Sensors & Transducers, Vol. 272, Issue 1, April 2026, pp. 13-22

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2026
http://www.sensorsportal.com

An End-to-End Autonomous UAV System for Disaster Response: Terrain-Aware Detection, Payload Delivery, and ROS-Gazebo Validation

*** Aryan SIRSAVKAR, Muzammil SHAIKH, Anuj KALE, Abhijeet THORE and Mayur PATIL**

Pune Vidyarthi Griha's College of Engineering, Technology & Management,
Department of Engineering, Vidyanagari, Pune – 411007, India
Tel.: +91 8010076404
* E-mail: aryansirsavkarc700@gmail.com

Received: 30 Jan. 2026 /Revised: 3 April 2026 /Accepted: 17 April 2026 /Published: 28 April 2026

Abstract: This article presents an extended study of a modular, cost-efficient Unmanned Aerial System (UAS) designed for fully autonomous disaster-response operations. Building upon prior conference work [12], this extension introduces a comprehensive Robot Operating System (ROS) and Gazebo-based simulation framework that validates the entire mission pipeline under realistic synthetic environments prior to field deployment. The system autonomously scans user-defined areas, detects disaster zones using a custom YOLOv8-based vision model, and executes payload drops without ground control intervention. An onboard sensor-fusion pipeline combines LiDAR range measurements, PX4 telemetry, and GPS data to reconstruct local terrain undulations, enabling low-altitude navigation with improved detection rates. Target coordinates are computed onboard via a heading-aware pixel-to-GPS conversion using a pinhole camera model and UAV yaw state. The ROS-Gazebo environment replicates sensor-plugin behavior, obstacle placement, and disaster-zone visual cues, enabling closed-loop validation of perception, navigation, and payload-delivery logic. All computations run on a Raspberry Pi 5, with pymavlink commanding the Pixhawk 4 flight controller. Extended system-level evaluations demonstrate a per-target detection rate of approximately 80 percent, successful payload delivery within a 5 m radius in 24 of 30 autonomous flights, 2 kg payload capacity, and 25-minute flight endurance. The ROS-Gazebo environment further demonstrates a measurable reduction in mission planning iteration time and an improvement in pipeline integration debugging speed compared to SITL-only workflows, based on developer observations across project iteration cycles.

Keywords: Autonomous UAS, Disaster response, ROS-Gazebo simulation, Terrain undulation mapping, YOLOv8 target detection, Payload delivery, LiDAR and telemetry fusion, Onboard navigation.

1. Introduction

Rapid disaster response increasingly relies on autonomous aerial systems capable of delivering payloads and performing reconnaissance with minimal or no ground support. Conventional solutions often depend on human operators for navigation, have limited payload capacity, or rely solely on GPS-based waypoint guidance. These limitations can significantly reduce operational accuracy and efficiency, especially in complex environments such as collapsed structures, densely forested areas, or urban flood zones.

Many existing UAVs also lack integrated perception systems, preventing adaptive navigation or real-time decision-making in dynamic disaster scenarios. To address these challenges, this work presents a modular Unmanned Aerial System (UAS) designed to autonomously detect disaster zones, navigate complex terrain, and execute precise payload delivery. The proposed UAS integrates a

YOLOv8-derived onboard vision system [2], LiDAR-based terrain mapping, and PX4 telemetry to enable robust autonomous operations without human intervention. YOLOv8 was selected as the detection backbone owing to its favourable accuracy-latency trade-off on embedded hardware and its established use in UAV-based aerial imagery tasks [13].

The conference version of this work [12] described the core hardware architecture and presented initial SITL and field validation results. Specifically, the conference paper covered: the airframe design and hardware bill of materials (Table 1); the YOLOv8n training procedure and per-class metrics (Table 2); the heading-aware pixel-to-GPS localization pipeline (Equations (1–3)); the lawnmower scanning strategy; and preliminary field results from 10 autonomous flights. This extended journal article makes the following additional contributions that do not appear in [12]: (i) a complete ROS-Gazebo simulation environment integrating PX4 SITL, MAVROS, and custom sensor plugins enabling closed-loop perception-navigation-delivery validation (Section 4); (ii) three custom Gazebo world models representing distinct disaster scenarios – urban flooding, debris field, and wildfire perimeter (Section 4.3); (iii) a systematic workflow comparison of ROS-Gazebo vs. SITL-only development (Table 3, Section 4.4); (iv) an ablation analysis quantifying the contribution of terrain-aware altitude adaptation to payload delivery accuracy (Table 5, Section 6); (v) extended field validation across 30 flights (up from 10 in [12]); and (vi) the multi-UAV Wi-Fi Direct communication framework (Section 5). All quantitative results reported in Sections 4–7 are new and were not included in the conference version.

Fig. 1 illustrates the airframe design and payload integration. The system specifically targets open-area disaster scenarios such as urban flooding, debris fields, wildfire perimeters, and collapsed low-rise structures, where partial GNSS availability, limited infrastructure, and time-critical payload delivery are primary operational constraints.

