An Autonomous Mission Framework for Quadrotor-Based Target Detection, Localization, and Inspection in Indoor Environments

Ahmed Ashry*, Zachary Bortoff†, and Derek A. Paley‡
University of Maryland (UMD), College Park, MD, 20742

This paper presents a framework that enables a quadrotor to systematically search, detect, localize, and inspect targets in GPS-denied, indoor environments. It introduces an autonomous inspection routine that navigates the vehicle to a sequence of close-range viewpoints around targets without manual intervention. The system combines a deterministic search strategy with vision-based navigation, localization, and a custom-trained real-time object detector, all running within the computational constraints of an onboard processor. The mission pipeline transitions between global search and inspection, empirically validated across multiple target layouts and mission scenarios. Controlled hardware trials in a netted arena confirm complete area coverage, successful target inspection, and fully autonomous operation. This work aims to advance the practical deployment of autonomous Unmanned Aerial Vehicles (UAVs) for first-responder search and rescue (SAR) applications where rapid assessment of inaccessible areas is critical.

I. Introduction

Natural disasters and industrial accidents frequently leave survivors trapped in cluttered, GPS-denied interiors where conventional first-responder tactics are slow and hazardous. The increasing frequency of natural disasters underscores the urgent need for automated, efficient, and safe Search and Rescue (SAR) solutions [1]. Research over the past decade has shown that light multirotor unmanned aerial vehicles (UAVs) have great potential in SAR operations, where it can slip through narrow apertures, stream videos in real-time for situational awareness, and even deliver medical payloads, reducing responder exposure and response time [2–5].

However, the practical deployment of UAVs in SAR operations, particularly in GPS-denied and cluttered indoor environments, presents significant challenges, where navigation, target detection, and inspection remain critical issues [6]. Vision-aided navigation drifts in texture-poor corridors, autonomous search routines stall near obstacles, and onboard processors struggle to fuse mapping, localization, and object detection at real-time rates. UAVs must autonomously explore unknown environments, localize targets, and conduct detailed inspections under constrained conditions [7, 8].

Most research efforts has focused on UAV applications in outdoor SAR missions that leverage large-scale area coverage and high-altitude imaging [9]. The community has then focused more towards enhancing UAV autonomy in more constrained environments (e.g., GPS-denied) by integrating motion planning, obstacle avoidance, target localization, vision-based navigation, mapping, and autonomous close-range inspection on size, weight, and power (SWaP) limited vehicles. Sandino *et al.* [10, 11] illustrated the development of an autonomous navigation system for UAVs. Their core objective was to equip small UAVs with the ability to autonomously detect, localize, and quantify victims in disaster scenarios, leveraging a Partially Observable Markov Decision Process (POMDP) solved with an Adaptive Belief Tree (ABT) algorithm. However, the autonomous decision making under environmental uncertainty and the modeling of target detection uncertainties from vision-based sensors remain difficult to tune on embedded hardware. Other approaches addressed the problem of limited computational resources by pushing the limits of what could be run onboard. Shastry *et al.* [12] demonstrated real-time indoor 3D mapping, target detection, and coarse localization on a resource-constrained platform, yet still relied on human operators for controlling the UAV and for fine-grained visual inspection. Tordesillas *et al.* [13] achieved aggressive obstacle avoidance with the FASTER framework but delegated object recognition to ground stations to preserve compute headroom. Several other prototypes offer indoor navigation and inspection but depend on motion-capture systems or manual piloting for navigation and inspection.

*Ph.D. Student, Department of Aerospace Engineering, University of Maryland, College Park, MD, 20742
†Ph.D. Student, Department of Aerospace Engineering, University of Maryland, College Park, MD, 20742
‡Willis H. Young Jr. Professor of Aerospace Engineering Education, Department of Aerospace Engineering and Institute for Systems Research, University of Maryland, College Park, MD 20742. AIAA Senior Member

To address these challenges, this paper presents and experimentally validates a complete indoor SAR pipeline that runs entirely on a SWaP-limited quadrotor, and is capable of (1) implementing an autonomous indoor inspection routine for vision-based target navigation and localization; (2) integrating a custom-trained object detection model for real-time target identification, optimized for UAV onboard processing; and (3) unifying these components in a comprehensive mission framework validated through multiple controlled hardware testing.

The remainder of the paper is organized as follows. Section II formulates the mission objectives and performance metrics. Section III reviews navigation and software foundations relevant to UAV-based indoor SAR projects. Section IV details the system architecture and implementation details. Section V reports experimental results and limitations and Section VI provides conclusions and outlines future extensions on this work.

II. Problem Statement

Consider a confined, GPS-denied indoor environment $\mathcal{E} \subset \mathbb{R}^3$ whose floor plan is approximately rectangular. Within $\mathcal{E}$ lie M unknown *targets** each fitted with a unique fiducial marker and acuity vision labels (e.g., Landolt labels) fixed to the target base. The test site is a netted flight arena measuring $3.6\,\text{m} \times 4.8\,\text{m}$ with a clear height of 4.5 m, equipped with a Vicon system that provides ground-truth trajectories for evaluation only.

A quadrotor equipped with a tilted downward-facing tracking camera, a flight controller, and a companion computer seeks to autonomously perform the following functions:

- **P1) Search**: Systematically traverse $\mathcal{E}$ following a coverage path-planning (CPP) strategy, so that every planar cell of a chosen grid resolution is observed;
- **P2) Detect & Localize**: Identify each fiducial marker, estimate its 3D pose, and report the corresponding target location in the local inertial frame with root-mean-square error $\varepsilon_{\text{loc}} \leq 5\,\text{cm}$, a limit derived from empirical tag-pose dispersion in preliminary flights;
- **P3) Inspect**: Navigate to a sequence of pre-recorded offset poses around each target and acquire closer inspection images until either (i) the running mean detection confidence of the onboard neural network exceeds 85 % over at least $N_{\min} = 50$ inferences or (ii) a maximum dwell time $T_{\max} = 20\,\text{s}$ elapses, in which case the location is flagged as an anomaly; and
- **P4) Report**: Return a mission log containing the set of localized targets $\{\mathbf{p}_i\}_{i=1}^{M}$, their image-based label classifications, and any anomaly markers for human follow-up.

The vehicle is constrained by (i) limited onboard compute and battery resources typical of SWaP-restricted platforms; (ii) sensor range and resolution that bounds the vision-based navigation and the detections and (iii) dynamic feasibility requirements that cap linear velocity and angular rates commanded to the PX4 flight controller.

Formally, let $\mathbf{x}(t) \in SE(3)$ denote the quadrotor state. The objective is to generate a complete, feasible trajectory $\mathbf{x}^*(t)$ over the mission horizon $[0,\, T_{\text{f}}]$ such that

$$
\begin{aligned}
&\text{(Coverage)} \quad C(\mathbf{x}^*) = 1, \\
&\text{(Localization)} \quad \|\hat{\mathbf{p}}_i - \mathbf{p}_i^{\text{GT}}\|_2 \leq \varepsilon_{\text{loc}}\ ,\ \forall i = 1, \ldots, M, \\
&\text{(Inspection)} \quad \bar{\mathcal{P}}_i \geq 0.85\ \vee\ t_i \geq T_{\max},
\end{aligned}
$$

where $C(\cdot)$ is a binary coverage metric, $\hat{\mathbf{p}}_i$ the estimated target position, $\mathbf{p}_i^{\text{GT}}$ its ground truth, $\bar{\mathcal{P}}_i$ the running mean detection confidence, and t_i the inspection dwell time. A solution that satisfies P1–P4 under these constraints constitutes successful completion of the indoor mission defined in this study. The remainder of the paper describes the algorithms and hardware that achieve this specification and evaluates their performance.

III. Background

A. Indoor Navigation and Localization

Autonomous UAV navigation in GPS-denied environments presents significant challenges due to the absence of absolute positioning information. Conventional UAVs rely on Global Positioning System (GPS) signals for localization, which are unavailable in indoor settings, underground spaces, and dense urban environments [14]. Therefore, alternative

*By targets, we mean bucket assemblies specified by the National Institute of Standards and Technology (NIST). More details may be found in Sec. IV.C.1.

localization and navigation methods are required for effective UAV operation in such conditions. Without satellite fixes, a UAV must estimate its six-degree-of-freedom state $\mathbf{x}(t) \in SE(3)$ from onboard sensors. Visual–Inertial Odometry (VIO) fuses high-rate inertial measurements with image features to bound drift along all axes [15]. VIO is advantageous in environments rich in visual features, and its relative simplicity allows for real-time estimation in such environments. Extended Kalman Filter (EKF) is a statistical approach that estimates the state of a dynamic system from a series of incomplete and noisy measurements. The PX4 flight stack supplies an EKF2 instance that blends accelerometer, gyroscope, and visual data [16]. VIO poses are injected with other sensory data, constraining low-frequency drift and enabling smooth position-hold even when visual tracks drop temporarily. Fiducial markers, such as AprilTags, serve as reliable landmarks for precise UAV localization. These markers provide high-contrast patterns that can be detected and decoded by onboard vision sensors [17]. AprilTags enable accurate position estimation by computing their relative pose with respect to the UAV. This makes them particularly useful for tasks requiring precise indoor navigation and localization.

