprint arXiv:2506.07509, 2025.

[16] ArduPilot Development Team. ArduPilot Software-In-The-Loop (SITL) Simulator Documentation. ArduPilot Documentation, 2025. Available: https://ardupilot.org

[17] Ollama Team. Ollama: Run Large Language Models Locally. Ollama Documentation, 2025. Available: https://ollama.com

[18] Open Interpreter Team. Open Interpreter: Let Language Models Run Code Locally. Open Interpreter Documentation, 2025. Available: https://github.com/OpenInterpreter/open-interpreter

[19] Bask Aerospace. AeroDrone MR4 Multirotor Aircraft Specifications. Bask Aerospace Product Documentation, 2025.

[20] Chang Y, Wang X, Wang J, Wu Y, Zhu K, Chen H, Yi X, Wang C, Wang Y, Ye W, Zhang Y, Chang Y, Yu P S, Yang Q and Xie X. A Survey on Evaluation of Large Language Models. ACM Transactions on Intelligent Systems and Technology, Vol. 15, No. 3, pp. 1–45, 2024.

[21] Huang W, Abbeel P, Pathak D and Mordatch I. Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents. Proc International Conference on Machine Learning (ICML), Baltimore, USA, pp. 9118–9147, 2022.

[22] C. Cui, Y. Ma, X. Cao, W. Ye, Y. Zhou, K. Liang, J. Chen, J. Lu, Z. Yang, K. Liao, et al. (2024-Jan.)*A survey on multimodal large language models for autonomous driving*.In Proceedings of the IEEE/CVF winter conference on applications of computer vision,Waikoloa, HI, USA, pp. 958–979.

[23] Civil Aviation Safety Authority. AC 101-01 v6.1: Remotely Piloted Aircraft Systems (RPAS) Licensing and Operations. CASA Advisory Circular AC 101-01, Canberra, Australia, 2025.

[24] ASTM International. ASTM F3266-18: Standard Guide for Training for Search and Rescue (SAR) Using Small Unmanned Aircraft Systems (sUAS). ASTM International, West Conshohocken, PA, USA, 2018.