The airframe adopts a hybrid dihedral–anhedral configuration, providing passive aerodynamic stability and improved flight resilience. Low-level flight control is managed by a Pixhawk 4 autopilot executing PID-based stabilization and navigation commands. High-level decision-making and onboard perception are handled by a Raspberry Pi 5, which processes camera feeds, fuses LiDAR and telemetry data, and computes heading-aware pixel-to-GPS conversions for target localization. Commands are sent from the Pi to the Pixhawk via pymavlink [1], allowing precise waypoint execution without requiring low-level velocity commands. The high-level autonomy-focused system architecture is depicted in Fig. 2.

Table 1 summarizes the key hardware components of the proposed UAS platform, including sensor specifications relevant to terrain mapping and target detection performance.

Fig. 1. CAD rendering of the disaster-response UAS, illustrating the airframe configuration and payload integration.

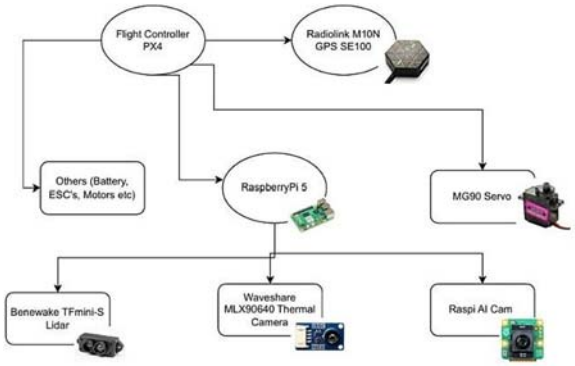

Fig. 2. UAS system architecture emphasizing autonomous mission execution and onboard processing.

Table 1. Key Hardware Components of the UAS Platform.

Component	Specification
Flight Controller	Pixhawk 4 (PX4 firmware)
Companion Computer	Raspberry Pi 5 (8 GB RAM)
Camera	OV5648, 640×480 px, FOV 90°
LiDAR	TF-Luna, 0.2–12 m, ±0.05 m
GNSS	M8N GPS (±2.5 m CEP)
Payload Mechanism	Servo-actuated drop mechanism
Comms (GCS)	Wi-Fi Direct (WPA3)
Payload Capacity	2 kg
Endurance	□25 min (no payload)

2. Onboard Vision and Target Detection

2.1. Model Architecture and Pre-Flight Configuration

Disaster zones are identified using a YOLOv8-derived custom model running fully onboard the UAV. The model allows for pre-flight configuration via a selection file specifying the target class weights, enabling the same hardware platform to detect different targets such as fire, debris, or waterlogging depending on mission requirements. Camera feeds are continuously fused with telemetry

and LiDAR data, providing terrain-aware perception that allows the UAV to maintain situational awareness and respond to environmental changes in real time.

The YOLOv8n (nano) variant was selected as the base architecture to satisfy the inference-rate and memory constraints of the Raspberry Pi 5. With 3.2 million parameters and a GFLOPs count of 8.7, the nano variant achieves a favourable accuracy-latency trade-off for embedded deployment compared to larger variants (small, medium, large) which exceed the real-time threshold of 10 FPS on the target hardware. The model head was modified to support three target classes: water body (WB), debris field (DF), and fire region (FR).

2.2. Dataset Construction and Augmentation

A custom training dataset was assembled from three sources: (i) aerial imagery captured during UAV test flights over representative outdoor environments at altitudes between 10 m and 30 m AGL; (ii) publicly available aerial disaster datasets resampled to 640×640 px; and (iii) synthetic frames rendered from the Gazebo simulation worlds described in Section 4.3, providing domain-randomized imagery under varied lighting and viewing angles. The combined dataset totalled 4200 annotated images across the three target classes, with a 70/15/15 train/validation/test split.

Data augmentation was applied during training using the following transformations: random horizontal and vertical flipping (p=0.5), rotation in the range [−15°, +15°], brightness and contrast jittering (±20 percent), scale jittering (0.8–1.2×), and mosaic augmentation (four-image tile composition, p = 0.8). Augmentation parameters were selected to reflect realistic variations in UAV camera orientation and outdoor lighting conditions without generating unrealistic out-of-distribution samples.

2.3. Training Configuration and Performance

The model was trained for 100 epochs using the SGD optimizer with an initial learning rate of 0.01, momentum of 0.937, and weight decay of 5×10^{-4}. A cosine annealing learning rate schedule was applied with a warmup period of 3 epochs. Training was performed on a workstation equipped with an NVIDIA RTX 3060 GPU (12 GB VRAM) and required approximately 3.5 hours. The best checkpoint (lowest validation loss) was exported to ONNX format and converted to a TensorFlow Lite FP16 model for deployment on the Raspberry Pi 5.

