Sensors and Materials, Vol. 37, No. 9 (2025) 4227–4247
MYU Tokyo

S & M 4178

Autonomous Obstacle Avoidance Path Programming Algorithm and Flight Validation for Fixed-wing Unmanned Aerial Vehicles in Formation

Meng Tse Lee,[1] Ming-Lung Chuang,[2*] and Po-Hsuan Yu[1]

[1]Department of Automation Engineering, National Formosa University,
No. 64, Wunhua Rd., Huwei Township, Yunlin County 632301, Taiwan R.O.C.
[2]Department of Power Mechanical Engineering, National Formosa University,
No. 64, Wunhua Rd., Huwei Township, Yunlin County 632301, Taiwan R.O.C.

(Received July 16, 2025; accepted September 22, 2025)

Keywords: fixed wing, path planning, obstacle avoidance, edge computing

With the continuous advancement of unmanned aerial vehicle (UAV) technology, UAVs have been widely adopted in both civilian and military applications, including geographic monitoring, disaster relief, aerial surveillance, logistics delivery, and tactical reconnaissance. However, single UAVs are increasingly unable to meet the demands of modern complex or large-scale missions owing to inherent limitations in flight endurance, sensing range, and processing capability. Consequently, deploying multiple UAVs for cooperative operations (i.e., UAV swarms) has become a critical development trend. During actual flight missions, whether operating individually or in swarms, UAVs may encounter unexpected no-fly zones or obstacles. Failure to reprogram and adjust flight paths in real time significantly reduces mission success rates and jeopardizes operational safety. To address this challenge, we propose a dynamic obstacle-avoidance path programming method based on the A* algorithm, integrated with onboard perception data. This system enables UAVs to autonomously calculate alternative bypass routes when encountering no-fly zones or obstacles and return to the original navigation waypoint afterward to ensure mission continuity. To enhance real-time performance and energy efficiency, the system utilizes a low-power onboard edge computing platform to perform immediate path reprogramming and rapidly transmit the updated route information to the flight controller for execution, thereby considerably reducing its dependence on ground stations and minimizing communication latency. Experimental results demonstrate that the proposed system can effectively perform real-time obstacle avoidance and maintain smooth flight trajectories across various simulated obstacle scenarios, showcasing excellent flexibility and reliability. The outcomes of this study significantly strengthen UAVs' autonomous decision-making capabilities in dynamic environments, enhancing safety and operational efficiency during complex missions. This approach holds substantial potential for future applications in multi-UAV collaborative inspections, real-time disaster area monitoring, and high-risk area operations.

*Corresponding author: e-mail: d1075101@gm.nfu.edu.tw
https://doi.org/10.18494/SAM5849

ISSN 0914-4935 © MYU K.K.
https://myukk.org/

1. Introduction

In recent years, unmanned aerial vehicle (UAV) technology has rapidly advanced, becoming a crucial representative of intelligent aerial platforms. Governments and industries worldwide have actively invested in the development and promotion of UAV systems.(1,2) With their high maneuverability, operational flexibility, and relatively low cost, UAVs have been widely applied in various civilian and military domains, including geographical monitoring, precision agriculture, disaster relief, infrastructure inspection in energy and transportation, commercial aerial photography, real-time logistics delivery, and unmanned combat aerial vehicles.

As the scope of UAV applications continues to expand, the demand for enhanced autonomous flight capabilities and intelligent decision-making is increasingly critical. Particularly in complex or high-risk environments, UAVs must be equipped with real-time, reliable navigation and obstacle avoidance mechanisms to ensure operational safety and mission success.(3) However, during mission execution, UAVs often face sudden no-fly zones and unexpected obstacles or must share airspace with other aerial vehicles. Without the ability to reprogram paths in real time, UAVs may experience collisions, mission interruptions, or severe safety incidents.

According to their flight characteristics, UAVs can be broadly classified into multi-copters and fixed-wing UAVs.(4) Multi-copters, despite their low speed, are capable of hovering and exhibit flexible multi-directional control and sharp turning capabilities, making them suitable for high-precision positioning and close-range obstacle avoidance tasks. In contrast, fixed-wing UAVs offer long endurance, high speed, and excellent range capabilities but lack hovering and sharp turning abilities. This limitation significantly restricts their maneuverability when facing obstacles and imposes high demands on path reprogramming and real-time obstacle avoidance.

Currently, most UAV systems rely on ground control stations to perform path programming after detecting obstacles or no-fly zones and upload the updated routes to the flight controllers for execution.(5) However, this approach suffers from communication delays, bandwidth limitations, and a heavy reliance on ground infrastructure. In dynamic environments or multi-UAV cooperative operations, these drawbacks often result in decision failures, impacting overall operational efficiency and safety. Thus, developing onboard path programming systems capable of autonomous perception, real-time decision-making, and rapid reprogramming has become a key technology for enhancing UAV autonomy.

Among path programming methods, the A* algorithm is widely used for UAV navigation tasks in static environments owing to its determinism and ability to find the shortest paths.(6) Nevertheless, in dynamic or complex terrain environments, traditional A* may exhibit low computational efficiency or produce paths with excessive sharp turns. To address these issues, various improvements have been proposed. For instance, Zhang *et al.* developed the multi-strategy fusion with minimum turning points optimization (MSF-MTPO) method, combining adaptive neighborhood A* search with smoothing strategies to reduce path turning points and enhance trajectory smoothness.(7) Chen *et al.* proposed a hybrid AANR*-DWA method, integrating an improved RRT* (AANR*) with the dynamic window approach (DWA), which balances global shortest paths and local real-time obstacle avoidance, significantly improving practical performance.(8)

In terms of multi-UAV cooperative formation control, the leader–follower strategy has become one of the mainstream approaches owing to its simple structure, clear control logic, and ease of implementation and expansion. In this strategy, the leader UAV is responsible for global path programming and navigation, while follower UAVs dynamically adjust their positions on the basis of the leader's real-time position, heading, and velocity information to maintain formation integrity.(9,10) However, this strategy heavily depends on the leader; if the leader encounters sensor or communication failures, the entire formation may collapse.(11) To mitigate this challenge, Liu *et al.* combined an improved A* algorithm, Q-learning, and genetic algorithms to propose an enhanced leader–follower strategy.(12) This allows the leader to update the formation in real time during obstacle avoidance and immediately distribute updated navigation information to followers, improving formation stability and adaptability.

