

TDP: CW-Lion (China), Outdoors Category

Jialiang Wang¹*, Yijun Huang¹, Jiwen Xu¹, Hang Zhen², Ze Qu³, Jiaxin Fan³

¹The Chinese University of Hong Kong, Hong Kong SAR, China

²Wuhan University, Hubei, China

³Tongji University, Shanghai, China

ABSTRACT

This Technical Description Paper outlines the CW-Lion team's solution for the IMAV 2025 outdoor competition. The large-payload quadrotor, featuring GNSS, IMU, camera, onboard computer and optional LiDAR sensors, enables autonomous flights in dynamic environments.

Mission 1 employs Fast-LIVO2 SLAM with local map management for light-weight 3D mapping, transmitting point cloud slices via ROS-bridge for ground visualization.

Mission 2 uses a two-stage pipeline for person retrieval from aerial video: tiled YOLOv8 detection for candidate generation, followed by CLIP-based semantic matching for precise identification.

Mission 3 proposes a distributed multi-UAV cooperative transportation scheme using virtual-node Lagrangian dynamics, enabling unified acceleration commands for trajectory tracking, formation maintenance, and payload lifting.

Mission 4 addresses water transportation with suspended load dynamics, swing suppression via feedback control, rope tension monitoring, and cooperative strategies for stable pickup, transport, and release.

1 PLATFORM OVERVIEW

To enable fully autonomous flight in outdoor environments, a large-payload MAV with GNSS receiver is designed. The MAV is constructed with firm and lightweight materials including carbon fiber and aluminum alloy, ensuring durability and flexibility. To meet the payload requirements in the transportation challenge, it is powered by a 21700 6S-1P 4000mAh 22.2V Li-ion Battery, facilitating a maximum take-off weight of 2500g and a payload capacity of 1800g. It offers a maximum flight time of 25 minutes outdoors, supporting extended operation periods. As is shown in Figure 1, the MAV is 440mm × 440mm × 160mm in size and consists of a large compartment beneath the fuselage, thereby enabling flexible add-ons such as LiDAR sensor for mapping challenge in mission 1 and suspended cables in mission 4.

*Email address: jialiang.wang@link.cuhk.edu.hk

The localization is achieved primarily through GNSS and IMU. It is equipped with a MicroAir MG902 GNSS receiver and a MicroAir NxtPX4v2 flight controller with dual BMI 088 IMU.

The onboard computing unit, an Nvidia Jetson Orin NX, features a double-core Nvidia ARM V8 CPU, 256-core Nvidia GPU, 16GB DDR4 memory, and 256GB storage, with a maximum power consumption of 25W. This setup supports real-time processing of sensor data and execution of complex algorithms, including deep learning for perception and decision-making.



Figure 1: Platform design

2 SOLUTION TO MISSION 1

To build a 3D map of the designated area, we employ the aircraft's Lidar, IMU, Camera, and GNSS to produce a colored point cloud map. We propose to run both a control algorithm and a SLAM algorithm on the aircraft, transmitting map point cloud slices to the ground via 5G/WIFI modules in real-time, allowing the mapping process to be monitored and visualized by the public through the ground control station. Fast-LIVO2 [1] will be used for SLAM on the aircraft, with Least Recently Used (LRU)-based local map management to address the limitations of the onboard processor and memory. A mapping algorithm will also run on the ground station, receiving point cloud slices and the aircraft's current pose for real-time visualization.

2.1 Fast-LIVO2 with LRU

Fast-LIVO2 assumes that the time offsets between three sensors (LiDAR, IMU, and camera) are known and can be pre-calibrated or synchronized. Initially, the asynchronously

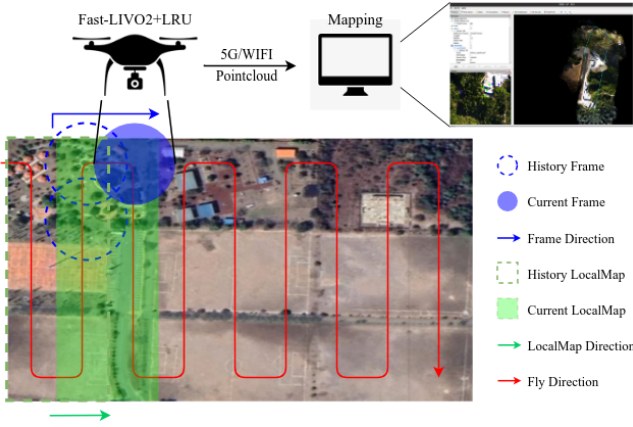


Figure 2: Overview of large-scale real-time UAV mapping

sampld LiDAR points are reassembled into scans corresponding to the camera sampling times through scan reorganization. Then, Fast-LIVO2 tightly couples LiDAR, image, and inertial measurements through an ESIKF (Extended Sequential Invariant Kalman Filter) with sequential state updates. The system state is sequentially updated first by LiDAR measurements, followed by image measurements, both of which employ a direct method based on a single unified voxel map.