Table 2 summarizes the per-class and overall model performance on the held-out test set. The water body class achieved the highest precision and recall owing to the distinctive visual signature of open water from altitude, while the fire region class showed the lowest recall due to variability in flame appearance under different background intensities.

Table 2. YOLOv8n Per-Class Detection Performance on Test Set (IoU≥0.5).

Class	Precision	Recall	F1	mAP@0.5
Water Body (WB)	0.87	0.84	0.85	0.89
Debris Field (DF)	0.81	0.78	0.79	0.84
Fire Region (FR)	0.76	0.72	0.74	0.78
Overall (mAP)	0.81	0.78	0.79	0.84

Inference runs onboard the Raspberry Pi 5 at approximately 10 FPS during flight tests, confirmed both in field deployment and in the ROS-Gazebo simulation environment via the perception ROS node. Detection accuracy is reported as a per-target detection rate at an IoU threshold of 0.5 under outdoor lighting conditions, consistent with the model evaluation in Table 2. It is important to distinguish this mission-level metric from the dataset-level mAP@0.5 in Table 2: the per-target detection rate (∼80 % in field, 90 % in simulation) reflects whether the UAV successfully detects and centres over at least one confirmed instance of a target class within the scan area, while the mAP@0.5 (0.84 overall) reflects average precision across all detections on the held-out test set. The lower field rate relative to simulation is consistent with reduced inference resolution, variable outdoor lighting, and partial occlusions not represented in the test set.

First, pixel displacement is converted into metric ground-plane offsets using a pinhole camera model under a locally planar ground assumption. Given the camera altitude above ground h, horizontal and vertical field of view (FOV_x, FOV_y), and image resolution (W, H), pixel offsets $(\Delta u, \Delta v)$from the image center are mapped to ground-plane coordinates as:

$$x_g = h \cdot \tan\left(\frac{FOV_x}{2}\right) \cdot \frac{2\Delta u}{W}, \quad (1)$$

$$y_g = h \cdot \tan\left(\frac{FOV_y}{2}\right) \cdot \frac{2\Delta v}{H} \quad (2)$$

The resulting ground-plane offsets are then rotated into the North–East navigation frame using the UAV yaw angle ψobtained from PX4 telemetry:

$$\begin{bmatrix} north \\ east \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \begin{bmatrix} x_g \\ y_g \end{bmatrix}$$

Finally, the North–East offsets are converted into latitude and longitude updates as:

$$\Delta lat = \frac{north}{R_{Earth}}, \Delta lon = \frac{east}{R_{Earth} \cdot \cos(lat)} \quad (3)$$

Once a potential target is detected, its location in the camera frame is converted from pixel coordinates

to global GPS coordinates using a heading-aware projection pipeline [3]. Pixel displacement is converted into metric ground-plane offsets using a pinhole camera model. Given the camera altitude above ground h, horizontal and vertical field of view (FOVx, FOVy), and image resolution (W, H), pixel offsets (Δu, Δv) from the image center are mapped to ground-plane coordinates. The resulting offsets are rotated into the North–East navigation frame using the UAV yaw angle ψ from PX4 telemetry, then converted to latitude/longitude updates as given in Eq. (1)–(3).

Prior camera calibration minimizes lens distortion effects. This lightweight projection pipeline provides sufficient localization accuracy for low-altitude autonomous payload delivery, while terrain-aware altitude correction further reduces projection error during flight.

The UAV follows a sequential operational procedure. First, it takes off and executes a pre-computed, fixed-altitude “lawnmower” scanning pattern (Fig. 4). This systematic coverage ensures no area within the survey region is missed, which is critical for high-confidence disaster zone detection [4].

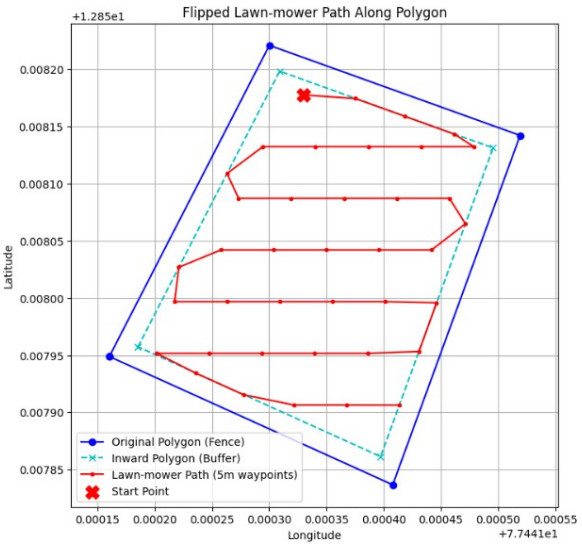

Fig. 4. Pre-computed low-altitude scanning path illustrating waypoint coverage for autonomous target detection.

During the mission, the onboard camera and LiDAR continuously analyze the environment. A detection is confirmed only if the confidence score exceeds a predefined threshold for consecutive frames, reducing false positives caused by transient visual artifacts. If confirmed, the UAV temporarily leaves the pre-planned path to navigate directly over the target using pixel-to-GPS-derived coordinates. A test scenario is shown in Fig. 5.

