

Integrating Manned Control Systems and Camera Feeds for Aerial Vehicle Control

Ahmad Qureshi
Cole Hersh
Drake Russ
Ryan Fong
Alexia Segarra
Lilly Gentry
Aarush Parvataneni
Etan Grant

Under the Mentorship Of:
Micheal W Marcellin

Electrical and Computer Engineering University of Arizona Tucson AZ, USA

ABSTRACT

We present a novel integration of Pixhawk autopilot technology with a manual control system utilizing a live camera feed for real-time mission control. TMotor FLAME and HobbyWing PLATINUM motors provide a powerful and stable source of lift and thrust. The Pixhawk (PX4) autopilot system provides a robust platform for autonomous missions in unmanned aerial vehicles (UAVs), offering precise control and navigation capabilities. Utilizing a Python library to transmit and interpret MAVlink messages, the aircraft responds to commands with meter-level accuracy. Instead of autonomous processing, the system relies on a real-time camera feed to provide operators with visual information, enabling manual adjustments during flight. This allows for responsive control in dynamic environments, ensuring adaptability to obstacles and changing conditions. We aim to highlight the synergy between the PX4 and manual control systems, demonstrating their combined potential to enhance UAV operations through real-time human oversight.

INTRODUCTION

Unmanned Aerial Vehicles (UAVs), particularly quadcopters, have seen a rapid evolution in both capability and autonomy, driven by advancements in embedded systems, sensing, and control algorithms. Automation has enabled UAVs to execute complex missions with minimal human intervention, finding applications across fields such as infrastructure inspection, search and rescue, agriculture, and defense. However, fully autonomous systems still face limitations in

dynamic, uncertain, or safety-critical environments — especially when human judgment is necessary to handle edge cases or mission-level decisions.

In light of these challenges, there is growing interest in hybrid control architectures that blend autonomous navigation with human-in-the-loop oversight. Our motivation stems from both competitive requirements and practical reliability considerations. A newly introduced competition rule mandates the inclusion of manual supervisory control alongside autonomous capabilities, reflecting an industry trend toward multi-agent coordination with centralized human monitoring. Moreover, human involvement adds a layer of safety and adaptability, particularly in mission scenarios that involve dynamic obstacles or require responsiveness to moving targets.

This paper presents the efforts of Arizona Autonomous Vehicles, an undergraduate research and competition team advised by Dr. Michael Marcellin at the University of Arizona. Our team is working to integrate real-time manual control into an otherwise autonomous UAV system. Our platform consists of a PX4-based flight stack, augmented by a companion computer running custom firmware to enable responsive, operator-influenced control. We introduce a live camera feed streamed to a ground station, allowing visual telemetry and mission oversight. This integration enables a human operator to supervise and intervene across multiple UAVs, bridging the gap between high-level autonomy and human judgment in real-world deployment.

Related Work

UAV control systems are generally divided into two paradigms: fully autonomous operation and manual piloting. Autonomous UAVs rely on preprogrammed missions, onboard navigation algorithms, and sensor feedback to execute complex tasks without continuous human intervention. Manual systems, by contrast, depend on human operators for all flight decisions, often through RC transmitters or ground control interfaces. While autonomy has enabled UAVs to operate in structured, GPS-denied, or high-risk environments, manual control remains indispensable in unpredictable or safety-critical contexts. Recent advances in hybrid systems have sought to combine the strengths of both approaches by enabling autonomous UAVs to accept human input during flight, though few frameworks are optimized for real-time collaborative control.

PX4 has emerged as a widely adopted open-source autopilot stack in both academic and commercial UAV applications. Combined with the MAVLink communication protocol, it provides a modular and extensible system for autonomous flight control. In our work, we leverage PX4 on a Pixhawk flight controller, interfaced via pymavlink and MAVSDK. These tools allow for high-level command structuring while abstracting the low-level complexities of

MAVLink. To streamline development and enhance clarity, we implemented a custom Python-based Plane class that encapsulates UAV behaviors such as takeoff, landing, and redirection in response to dynamic conditions or foreign objects.