To construct the LiDAR measurement model in the ESIKF update, point-to-plane residuals from the frame to the map are calculated. The error model for the LiDAR update is given by:

$$\underbrace{\mathbf{0}}_{\mathbf{y}_l} = \underbrace{(\mathbf{n}_j \boxminus \delta \mathbf{n}_j)^T ({}^G \mathbf{T}_I^I \mathbf{T}_L ({}^L \mathbf{p}_j - \delta {}^L \mathbf{p}_j) - (\mathbf{q}_j - \delta \mathbf{q}_j))}_{\mathbf{h}_l(\mathbf{x}, \mathbf{v}_l)} \quad (1)$$

where ${}^L \mathbf{p}_j$ is the measured position of point j in the LiDAR frame, and $\mathbf{q}_j$ corresponds to the center point of the plane, and $\mathbf{n}_j$ is the normal vector of the plane. The measurement noise $\mathbf{v}_l = (\delta {}^L \mathbf{p}_j, \delta \mathbf{n}_j, \delta \mathbf{q}_j)$ consists of the noise associated with the LiDAR point, the normal vector, and the plane center, respectively.

In the visual measurement model, the algorithm utilizes visible voxel queries and on-demand raycasting to retrieve visual map points located within the current field of view (FoV) from the map. Once extracted, any abnormal visual map points, such as occluded ones or those with depth discontinuities, are identified and discarded. The photometric error between the frame and the map is then calculated to update the visual measurements. The Sparse-Direct Visual

Measurement Model is expressed as follows:

$$\underbrace{\mathbf{0}}_{\mathbf{y}_c} = \underbrace{\tau_k(\mathbf{I}_k(\mathbf{u}_i + \Delta \mathbf{u}) - \delta \mathbf{I}_k) - \tau_r(\mathbf{I}_r(\mathbf{u}'_i + \mathbf{A}_i^T \Delta \mathbf{u}) - \delta \mathbf{I}_r)}_{\mathbf{h}_c(\mathbf{x}, \mathbf{v}_c)} \quad (2)$$

where $\mathbf{A}_i^T$ is the affine warping matrix that transforms pixels from the i -th current patch to the reference patch, $\Delta \mathbf{u}$ is the relative pixel position to the center $\mathbf{u}_i$ within the current patch. The measurement noise $\mathbf{v}_c = (\delta \mathbf{I}_k, \delta \mathbf{I}_r)$ consists of the noise associated with the reference and current frame pixel values, which can originate from various sources such as shot noise and Analog-to-Digital Converter (ADC) noise in the camera CMOS.

Due to space constraints, a detailed discussion is omitted here. For a more comprehensive explanation of the underlying principles of Fast-LIVO2, the reader is referred to [1] [2] [3].

In practical applications, we primarily improve the map management of Fast-LIVO2 by employing the LRU caching strategy. In Fast-LIVO2, the concept of a local map is introduced to manage memory usage efficiently. The local map only stores data within a large region around the LiDAR's current position, and its size is kept fixed. As the LiDAR moves, the map is shifted to ensure it always covers the relevant area, with any out-of-bounds memory being reset. While this approach prevents the map from growing unbounded, it is not highly efficient. The need to frequently check whether voxel hashes exceed the boundaries and update the map accordingly can introduce unnecessary computational overhead.

LRU is a commonly used page replacement strategy, which selects the page that has not been accessed for the longest time for elimination. The algorithm assigns each page an access field to record the time t elapsed since the page was last accessed. When a page needs to be replaced, the page with the largest t value, i.e., the least recently used page, is chosen for eviction. Faster-LIO [4] has previously utilized this algorithm to manage voxel hash maps.

We adopt a similar management approach for the global voxel map in Fast-LIVO2. For the originally scattered hashed voxels, we use a linked list for management. During each update, the most recently accessed voxel is inserted at the front of the list. We set a fixed length for the linked list, and any voxels exceeding this length will be discarded. Based on the trajectory shown in Figure 2, our goal is to maintain a local map, indicated by the green box, which can meet the scan-to-map registration requirements in a repeatedly traversed trajectory, while ensuring the memory usage remains manageable for the onboard computer.

2.2 Aircraft-Ground Control Station Communication

To enable online preview of the built map, we propose to use Rosbridge to transport standard ROS messages to the ground control station. Rosbridge is a protocol for sending

JSON based commands to ROS, which includes subscribing and publishing topics, service calls, getting and setting params etc.

While Rosbridge provides the JSON_i-ROS conversion, it leaves the transport layer to others. We use the standard `Rosbridge_server` package to set up a WebSocket connection so that ground control station can directly communicate with the onboard ROS. Once a 5G/WIFI connection to the onboard system is established, the ground control stations receives the local point cloud map maintained by LRU and visualizes it to the screen. Whenever LRU updates the current local map, the ground control station appends the new point cloud to the existing map, thereby eliminating the need to transport the whole map at once.