Additionally, the PANTHER framework utilizes a perception-guided, dynamic, real-time path reprogramming method that maintains obstacle tracking even in high-speed environments, making it suitable for missions involving strong winds and complex terrains.(13) The hyper-efficient perception and planning (HEPP) system proposed by Lu *et al.* focuses on high-speed flight scenarios and incorporates incremental map generation, topological search, and time-optimal trajectory programming, reducing system latency to approximately 20% of traditional methods while maintaining near-optimal flight performance.(14)

In response to these challenges, we propose a real-time obstacle avoidance and dynamic path programming system based on edge computing, integrating the computational core directly into the onboard platform. This design enables UAVs to receive perception data in real time, autonomously compute alternative routes, and return to the original navigation path after successful avoidance. To address energy efficiency and endurance requirements, the system employs a microcontroller unit (MCU) as the core processor instead of high-power embedded platforms (such as Nvidia Jetson or Raspberry Pi), significantly reducing power consumption and extending flight duration, especially for fixed-wing UAV applications.

In summary, UAV obstacle avoidance and path reprogramming technologies have evolved from traditional static algorithms to integrated multi-strategy approaches combining hybrid path searching, intelligent decision-making, and hardware adaptability, effectively addressing fast-changing and highly complex mission demands. On the basis of this, we integrated the A* search algorithm activated when sensors detect obstacles with fixed-wing UAV swarm formation control techniques to develop a cooperative obstacle avoidance system characterized by real-time performance, low power consumption, and high flexibility. This system is expected to further enhance UAV autonomy, reliability, and operational efficiency in dynamic environments, laying a technological foundation for future multi-UAV cooperative missions and intelligent applications.

2. Experimental Vehicle and System Architecture

2.1 System architecture

In this study, a formation flight scenario involving two fixed-wing UAVs was designed to simulate situations where the UAV swarm encounters sudden no-fly zones either collectively or individually during flight. The scenarios considered include cases where the entire swarm must simultaneously avoid a no-fly zone as well as cases where only a single wingman UAV needs to perform obstacle avoidance. All no-fly zones are dynamically added during flight by the ground control station and transmitted in real time to the UAV mission lists.

Since it is necessary to rapidly update the flight paths and complete avoidance maneuvers during flight, each UAV must be capable of real-time autonomous decision-making and path reprogramming in the air, without continuous reliance on centralized computations and path updates from the ground station. To achieve this objective, we developed an edge computing module (companion computer) that integrates the A* search algorithm with formation control logic, enabling UAVs to independently perform perception, decision-making, and path reprogramming during flight. This architecture not only enhances system autonomy but also effectively reduces communication delays and dependence on external infrastructure, making it especially suitable for highly dynamic and high-risk environments.

The overall system architecture, illustrated in Fig. 1, consists of two main components: the ground control station and the UAVs with the edge computing module. The ground control station is responsible for monitoring real-time UAV statuses (such as position, speed, and formation integrity) and can dynamically add or update no-fly zone information during mission execution, transmitting updates to the UAV swarm via a wireless link. Inter-UAV communication is achieved using RF modules configured in a mesh topology, enabling the real-time sharing of positional information and collaborative command exchange among UAVs. This further strengthens formation stability and global situational awareness. Each UAV is equipped with an edge computing module, which performs the following key functions:

(1) It integrates and manages positional information received from other UAVs in the formation, ensuring coordinated and dynamic collaboration within the swarm.

(2) It dynamically calculates and updates the desired formation positions for each UAV based on the current geometric configuration, and transmits these to the flight controller to control attitude and heading.

(3) Upon receiving no-fly zone information from the ground control station, the module autonomously determines whether the UAV's current path intersects with the restricted area. If so, it immediately employs the built-in A* search algorithm to compute a new safe path and transmits this updated route to the flight controller, enabling real-time obstacle avoidance and preventing unintended entry into no-fly zones or collisions with obstacles.

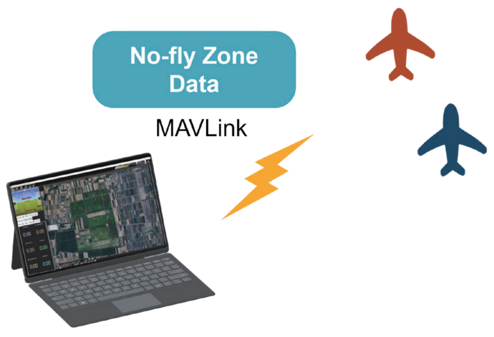

Fig. 1. (Color online) System architecture diagram.

This system architecture, which combines edge computing with distributed cooperative control, significantly enhances mission flexibility, reliability, and scalability for UAV swarms. It provides a robust technical foundation for future applications in multi-UAV autonomous cooperation, disaster response, and complex environmental monitoring.

2.2 Experimental vehicle

The experimental platform utilized in this study is the fixed-wing UAV "Sky Surfer," as shown in Fig. 2. The Sky Surfer UAV has a wingspan of 1400 mm, a fuselage length of 925 mm, and an approximate weight of 1.2 kg. It is powered by an electric brushless motor with a propeller and achieves a typical cruising speed of around 12–18 m/s. This aircraft features a lightweight structure, excellent flight stability, and long endurance, making it well suited for formation flight and obstacle avoidance technology validation.

The overall system architecture, illustrated in Fig. 3, consists of two core components: the onboard flight controller and the companion (edge computing) module. The flight controller

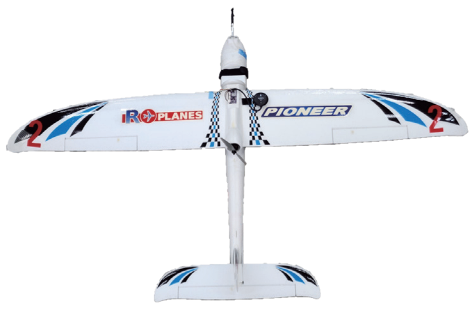

Fig. 2. (Color online) Sky Surfer experimental vehicle.