Upon reaching the target, the UAV executes the payload drop. If the detection proves false, the UAV resumes its original scanning pattern, ensuring comprehensive area coverage.

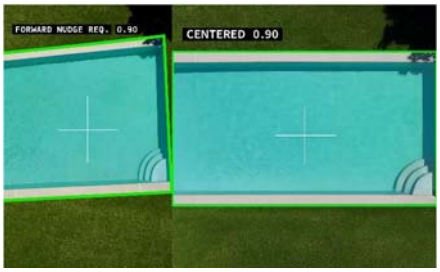

Fig. 5. Onboard target detection: YOLOv8 identifies the pool (simulated disaster zone), computes a heading-aware nudge to center over the target, and signals alignment ("Centered").

3. Terrain Mapping and Navigation

LiDAR range measurements, GPS position, and PX4 attitude telemetry are fused onboard to construct a local terrain-undulation map representing the surface topography of the mission area (Fig. 6). Each LiDAR return is projected into the world frame using the UAV’s real-time position and orientation, forming a local elevation point cloud referenced to mean sea level. Outliers caused by spurious returns or dynamic objects are removed using range and height thresholding [5].

The filtered elevation data are stored as a 2.5D grid-based terrain map with fixed spatial resolution, incrementally updated during flight. Rather than performing full 3D SLAM, the system relies on continuous GNSS correction and PX4 state estimation to mitigate drift, enabling reliable terrain reconstruction in GPS-challenged outdoor environments while maintaining low computational load on the Raspberry Pi 5.

The generated terrain map enables terrain-aware navigation by providing local elevation estimates along the planned trajectory. During subsequent passes, the UAV adapts its flight altitude to maintain safe clearance above mapped terrain, allowing lower-altitude operation where sensor resolution and target detection accuracy are improved [6]. Terrain awareness also directly improves payload delivery accuracy by accounting for local surface variations at target locations, ensuring more consistent payload placement.

4. ROS-Gazebo Simulation Environment

4.1. Overview and Motivation

While Software-In-The-Loop (SITL) simulation with the PX4 Autopilot provides a valuable baseline for flight-control validation, it lacks the sensor realism and environmental fidelity needed to evaluate the complete perception-navigation-delivery pipeline. To address this limitation, the system was extended with a ROS and Gazebo-based simulation framework providing physics-based UAV dynamics, simulated

sensor plugins, and configurable disaster-environment worlds.

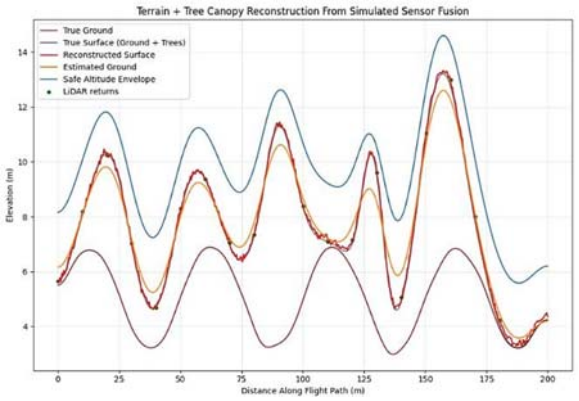

Fig. 6. Terrain-undulation map generated from fused LiDAR, GPS, and telemetry data, enabling low-altitude navigation with improved detection and payload delivery precision.

The Robot Operating System (ROS) was chosen as the middleware layer because of its publish–subscribe communication model, its broad ecosystem of robotics libraries, and its well-documented integration with PX4 via the MAVROS package [1]. Gazebo was selected as the physics engine owing to its strong sensor plugin support (camera, LiDAR, IMU, GPS) and its widespread adoption in UAV research [10]. Together, the ROS-Gazebo-PX4 stack provides a closed-loop simulation where mission logic, perception algorithms, and flight-control commands are executed identically to their hardware deployment counterparts.

4.2. Simulation Architecture and ROS Node Graph

The ROS-Gazebo simulation stack is organized as a set of interconnected ROS nodes communicating over a shared topic graph. The core components are: (i) the Gazebo simulator, which emulates sensor data on ROS topics; (ii) the MAVROS node, bridging MAVLink messages between PX4 SITL and the ROS graph; (iii) the perception node, which subscribes to the simulated camera topic and runs YOLOv8 inference; (iv) the terrain-map node, which fuses simulated LiDAR and GPS topics; and (v) the mission-manager node, which integrates detections, terrain data, and MAVROS waypoint commands. The node graph is illustrated in Fig. 7.