B. Software Infrastructure for Autonomous Drones

Robot Operating System (ROS) is an open-source framework widely adopted in autonomous robotics research. It provides a modular architecture for distributed computation, facilitating the development and integration of UAV navigation, perception, and control systems [18]. The system architecture consists of multiple nodes communicating asynchronously, which enables seamless data processing and control execution. Micro Air Vehicle Link (MAVLink) is a lightweight messaging protocol designed for real-time communication between UAV autopilots and companion computers. MAVROS, an extension of MAVLink within ROS, bridges ROS-based high-level autonomy algorithms with PX4-based UAV flight controllers. This enables waypoint navigation, sensor data streaming, and control command execution [19]. Software-in-the-Loop (SITL) Simulation is a crucial tool for validating UAV autonomy algorithms in a risk-free virtual environment. It enables UAV flight software to interact with a simulated world, facilitating pre-flight testing of control strategies, state estimation algorithms, and perception pipelines [20]. PX4 SITL integrates with Gazebo, which allows for realistic simulation of UAV dynamics, sensor feedback, and environmental interactions.

IV. System Architecture & Methodology

A. System Overview

The proposed system enables a UAV to autonomously search for and inspect targets in an indoor environment. The system architecture, illustrated in Fig. 1, consists of three primary subsystems: Perception, Planning, and Vehicle Control. Each subsystem is responsible for specific tasks that collectively ensure full autonomy in search and inspection operations.

The *Perception System* processes sensor data to estimate the UAV's state, detect targets, and identify visual markers. It incorporates a state estimation module that fuses inertial and vision-based measurements for localization, an AprilTag detection module for fiducial marker recognition and pose estimation, and an object detection module that autonomously identifies vision markers inside target locations.

The *Planning System* governs the UAV's decision-making, switching between search and inspection modes. The search phase executes a predefined lawnmower coverage strategy to ensure complete area coverage, while the inspection phase autonomously navigates to optimal viewpoints for analyzing detected targets.

The *Vehicle Control System* executes the planned waypoints using a PX4-based flight controller, ensuring precise trajectory following. The integration of these modules enables fully autonomous navigation, target detection, and structured inspections without manual piloting.

B. Autonomous Coverage Strategy

The UAV autonomously executes a predefined search strategy to systematically cover the target environment while detecting objects of interest. The selected approach follows a lawnmower coverage pattern, ensuring full area traversal with minimal redundancy. This strategy is widely used due to its efficiency in structured environments with known boundaries [21]. Compared to probabilistic approaches such as random walks, the lawnmower pattern follows a back-and-forth sweeping motion, which offers a deterministic and exhaustive search method and reduces the likelihood of missing potential targets [22, 23].

Waypoint generation for the search pattern is based on user-defined parameters, including area dimensions, desired

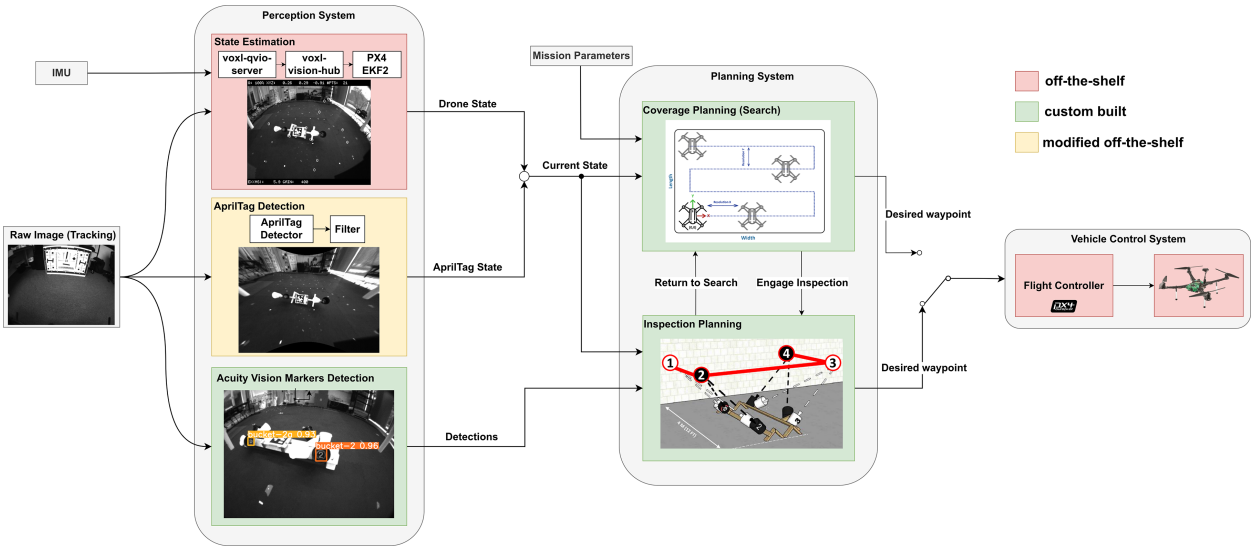


Fig. 1 System architecture illustrating the integration of perception, planning, and control subsystems.

flight altitude, and camera resolution along each axis. These parameters determine the spacing between adjacent passes, balancing the trade-off between coverage efficiency and localization accuracy. The UAV follows a sequence of parallel waypoints, reversing direction at each boundary to maximize area coverage. The system computes waypoints by partitioning the search area into a structured grid. Each waypoint represents a predefined pose consisting of position and orientation setpoints. These waypoints are published as ROS messages and transmitted via `MAVROS` to the PX4 flight controller, where they are converted from the East-North-Up (ENU) coordinate frame to the North-East-Down (NED) frame used by PX4. In this project, the UAV maintains a constant heading throughout the search mission, following a fixed yaw angle to ensure a consistent field of view. The execution process follows a state-based logic, as illustrated in Fig. 2. The UAV continuously monitors its real-time position, comparing its current pose—obtained from the onboard state estimation module—against the active waypoint. When the UAV reaches a waypoint within a user-defined tolerance threshold, it transitions to the next target in the sequence. This iterative process continues until all waypoints are traversed, at which point the UAV either returns to the home position or transitions to the next phase of the mission.

C. Target Detection, Localization, and Identification

1. Target Specifications

The experimental targets are based on standardized configurations set by NIST for UAV-based first responder applications. These targets consist of different bucket assemblies, designed to evaluate the drone sensors' acuity and ability to detect objects in constrained environments. The bucket alignments vary in configuration, with vision acuity markers placed inside them as distinguishing labels. These labels are critical for evaluating sensor acuity and serve as the primary objects of interest for the onboard object detection pipeline. NIST's intention with these targets is to establish a standardized method for testing and evaluating robotic platforms in emergency situations. We choose these targets to evaluate our project's mission.

Figure 3a illustrates a bucket alignment used in a confined test scenario, depicting a collapsed and confined environments. The figure highlights the bucket alignment amidst crushed cars and semi-collapsed structures, showcasing the aerial vehicle's potential in such challenging environments. Figure 3b shows a dual bucket alignment, with a closer look inside. These small printed markers challenge the acuity of the onboard sensors in various lighting conditions and the ability to closely inspect target. The ease of replicating these acuity vision labels is what justifies their use in testing our autonomous target inspection system. Figure 4 presents a model of one such bucket alignment, alongside an actual alignment used in our laboratory tests. There are various configurations, differing in bucket placement, label designs, and orientations, which makes it necessary that the autonomous agent not only localizes targets within an area but also identifies and distinguishes between different targets. This aspect will be explored in greater detail in the following sections.