3 SOLUTION TO MISSION 2

To address the challenge of finding a specific person from a live aerial video feed using a natural language description, we propose a robust, two-stage "detect-then-match" pipeline. This task is particularly demanding due to a confluence of factors inherent to aerial surveillance: large variations in altitude and oblique camera angles which alter a person's appearance, the small apparent size of human targets within a wide field of view, and potential occlusions from infrastructure or foliage. Our system's primary **input** is a high-resolution video stream from the UAV's camera, coupled with a user-provided text query (e.g., "a person wearing a red helmet"). The final **output** is the 2D bounding box of the identified person, enabling downstream tasks like target tracking or UAV guidance.

The overall architecture of our proposed method, depicted in Figure 3, is explicitly designed to prioritize accuracy and robustness over raw speed. It first employs a specialized detection strategy to generate a comprehensive set of human candidates, maximizing recall to ensure the true target is not missed. Subsequently, it uses a powerful vision-language model to perform fine-grained re-identification, filtering the candidates to achieve high precision. The entire workflow is formally summarized in Algorithm 1.

The first stage of our pipeline is dedicated to maximizing detection recall. Standard object detectors, when applied to high-resolution aerial imagery, often fail to detect small objects. This is because the initial downsampling, required to fit the large image into the model's fixed-size input, can cause small targets to diminish to just a few pixels or disappear entirely. To overcome this critical limitation, we employ a state-of-the-art detector, YOLOv8 [5], chosen for its excellent balance of speed and accuracy, and augment it with a Slicing Aided Hyper Inference (SAHI) strategy [6]. Instead of processing the entire frame at once, the SAHI framework intelligently divides the input image into multiple overlapping smaller tiles. The YOLOv8 model then runs inference on each tile independently, where the relative size of objects is effectively magnified. This significantly improves the detection rate for small-scale humans. The detections from all

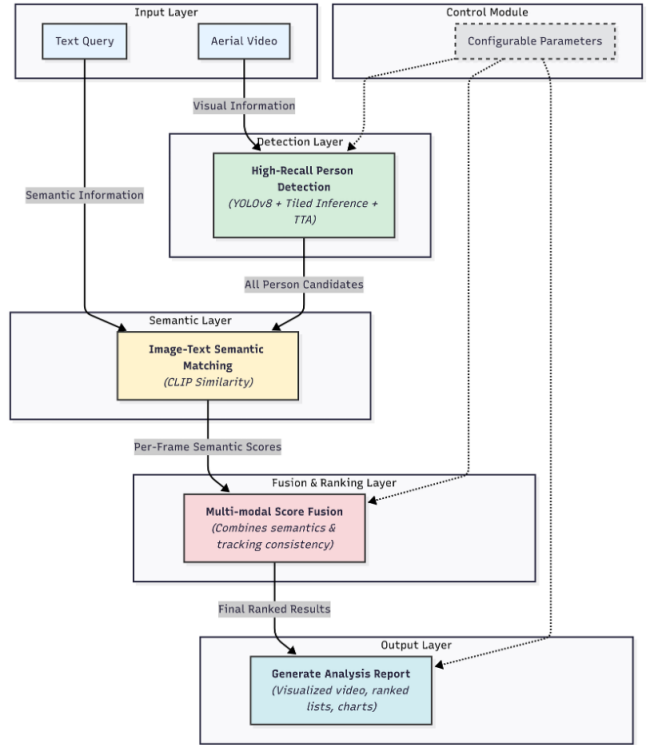


Figure 3: System overview of our proposed two-stage person retrieval pipeline. Stage 1 (Tiled Detection) maximizes recall by finding all potential candidates using SAHI and YOLOv8. Stage 2 (Semantic Re-ID) uses CLIP to filter these candidates against the text query to ensure precision.

tiles are subsequently merged back into the original image coordinates, and a final non-maximum suppression (NMS) pass is applied to eliminate duplicate bounding boxes arising from the overlapping regions. The output of this stage is a high-confidence set of bounding boxes for all detected persons, which serve as candidates for the next stage.

With a comprehensive set of candidate detections, the second stage performs semantic re-identification to determine which candidate, if any, matches the user's text query. For this task, we leverage the powerful zero-shot capabilities of the Contrastive Language-Image Pre-Training (CLIP) model [7]. Pre-trained on a vast dataset of internet-scale image-text pairs, CLIP learns to project both images and text into a shared multimodal embedding space. This allows it to compute a meaningful similarity score between an arbitrary image region and a text description, even if it has never been explicitly trained on the specific object attributes in the query (e.g., "red helmet"). For each candidate bounding box B_i from the first stage, we crop the corresponding image region C_i from the full frame. The CLIP model then computes the cosine similarity between the image embedding of the crop ($\mathbf{v}_i$) and the text embedding of the user's query ($\mathbf{t}$), calculated

as: $\text{Score}(C_i) = (\mathbf{v}_i \cdot \mathbf{t}) / (\|\mathbf{v}_i\| \|\mathbf{t}\|)$. The candidate with the highest overall score is selected as the final target, effectively filtering the initial detections based on nuanced semantic attributes and providing robustness against novel or complex descriptions.