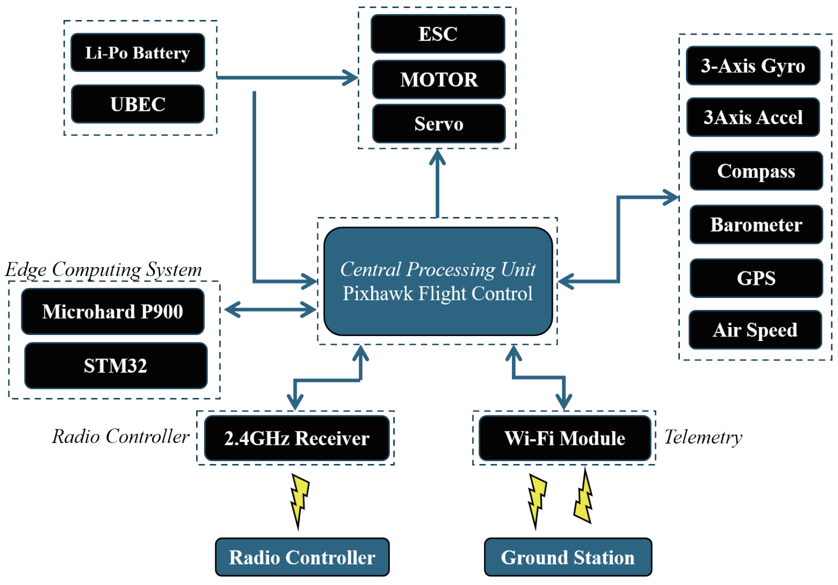

Fig. 3. (Color online) Subsystem of vehicle architecture diagram.

integrates a barometer, a gyroscope, an accelerometer, and other inertial measurement unit sensors, and it can further incorporate external modules such as a GPS receiver and an airspeed sensor. This setup provides high-precision positioning and the accurate estimation of flight attitude. The primary functions of the flight controller include attitude stabilization, speed control, and navigation tasks. It ensures safe and stable flight through closed-loop control mechanisms.

For the edge computing module, we adopted the STM32H747 as the main computational core. The STM32H747 is a high-performance dual-core MCU characterized by strong computational capability and low power consumption, making it highly suitable for real-time processing on energy-constrained UAV platforms. The edge computing module is responsible for receiving real-time positional data from the leader UAV and other wingman UAVs. It then calculates each UAV's navigation path and target position on the basis of current environmental data and formation control strategies.

Additionally, the edge computing module transmits the generated navigation control commands to the flight controller via the Micro Air Vehicle Link (MAVLink) communication protocol. MAVLink is a lightweight communication protocol widely used in both commercial and research UAV platforms, supporting various data packets and command types. Upon receiving the updated waypoints, the flight controller autonomously guides the UAV to accurately reach the designated target positions while maintaining the formation structure. The physical configuration of the edge computing module is shown in Fig. 4. The design emphasizes modularity, lightweight construction, and ease of maintenance. In addition to the main STM32H747 core, it includes a power management module, data transmission interfaces, and essential communication modules to ensure stable computation and communication capabilities during extended flight operations.

In summary, by integrating a high-performance MCU with a precise flight controller, the UAV platform developed in this study successfully achieves real-time autonomous obstacle avoidance and formation cooperation. This provides a solid technical foundation for subsequent experiments involving dynamic obstacle avoidance in multi-UAV formations.

3. Design and Principle of Path Programming

The overall system architecture developed in this study can be categorized into two core functional modules: formation control and no-fly zone avoidance. For formation control, all UAVs within the swarm continuously share real-time positional information through inter-UAV wireless communication. The edge computing module (companion computer) integrates these

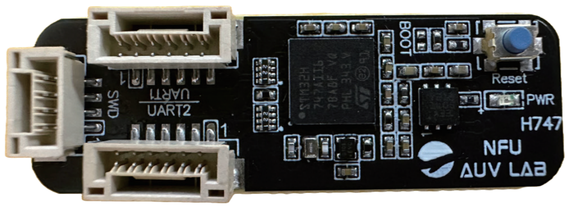

Fig. 4. (Color online) Edge computing system.

data and calculates the corresponding formation waypoints. On the basis of each UAV's assigned role within the formation (e.g., leader or wingman), the edge computing module dynamically generates specific target waypoints to maintain the predefined geometric configuration and ensure stable flight. Once calculated, the navigation control commands are transmitted in real time to the flight controller via the MAVLink communication protocol, enabling precise attitude and heading adjustments to ensure that each UAV accurately follows and maintains the formation.

Regarding the no-fly zone avoidance strategy, in this study, we adopted the A* search algorithm, a type of heuristic algorithm widely used in dynamic path programming for robotics and UAVs, owing to its effective evaluation function that considers both cost and heuristic values to efficiently determine the optimal path from the start point to the target point. In the scenario designed for this study, when a UAV receives updated no-fly zone information from the ground control station during flight, the edge computing module immediately assesses whether the UAV's current trajectory would intersect the restricted area. If so, the A* algorithm is triggered to perform real-time path reprogramming.

Specifically, the starting point for the A* algorithm is set as the UAV's current real-time position that requires obstacle avoidance, while the target point is the originally assigned mission waypoint, provided it lies outside the no-fly zone. If the original mission waypoint is located within a no-fly zone, the system automatically searches for the next waypoint in the mission waypoint list to serve as the new target, thereby ensuring flight safety and preventing entry into restricted areas. The updated navigation path is then transmitted in real time to the flight controller for execution, allowing the UAV to smoothly bypass obstacles and rejoin the mission trajectory in a safe zone.

By integrating formation control with real-time obstacle avoidance strategies, the proposed multi-UAV cooperative system is capable of autonomous decision-making and collaborative flight operations in dynamic environments, significantly enhancing swarm robustness and overall mission efficiency.

3.1 A* Search

In this study, when a UAV flies from the start point (point S) to the end point (point E), it must activate an obstacle avoidance programming mechanism in real time to regenerate a safe flight path if it detects that it will traverse one or more no-fly zones along the route.