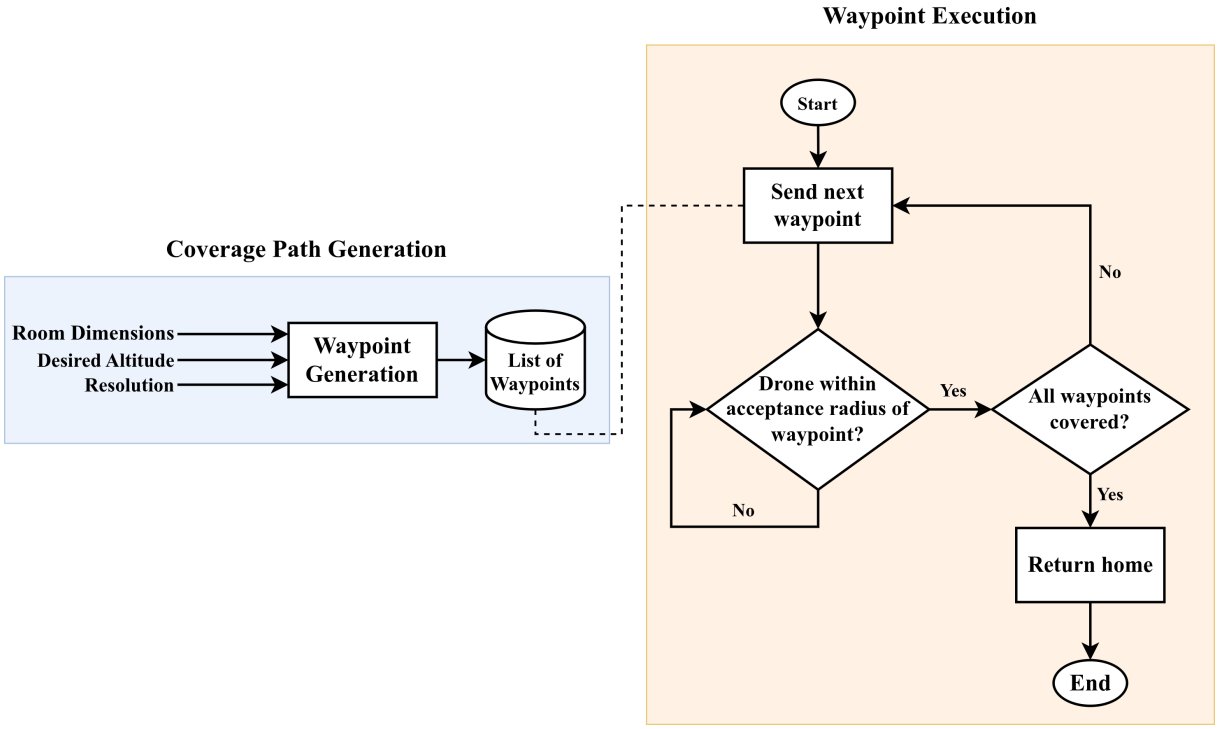
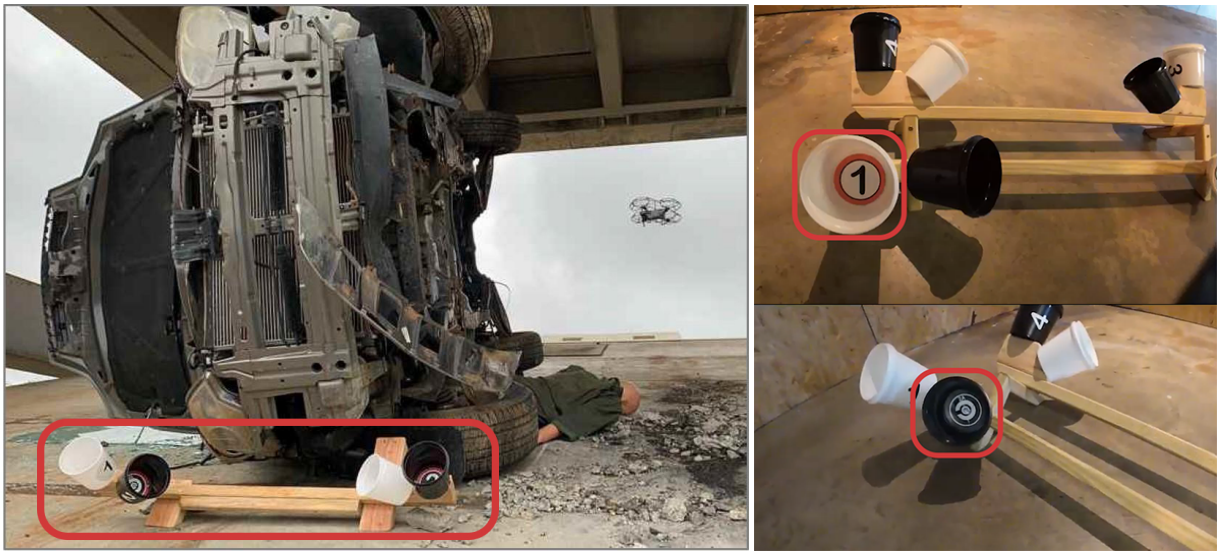


Fig. 2 Flowchart illustrating the search waypoint generation and execution process.



(a) Bucket alignment in a disaster scenario **(b) Acuity vision labels**

Fig. 3 Closer look at the chosen target alignments used for testing (adapted from NIST website [24]).

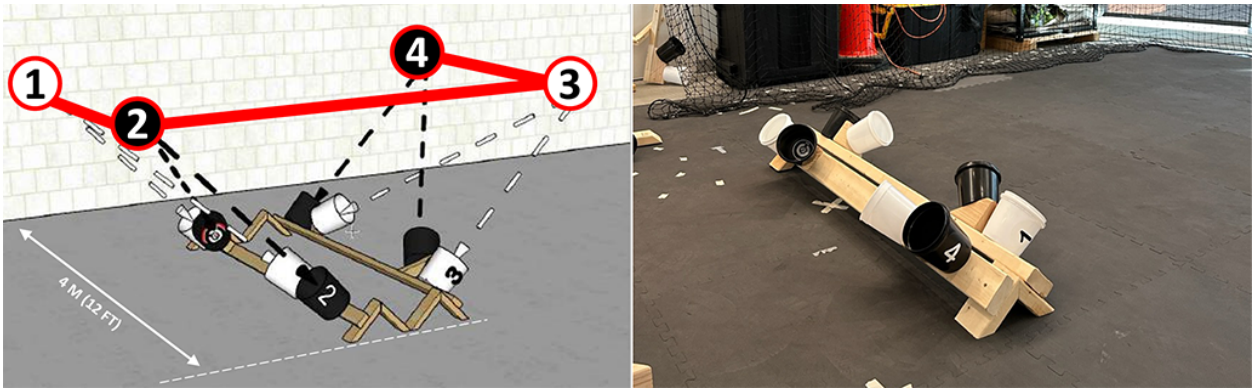


Fig. 4 Model vs. actual configuration of target alignment - Left: a graphical representation of "Ground" bucket alignment (adapted from NIST website [24]). Right: the corresponding real-world laboratory setup used for mission testing.

2. Target Localization

To obtain precise target localization, the UAV utilizes AprilTags, a widely adopted fiducial marker system in robotic applications due to its robustness against variations in lighting and scale [17]. These markers provide unique IDs and facilitate accurate pose estimation through computer vision processing. The AprilTag detection algorithm processes images captured by the UAV's downward-facing tracking camera. The detected tag's pose is estimated relative to the camera frame. Pose refers to a 3D pose encompassing both the position and orientation of an object. Throughout this paper, poses are represented as homogeneous transformations expressed as 4×4 matrices that combine rotation and translation. The standard form is

$$ {}^{A}\mathbf{T}^{B} = \begin{bmatrix} {}^{A}\mathbf{R}^{B} & \mathbf{p}_{B} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}, $$

where ${}^{A}\mathbf{T}^{B}$ denotes the pose of frame B with respect to frame A, ${}^{A}\mathbf{R}^{B} \in SO(3)$ is the rotation matrix, and $\mathbf{p}_{B} \in \mathbb{R}^{3}$ is the position vector expressed in frame B.

The detected tag pose provides the transform ${}^{C}\mathbf{T}^{A}$ from the tag frame A to the camera frame C. Because the tracking camera is rigidly mounted, its static transform with respect to the UAV body frame, ${}^{B}\mathbf{T}^{C}$, is known from calibration and hardware specification. The UAV's pose in the local inertial frame, ${}^{I}\mathbf{T}^{B}$, is delivered by the onboard state estimation system (see Fig. 5 for the different coordinate frames used). Concatenating these transforms yields the absolute pose of the tag in the inertial frame:

$$ {}^{I}\mathbf{T}^{A} = {}^{I}\mathbf{T}^{B}\,{}^{B}\mathbf{T}^{C}\,{}^{C}\mathbf{T}^{A}. $$

For computational efficiency, the system is designed to bypass processing of five consecutive frames, focusing instead on every sixth frame. This frequency is adjustable to suit specific requirements, acknowledging that continuous processing of every frame is generally unnecessary and would excessively burden the CPU. Moreover, to mitigate noise and ensure reliability, the system applies a filtering technique to refine localization accuracy. When a tag is detected, the UAV hovers briefly, collecting multiple pose estimates over a short window. These estimates are then averaged, rejecting outliers beyond a predefined threshold. This stabilization enhances consistency and improves subsequent mission phases such as inspection.

In our experiments, AprilTags, each encoded with a unique ID, are attached to bucket assemblies to differentiate targets during the search phase, as depicted in Fig. 6. It demonstrates the three principal target alignments tested: Target 0 (T0), Target 1 (T1), and Target 2 (T2), as shown from left to right.

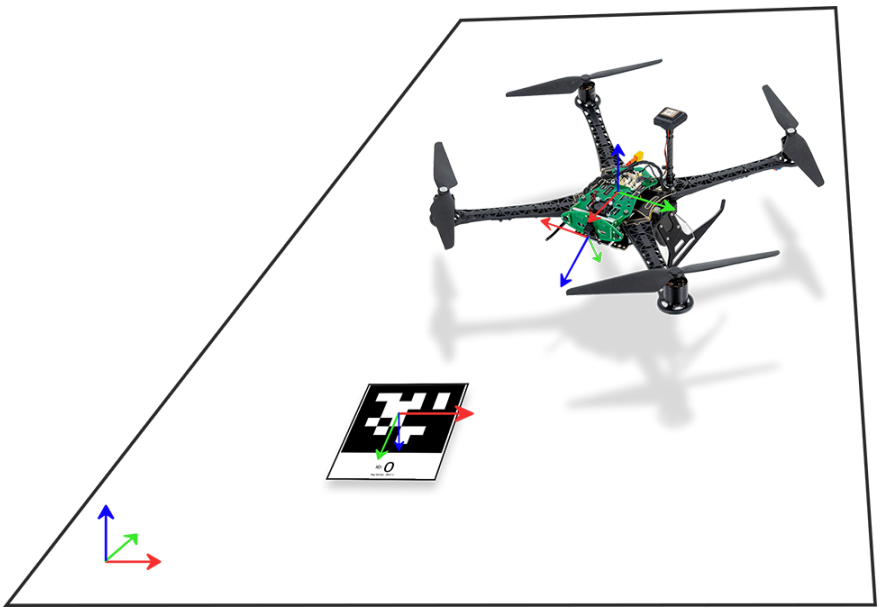


Fig. 5 Visualization of the various reference frames used during the mission.