Camera-based navigation and situational awareness are also widely studied in aerial robotics. While many systems apply computer vision (CV) for object tracking, obstacle avoidance, or SLAM, fewer provide shared visual interfaces that allow human operators to view and interpret the same visual data stream used by the UAV. Our platform incorporates a gimballed onboard camera, streaming over a LAN via a large dish network to a ground station. This setup supports both autonomous CV processing and manual supervision, allowing the operator to intervene or override autonomous behavior as needed. This dual-use vision stream facilitates not only improved safety, but also the training and debugging of CV models in real-time conditions.

Despite the availability of sophisticated autonomous and manual control frameworks, there remains a gap in systems that offer responsive, scalable hybrid control — particularly those that allow a single operator to manage and redirect multiple UAVs in flight. Our integration addresses this gap by prioritizing real-time human oversight, mission-level adaptability, and future scalability, while remaining relatively lightweight and modular in implementation.

System Architecture

Hardware Components

The propulsion system of our Mugin VTOL kit integrates high-performance components to ensure stable lift and thrust during both vertical and horizontal flight. For the vertical lift motors, we have selected the TMotor FLAME 100A 14s Electronic Speed Controllers (ESCs) paired with TMotor MN8014 Antigravity 180kv brushless motors. The FLAME series ESCs are renowned for their reliability and precise throttle control, capable of handling high current loads required by heavy-lift UAVs. Operating at voltages up to 14s (58.8V), these ESCs feature advanced algorithms for optimizing response time and motor efficiency, which is paramount in maintaining stability during VTOL transitions. The MN8014 Antigravity motors offer a low Kv rating (180kv), which enables them to spin large propellers at lower speeds, producing ample thrust while maintaining high efficiency and minimizing heat generation—ideal for prolonged vertical flight operations.

For the fixed-wing propulsion, our configuration employs HobbyWing PLATINUM 130A HV ESCs in combination with Dualsky XM6360EA V3 motors. The PLATINUM 130A HV ESC supports high-voltage (up to 12s) setups, delivering smooth and precise throttle response required for the nuanced demands of horizontal, fixed-wing flight. Its high current rating ensures consistent performance during high-power maneuvers, while robust safety features protect

against voltage spikes and thermal overload. Paired with the Dualsky XM6360EA V3—a motor designed for efficiency and power in larger UAV applications—this setup provides reliable forward thrust with minimal energy loss. The synergy between these components allows for optimal energy management, smooth acceleration, and the overall system efficiency required for extended endurances and dynamic mission profiles. This carefully chosen arrangement of motors and ESCs not only guarantees the necessary lift and propulsion but also ensures precise operator control, aligning well with the hybrid autonomous-manual mission concept.

The Mugin VTOL kit's demanding power requirements are met with a robust battery setup utilizing four ThunderPowerRC TP16AH-6SUAV lithium polymer (LiPo) batteries. Each TP16AH-6SUAV cell is a 6-cell (6S) configuration, providing a nominal voltage of 22.2V and a capacity of 16,000mAh (16Ah). By wiring two of these batteries in series, the system achieves a total voltage of 44.4V, which is crucial for delivering sufficient power to high-voltage motors and ESCs, ensuring optimal performance under heavy loads. The remaining two batteries are connected in parallel to this series pair, effectively doubling the available capacity to 32,000mAh while maintaining the 44.4V output. This series-parallel arrangement strikes a balance between delivering high voltage—essential for efficient, high-thrust operations—and providing extended endurance through increased current capacity.

The total energy storage afforded by this configuration not only supports the substantial current demands of both lift and propulsion systems but also enhances flight times, key for complex or long-range missions. Given that the ESCs and motors can draw significant peak currents, especially during VTOL transition phases or full-throttle cruise, the high current handling capacity and discharge rates of the ThunderPowerRC cells are vital. Each battery is rated for safe operation under high-load conditions, minimizing voltage sag and thermal buildup even at sustained high draw. By matching the power system's voltage and current characteristics to the motor and ESC specifications, the UAV is able to maintain stable, reliable performance while protecting components from overcurrent conditions or power irregularities.

Camera system: specifications, mounting, and field of view

SOFTWARE STACK

The UAV's core flight operations are managed by the PX4 autopilot stack, an open-source platform responsible for critical functions such as navigation, stabilization, and automatic fail-safes. PX4 uses sensor fusion to maintain aircraft attitude and position, execute waypoint missions, and respond autonomously to emergency situations like communication loss or low

battery. This autonomous capability forms the backbone of the system, enabling the UAV to perform complex tasks with minimal operator intervention.