As illustrated in Fig. 5, upon detecting a no-fly zone ahead, the edge computing module sets the UAV's current real-time position as the point S for the A* algorithm and designates the final navigation target (point E) as the goal point, provided that it is located outside the no-fly zone.

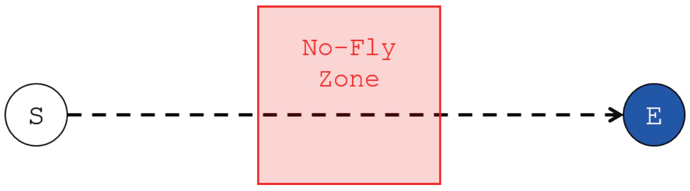

Fig. 5. (Color online) Path through no-fly zone.

By recalculating an alternative path, the UAV can effectively bypass the no-fly zone and, after completing the avoidance maneuver, smoothly return to the original mission trajectory before finally arriving at the designated end point. This approach ensures overall flight safety and the integrity of mission execution.

At this stage, the system performs an outward expansion of the boundary vertices of the no-fly zone, generating additional corner points by extending these vertices outward. This approach ensures that the reprogrammed path can safely bypass the no-fly zone at a sufficient distance.

The expansion distance can be parameterized and adjusted on the basis of the dynamic characteristics of different UAV platforms, such as turning radius, control response speed, and flight stability. This parameterization enhances the flexibility and adaptability of the obstacle avoidance programming. A schematic of the corner point extension strategy, illustrating how the path circumvents the no-fly zone with a safety margin, is shown in Fig. 6.

After completing the addition and outward extension of the corner points, the system generates a continuous path starting from the point S, sequentially connecting all generated corner points to the point E. During this path generation process, strict constraints are enforced to ensure that the path does not cross into any no-fly zones, thereby guaranteeing flight safety.

The final generated path represents a feasible and safe bypass trajectory for the UAV, allowing it to effectively avoid the no-fly zones and ultimately reach the designated target point. A schematic illustration of the final generated path, clearly showing the connected corner points and the resulting programmed flight trajectory, is presented in Fig. 7.

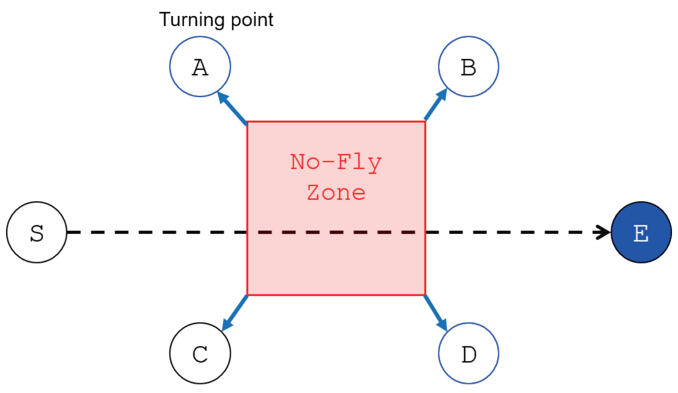

Fig. 6. (Color online) Addition of turning point.

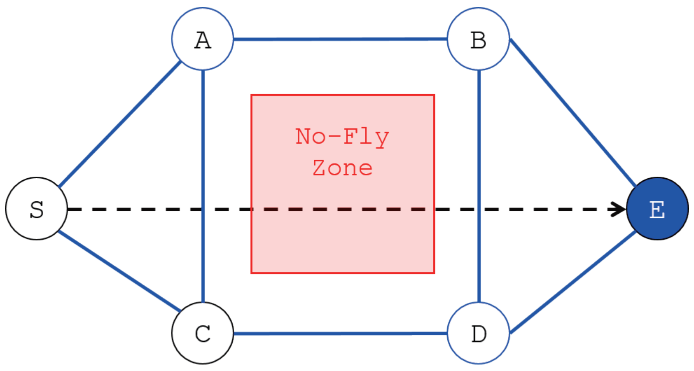

Fig. 7. (Color online) Possible path.

Next, the system performs path searching and programming using the A* algorithm. The core evaluation function of A*, denoted as $f(n)$, is shown in Eq. (1). This function is composed of two parts: the actual cost $g(n)$ and the heuristic cost $h(n)$.

Specifically, $g(n)$ represents the accumulated actual travel cost from the S point to any node n, reflecting the cost already incurred along the traversed path. On the other hand, $h(n)$ estimates the remaining cost from node n to the point E, typically calculated as the Euclidean distance, serving as the heuristic function to approximate the shortest potential remaining distance to the goal.

By adopting the strategy $f(n) = g(n) + h(n)$, the A* algorithm simultaneously considers both the actual cost incurred and the estimated future cost, enabling it to identify the optimal feasible path with the minimum total cost among numerous possible trajectories.

$$f(n) = g(n) + h(n) \tag{1}$$

The execution steps of the A* search algorithm are as follows:

Step 1: Add the start point to the open set, denoted as Open().

Step 2: Check whether the goal point is in Open(), or if Open() is empty. If either condition is true, proceed to Step 4. Otherwise, continue to the next step.

Step 3: From Open(), select the node C with the lowest evaluation function value $f(n)$. Remove C from Open() and expand all its child nodes m() that are not in Open(), the closed set Closed(), or the dead-end set DE().

Step 4: If the set m() is empty, add C to DE(). Otherwise, add C to Closed() and add m() to Open(). Then, return to Step 2.

Step 5: Check whether the goal point is in Open(). If yes, backtrack through Closed() to reconstruct the solution path and return the final path and its total cost. If not, it indicates that no feasible solution exists.

After path programming using the A* algorithm, the final optimized avoidance path is illustrated in Fig. 8. This path starts from the initial point S, sequentially passes through intermediate corner points C and D, and ultimately reaches the target point E.

Designed with both safety and efficiency in mind, the generated path effectively circumvents the no-fly zone while minimizing the overall flight distance. This ensures that the UAV can maintain stable and reliable navigation performance even in dynamic environments.