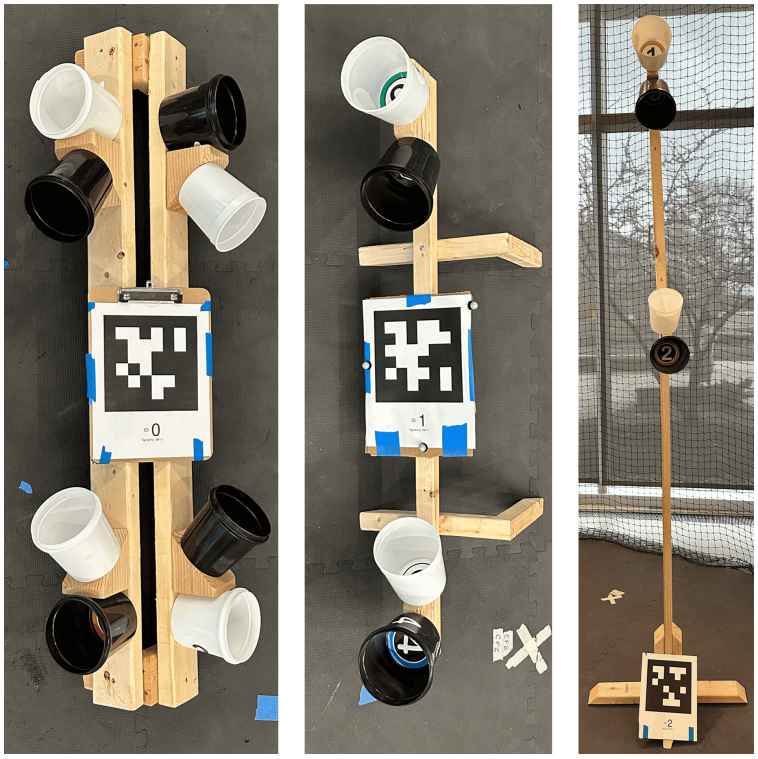


Fig. 6 Multiple target identification with different tag IDs. From left to right: Target 0 (T0), Target 1 (T1), Target 2 (T2).

3. Object Detection Integration

To test a complete mission with target detection and visual inspection of closer details, a deep learning-based object detection model is deployed to detect and classify the vision acuity markers inside the buckets. In our pipeline this capability is delivered by You Only Look Once (YOLO), which is a real-time object detection model known for its speed and accuracy [25]. In this project, YOLOv5s is used due to its lightweight architecture, which is well-suited for deployment on the UAV's onboard companion computer.

The model was trained on a custom dataset of images capturing vision acuity markers from various angles and lighting conditions. Image data were collected with the downward–facing tracking camera while the drone was flown and hand-held over the bucket configurations. ROS bag files were recorded under different laboratory lighting, yielding 1511 RGB frames at 1280×720 resolution. Eight marker classes—four alphanumeric vision-acuity labels and four Landolt rings—were annotated in Roboflow. To increase invariance to pose and illumination, data augmentation injected up to 1.96 % Gaussian noise, random crops of 0 % to 28 % of the field of view, rotations in $\pm 20°$, and brightness shifts of $\pm 17\%$. Augmentation expanded the dataset to 2626 images, which were split 85/13/2% into training, validation, and held-out test sets (2230/354/42 samples respectively) as shown in Fig. 7.

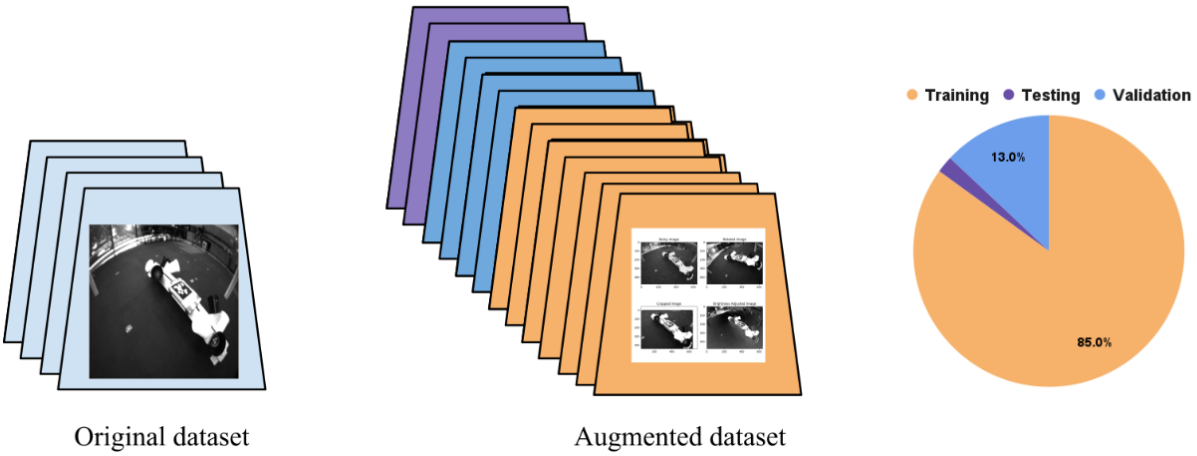


Fig. 7 2x Augmentation of original dataset from 1511 to 2626. Division into Training, Testing and Validation.

Transfer learning from the original COCO-trained YOLOv5s weights was carried out on Google Colab (Tesla T4, PyTorch 2.2.1). Images were down-sampled to 416×416 pixels; batches of 16 were used for 150 epochs with the default SGD optimizer (initial learning rate 0.01, cosine decay). Training and validation losses converged smoothly, and precision/recall stabilized above 0.95 after epoch 80.

During operation, the object detection pipeline processes images from the UAV's tracking camera and identifies specific labels affixed to the interior of the bucket targets. The model outputs bounding boxes, class labels, and confidence scores, which are used to distinguish between different target categories. Before flashing the detector to the UAV onbaord computer, the best checkpoint was exported and evaluated offboard first. Streaming the tracking camera topic over Wi-Fi to the ground station incurred a round-trip latency of 175.47 ms, but verified that the ROS-wrapped object detection inferred class IDs, confidence scores, and pixel coordinates correctly.

The decisive step was migrating inference to the VOXL1 Snapdragon 821 SoC onboard computer. The model was converted to TensorFlow-Lite, quantized post-training, and flashed. With GPU delegation enabled, the average inference time fell to 71.3 ms, a 2.5x speed-up over offboard processing without saturating the VOXL thermal budget. Figure 8 contrasts onboard and offboard detection execution time. Onboard inference provides real-time detections that are synchronized with the UAV's actual location, whereas offboard detections, executed on the ground station, experience latency. During flight the detection results are published as ROS messages, to which the inspection state-machine subscribes.

Fig. 8 Onboard vs. offboard inference delay. The UAV's actual position (left) is compared to the object detection running on the ground station (offboard) and on the drone's companion computer (onboard).

D. Autonomous Inspection Routine

Once a target (i.e., bucket assembly) is detected and localized in the local inertial frame (Sec. IV.C.2), the mission transitions from global coverage to close-range visual inspection to expose the labels attached to the bucket base to the onboard object detector. This task is formalized as a deterministic pose-offset problem that places the tracking camera at vantage points from which every bucket interior is clearly visible.

During system calibration, the operator manually station-keeps the quadrotor so that the tracking camera sees directly into each bucket aperture. For every satisfactory viewpoint, the AprilTag affixed to the target remains visible, which allows the onboard AprilTag detector to output a precise relative pose ${}^{C}\mathbf{T}^{A}$ (tag frame A with respect to the camera frame C). Then, using the static transform ${}^{B}\mathbf{T}^{C}$, a helper program converts this into the body-frame representation

$$ {}^{B}\mathbf{T}^{A} = {}^{B}\mathbf{T}^{C}\ {}^{C}\mathbf{T}^{A}. \tag{1} $$

When the operator presses a key, the desired offset body pose with respect to the AprilTag,

$$ {}^{A}\mathbf{T}^{B^\star} := \left({}^{B}\mathbf{T}^{A}\right)^{-1}, $$

is written to a JSON file keyed by the respective tag's `TagID`. For each target, we have pre-defined K desired inspection views recorded as

$$ {}^{A}\mathbf{T}^{B_k^\star} = \begin{bmatrix} \mathbf{R}_z(\psi_k^\star) & \mathbf{p}_k^\star \\ \mathbf{0}^\top & 1 \end{bmatrix}, \qquad k = 1, \ldots, K, $$

where $\mathbf{p}_k^\star$ offsets the body by the required distance from the label k and $\psi_k^\star$ aligns its optical axis with that label's normal. The number of K offsets relies on the particular bucket assembly detected (recall Fig. 6).

Given ${}^{I}\mathbf{T}^{A}(t)$–the live tag pose obtained exactly as in Sec. IV.C.2–the inertial frame setpoint for view k is obtained by concatenation:

$$ {}^{I}\mathbf{T}^{B_k^\star}(t) = {}^{I}\mathbf{T}^{A}(t)\ {}^{A}\mathbf{T}^{B_k^\star}. \tag{2} $$

During flight, detection of an AprilTag triggers retrieval of the corresponding sequence of offset poses $\left\{{}^{A}\mathbf{T}^{B_k^\star}\right\}_{k=1}^{K}$. The system then computes the desired inertial set-point via (2). PX4's `OFFBOARD` mode accepts either an absolute yaw angle or a yaw rate. Following empirical tuning, the latter produced markedly steadier footage: abrupt angle steps often induced overshoot, whereas small, continuous yaw-rate commands yielded clean label imagery. Given the present yaw $\psi(t)$ and the desired yaw $\psi^\star(t)$, extracted from (2), a helper function computes the yaw rate

$$\dot{\psi}_{\text{cmd}} = \kappa \ \text{wrap}_{[-\pi,\pi)}\big(\psi^\star(t) - \psi(t)\big),$$

where wrap delivers the signed minimal angular difference. The gain κ is tuned empirically, where smooth convergence is achieved for κ in the range 0.4–0.6s^{-1}. The same `PositionTarget` message's three translational components are filled with the position extracted from ${}^{A}\mathbf{T}^{B_k^\star}$. Tolerance checks of $|\psi^\star - \psi| \leq 10^\circ$ and $\|\mathbf{p}^\star - \mathbf{p}\| \leq 5\text{cm}$ are applied to check whether a desired waypoint is achieved, and these threshold values are user-defined that may be adjusted per mission.

