**Senior Design Project: Rescuer**

Keon Augustine, Leule Bantywalu, Tehya Gaines, Jaden Samuels

Howard University

Course Number: EECE 404

Dr. Charles Kim

April 23, 2025

90/100

**Abstract**

The Rescuer project presents the design and implementation of an autonomous search-and-recovery system designed to aid in locating lost pets using Unmanned Ariel Vehicles (UAV) and Unmanned Ground Vehicles (UGV). Using real-time object detection, autonomous flight, and GPS geolocation, the UAV can identify visual targets and transmit precise location data to a UGV. This coordinated system offers an efficient alternative to manual search efforts, reducing the time, labor, and uncertainty associated with traditional pet recovery methods.

The system's architecture is centered around a Raspberry Pi 5 and Pixhawk flight controller, with the UAV integrating a high-resolution ArduCam IMX477 camera, GPS module, and electronic speed controls. Software components include Mission Planner for autonomous flight path design and Python-based algorithms utilizing OpenCV for visual detection. Over a three-month agile development cycle, the team achieved several key milestones, including stable hardware integration, reliable marker recognition at a distance of 30 feet, and a final prototype demonstration to university and industry advisors. Future work aims to expand from marker recognition to general object detection and enable full end-to-end autonomy where the UGV can independently respond to UAV-provided coordinates.

**Problem Statement**

Many pet owners struggle to find lost pets; they often end up in expansive areas, making manual search time-consuming and inefficient. A scouting UAV with an integrated camera equipped with image processing and recognition abilities can help efficiently locate missing pets. Once the pet is identified, the UAV will capture the GPS coordinates of the location and transmit this data to other unmanned systems.

This approach leverages autonomous flight, real-time object detection, and geolocation to streamline the search process, reduce human labor, and enable faster response. By removing the need for manual searching and enabling data-driven coordination, the system offers a more scalable and effective solution for locating lost pets.

-5 Lack of non-technical constraints and compliance

**Design Requirements**

The system is composed of a UAV and UGV, each equipped with its own distinct set of hardware and software components. The UAV serves as the primary scouting unit and is built around a Raspberry Pi 5, which functions as the onboard processor for image recognition and data handling. Flight control is managed by a Pixhawk controller, which works in conjunction with an Electronic Speed Controller (ESC), brushless motors, and propellers to enable stable autonomous flight. Additional hardware includes a GPS module for real-time geolocation and an ArduCam IMX477 camera module, which provides high-resolution video input for image detection. Power delivery is supported through a dedicated module

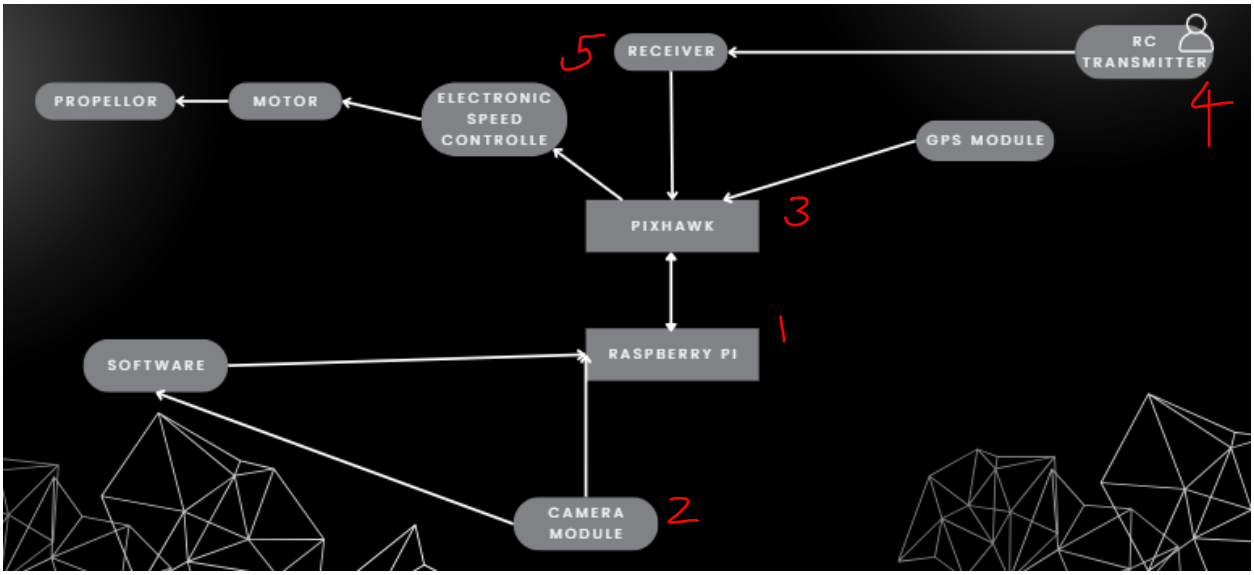
compatible with the UAV's high-performance demands, and an RC transmitter (UAV controller) allows for manual override during development and safety testing.