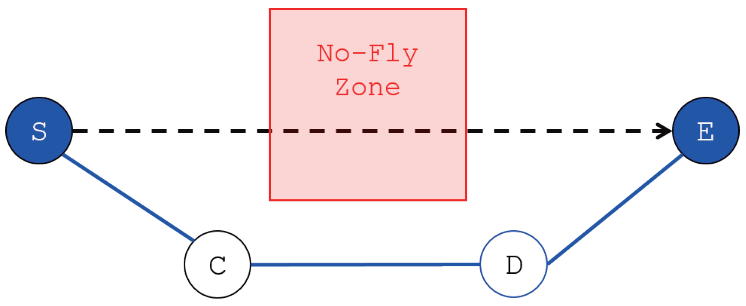

Fig. 8. (Color online) Final path to avoid no-fly zone.

3.2 Formation control

In the formation flight architecture designed in this study, the leader UAV is equipped with a low-power edge computing module that continuously requests critical flight information—including position, velocity, and attitude—from its flight controller via MAVLink message packets in real time.

The acquired data are first processed by the edge computing module onboard the leader UAV and then transmitted in real time to all wingman UAVs through a wireless edge communication module configured in a mesh topology.

After receiving flight information from the leader UAV, each wingman transmits the data to its onboard low-power edge computing module for processing. On the basis of the received formation data and its own current position, the module calculates the required navigation control commands.

Subsequently, the wingman uses MAVLink command packets to transmit these computed navigation commands to its flight controller, enabling automatic adjustments of flight attitude and heading to precisely reach the designated formation position.

Through this design, the entire formation can autonomously maintain its configuration and perform flight control without human intervention, achieving fully autonomous formation flight. This architecture significantly enhances the flexibility, reliability, and adaptability of the formation system, making it particularly suitable for complex environments and highly dynamic mission scenarios. The overall system workflow and architecture are illustrated in Fig. 9, which clearly depicts the information exchange, computational process, and control logic between the leader and the wingman.

In the formation position calculation framework designed in this study, the leader UAV periodically transmits flight parameters to the wingmen, including its own geographic coordinates (latitude and longitude), flight heading, and current flight altitude.

Upon receiving these coordinates, the wingmen use the World Geodetic System (WGS84) as the base geographic coordinate system. To facilitate precise subsequent calculations of distances and heading angles, the received WGS84 coordinates are first converted to the Universal Transverse Mercator (UTM) coordinate system.

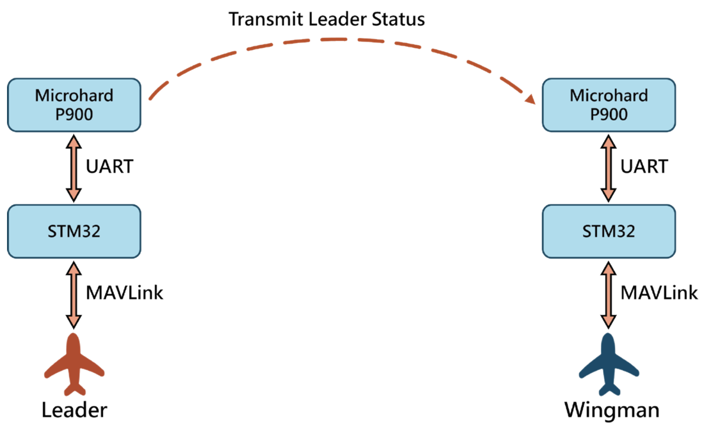

Fig. 9. (Color online) Formation flight control system architecture.

After completing the projection conversion, each wingman calculates its desired relative position within the formation based on the predefined relative spacing (e.g., formation separation distance) and relative direction angle. The target relative position is decomposed into local variations in the north and east directions, as shown in Eqs. (1) and (2).

Subsequently, these local variations are added to the leader's UTM coordinates, as shown in Eqs (3)–(5), to compute the target position that each wingman should reach within the formation.

After completing the calculations, each wingman converts the computed target UTM coordinates back to the WGS84 coordinate system for use by the navigation module. Additionally, since each wingman must maintain the same flight heading as the leader within the formation, there is no need for extra heading calculations; the heading is simply synchronized directly with that of the leader. Regarding altitude control, the target flight altitude for each wingman is calculated by adding a predefined vertical offset to the current altitude of the leader. This ensures the three-dimensional integrity and layered structure of the formation.

Finally, the wingman inputs the computed navigation position (including latitude, longitude, altitude, and heading) as the target waypoint into the navigation control program. Using the low-power edge computing module, real-time control commands are transmitted to the flight controller, enabling precise position control and fully autonomous formation flight.

$$d_{n-s} = dx\sin(hdg) + dy\cos(hdg), \tag{2}$$

$$d_{e-w} = dx\cos(hdg) - dy\sin(hdg), \tag{3}$$

$$target\_lat = leader\_lat + [(dn - s)/radius\_of\_earth], \tag{4}$$

$$target\_lon = leader\_lon + [(de - w)/radius\_of\_earth], \tag{5}$$

$$target\_alt = leader\_alt + dz, \tag{6}$$

To meet the control requirements of fixed-wing UAVs, we integrated both an L1 controller and a Total Energy Control System (TECS) for navigation and attitude control. The L1 controller primarily manages horizontal path navigation by converting the latitude and longitude coordinates of the start and target waypoints into horizontal lateral acceleration signals. This enables adjustments in roll and yaw, ensuring that the UAV can precisely follow the programmed flight path. In contrast, the TECS focuses on vertical axis control by balancing kinetic energy (speed) and potential energy (altitude) to manage pitch and throttle commands. This allows the UAV to smoothly achieve and maintain the desired altitude, while effectively compensating for external disturbances or altitude variation demands during flight.

On the basis of this framework, each fixed-wing wingman UAV inputs the target formation position and target velocity parameters calculated by its onboard low-power edge computing module into the navigation program. These commands are then transmitted to the flight controller via MAVLink packets, allowing the UAV to autonomously navigate to its designated position within the formation and maintain synchronized coordination with the leader UAV, thus achieving fully autonomous formation flight control.

After establishing the basic formation control, a Proportional-Integral-Derivative controller is introduced into the navigation control system to further refine and maintain the relative distance between the leader and the wingman UAVs. This control strategy integrates a trapezoidal integral formula, a saturation controller, and a switching controller, effectively improving the accuracy and stability of speed regulation while preventing excessive deviations caused by integral error accumulation.