For each newly detected tag the inspection state-machine performs the following loop:

1) **Publish setpoint**: the first offset in the list is streamed to PX4 at 20Hz.
2) **Reach tolerance**: when the position-and-yaw errors fall beneath the thresholds stated earlier, the controller deems the viewpoint reached and starts a dwell timer.
3) **Evaluate stopping conditions**: while dwelling, the node maintains a running mean of the object detection confidences. If the mean exceeds 85% over at least $N_{\min} = 50$ inferences, that label is considered successfully inspected. Otherwise, if the dwell timer reaches $T_{\max} = 20\text{s}$, the label is flagged *anomalous*.
4) **Advance**: the next offset in the list is loaded and steps 1–3 repeat until all stored viewpoints for that target have been serviced.
5) **Mark inspected**: after the final offset is processed, the tag ID is entered into an "inspected" registry so that subsequent detections skip the inspection routine.

Because each offset is recorded and expressed in the tag frame, they remain valid even when identical bucket assemblies appear at arbitrary orientations throughout the arena. Adding new assemblies requires only re-running the viewpoints recording script.

E. Mission–Level Framework

Figure 9 condenses the logic that binds the coverage path planner (§IV.B), target localization (§IV.C.2), the inspection routine (§IV.D), and the onboard object detector (§IV.C.3) into one autonomous mission. The executive is realized as a ROS state machine with two principal modes; *Search* and *Inspect*. All transitions depend exclusively on information already introduced in earlier sections; what follows therefore focuses on *how* these modules interact.

At time t the system receives a combined state message $\mathcal{S}(t) = \left\{{}^{I}\mathbf{T}^{B}(t),\ {}^{C}\mathbf{T}^{A}(t),\ \bar{\mathcal{P}}(t),\ \mathcal{F}_{\text{timer}}(t)\right\}$, where ${}^{I}\mathbf{T}^{B}(t)$ is the current drone pose, ${}^{C}\mathbf{T}^{A}(t)$ the latest tag pose (or `NULL`), $\bar{\mathcal{P}}(t)$ the running mean object detector confidence during inspection, and $\mathcal{F}_{\text{timer}}(t)$ the elapsed time at the present inspection offset. The search subsystem uses only ${}^{I}\mathbf{T}^{B}$ to advance along the lawnmower path, whereas the inspection mode monitors $\bar{\mathcal{P}}$ and $\mathcal{F}_{\text{timer}}$ to decide when to move on and to enforce conditions stated in the problem statement II.

During *Search*, the program publishes waypoints w_i at 20Hz; PX4 executes it through `OFFBOARD`. Concurrently, the AprilTag node analyses every sixth camera frame (tunable) to recover ${}^{C}\mathbf{T}^{A}$. A non-`NULL` detection whose tag ID is *not* in the inspected registry triggers the *switch-to-Inspect* transition. Before switching, the executive pauses and hovers so the pose filter can average outliers and yield a filtered tag estimate within the acceptable threshold.

In *Inspect*, the node loads the tag-specific list of desired offsets from the JSON file saved offline and executes the loop detailed in Sec. IV.D. Two additional bookkeeping actions complete the mission-level integration: If the dwell timer reaches $T_{\max}$ before $\bar{\mathcal{P}} \geq 0.85$ for at least $N_{\min} = 50$ inference instances, the program stores the current inertial pose, the detector statistics, and a timestamp in an `anomaly.csv` file for human follow-up. Once all K offsets for the tag are inspected, the tag ID is appended to the inspected registry and the state machine reverts to *Search*. Because the coverage waypoints are numbered sequentially along each lawnmower strip, the planner simply resumes at the next search waypoint index w_{i+1}, which guarantees eventual coverage of $\mathcal{E}$ even when inspection interrupts the original path.

The mission terminates autonomously once the coverage metric $C(\cdot) = 1$ is satisfied with all search waypoints traversed, *and* every detected tag resides in the inspected registry, at which point the executive sends a `Return Home` command. Additionally, a watchdog monitors PX4's battery message; when remaining capacity drops below 25% the executive overrides all states and commands a straight-line return-to-launch.

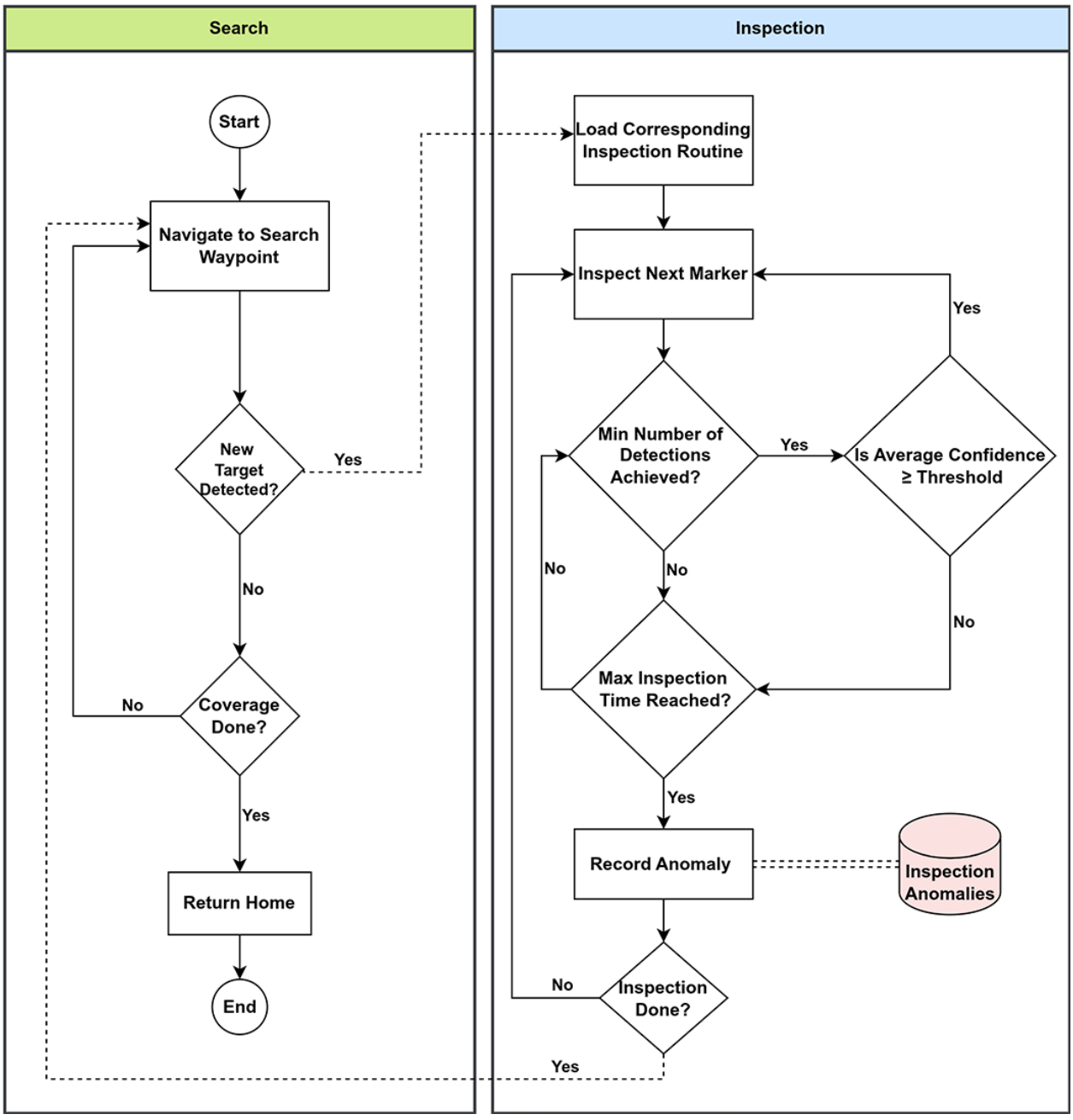


Fig. 9 Flowchart for the framework between search and inspection.

V. Experimental Evaluation

This section evaluates the implementation and performance of the methodologies described in Sec. IV. We begin by describing the hardware platform and experimental testbed, followed by evaluating individual system components, and conclude with a comprehensive assessment of the full integrated system.

A. Hardware Platform & Experimental Testbed

The UAV platform utilized in this study is the ModalAI M500, a quadcopter designed for autonomous indoor and outdoor operations [26]. It features the VOXL Flight Deck, which integrates a PX4-based flight controller with a Qualcomm Snapdragon-based companion computer, running a Linux environment for onboard computational tasks. The system includes a forward-facing high-resolution camera, a downward-facing tracking camera, and an integrated VIO system for precise state estimation.

Communication between the UAV and the ground station is achieved via Wi-Fi, which allows real-time command execution and telemetry monitoring. The modular architecture facilitates seamless integration with ROS to support algorithm development and validation. Figure 10 illustrates the key components of the ModalAI M500 UAV, including its sensor suite, propulsion system, and onboard processing unit.