Algorithm 1: Two-Stage Person Retrieval Pipeline

Data: High-resolution video frame F , Text query Q

Result: Bounding box of the target person B_{target}

begin

$Tiles \leftarrow \text{GenerateOverlappingTiles}(F)$

$AllDetections \leftarrow \emptyset$

foreach $tile$ **in** $Tiles$ **do**

$Detection_{\text{tile}} \leftarrow \text{YOLOv8.detect}(tile)$

$AllDetections \leftarrow$

$AllDetections \cup Detection_{\text{tile}}$

$CandidateBoxes \leftarrow$

$\text{MergeAndNMS}(AllDetections)$

// Stage 2: Semantic
Re-Identification

$max_score \leftarrow -\infty$

$B_{\text{target}} \leftarrow \text{null}$

$\mathbf{t} \leftarrow \text{CLIP.EncodeText}(Q)$

foreach box **in** $CandidateBoxes$ **do**

$Crop \leftarrow \text{CropImage}(F, box)$

$\mathbf{v} \leftarrow \text{CLIP.EncodeImage}(Crop)$

$score \leftarrow \text{CosineSimilarity}(\mathbf{v}, \mathbf{t})$

if $score > max_score$ **then**

$max_score \leftarrow score$

$B_{\text{target}} \leftarrow box$

return B_{target}

In terms of expected performance, this specialized two-stage pipeline is designed for maximum accuracy. This high level of accuracy, however, comes at the cost of significant computational overhead, as each of the candidates must be processed sequentially by the CLIP model. This makes the approach highly suitable for accuracy-critical, non-real-time analysis performed on powerful ground station hardware where precision is paramount.

For scenarios demanding higher real-time feedback, we also evaluate a flexible onboard alternative using the end-to-end YOLO-World model [8]. This model performs open-vocabulary detection in a single pass, directly identifying objects based on text prompts. As illustrated in Figure 4, this approach offers exceptional real-time processing speeds. However, this speed comes at a clear cost to precision. For the given query, the model generates multiple false positives, incorrectly labeling other people and even inanimate objects, while assigning only a moderate confidence score to the correct target.

In stark contrast, Figure 5 demonstrates the superior ac-

curacy and reliability of our proposed two-stage methodology. In the first stage, our system correctly identifies all individuals in the scene as potential candidates. In the second stage, the CLIP-based re-identification process analyzes each candidate and correctly assigns the highest semantic similarity score to the person wearing the red helmet, decisively identifying them as the target. Crucially, all other candidates receive lower scores, and no false positives are produced. This empirical comparison highlights the effectiveness of our design choice. The dual-solution architecture—a high-accuracy ground-based system and a high-speed onboard alternative—provides operational flexibility, allowing the system to dynamically adapt to different mission requirements and communication link qualities.

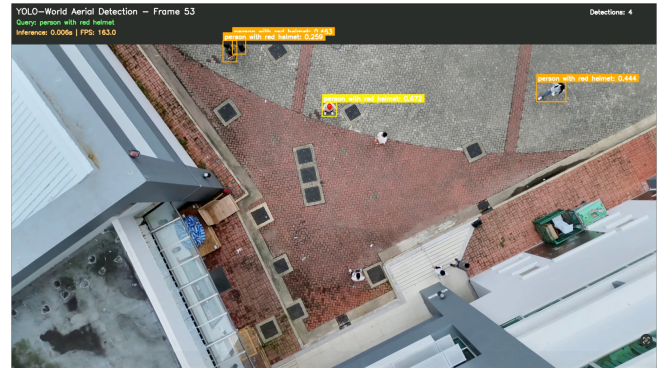


Figure 4: Result from the single-stage YOLO-World model. While offering real-time performance, it produces several false positives for the text query and identifies the correct target with only moderate confidence.

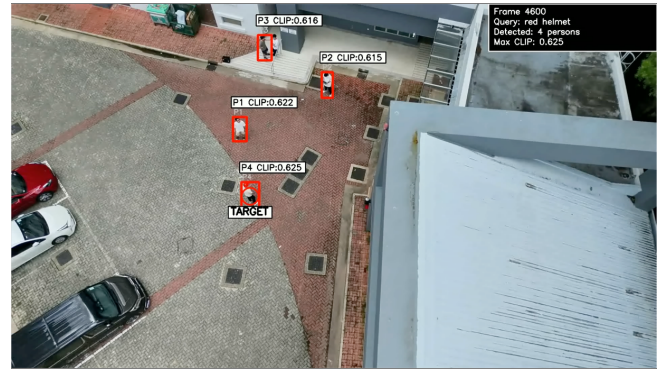


Figure 5: Result from our proposed two-stage pipeline. It correctly identifies all human candidates in the first stage. In the second stage, it precisely matches the text query to the correct person by assigning the highest semantic similarity score, demonstrating superior accuracy and eliminating false positives.