Additionally, the system dynamically switches between different control modes and adjusts control gains on the basis of the current distance error magnitude, enabling rapid response and smooth convergence, thereby enhancing the overall robustness of formation maintenance.

During formation flight, the wingman continuously computes the real-time distance error $E(k)$ relative to the leader UAV and derives the speed increment $U(k)$, as shown in Eq. (7). Subsequently, this speed increment is combined with the leader UAV's current flight speed to calculate the target flight speed $V(k)$ for the wingman, as expressed in Eq. (8). With this design, the wingman automatically increases its speed when the distance error increases to catch up with the formation, and decreases its speed when too close, thus maintaining the prescribed formation position and ensuring the overall consistency and safety of the formation.

$$\begin{aligned} U(k) = K_p\,[E(K) - E(K-1) + K_i\,\{[\,E(K) - E(K-1)]/2\} \\ + K_d\,[E(K) - 2E(K-1) + E(K-2)] \end{aligned} \tag{7}$$

$$V(k) = leader\_v + U(k), \tag{8}$$

3.3 Integration of formation control and obstacle avoidance

After the design of the two core functionalities—namely, A* obstacle avoidance path programming and formation control—was completed, we integrated them into a single edge computing module for collaborative application. To achieve this integration, the STM32H747 module was adopted as the core computing platform. The STM32H747 features a dual-core architecture, comprising a Cortex-M7 core (operating at 480 MHz) and a Cortex-M4 core (operating at 240 MHz), offering high computational performance with low power consumption. These characteristics make it especially suitable for real-time UAV computing scenarios.

In this architecture, the Cortex-M7 core is dedicated to executing the A* algorithm for path searching and computation, ensuring the rapid generation of new safe paths when encountering no-fly zones. The Cortex-M4 core is responsible for handling formation control logic, data exchange with the flight controller, and real-time communication and coordination tasks. Data exchange and synchronization between the two cores are managed through shared memory.

From a program flow perspective, the system first uses the MAVLink communication protocol to retrieve the current mission waypoints and the list of known no-fly zones from the flight controller, storing this information in memory arrays. Once the UAV switches to the autonomous mission mode, the Cortex-M4 core continuously evaluates whether the flight path from the current position to the next waypoint intersects with any no-fly zones. If a potential intersection is detected, the M4 core transmits the no-fly zone information and target waypoint data to the M7 core via shared memory (as illustrated in Fig. 10), triggering the A* obstacle

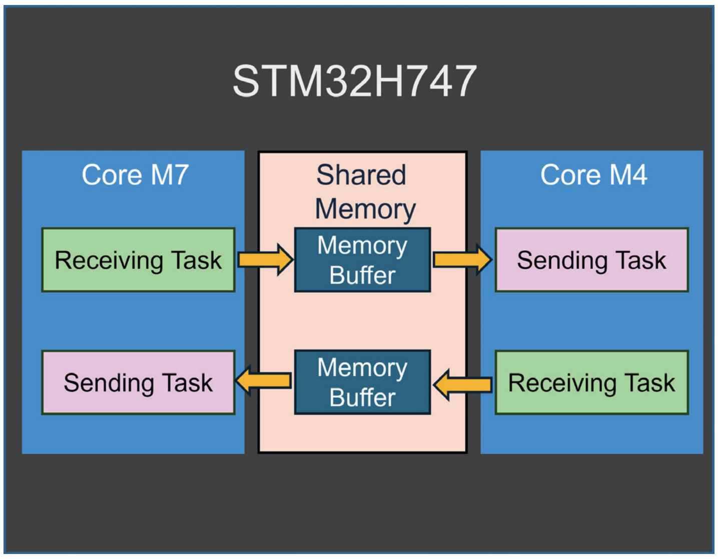

Fig. 10. (Color online) Data transmission between two cores.

avoidance computation. After completing the computation, the M7 core generates a new set of alternative waypoints and returns it to the M4 core through shared memory.

Upon receiving the new waypoints, the M4 core inserts them into the mission waypoint list and transmits them to the flight controller via the MAVLink protocol, instructing the UAV to execute the updated flight path and safely bypass the no-fly zones. Furthermore, to ensure multi-tasking and real-time execution, the M4 core operates under a real-time operating system (RTOS), managing multiple threads. This multi-threaded design enables simultaneous real-time communication with the flight controller, the calculation of formation positions, and the transmission of control commands, thereby guaranteeing the system's real-time responsiveness and operational stability.

The overall system control flow is illustrated in Fig. 11, clearly depicting the complete operational mechanism from data acquisition, evaluation, and computation to final command delivery.

In addition, if it is necessary to update no-fly zone data during UAV flight, the ground control station can directly transmit new no-fly zone information to the flight controller via the data link and then forward the update commands to the onboard edge computing module using the MAVLink routing mechanism.

In the system design of this study, the MAVLink routing configuration is as follows: the component ID of the edge computing module is set to 2, and that of the flight controller is set to 1. Both share the same system ID (System ID = 1). This configuration indicates that both components are considered part of the same vehicle system, with the edge computing module treated as an independent component within the system.