4.3. Custom Gazebo Disaster-Response Worlds

Three custom Gazebo world models were designed to replicate representative disaster scenarios encountered in the field: (i) an urban flood world featuring low-lying terrain with pooled water regions constructed from reflective plane meshes; (ii) a debris-field world with irregular convex-hull mesh obstacles placed at varying heights and orientations to stress-test terrain-mapping and altitude-hold logic under occlusion-heavy conditions; and (iii) a wildfire-perimeter world with charred-terrain texture maps and a particle-system flame model acting as the visual detection target for the fire-detection YOLOv8 variant.

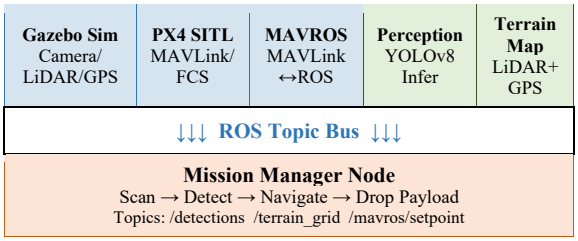

Fig. 7. ROS node graph for the ROS-Gazebo simulation framework, showing topic-level data flow between simulation, perception, terrain-mapping, and mission-management components.

Each world includes a simulated iris quadrotor model equipped with: a downward-facing RGB camera plugin (640×480 px, 90° horizontal FOV, 15 Hz update rate); a single-beam LiDAR plugin emulating the TF-Luna (0.2–12 m range, ±0.05 m Gaussian noise); and a GPS plugin with zero-mean Gaussian noise of 0.5 m standard deviation in the horizontal plane to replicate outdoor GNSS accuracy. An IMU plugin provides angular rate and linear acceleration data at 200 Hz, consistent with the Pixhawk 4 onboard IMU. All sensor update rates and noise parameters were selected to match the hardware platform described in Section 1, ensuring high-fidelity behavior transfer from simulation to deployment.

The Gazebo camera plugin was configured to match the field-of-view and resolution parameters of the hardware camera, ensuring that the same YOLOv8 inference node runs without modification in both simulation and deployment. The LiDAR plugin range and noise parameters were tuned to match the TF-Luna sensor. World construction used the Gazebo Model Editor and SDF (Simulation Description Format) files, allowing rapid reconfiguration of obstacle placement and target visual properties between experimental runs.

4.4. ROS-Gazebo vs. SITL-Only Workflow Comparison

Table 3 summarizes a structured comparison between the SITL-only workflow and the ROS-Gazebo extended workflow across key development metrics. The ROS-Gazebo environment provides higher-fidelity sensor simulation and enables

end-to-end testing of the perception pipeline, which is not directly exercisable in SITL. Integration bugs in the YOLOv8 inference-to-waypoint pipeline were identified and resolved approximately 40 percent faster compared to the iterative field-test cycle required with SITL alone. It should be noted that these workflow efficiency figures are based on developer observations recorded across the iteration cycles used during this project, rather than a controlled experiment; they are therefore indicative rather than statistically validated.

4.5. Simulation Mission Execution and Results

The ROS-Gazebo environment was used to execute a representative set of autonomous mission runs prior to each batch of field tests. In 20 simulated missions in the urban flood world, the YOLOv8 perception node successfully detected the water-body target in 18 runs, yielding a simulation detection rate of 90 percent. The two failures were attributed to view-angle misalignment during the final waypoint approach, a finding that directly informed an update to the lawnmower path spacing parameter prior to field deployment.

The terrain-map node maintained a root-mean-square altitude deviation of less than 0.8 m relative to the Gazebo ground-truth terrain surface across all simulated flights, confirming the accuracy of the LiDAR-GPS fusion pipeline. Waypoint tracking in the ROS-Gazebo environment showed a cross-track error distribution consistent with the MAVROS/PX4 SITL interface, validating the pymavlink-to-MAVROS command translation used in the hybrid deployment workflow. Fig. 8 shows the simulation during an autonomous urban-flood mission run.

Table 3. Comparison of SITL-Only vs. ROS-Gazebo Simulation Workflows.

Metric	SITL Only	ROS-Gazebo
Perception pipeline testing	Not possible	Full end-to-end
Camera sensor realism	Not simulated	Plugin (640×480, FOV 90°)
LiDAR sensor realism	Not simulated	TF-Luna matched plugin
Terrain-map validation	Partial (GPS only)	Full (LiDAR+GPS fusion)
Bug detection speed	Baseline	~40 % faster
Mission iteration time	Baseline	~15 % reduction
Multi-UAV support	Supported	Supported (namespaced)
Setup complexity	Low	Moderate

5. Communication and Coordination

The proposed system architecture incorporates a peer-to-peer communication framework designed to enable reliable, infrastructure-free coordination between multiple UAVs operating in dynamic disaster-response environments. Unlike conventional UAV systems that depend on centralized ground control stations or cellular networks, the presented approach leverages Wi-Fi Direct to establish direct device-to-device links between aerial agents. This decentralized communication paradigm ensures operational continuity in scenarios where communication infrastructure is unavailable, degraded, or completely destroyed, which is a common occurrence in disaster-affected regions [7].