4 SOLUTION TO MISSION 3

The cooperative transportation system is described as a robot system consisting of n flying robots and a suspended load. Each robot is modeled as a double integrator. The connection points of the flying robots and the load are located at the center of mass. Since the elastic deformation of the cable is insignificant, the cable is treated as being in a stressed state. The dynamics of the robot system are described by the Lagrange equation as

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{\mathbf{x}}_i} - \frac{\partial L}{\partial \mathbf{x}_i} = \mathbf{u}_i, i = 1, 2, \dots, n+1 \quad (3)$$

where $\mathbf{x}_i$ are the coordinates of the i th flying robot; $\mathbf{x}_{n+1}$ are the coordinates of the suspended load; L is the Lagrange function of the cooperative robot system; and $\mathbf{u}_i$ is the control input of the i th flying robot. In this paper, we design a cooperative transportation system to derive a generalized control law for each flying robot. Spatial coordinates of virtual nodes are defined as $\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_{n+2} \in \mathbb{R}^3$. Each virtual node is associated with a corresponding flying robot. These virtual nodes generate interactive forces between neighboring nodes, guiding the overall system toward a stable configuration. The corresponding node system in Equation 3 is described as

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{\mathbf{q}}_i} - \frac{\partial L}{\partial \mathbf{q}_i} + \frac{\partial \psi}{\partial \dot{\mathbf{q}}_i} = 0, i = 1, 2, \dots, n+1 \quad (4)$$

where $\mathbf{q}_i$ is the coordinates of the i th virtual node and it is calculated as

$$\mathbf{q}_i = \mathbf{x}_i + \frac{h_c - h_i}{h_i - h_{n+1}} (\mathbf{x}_i - \mathbf{x}_{n+1}), i = 1, 2, \dots, n, \quad (5)$$

$\mathbf{q}_{n+1}$ is the coordinates of the suspended load. h_c , h_i , and h_{n+1} are the pre-designed altitude, the altitude of the i th flying robot, and the altitude of the suspended load, respectively. ψ in Equation 4 is a dissipative function.

The general description of the interaction force is

$$\mathbf{f}_i = \mathbf{h}_i(\mathbf{q}_i, \mathbf{x}_i, \mathbf{x}_j, \dot{\mathbf{q}}_i, \dot{\mathbf{x}}_i, \dot{\mathbf{x}}_j), i = 1, 2, \dots, n+1. \quad (6)$$

Considering the perception range of the onboard sensors, interactive force is generated within the perception range of the onboard sensors of robots as

$$\mathbf{h}_i(\mathbf{q}_i, \mathbf{x}_i, \mathbf{x}_j, \dot{\mathbf{q}}_i, \dot{\mathbf{x}}_i, \dot{\mathbf{x}}_j) = \begin{cases} \mathbf{h}_i(\mathbf{q}_i, \mathbf{x}_i, \mathbf{x}_j, \dot{\mathbf{q}}_i, \dot{\mathbf{x}}_i, \dot{\mathbf{x}}_j), \\ \|\mathbf{x}_i - \mathbf{x}_j\|_2 < r \\ \mathbf{0}, & \text{otherwise} \end{cases}, \quad (7)$$

where r denotes the perception range of flying robots. Based on the design of dissipative force, the dynamics of the cooperative robot system are further described as

$$\mathbf{M}(\ddot{\mathbf{q}}_i, \ddot{\mathbf{x}}_i) + \mathbf{G}(\mathbf{q}_i, \mathbf{x}_i) = \mathbf{u}_i. \quad (8)$$

A controller is designed to address the practical challenges encountered in cooperative transportation based on the

dissipative node-robot-load system in Equation 8. Each robot employs a control law that manages both load transportation and formation maintenance. Acceleration command of the i th flying robot is described as

$$\mathbf{a}_{\text{cmd},i} = \mathbf{a}_{\text{expected},i} + \frac{1}{m_i} \mathbf{u}_i + \frac{1}{m_i} \mathbf{f}_i + \mathbf{g}. \quad (9)$$

The complete algorithmic implementation is formalized in Algorithm 2.

Algorithm 2: Multi-UAV Cooperative Transportation Control

Data: Initial UAV positions $\mathbf{x}_i(0)$, velocities $\dot{\mathbf{x}}_i(0)$

Load state $\mathbf{x}_{n+1}(0)$, $\dot{\mathbf{x}}_{n+1}(0)$

Expected acceleration $\mathbf{a}_{\text{expected},i}$

Sensor perception range r

Result: Acceleration commands $\mathbf{a}_{\text{cmd},i}$

begin

Initialize virtual nodes:

foreach UAV $i \in [1, n]$ **do**

$\mathbf{q}_i \leftarrow \mathbf{x}_i + \frac{h_c - h_i}{h_i - h_{n+1}} (\mathbf{x}_i - \mathbf{x}_{n+1})$

$\mathbf{q}_{n+1} \leftarrow \mathbf{x}_{n+1}$

while $\neg \text{mission_complete}$ **do**

foreach UAV $i \in [1, n]$ **do**

$\mathbf{h}_i \leftarrow \mathbf{0}$

foreach Node $j \in [1, n+1]$ **do**

if $\|\mathbf{x}_i - \mathbf{x}_j\|_2 < r$ **then**

$\mathbf{u}_i \leftarrow \mathbf{h}_i(\mathbf{q}_i, \mathbf{x}_i, \mathbf{x}_j, \dot{\mathbf{q}}_i, \dot{\mathbf{x}}_i, \dot{\mathbf{x}}_j)$