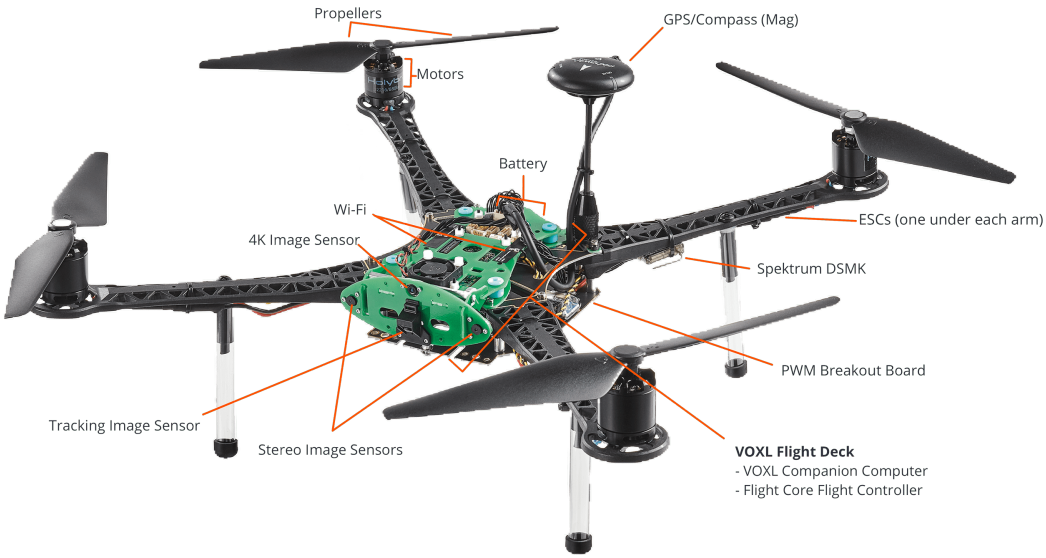


Fig. 10 ModalAI M500 UAV with labeled components (Adapted from ModalAI website).

B. Subsystem-Level Performance

In the *Search* phase, despite of the drone's onboard state estimation exhibiting minor discrepancies compared to the ground truth position recorded by the motion capture system, the actual flight path remains within the acceptable boundaries of the designated search area. The observed deviation from the planned trajectory is attributed to the known issue of drift in VIO-based navigation, which accumulates positional error over time.

AprilTag pose accuracy is pivotal since each inspection offset is defined in the tag frame. During experiments, the drone's movement, combined with motion blur and varying lighting conditions, initially created challenges in accurately determining the marker's pose from the first detection. Figure 11 compares the first pose estimate, the filtered estimate obtained after hovering to collect 50 samples and remove outliers, and ground truth. The drone's first estimation of the tag's position showed a discrepancy of approximately 5cm from the ground truth. Filtering reduces the translational root-mean-square error from 5.2 cm to 2.4 cm, meeting the $\varepsilon_{\text{loc}} = 5$ cm requirement stated in Sec. II.

To further illustrate the relationship between tag localization quality and inspection performance, Fig. 12 plots

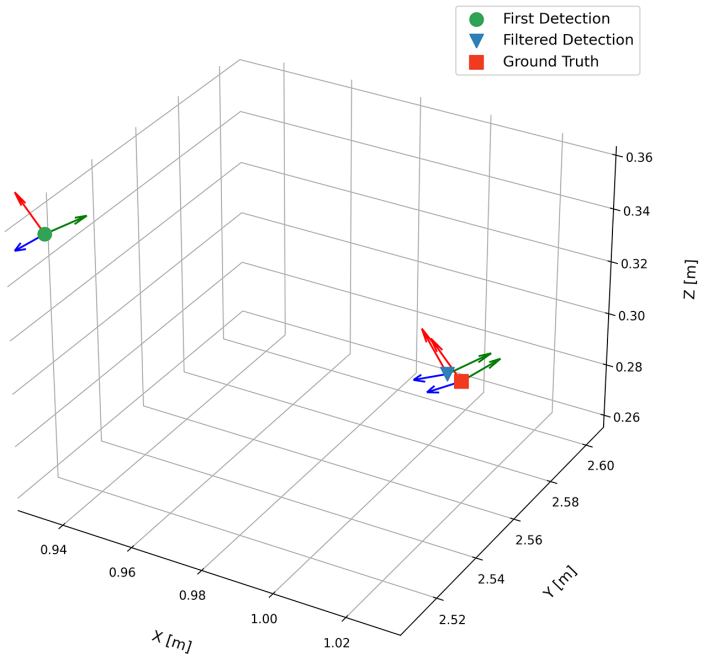


Fig. 11 3D AprilTag pose comparison between first estimate, filtered estimate, and ground truth.

detector confidence $\mathcal{P}$ over inspection time for the eight buckets of Target 0 (T0[†]). The acceptance threshold for the average confidence 85% is shown as a dashed line with a minimum required number of readings set at 50 as detailed earlier. Considering the object detection model reports inferences at approximately 10Hz, it takes around 5 to 6 seconds to gather the necessary readings to evaluate against the confidence threshold.

Figure 12a shows inspection windows across the 8 buckets where detection confidence consistently exceeds the 85% threshold. This resulted in an even time allocation for each inspection window, averaging around 14 seconds. This timing includes approximately 5-6 seconds of inspection per bucket, followed by an orientation change to inspect the next bucket, and navigation to the next set of buckets for the proceeding inspection window. Buckets B2 and B2a occasionally fail the threshold and are flagged anomalous when the dwell timer reaches $T_{\max} = 20\,\text{s}$ as shown in Fig. 12b.

Figure 13 visually contrasts the inspection positioning for B2 with the filtered (a) vs the initial (b) tag pose estimates. The precision of localization directly impacts inspection effectiveness; the filtered estimate leads to a more centered and clear view of the bucket, whereas the initial estimate results in a less focused and consequently lower confidence inspection.

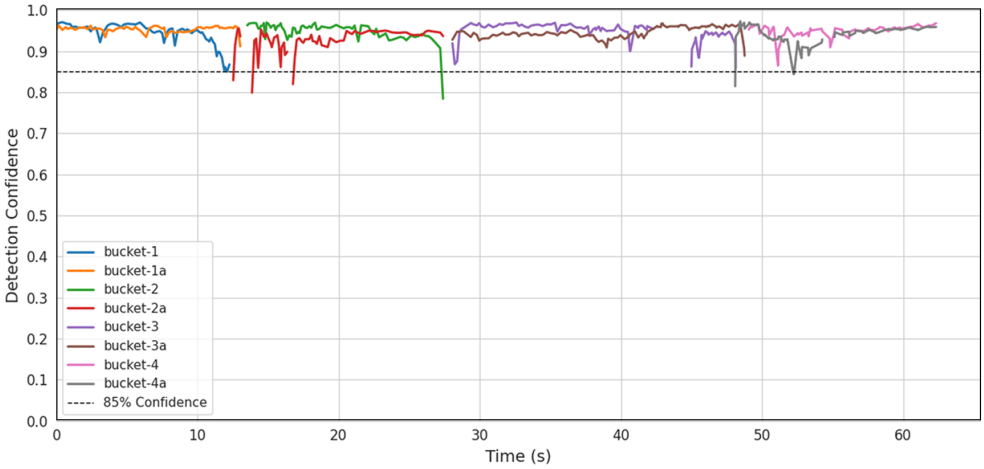
C. Full System Evaluation

To evaluate the integrated system performance, we tested the complete mission by placing three targets randomly in the testing arena for autonomous search and inspection. Figure 14 illustrates the system's functionality, showing the drone's actual trajectory in the lab's netted area alongside a 3D plot that demonstrates the transition between search and inspection phases. The 3D trajectory visualizes the drone's systematic movement from searching for targets to shifting into inspection mode upon target localization.

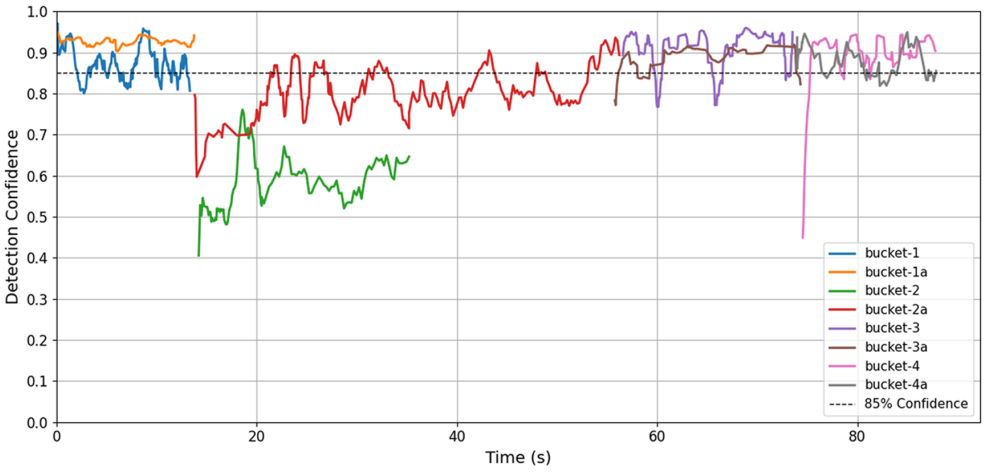
Figure 15 shows a residence-time heat-map that highlights prolonged dwells at anomalous buckets. Such insights can guide future mission planning, enabling more efficient path planning and resource allocation during actual deployments in SAR missions.