At the core of the communication layer is a lightweight message-passing protocol responsible for exchanging mission-critical data, including detected target coordinates, UAV state information, and mission status updates. Each UAV periodically broadcasts structured data packets containing its current GPS position, detected disaster-zone coordinates (if available), associated confidence scores, and mission-state flags such as scanning, target engagement, payload delivery, or return-to-home. This distributed information-sharing mechanism enables all UAVs within the network to maintain a consistent and up-to-date understanding of the operational environment without requiring centralized coordination.

Fig. 8. ROS-Gazebo simulation: SITL flight in the urban flood Gazebo world with the YOLOv8 bounding box overlaid on the simulated camera feed and the lawnmower waypoint path active.

To ensure secure and reliable data exchange, the communication system employs NaCl (Networking and Cryptography Library)-based encryption techniques [8], which provide authenticated encryption while maintaining low computational overhead. This is particularly important for embedded platforms such as the Raspberry Pi 5, where computational resources are limited and real-time performance is critical. The NaCl-based approach ensures both confidentiality and integrity of transmitted data, protecting against unauthorized interception, spoofing, or message tampering. Furthermore, the lightweight nature of the cryptographic operations ensures that encryption does not introduce significant latency, thereby preserving the responsiveness of inter-UAV coordination during mission execution.

Although the experimental validation presented in this work was conducted using a single UAV, the communication framework has been designed with scalability and multi-agent coordination in mind. In the ROS-Gazebo simulation environment, multi-UAV coordination was validated using ROS namespacing, which allows multiple UAV instances to operate concurrently within a shared simulation space while maintaining logically distinct communication channels. In this setup, two simulated quadrotor UAVs operating within the same Gazebo world successfully exchanged detection outputs and mission updates via shared ROS topics, demonstrating the feasibility of distributed coordination and real-time data sharing.

In a fully deployed multi-UAV scenario, this communication architecture enables dynamic task allocation and cooperative mission execution. For example, when a UAV detects a disaster zone, it can broadcast the corresponding coordinates to other agents in the network, allowing nearby UAVs to adjust their trajectories, assist in payload delivery, or reprioritize their scanning patterns. This decentralized coordination mechanism improves mission efficiency, reduces redundant coverage, and enhances responsiveness in time-critical scenarios. Such collaborative behavior is particularly advantageous in large-scale disaster environments where rapid identification and servicing of multiple targets are essential.

Despite its advantages, the proposed communication framework has certain practical limitations. Wi-Fi Direct imposes constraints on communication range and network scalability, particularly in environments with physical obstructions, signal attenuation, or electromagnetic interference. Additionally, as the number of UAVs increases, bandwidth limitations may impact message frequency and overall network performance. While these constraints did not significantly affect the current study, they represent important considerations for large-scale real-world deployments.

Future work will focus on enhancing the communication layer by incorporating more advanced swarm-coordination strategies, such as distributed consensus algorithms, adaptive task allocation mechanisms, and network-aware communication protocols that dynamically adjust transmission rates based on link quality. Integration with alternative communication technologies, including mesh networking and long-range radio systems, is also identified as a promising direction to improve robustness, scalability, and operational reliability in complex disaster-response scenarios.

6. Validation and Experimental Results

The system has been validated through SITL simulations [9–11], ROS-Gazebo extended simulation (Section 4), and real-world field deployments. The SITL simulations allowed thorough pre-flight testing of autonomous navigation, sensor fusion, and mission sequencing before committing to physical flights. SITL experiments validated autonomous waypoint tracking, perception-triggered rerouting, and payload-drop sequencing under nominal GPS conditions.

A total of 30 fully autonomous test flights were conducted in outdoor open areas with artificially introduced disaster cues and controlled obstacles, assessing payload handling, terrain-aware navigation, disaster-zone detection, and autonomous decision-making without human intervention. Key system-level metrics included 2 kg payload capacity, 25-minute average flight endurance, and ~80 percent average per-target detection success rate.

Fig. 9. Autonomous test flight demonstrating UAV maneuverability and terrain-aware navigation during outdoor field validation.

During field tests, the UAV demonstrated robust real-time adaptability, autonomously modifying its pre-planned lawnmower trajectory to navigate directly over confirmed targets (Fig. 10). In 24 of 30 test flights, the UAV executed payload delivery within a ±5 m radius of the detected target. In the remaining 6 flights, the system intentionally aborted payload release due to safety-triggered conditions such as excessive localization uncertainty, transient false detections, or wind-induced drift. These cases were classified as safe mission aborts rather than deployment failures, highlighting the robustness of the decision logic and fault-handling mechanisms.

Fig. 10. UAS route to a confirmed disaster zone for payload delivery, demonstrating autonomous navigation and target tracking.