$\mathbf{a}_{\text{cmd},i} \leftarrow \mathbf{a}_{\text{expected},i} + \frac{1}{m_i} \mathbf{u}_i + \frac{1}{m_i} \mathbf{f}_i + \mathbf{g}$

 Update state: $\dot{\mathbf{x}}_i \leftarrow \mathbf{a}_{\text{cmd},i}$,

$\dot{\mathbf{x}}_i \leftarrow \dot{\mathbf{x}}_i + \dot{\mathbf{x}}_i \Delta t$, $\mathbf{x}_i \leftarrow \mathbf{x}_i + \dot{\mathbf{x}}_i \Delta t$

 Update load state $\mathbf{x}_{n+1}$, $\dot{\mathbf{x}}_{n+1}$

To evaluate the effectiveness of the proposed multi-UAV cooperative transportation algorithm, we conducted simulations in the Gazebo environment using three quadrotor UAVs (See Figure 6-8). In this setup, the UAVs were required to collaboratively transport a payload while maintaining stability and coordination. The simulation environment provided a realistic physics-based platform, enabling us to validate both the trajectory planning and the cooperative control strategies. Through these experiments, we demonstrated that the proposed algorithm can achieve reliable payload transportation with three UAVs, ensuring robust performance in terms of stability, synchronization, and load distribution.

5 SOLUTION TO OUTDOOR MISSION 4

The UAV-based water transportation task can be organized into a systematic framework that integrates perception, suspended load handling, trajectory planning, and delivery. The system is anchored by two spatial references: a pickup

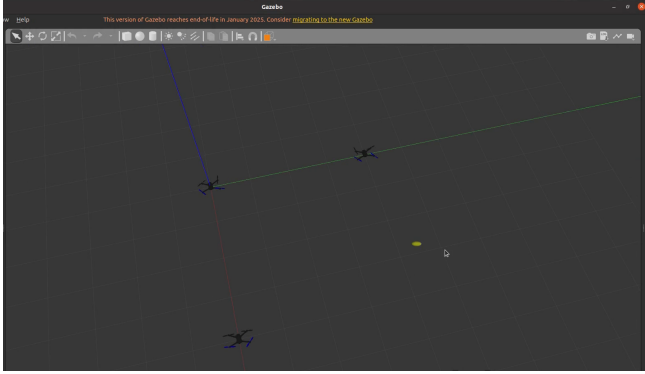


Figure 6: Initial condition of three UAVs

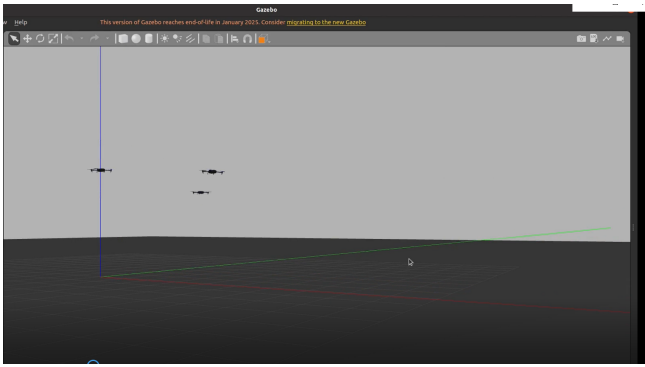


Figure 7: Middle stage of three UAVs

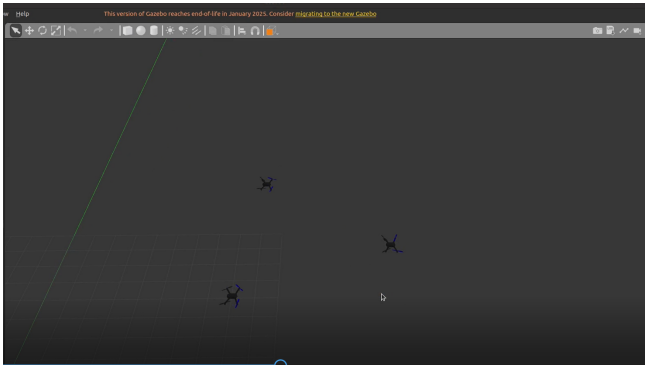


Figure 8: The stable configuration of three UAVs

point above the water source and a drop-off point at the designated container. To enable safe maneuvering, hovering waypoints are defined at both locations, and a minimum rope length of 2.5 m ensures that the bucket can be immersed without risk of rotor-surface interference. The UAV relies on inertial and altitude sensors, complemented by vision or ranging modules, to detect bucket contact and monitor filling. The mission objective is to achieve reliable pickup, stable transport, and accurate release. To this end, the framework is structured into three main modules: (i) suspended load dynam-

ics and swing suppression, (ii) rope tension monitoring with thrust margin assurance, and (iii) cooperative transport strategies. The complete algorithmic implementation is formalized in Algorithm 3.