Figure 16 presents multiple mission scenarios with different target layouts and their corresponding executed 2D paths. This diversity in target placement provides a way to assess how the drone adapts its search and inspection strategies in varied configurations. Table 1 quantifies the outcomes of each mission, summarizing total mission time,

[†]Throughout the remainder of this section, 'T' will denote 'Target' (e.g., Target 1 as T1) and 'B' will denote 'bucket' (e.g., bucket 2a as B2a). For further details on each target's specifics, see Fig. 6.



(a)



(b)

Fig. 12 Detection confidence during inspection windows: (a) successful cases; (b) Bucket B2/B2a anomalies

(a) (b)

Fig. 13 Inspection framing comparison: filtered estimate (left) versus first AprilTag estimate (right).

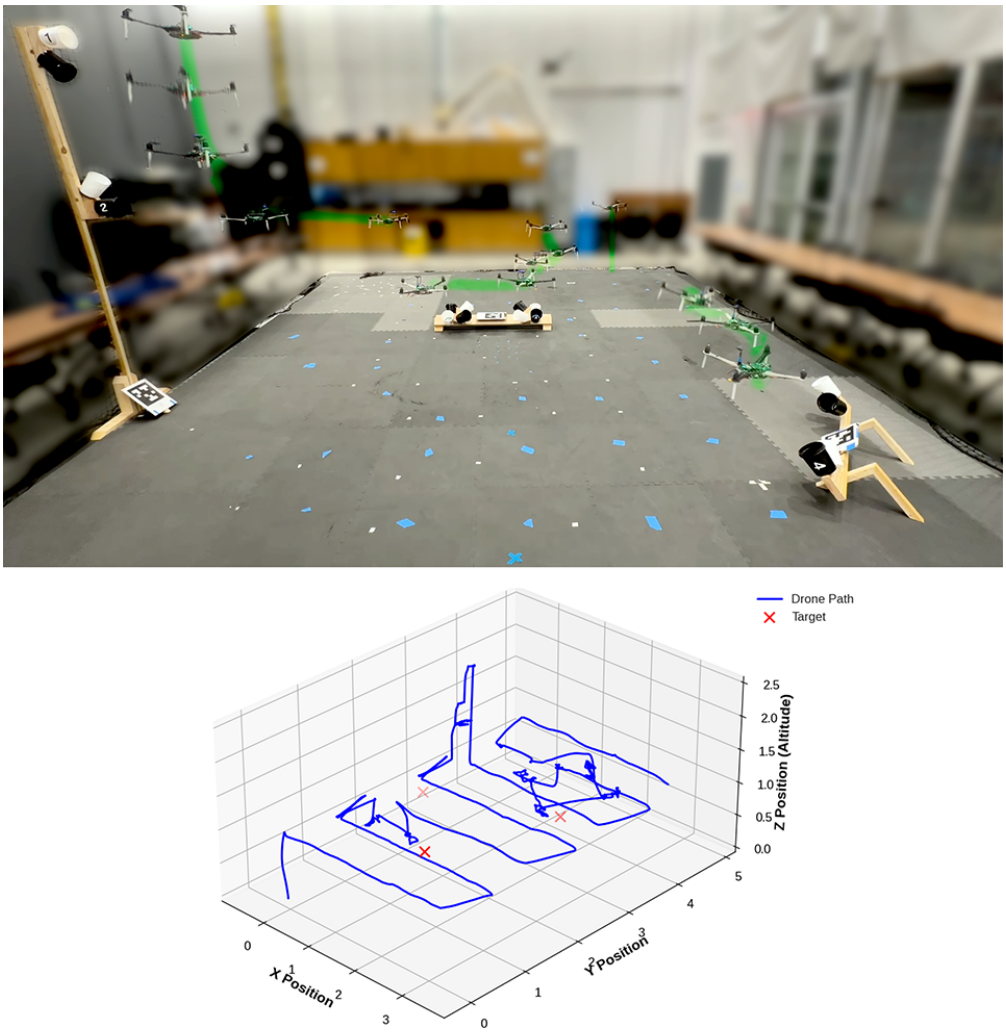


Fig. 14 Actual and 3D representation of the drone's complete mission trajectory - showing transition between search and inspection phases.

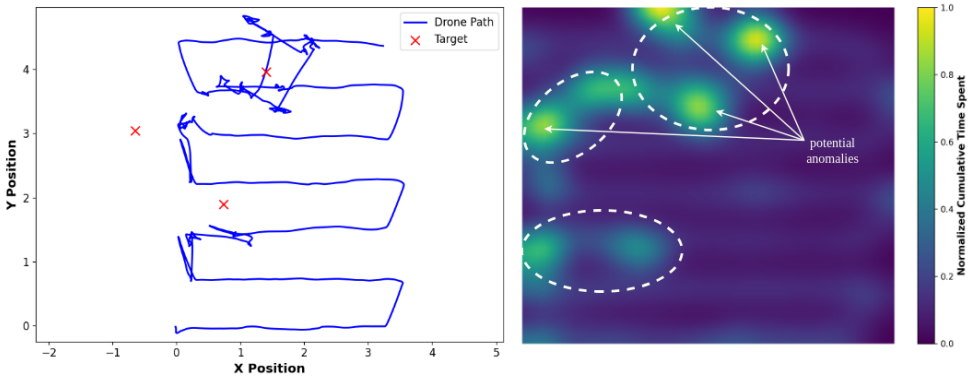


Fig. 15 Full mission 2D path with corresponding heatmap highlighting time spent in different areas and identifying potential anomalies.

inspection times per target, mean inspection time for each target across all four scenarios, number of detected markers that satisfied $\bar{\mathcal{P}} \geq 85\%$, and anomaly counts.

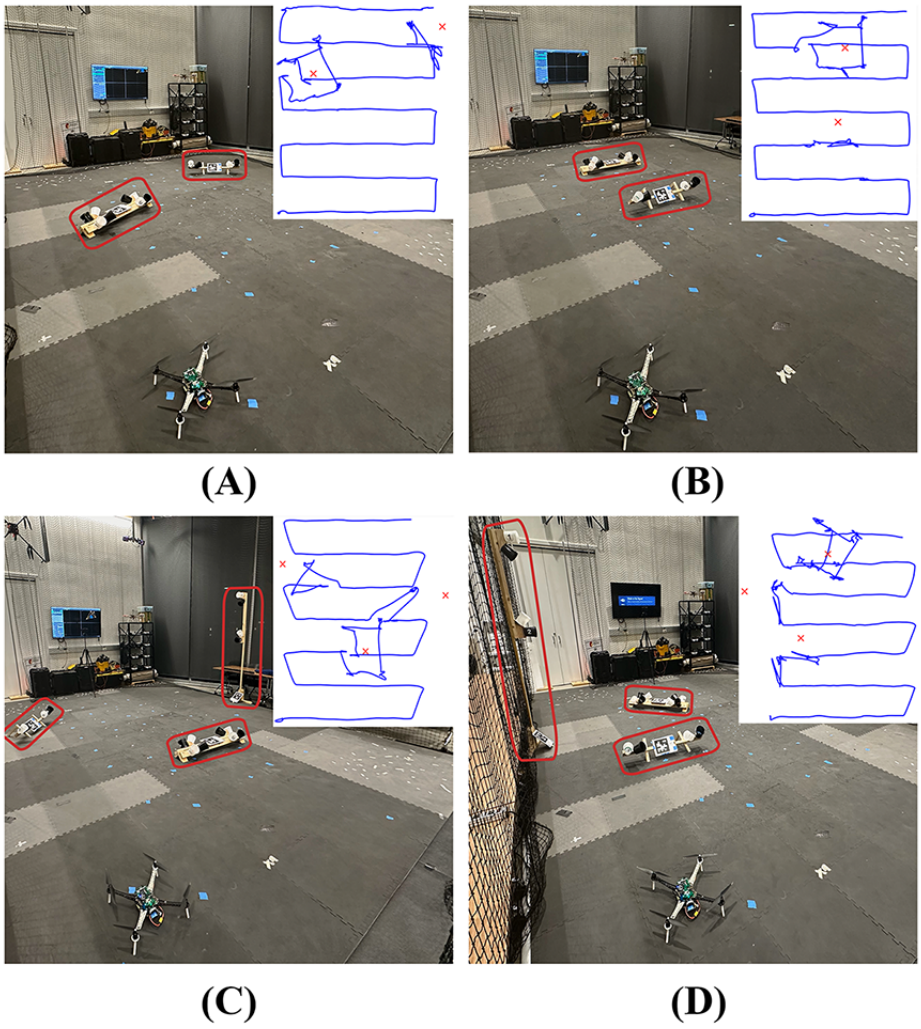


Fig. 16 Missions A–D showing different target layouts with their executed trajectories shown in 2D.

Target arrangement significantly influences mission outcomes. For mission scenarios with two targets, aligned targets in Mission B led to faster operational time (302.3 s) and zero anomalies, unlike Mission A which incurred three anomalies, suggesting environmental effects on system performance. The consistent detection accuracy across scenarios underscores the robustness of the detection model, though longer inspection times in some missions indicate potential for system refinement. Table 1 also reveals the effect of target orientation on inspection times. For T0, Mission B was anomaly-free with a total inspection time of 82.7 seconds. Conversely, Missions A and C each reported an anomaly in bucket B2, and Mission D reported two anomalies, extending the inspection duration to 113.61 seconds. T1 required more inspection time in Mission A, resulting in two anomalies, while other missions reported none. Inspection times for T2 in Missions C and D were comparable, with no anomalies noted. Target positioning provides additional context. Missions A and B, hosting the same targets but with varying inclinations, demonstrate how shadows affect detection—darker buckets like B2, B2a, B4, and B4a prove more challenging due to diminished visibility in shadowed areas. Across all missions, detection accuracy averaged 86.4% and all missions met the $C(\cdot) = 1$ coverage criterion without operator intervention.