Complementing PX4, the higher-level control and mission management are implemented in Python using a combination of pymavlink and MAVSDK, augmented by an in-house **Plane** class that serves as the primary interface between the software and the UAV hardware. The **Plane** class initializes a MAVLink connection to the Pixhawk via pymavlink over a USB link to a radio transmitter. It polls telemetry data—including position, velocity, and flight status—at a frequency of 10 Hz, ensuring that up-to-date situational awareness is available for both autonomous decision-making and operator oversight.

The **Plane** class encapsulates core UAV commands such as takeoff, landing, and flight mode switching between quadcopter hover and forward flight. It also manages mission waypoint execution through a separate **Waypoints** class, which loads waypoint data from structured text files and constructs flight missions using MAVSDK’s mission APIs. To ensure safe operation, a polygon geofence is defined using MAVSDK, creating virtual boundaries for the UAV. Although the UAV is programmed to hold position if the geofence boundary is reached, path-planning within the mission class proactively avoids such violations by pre-validating flight paths.

A key feature of the system is the integration of manual control capabilities that always take priority over autonomous commands. Operator inputs are continuously monitored and, when present, immediately override autopilot behavior to ensure real-time responsiveness and safety. This prioritization is enforced within the **Plane** class’s decision logic, which seamlessly switches control authority to the human operator whenever manual commands are detected. This guarantees that human oversight remains paramount in all dynamic or unforeseen situations.

Supporting manual control is a live video feed from a gimballed onboard camera streamed via a LAN network to a ground station. This feed provides the operator with real-time visual feedback, mirroring the input seen by onboard computer vision algorithms. The shared visual stream reduces latency and cognitive load, enabling the operator to effectively supervise the UAV’s environment and intervene when necessary. The combined approach of autonomous navigation, geofencing, and human-in-the-loop control results in a flexible and responsive UAV platform well-suited for competitive missions and complex operational environments.

The software stack avoids reliance on traditional ground control stations like MAVProxy or QGroundControl during active missions. Instead, it leverages direct control via pymavlink to maintain fine-grained command timing and message handling, facilitating rapid response to both autonomous triggers and operator interventions.

IMPLEMENTATION

The UAV platform is built around the PX4 autopilot, deployed on a Pixhawk-compatible flight controller, and supported by a low-spec companion laptop. The system is mounted on a

multirotor frame built from a Mughin kit, chosen for its modularity and compatibility with PX4-based components. Careful system setup and calibration are essential to ensure reliability, especially given the hybrid control architecture that combines autonomous flight with real-time operator input.

Before each flight, we perform a series of calibration routines to guarantee sensor accuracy and control consistency. Manual calibration procedures are conducted for the onboard accelerometer, gyroscope, and magnetometer using PX4's built-in calibration tools. These steps are operator-driven to ensure precision, particularly in environments where magnetic or inertial interference is a concern. In parallel, automatic calibration routines—such as ESC and radio calibration—are executed by PX4 on startup, streamlining system readiness. The gimballed camera is also aligned and tested to confirm stability and image orientation before flight.

Communication between the ground station and the aircraft occurs via MAVLink over UDP, transmitted through a USB-linked radio module. PX4 internally tracks communication latency using timestamp-based mechanisms embedded in the MAVLink message structure. Our ground-side Python interface—via pymavlink—monitors these timestamps in real time, with typical round-trip latency consistently measured at under 0.05 seconds. This low latency ensures that manual control commands are executed without delay, a critical requirement for our operator-priority design.

The heart of this interaction is the in-house `Plane` class, which maintains continuous state polling at 10 Hz and provides immediate response to manual commands. If operator input is detected, it automatically and unequivocally overrides any ongoing autonomous behavior, interrupting or redirecting the mission as necessary. This prioritization is fundamental to our control philosophy, ensuring human oversight remains authoritative in all cases—particularly in dynamic or unpredictable mission contexts.

Field testing is conducted at the Tucson International Modelplex Park Association (TIMPA), a large RC flight facility located in Avra Valley, west of Tucson. This site offers over 160 acres of open, unobstructed airspace, ideal for extended autonomous flight experiments without urban interference. The aircraft operates within a defined geofenced boundary enforced through MAVSDK's polygon tools and our `Mission` class's built-in path validation logic. These constraints prevent boundary violations proactively, rather than reactively.