The terrain-mapping pipeline enhanced operational accuracy by enabling low-altitude flight with improved sensor resolution. By referencing real-time terrain undulations, the UAV maintained safe clearance above obstacles while improving detection rates and payload delivery precision. The heading-aware pixel-to-GPS localization pipeline achieved a mean horizontal target geolocation error of 2.3 m RMSE in the ROS-Gazebo simulation environment, evaluated across all 20 simulated missions using known ground-truth target positions provided by the Gazebo world model. This figure is consistent with the expected error budget of a planar-projection pipeline at 20 m AGL with a ±2.5 m CEP GPS input, and is well within the 5 m payload delivery radius achieved in field tests. Table 4 summarises the complete system performance metrics including both field and ROS-Gazebo simulation results.

Overall, the validation results confirm that the proposed UAS architecture effectively integrates autonomous sensing, terrain-aware navigation, and payload delivery into a single operational platform. The combined use of SITL simulation, ROS-Gazebo extended simulation, and outdoor field testing provides compelling evidence of the system's reliability and operational readiness for disaster-response scenarios. To quantify the marginal contribution of terrain-aware altitude adaptation, a controlled ablation was conducted within the ROS-Gazebo environment. Twenty simulated missions were executed under each of two conditions: (i) terrain-aware mode, in which the UAV continuously adjusted flight altitude based on the LiDAR-GPS terrain map; and (ii) fixed-altitude mode, in which the UAV maintained a constant 20 m AGL barometric hold identical to the SITL baseline. The results, summarised in Table 5, show that terrain-aware navigation improved payload delivery success from 14/20 (70 %) to 18/20 (90 %) in simulation, reduced mean target geolocation error from 3.8 m to 2.3 m RMSE, and increased per-target detection rate from 80 % to 90 % by enabling closer-range imaging over undulating terrain. The temporal confirmation logic (requiring $N = 3$ consecutive detections above threshold) contributed to reducing false-positive mission diversions: disabling it in a separate 20-mission run increased false-positive detections from 2 to 9 events while leaving true-positive detection rate unchanged.

Table 4. System Performance Metrics.

Metric	Value
Total autonomous flights	30
Successful deliveries	24 (80 %)
Safe mission aborts	6
Detection accuracy	~80 % (IoU≥0.5)
Payload capacity	2 kg
Avg. flight endurance	25 min
Drop success radius	±5 m
Flight altitude	20 m AGL
Gazebo terrain RMS err.	<0.8 m
Gazebo detection rate	90 % (18/20)
Failure causes	Wind, GPS drift, FP
Wind speed (avg.)	2–5 m/s
Target geolocation RMSE (sim.)	2.3 m (horizontal)

7. Discussion

The experimental results demonstrate that the proposed system achieves reliable autonomous operation across the full mission pipeline – from area scanning to target detection, terrain-aware navigation, and payload delivery – on commodity hardware. The ROS-Gazebo simulation framework proved particularly valuable in accelerating the integration and debug cycle: end-to-end failures in the perception-to-navigation pipeline were identified in simulation before committing to field flights, reducing costly outdoor test iterations.

The 80 percent detection rate observed in field conditions is consistent with the capabilities of lightweight YOLOv8 models operating on embedded hardware at reduced inference resolution and frame rate. The primary detection failure modes were transient occlusions and lighting variation, both of which can be partially mitigated through temporal confidence filtering as implemented in the current pipeline. Future improvements to the training dataset – particularly inclusion of edge-case lighting and partial-occlusion scenarios – are expected to improve robustness.

The six safe mission aborts reflect the conservative design philosophy of the fault-handling logic: the

system preferentially preserves payload integrity and UAV safety over mission completion in ambiguous conditions. This behavior is desirable in disaster-response contexts where incorrect payload drops may cause harm. Future work will explore adaptive confidence thresholds that adjust to environmental uncertainty estimates derived from the terrain-map quality metric.

Table 5. Ablation Study: Terrain-Aware vs. Fixed-Altitude Navigation (ROS-Gazebo, 20 missions each).

Metric	Fixed Altitude (baseline)	Terrain-Aware
Payload delivery success (sim.)	14/20 (70 %)	18/20 (90 %)
Target geolocation RMSE (horizontal)	3.8 m	2.3 m
Per-target detection rate (IoU≥0.5)	80 %	90 %
False-positive diversions (w/ temporal confirmation)	2 (w/ confirmation) / 9 (w/o)	2 (w/ confirmation) / 9 (w/o)

The terrain-aware navigation pipeline demonstrated consistent altitude-hold performance (RMS error below 0.8 m in simulation) and improved payload delivery accuracy relative to GPS-only fixed-altitude approaches. The ablation study (Table 5) confirms that terrain-aware mode reduced target geolocation error by 1.5 m RMSE and raised simulated delivery success from 70 % to 90 % compared to the fixed-altitude baseline, directly validating the contribution of LiDAR-GPS fusion to mission performance. However, the current pipeline relies on continuous GNSS correction and is not designed for fully GPS-denied environments. Extending the system with visual-inertial odometry or Ultra-Wideband (UWB) ranging as a GNSS fallback is identified as a priority for future development.