Algorithm 3: UAV Load and Release Control

Data: UAV positions $\mathbf{x}_i$, velocities $\dot{\mathbf{x}}_i$
 Load position $\mathbf{x}_{load}$, release position $\mathbf{x}_{release}$
 Rope length L , load mass m_{load}
 Swing angle θ , angular velocity $\dot{\theta}$
Result: Acceleration commands $\mathbf{a}_{cmd,i}$

begin

Set swing suppression gains k_θ, k_ω based on
 $\omega_0 = \sqrt{g/L}$

foreach UAV $i \in [1, n]$ **do**

$\mathbf{w}_{load,i} \leftarrow \mathbf{x}_{load} + [0, 0, h_{hover}]^T$

$\mathbf{w}_{release,i} \leftarrow \mathbf{x}_{release} + [0, 0, h_{hover}]^T$

Initialize trajectory planner for $\mathbf{x}_{ref,i}(t)$,

$\dot{\mathbf{x}}_{ref,i}(t), \ddot{\mathbf{x}}_{ref,i}(t)$

while mission not complete **do**

foreach UAV $i \in [1, n]$ **do**

switch mission.state **do**

case Load **do**

Move to $\mathbf{w}_{load,i}$, descend,
 monitor loading, lift

case Transporting **do**

$\mathbf{a}_{cmd,i} \leftarrow \ddot{\mathbf{x}}_{ref,i} - k_\theta \theta_i - k_\omega \dot{\theta}_i$

$T_i \leftarrow m_{load}(g + a_{z,i}) + m_{load} \frac{v_{1,i}^2}{L}$

if $T_i > \gamma(m_{uav,i} + m_{load})g$ **then**
 Adjust acceleration to reduce
 tension

case Release **do**

Move to $\mathbf{w}_{release,i}$, descend,
 monitor loading, lift

if cooperative mode **then**

Compute shared $\mathbf{a}_{ref}$, corrections

$\Delta \mathbf{a}_i$

$\mathbf{a}_{cmd,i} \leftarrow \mathbf{a}_{ref} + \Delta \mathbf{a}_i$

Apply $\mathbf{a}_{cmd,i}$ and update UAV state

Update load state

5.1 Suspended Load Dynamics and Swing Suppression

The suspended bucket forms a classic underactuated system, in which UAV motion directly excites oscillations through the rope. This coupling makes the UAV-bucket system more complex than free-flight dynamics, as horizontal commands induce pendulum-like motions at the natural frequency $\omega_0 = \sqrt{g/L}$. Without compensation, these oscillations degrade delivery precision, increase mission time, and may destabilize the UAV if resonance is excited [9].

A small-angle model of the suspended load yields

$$\ddot{\theta} + 2\zeta\omega_0\dot{\theta} + \omega_0^2\theta = -\frac{1}{L}a_{\perp} \quad (10)$$

where θ is the swing angle and $a_{\perp}$ the UAV's lateral acceleration. This equation shows that sudden or aggressive maneuvers are the main source of excitation. To counteract this, a swing suppression strategy is introduced. The commanded acceleration is modified as

$$a_{\text{cmd}} = a_{\text{traj}} - k_{\theta}\theta - k_{\omega}\dot{\theta} \quad (11)$$

with k_{θ} and k_{ω} tuned to the pendulum frequency. The correction applies negative feedback on both displacement and velocity of the swing, creating an active damping effect.

In practice, swing estimation can be achieved using on-board IMU data, visual tracking of the bucket, or indirect inference from rope tension variations. Implementation requires no additional actuators, making it lightweight and feasible. During critical stages—such as lifting from water and turning near obstacles—this strategy significantly reduces swing amplitude, leading to more stable transport and more accurate drop-off.

5.2 Rope Tension Monitoring and Thrust Margin Assurance

Swing suppression alone is insufficient without monitoring the forces transmitted through the rope. Rope tension is the most direct indicator of load safety, as it reflects both static payload weight and dynamic excitations. For a suspended mass m , the approximate tension can be expressed as

$$T \approx m(g + a_z) + m\frac{v_{\perp}^2}{L} \quad (12)$$

where a_z is vertical acceleration and $v_{\perp}$ the lateral velocity of the bucket. The second term illustrates how oscillations amplify tension beyond static weight. This effect is especially pronounced when the UAV accelerates laterally or in the presence of crosswinds.

To guarantee feasibility, the UAV must satisfy a thrust margin criterion:

$$T_{\text{max}} \geq \gamma(m_{\text{uav}} + m)g \quad (13)$$

where T_{max} is maximum thrust and γ a safety factor typically chosen between 1.5 and 2.0. This ensures sufficient reserve thrust for disturbance rejection and maneuvering.

In practice, tension can be estimated via load-cell sensors embedded at the rope attachment or inferred from UAV motor current combined with dynamics models. Real-time tension monitoring serves two purposes: (i) preventing overload that may exceed actuator limits, and (ii) detecting anomalies such as sudden reductions (indicating payload release) or spikes (indicating unsafe maneuvers) [10]. By integrating predictive thrust margins with continuous monitoring, this module ensures that UAV operations remain within a safe and reliable envelope.