The UAV's software infrastructure combines both navigation and computer vision. Mission Planner was used for autonomous flight path planning, including takeoff, waypoints, and landing procedures. Python was used to implement the ArUco marker recognition algorithms, using OpenCV libraries to detect, track, and respond to visual targets in real time. This hardware-software integration allows the UAV to operate autonomously while remaining responsive to external commands and environmental input.

-5 Deficiency in patent-style description

**Solution Design**

The UAV operates using a modular system with its hardware centered around the Raspberry Pi, which serves as the core processor for image recognition, navigation coordination, and communication with the flight controller. The camera module, along with the algorithms responsible for image detection and autonomous flight, are directly integrated with the Raspberry Pi. These algorithms work in tandem with the Pixhawk flight controller, which interprets the data and manages the electronic speed controllers that control the motors and stabilize the drone during flight. Additionally, the Pixhawk supports input from an RC transmitter, allowing for manual control when needed. Signals from the transmitter are received by an onboard receiver and relayed through the Pixhawk to guide the drone's movement and behavior.




Several key constraints were considered while developing and testing the design. Due to Washington, D.C.'s designation as a no-fly zone, drone testing was restricted to indoor environments and select local parks outside the restricted area. Additionally, the UAV was required to include a manual override or "kill switch" to ensure safe landing in the event of an autonomous system failure. This feature maintains full autonomous functionality while providing a critical layer of manual control for safety and compliance.

## Project Implementation

To structure the development of the Rescuer project, the team followed an agile workflow over the course of three months. This approach allowed for iterative development through defined sprints, each with clear objectives, deliverables, and checkpoints. By breaking down the project into incremental stages, the team was able to maintain steady progress while adapting to unforeseen technical challenges and design constraints.

Sprint 1, which spanned from January 28 to February 10, was dedicated to laying the groundwork for the project. During this phase, the team collected all necessary hardware components, researched various object detection algorithms, and began drafting the UAV's autonomous flight plan. Particular attention was given to the takeoff and landing procedures, which would later be refined through testing. Early exploration of flight control systems and drone configuration tools also took place during this sprint.

In Sprint 2, from February 17 to March 10, the focus shifted toward system integration and testing. The team selected ArUco markers as the primary objects for recognition due to their high detection accuracy and compatibility with open-source computer vision libraries. Using the Arducam and Raspberry Pi, the team began developing and validating the image recognition pipeline. Concurrently, Mission Planner software was used extensively to design and simulate autonomous flight paths, fine-tune GPS responses, and understand the UAV's behavior under different mission profiles.

Sprint 3, which ran from March 17 to March 31, was dedicated to final integration and debugging. During this critical phase, the UAV's camera, image recognition system, and flight controller were brought together into a cohesive unit. The team worked through power, stability, and data transmission issues to ensure that the UAV could complete its mission reliably. The sprint culminated in a successful demonstration of the working prototype to faculty at Howard University and advisors from RTX, showcasing the system's ability to detect ArUco markers, capture GPS data, and prepare that information for transfer to the UGV.

## Project Implementation Process

Throughout the development process, the team achieved several important technical milestones that were critical to the success of the project. Initial camera testing confirmed that the ArUco marker detection algorithm functioned reliably under varying lighting conditions and from multiple angles and distances. This early success laid the foundation for more advanced image recognition tasks later in development.

After assembling the UAV from the provided kit, we identified structural issues with the drone's landing gear. The original legs attached to the frame caused instability during landing, resulting in excessive bouncing and inconsistent touchdowns. To address this, we removed the legs and implemented custom 1-inch foam padding on the base of the drone. This modification significantly

improved landing stability, providing a soft and controlled descent that minimized hardware stress and risk of damage.

Once the UAV structure was optimized, the Raspberry Pi was mounted and connected to the Pixhawk flight controller. This step was essential for integrating the image recognition and flight control systems. We conducted power and signal testing between the two components, confirming stable communication and responsiveness during test sequences.

Further testing of the Arducam-based detection algorithm demonstrated the camera's ability to identify ArUco markers from a distance of up to 30 feet. This range exceeded expectations and validated the camera's placement, resolution, and field-of-view settings. These implementation milestones marked significant progress toward building a fully functional UAV platform, capable of autonomous image detection and GPS communication with the UGV.

## Conclusions

The Rescuer project successfully developed an autonomous UAV system capable of detecting visual targets, capturing their geolocation, and relaying this data to a UGV for coordinated response. The system demonstrated consistent performance in recognizing ArUco markers and executing GPS-based communication between the aerial and ground units. This approach significantly reduces the time, labor, and uncertainty involved in manual search-and-rescue efforts, offering strong potential for broader real-world applications.

Looking ahead, future development will focus on expanding the UAV's capabilities from marker-based recognition to full image processing and object detection, enabling the system to identify specific animals based on attributes such as size, color, or breed. The current software supports autonomous flight, marker detection, and GPS transmission; continued enhancements will integrate and refine these components into a seamless end-to-end system. Ultimately, the goal is to enable the UGV to autonomously navigate to the UAV-detected coordinates, completing a fully autonomous and responsive search-and-recovery solution.

## References

-5 References?