Table 1 Tabulated results showing inspection times, target detection efficiency, and anomalies identified for Targets 0 to 2 (T0-T2), across multiple missions.

Mission ID	Mission Time	Target 0 (T0)		Target 1 (T1)		Target 2 (T2)		Markers Detected	Anomalies Identified
		Inspection Time	Mean	Inspection Time	Mean	Inspection Time	Mean		
A	317.53 s (5.29 min)	94.88 s (1.58 min)	95.93 s (1.60 min)	54.87 s (0.91 min)	40.89 s (0.68 min)	—	28.54 s (0.48 min)	9/12 (75%)	3 (B2 at T0, B4 & B4a at T1)
B	302.34 s (5.04 min)	82.70 s (1.38 min)		34.65 s (0.58 min)		—		12/12 (100%)	0
C	322.35 s (5.37 min)	92.54 s (1.54 min)		40.31 s (0.67 min)		29.48 s (0.49 min		14/15 (93.3%)	1 (B2 at T0)
D	330.37 s (5.51 min)	113.61 s (1.89 min)		33.72 s (0.56 min)		27.60 s (0.46 min)		13/15 (86.7%)	2 (B2 & B4 at T0)

VI. Conclusion

This paper has presented a framework for autonomous UAV operation in indoor SAR missions with a focus on reliable close range target inspection. A deterministic inspection routine has enabled the vehicle to align itself with fiducial-marked targets and acquire images from multiple viewpoints without manual intervention. A custom-trained object detection model is deployed on the companion computer for real time object recognition, which streamlines the inspection process and the overall mission flow. The system's performance was validated in repeated hardware trials under varied target layouts. Ongoing and future work focuses on expanding capability and robustness by adding additional sensing modalities such as depth, upgrading to a higher-resolution gimballed camera for extra degrees of freedom, and integrating autonomous exploration and 3D mapping to boost performance in larger, cluttered, GPS-denied environments.

References

[1] American Red Cross, "Drones in the RCRC — americanredcross.github.io," `https://americanredcross.github.io/rcrc-drones/index.html`, 2020.

[2] Álvarez García, C., Cámara-Anguita, S., López-Hens, J. M., Granero-Moya, N., López-Franco, M. D., María-Comino-Sanz, I., Sanz-Martos, S., and Pancorbo-Hidalgo, P. L., "Development of the aerial remote triage system using drones in mass casualty scenarios: A survey of international experts," *PLOS ONE*, Vol. 16, 2021, p. e0242947. https://doi.org/10.1371/JOURNAL.PONE.0242947.

[3] Huang, H. W., Chen, J., Chai, P. R., Ehmke, C., Rupp, P., Dadabhoy, F. Z., Feng, A., Li, C., Thomas, A. J., da Silva, M., Boyer, E. W., and Traverso, G., "Mobile robotic platform for contactless vital sign monitoring," *Cyborg and Bionic Systems*, Vol. 2022, 2022. https://doi.org/10.34133/2022/9780497.

[4] Dalamagkidis, K., Valavanis, K. P., and Piegl, L. A., *On integrating unmanned aircraft systems into the national airspace system: Issues, challenges, operational restrictions, certification, and recommendations*, Intelligent Systems, Control and Automation: Science and Engineering, Vol. 54, Springer Dordrecht, 2012. https://doi.org/10.1007/978-94-007-2479-2.

[5] Karaca, Y., Cicek, M., Tatli, O., Sahin, A., Pasli, S., Beser, M. F., and Turedi, S., "The potential use of unmanned aircraft systems (drones) in mountain search and rescue operations," *The American Journal of Emergency Medicine*, Vol. 36, 2018, pp. 583–588. https://doi.org/10.1016/J.AJEM.2017.09.025.

[6] Petrlík, M., Báča, T., Heřt, D., Vrba, M., Krajník, T., and Saska, M., "A robust UAV system for operations in a constrained environment," *IEEE Robotics and Automation Letters*, Vol. 5, No. 2, 2020, pp. 2169–2176. https://doi.org/10.1109/LRA.2020.2970980.

[7] Alhafnawi, M., Bany Salameh, H. A., Masadeh, A., Al-Obiedollah, H., Ayyash, M., El-Khazali, R., and Elgala, H., "A survey of indoor and outdoor UAV-based target tracking systems: Current status, challenges, technologies, and future directions," *IEEE Access*, Vol. 11, 2023, pp. 68324–68339. https://doi.org/10.1109/ACCESS.2023.3292302.

[8] Al-Naji, A., Perera, A. G., Mohammed, S. L., and Chahl, J., "Life signs detector using a drone in disaster zones," *Remote Sensing 2019, Vol. 11, Page 2441*, Vol. 11, 2019, p. 2441. https://doi.org/10.3390/RS11202441.

[9] Amrallah, A., Mohamed, E. M., Tran, G. K., and Sakaguchi, K., "UAV trajectory optimization in a post-disaster area using dual energy-aware bandits," *Sensors 2023, Vol. 23, Page 1402*, Vol. 23, 2023, p. 1402. https://doi.org/10.3390/S23031402.

[10] Sandino, J., Vanegas, F., Gonzalez, F., and Maire, F., "Autonomous UAV navigation for active perception of targets in uncertain and cluttered environments," *2020 IEEE Aerospace Conference*, 2020, pp. 1–12. https://doi.org/10.1109/AERO47225.2020.9172808.

[11] Sandino, J., Vanegas, F., Maire, F., Caccetta, P., Sanderson, C., and Gonzalez, F., "UAV framework for autonomous onboard navigation and people/object detection in cluttered indoor environments," *Remote Sensing 2020, Vol. 12, Page 3386*, Vol. 12, 2020, p. 3386. https://doi.org/10.3390/RS12203386.

[12] Shastry, A. K., Cui, W., Abdi, S. S., Poojari, S. S., Ashry, A., Wei, Q., Luterman, A., Ved, V., and Paley, D. A., "Indoor aerial 3D mapping and target localization with a custom-built multispectral visual-inertial sensor system," *AIAA Science and Technology Forum and Exposition, AIAA SciTech Forum 2025*, 2025. https://doi.org/10.2514/6.2025-1505.

[13] Tordesillas, J., Lopez, B. T., and How, J. P., "FASTER: fast and safe trajectory planner for flights in unknown environments," *IEEE International Conference on Intelligent Robots and Systems*, 2019, pp. 1934–1940. https://doi.org/10.1109/IROS40897.2019.8968021.

[14] Acharya, A. D., Bhandari, S., and Aliyazicioglu, Z., "Autonomous navigation of a quadcopter in indoor environment," *AIAA Scitech 2019 Forum*, 2019. https://doi.org/10.2514/6.2019-1057.

[15] Scaramuzza, D., and Zhang, Z., "Visual-inertial odometry of aerial robots," *CoRR*, Vol. abs/1906.03289, 2019. URL http://arxiv.org/abs/1906.03289.

[16] Driessen, S. P., Janssen, N. H., Wang, L., Palmer, J. L., and Nijmeijer, H., "Experimentally validated extended kalman filter for UAV state estimation using low-cost sensors," *IFAC-PapersOnLine*, Vol. 51, 2018, pp. 43–48. https://doi.org/10.1016/J.IFACOL.2018.09.088.

[17] Olson, E., "AprilTag: A robust and flexible visual fiducial system," *Proceedings - IEEE International Conference on Robotics and Automation*, 2011, pp. 3400–3407. https://doi.org/10.1109/ICRA.2011.5979561.

[18] Open Source Robotics Foundation, "Documentation - ROS Wiki — wiki.ros.org," https://wiki.ros.org/, 2025.

[19] MAVLink, "MAVLink Developer Guide | MAVLink Guide," https://mavlink.io/en/, 2025.

[20] PX4 Autopilot Developers, "Simulation | PX4 User Guide (v1.12) — docs.px4.io," https://docs.px4.io/v1.12/en/simulation/, 2025.

[21] Choset, H., and Pignon, P., "Coverage path planning: The boustrophedon cellular decomposition," *Field and Service Robotics*, 1998, pp. 203–209. https://doi.org/10.1007/978-1-4471-1273-0_32.

[22] Das, C., Becker, A., and Bretl, T., "Probably approximately correct coverage for robots with uncertainty," *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011, pp. 1160–1166. https://doi.org/10.1109/IROS.2011.6094695.

[23] Tan, C. S., Mohd-Mokhtar, R., and Arshad, M. R., "A comprehensive review of coverage path planning in robotics using classical and heuristic algorithms," *IEEE Access*, Vol. 9, 2021, pp. 119310–119342. https://doi.org/10.1109/ACCESS.2021.3108177.

[24] National Institute of Standards and Technology (NIST), "2022 First responder UAS indoor challenge," https://www.nist.gov/ctl/pscr/open-innovation-prize-challenges/past-prize-challenges/2022-first-responder-uas-indoor, Apr. 2022.

[25] Jocher, G., "Ultralytics YOLOv5," https://github.com/ultralytics/yolov5, 2020.

[26] ModalAI, "Home — docs.modalai.com," https://docs.modalai.com/, 2025.