Therefore, when it is necessary to send update commands from the ground control station to the edge computing module, the existing data link between the ground station and the flight controller can be used. By specifying the target component ID as 2 in the command packet, the

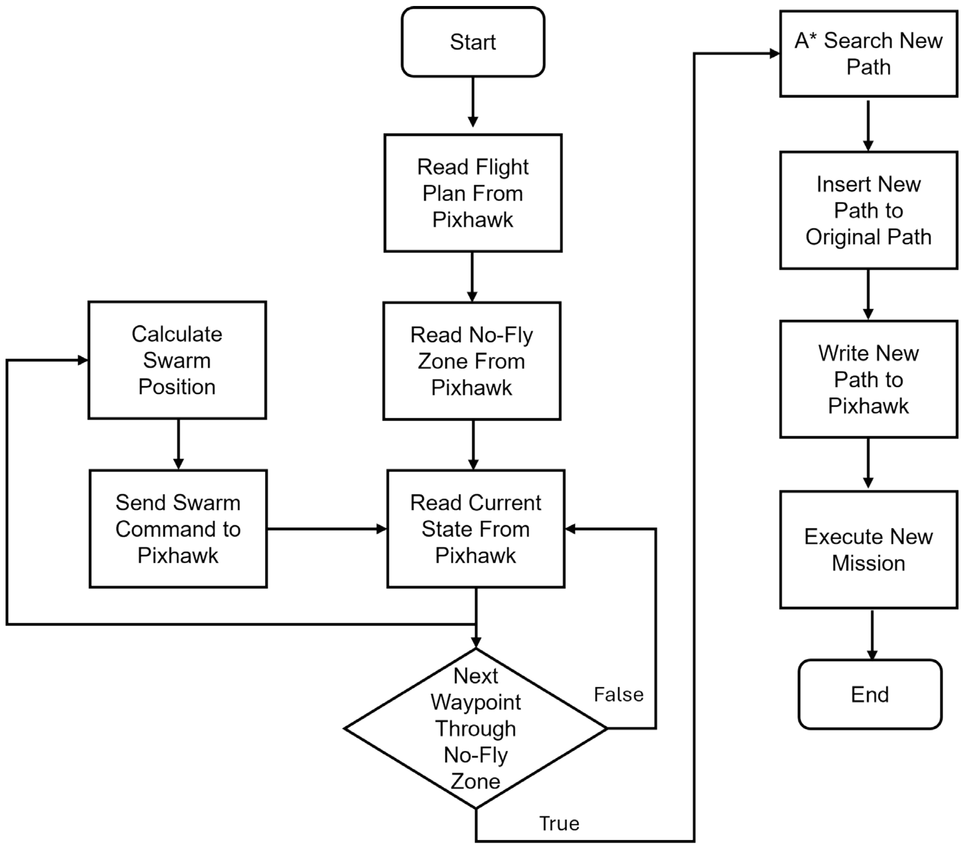
Start
Read Flight Plan From Pixhawk
Read No-Fly Zone From Pixhawk
Calculate Swarm Position
Send Swarm Command to Pixhawk
Read Current State From Pixhawk
Next Waypoint Through No-Fly Zone
False
True
A* Search New Path
Insert New Path to Original Path
Write New Path to Pixhawk
Execute New Mission
End

Fig. 11. System flow chart.

command is first received by the flight controller and then routed directly to the edge computing module, enabling seamless internal communication and data updates. The overall architecture and the data transmission flow among the ground control station, flight controller, and edge computing module are illustrated in Fig. 12.

Upon receiving the update command from the ground station, the M4 core in the edge computing module immediately initiates the command processing procedure, retrieves the latest no-fly zone data from the flight controller, and stores it in internal memory. This updated data can then be used by the A* algorithm for real-time path assessment and reprogramming. This mechanism ensures that the UAV can promptly avoid newly added or modified no-fly zones during the mission, significantly enhancing flight safety and mission reliability.

4. Path Programming and Experiment Results

In this study, the field flight experiments were divided into two main parts: a single-UAV obstacle avoidance experiment and a two-UAV formation obstacle avoidance experiment. This design aimed to validate the system's obstacle avoidance performance and robustness under different operational modes.

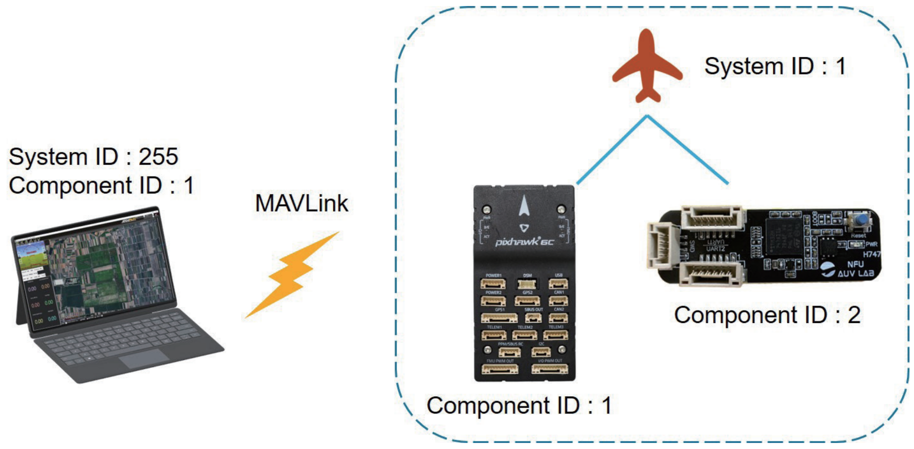

Fig. 12. (Color online) MAVLink routing.

In the experimental scenario, the UAVs execute a predefined autonomous flight route (as shown in Fig. 13), simulating a realistic mission flight process. During flight, the ground control station uploads new no-fly zone information in real time. This no-fly zone is intentionally designed to overlap part of the original autonomous flight path to test whether the edge computing module can promptly receive the updated no-fly zone information and utilize the A* algorithm to reprogram a safe bypass trajectory, thereby avoiding the no-fly zone and ensuring flight safety.

In the single-UAV experiment, the focus is on verifying whether an individual UAV can successfully receive the updated no-fly zone data and perform real-time path reprogramming. In contrast, in the two-UAV formation experiment, in addition to validating the obstacle avoidance function, it is essential to ensure that the wingman can maintain formation with the leader UAV according to the formation control logic, synchronously flying and jointly avoiding the no-fly zone. This verification effectively demonstrates the system's required real-time responsiveness, autonomy, and multi-UAV cooperative capability needed for practical mission scenarios.

4.1 Single-UAV obstacle avoidance path programming experiment

This experimental objective is to verify whether a single UAV can perform real-time path reprogramming and obstacle avoidance through the edge computing module when encountering newly added no-fly zones during its mission flight. The experimental procedure is as follows: after takeoff, the UAV autonomously flies along the predefined mission waypoints. During the flight, the ground control station uploads a new no-fly zone in real time (as shown in Fig. 14), which intentionally overlaps part of the originally programmed route.