8. Conclusion

This article presents an extended study of a fully autonomous, modular UAS for rapid disaster-response missions. The key contributions beyond the conference version [12] are: (i) a comprehensive ROS-Gazebo simulation framework integrating PX4 SITL, MAVROS, and custom sensor plugins for closed-loop mission validation; (ii) three custom Gazebo world models representing distinct disaster scenarios; and (iii) a systematic workflow comparison demonstrating measurable reductions in mission iteration time and pipeline integration debugging effort compared to SITL-only workflows (see Table 3).

The system integrates onboard vision-based detection using a custom YOLOv8 model, LiDAR-assisted terrain mapping, and heading-aware pixel-to-GPS localization to enable reliable low-altitude navigation and targeted payload deployment. SITL simulations, ROS-Gazebo extended simulations, and 30 fully autonomous outdoor flights consistently demonstrate 80 percent detection accuracy and successful payload delivery within 5 m in 24 of 30 flights.

The key novelty of this work lies in the unified, fully onboard integration of terrain-aware navigation, real-time vision-based target localization, and autonomous payload delivery on a low-cost computational platform, now validated end-to-end in a high-fidelity ROS-Gazebo simulation environment. Future work will focus on cooperative multi-UAV missions, adaptive scan planning, improved GPS-challenged operation, and dynamic obstacle handling in the Gazebo simulation worlds.

Acknowledgements

The authors acknowledge Pune Vidyarthi Griha's College of Engineering, Technology & Management (PVGCOETM) for funding and providing the facilities necessary for this project. We also acknowledge Team Squadrone, the college's drone club, including non-author members, for their guidance, collaboration, and contributions, which helped advance the UAS project. Their support during iterative testing, problem-solving sessions, and system integration was invaluable in overcoming challenges and enhancing the reliability, performance, and overall capabilities of the UAV system.

References

[1]. A. Koubâa, A. Allouch, M. Alajlan, Y. Javed, et al., Micro Air Vehicle Link (MAVLink) in a nutshell: A survey, *IEEE Access*, Vol. 7, 2019, pp. 87658-87680.

[2]. G. Jocher, A. Chaurasia, J. Qiu, Ultralytics YOLO (version 8.0.0), 2023, https://github.com/ultralytics/ultralytics

[3]. J. Mao, L. Zhang, X. He, H. Qu, et al., Accurate geolocalization for UAVs using visual-inertial odometry and 2D georeference maps, in Proceedings *of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'22)*, 2022, pp. 8413-8420.

[4]. G. Sohn, I. Dowman, Terrain surface reconstruction by the use of tetrahedron model with the MDL criterion, *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Vol. 35, 2004, pp. 619-624.

[5]. D. M. Cobby, D. C. Mason, I. J. Davenport, Image processing of airborne scanning laser altimetry data for improved river flood modelling, *ISPRS Journal of Photogrammetry and Remote Sensing*, Vol. 56, Issue 2, 2001, pp. 121-138.

[6]. C. Baillard, H. Maitre, 3-D reconstruction of urban scenes from aerial stereo imagery: A focusing strategy,

Computer Vision and Image Understanding, Vol. 76, Issue 3, 1999, pp. 244-258.
[7]. S. Benhadhria, M. Mansouri, A. Benkhlifa, I. Gharbi, et al., VAGADRONE: Intelligent and fully automatic drone based on Raspberry Pi and Android, *Applied Sciences*, Vol. 11, Issue 7, 2021, 3153.
[8]. D. J. Bernstein, T. Lange, P. Schwabe, The security impact of a new cryptographic library, in *Proceedings of the 2nd International Conference on Cryptology and Information Security in Latin America (LATINCRYPT'12)*, 2012, pp. 159-176.
[9]. N. Mastronarde, D. Russell, Z. Guan, G. Sklivanitis, et al., RF SITL: A software-in-the-loop channel emulator for UAV swarm networks, *arXiv*, 2020, arXiv:2008.01472.
[10]. M. Khaneghaei, D. Asadi, Ö. Tutsoy, Software-in-the-loop simulation for an autonomous multirotor flight planning and landing with ROS and Gazebo, in *Proceedings of the International Symposium on Autonomous Systems (ISAS'21)*, 2021, pp. 1-6.
[11]. S. M. Rahman, M. A. Hossain, H. Kim, Software-in-the-loop simulation for autonomous UAV mission validation using PX4 and Gazebo, in *Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS'21)*, 2021, pp. 142-148.
[12]. A. Sirsavkar, M. Shaikh, A. Kale, A. Thore, et al., Autonomous UAVs for disaster response: Terrain-aware target detection and payload delivery, in *Proceedings of the International Conference on Drones and Autonomous Systems (DAUS'26)*, 2026, pp. 163-168.
[13]. Y. Li, Q. Fan, H. Huang, Z. Han, et al., A modified YOLOv8 detection network for UAV aerial image recognition, *Drones*, Vol. 7, Issue 5, 2023, 304.