5.3 Cooperative Multi-UAV Transport

When payload demands or environmental disturbances exceed the capability of a single UAV, cooperative transport becomes a natural extension. Recent work has demonstrated that cooperative aerial transport is particularly effective for handling larger payloads while maintaining formation stability and actively suppressing load swing [11]. By connecting multiple UAVs to the bucket via crossbars or dual suspension points, the system distributes the total load and constrains lateral oscillations, thereby improving both efficiency and robustness.

Coordination is achieved through unified command laws. Each UAV follows a shared reference trajectory, while adjustments are applied to account for load-sharing geometry:

$$a_{\text{cmd},i} = a_{\text{ref}} + \Delta a_i \quad (14)$$

where a_{ref} is the common trajectory and Δa_i the corrective allocation for UAV i . This framework prevents internal conflicts and ensures synchronized motion.

Beyond load capacity, cooperative transport enhances overall reliability. If one UAV degrades or fails, the remaining units can stabilize the load temporarily, providing time for controlled release or safe landing. Such cooperative schemes are particularly valuable in strong-wind environments, where lateral stability is difficult for a single vehicle. Moreover, the framework is scalable: with precise formation control, more than two UAVs can collaborate to transport a larger bucket.

Through the integration of the above three modules, the load and release mission achieves both robustness and adaptability. Suspended load dynamics modeling ensures effective swing suppression, rope tension monitoring guarantees safe thrust margins, and cooperative extensions expand the operational envelope. Together, these components form a cohesive solution capable of reliable pickup, stable transport, and precise delivery across a wide range of operational scenarios.

6 CONCLUSION

This Technical Description Paper presents the CW-Lion team's solutions for the IMAV 2025 outdoor competition, leveraging a robust quadrotor platform equipped with GNSS, LiDAR, cameras, and advanced computing for autonomous operations.

For Mission 1, Fast-LIVO2 SLAM with LRU map management enables real-time 3D mapping with efficient data transmission. Mission 2 achieves precise person identification through a two-stage YOLOv8-CLIP pipeline, balancing recall and accuracy. Mission 3 introduces a virtual-node Lagrangian framework for multi-UAV cooperative transportation, ensuring stable payload handling and scalability, validated via Gazebo simulations. Mission 4 addresses water transportation with swing suppression, tension monitoring, and cooperative strategies for reliable delivery.

Preliminary results indicate high mapping accuracy, robust target detection, stable payload transport, and precise

load release. This integrated approach demonstrates a versatile, high-performance system tailored to the competition's demands, with potential for broader search and rescue applications.

REFERENCES

- [1] Chunran Zheng, Wei Xu, Zuhao Zou, Tong Hua, Chongjian Yuan, Dongjiao He, Bingyang Zhou, Zheng Liu, Jiarong Lin, Fangcheng Zhu, Yunfan Ren, Rong Wang, Fanle Meng, and Fu Zhang. Fast-livo2: Fast, direct lidar-inertial-visual odometry. *IEEE Transactions on Robotics*, 41:326–346, 2025.
- [2] Chunran Zheng, Qingyan Zhu, Wei Xu, Xiyuan Liu, Qizhi Guo, and Fu Zhang. Fast-livo: Fast and tightly-coupled sparse-direct lidar-inertial-visual odometry. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4003–4009, 2022.
- [3] Bingyang Zhou, Chunran Zheng, Ziming Wang, Fangcheng Zhu, Yixi Cai, and Fu Zhang. Fast-livo2 on resource-constrained platforms: Lidar-inertial-visual odometry with efficient memory and computation. *IEEE Robotics and Automation Letters*, 10(8):7931–7938, 2025.
- [4] Chungge Bai, Tao Xiao, Yajie Chen, Haoqian Wang, Fang Zhang, and Xiang Gao. Faster-liv: Lightweight tightly coupled lidar-inertial odometry using parallel sparse incremental voxels. *IEEE Robotics and Automation Letters*, 7(2):4861–4868, 2022.
- [5] R. Varghese and S. M. YOLOv8: A novel object detection algorithm with enhanced performance and robustness. In *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*, pages 1–6, 2024.
- [6] Fatih Cagatay Akyon, Sinan Onur Altinuc, and Alptekin Temizel. Slicing aided hyper inference and fine-tuning for small object detection. *arXiv preprint arXiv:2202.06934*, 2022.
- [7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 18–24 Jul 2021.
- [8] Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. YOLO-World: Real-time open-vocabulary object detection. *arXiv preprint arXiv:2401.17270*, 2024.
- [9] Feng Ding, Chong Sun, and Shunfan He. Anti-swing control for quadrotor-slung load transportation system with underactuated state constraints. *Sensors*, 23(21), 2023.
- [10] Fatima Oliva-Palomo, Diego Mercado-Ravell, and Pedro Castillo. Aerial transportation control of suspended payloads with multiple agents. *Journal of the Franklin Institute*, 361(7):106787, May 2024.
- [11] Elia Costantini, Emanuele Luigi de Angelis, and Fabrizio Giulietti. Two-time-scale cooperative uav transportation of a cable-suspended load: A minimal swing approach. *Drones*, 9(8), 2025.