Each field test follows strict safety procedures. The aircraft is equipped with redundant kill switches—one software-triggered via the companion computer and one hardware-triggered via the transmitter—allowing for immediate shutdown in the event of flight anomalies. While the system is currently configured for single-UAV deployment, the architecture is designed to scale to multi-agent control scenarios. Mission durations are typically under two hours, limited primarily by battery capacity, and each session includes full pre- and post-flight inspection and logging.

Through this integrated setup—combining precise calibration, low-latency control, and disciplined field testing—the system has demonstrated robust and responsive behavior, laying the groundwork for more complex missions and multi-agent operations in future competition deployments.

BENEFITS OF COMBINING MANUAL AND AUTONOMOUS CONTROL

A benefit of combining autonomous and human control is that it allows the autonomous systems to be tested more safely. Autonomous systems of this level are still a relatively new and are operating in complex environments, so there are many edge cases that need to be worked on. Adding partial human control provides a greater level of adaptability and speed until the autonomous system is improved enough to function on its own.

Adding human control also makes it much clearer who is responsible should something go wrong. Because the development and use of the plane is being done by the same group, this is not very applicable here. However, if a company were to develop and sell an autonomous product without human supervision or control, it would be harder to determine which party is responsible when something goes wrong.

LIMITATIONS

One of the limitations on the speed of the plane's progress is how often tests are able to be carried out. In order to make sure the tests are done in a safe place, the tests are done in TIMPA, which is a 40 minute drive away from the club room, which limits the available days to weekends only. Additionally, only a couple of people are currently allowed to fly the plane and a trailer is needed to transport it, so the days that are available are relatively low in number.

A potential improvement is the addition of two forward facing cameras on the plane. Using stereo vision, these could be used to look for obstacles like trees or other planes. In the past, this club has worked on an implementation of this with python, openCV, and two webcams. However, progress on it has stopped due to a change of focus following the rule changes.

CONCLUSIONS

This integration is particularly impactful in environments characterized by dynamic obstacles, uncertain conditions, or real-time mission updates—contexts that are increasingly common in both research and real-world applications. The inclusion of a shared visual feedback loop, which allows the operator to view the same gimballed camera feed used by onboard systems, further reinforces the collaborative control structure. It empowers the human operator not only to

intervene, but also to interpret, anticipate, and adjust to environmental factors in ways current autonomous algorithms often cannot.

Beyond its immediate utility in competitions, this work contributes to the broader field of aerial robotics, offering a scalable and modular framework for integrating human oversight into autonomous platforms. As drone applications expand into multi-agent systems, urban deployments, and autonomous inspection tasks, the ability to balance autonomy with real-time human adaptability will become increasingly vital. Our system lays a foundation for that future by demonstrating that reliable autonomous flight need not come at the expense of human authority—it can be strengthened by it.

REFERENCES

- [1] M. Lorenz Meier et al., “MAVLink Developer Guide,” [Online]. Available: <https://mavlink.io/en/>
[Accessed: Jun. 30, 2025].
- [2] ArduPilot Developers, “pymavlink: MAVLink bindings for Python,” GitHub repository. [Online]. Available: <https://github.com/ArduPilot/pymavlink>
[Accessed: Jun. 30, 2025].
- [3] PX4 Development Team, “PX4 System Architecture Overview,” PX4 Autopilot Documentation. [Online]. Available: https://docs.px4.io/main/en/concept/px4_systems_architecture.html
[Accessed: Jun. 30, 2025].
- [4] Wikipedia Contributors, “PX4 autopilot,” Wikipedia, The Free Encyclopedia. [Online]. Available: https://en.wikipedia.org/wiki/PX4_autopilot
[Accessed: Jun. 30, 2025].
- [5] M. L. Meier, “Using pymavlink,” MAVLink Documentation. [Online]. Available: https://mavlink.io/en/mavgen_python/
[Accessed: Jun. 30, 2025].
- [6] S. Mahmoud, L. G. Jaimes, and S. A. Hosein, “MAVLink: A Survey,” arXiv preprint arXiv:1906.10641, 2019. [Online]. Available: <https://arxiv.org/abs/1906.10641>
[Accessed: Jun. 30, 2025].