Upon receiving the new no-fly zone data, the edge computing module must immediately assess whether the current flight path intersects with the restricted area. If an intersection is detected, the module activates the A* algorithm to perform path reprogramming, calculating a safe alternative route that avoids the no-fly zone. The updated route is then sent to the flight controller via MAVLink commands.

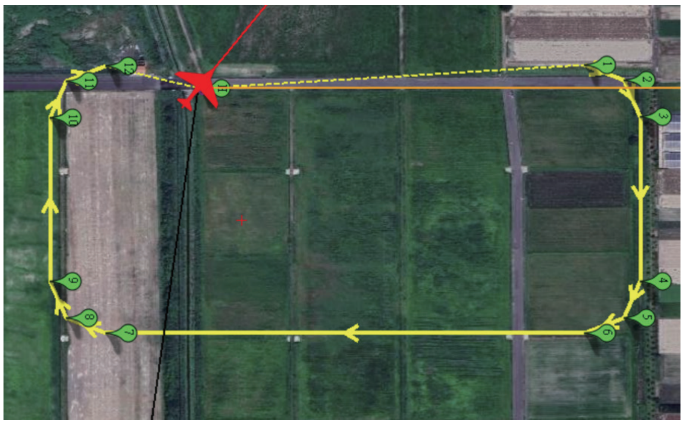

Fig. 13. (Color online) Default flight path.

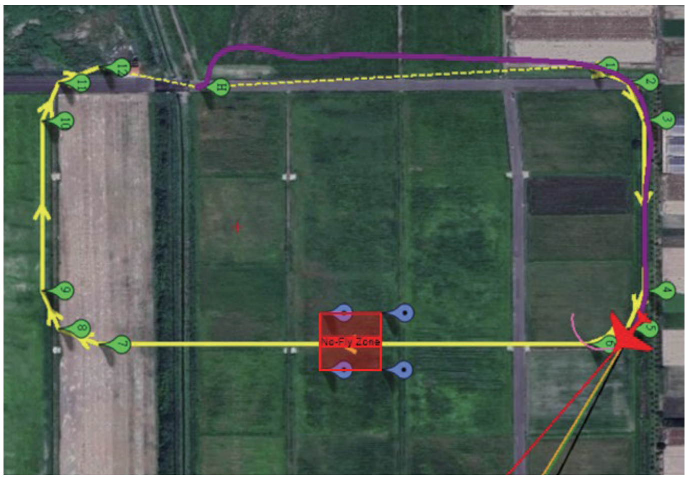

Fig. 14. (Color online) Uploading of new no-fly zone to UAV.

Ultimately, the UAV is expected to successfully bypass the no-fly zone according to the new waypoints and safely proceed to subsequent waypoints, thereby verifying the system's real-time obstacle avoidance capability and autonomous decision-making performance.

Figure 15 shows the mission waypoint data retrieved from the UAV by the ground control station after completing obstacle avoidance. It can be observed that the edge computing module successfully inserted two new corner waypoints into the original flight path and promptly updated these new waypoints to the flight controller, thereby guiding the UAV to execute the mission according to the newly programmed route. The actual flight trajectory executed by the UAV is depicted in Fig. 16. In this figure, the blue segments represent the flight path taken by the UAV when deviating from the original route and transitioning to the newly generated path.

It is clearly evident from the figure that the UAV was able to smoothly and accurately perform obstacle avoidance following the real-time reprogrammed path and seamlessly rejoin the subsequent mission route. This validates the effectiveness and stability of the proposed edge computing module in real-time obstacle avoidance decision-making and path control.

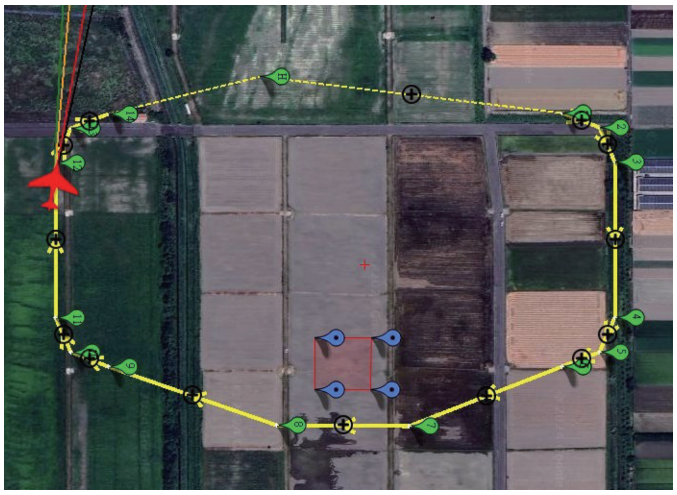
Fig. 15. (Color online) New path after avoidance of no-fly zone.

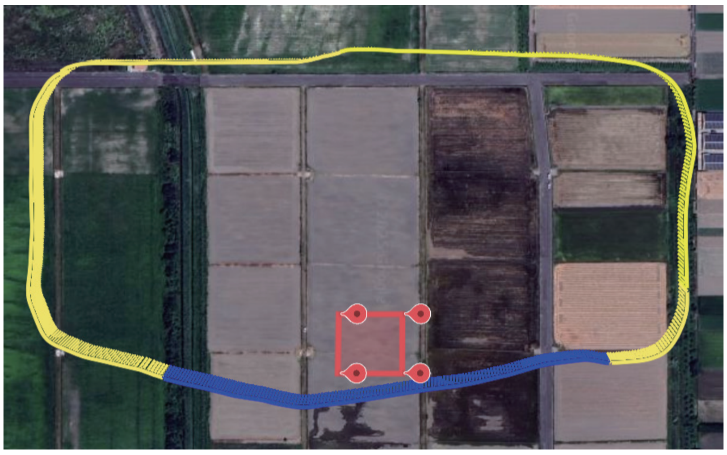
Fig. 16. (Color online) Track of experiment.

4.2 Multi-UAV formation obstacle avoidance path programming experiment

After successfully verifying the capability of the edge computing module to perform real-time path reprogramming and obstacle avoidance in the single-UAV experiment, we further validated the system under multi-UAV cooperative scenarios to evaluate the overall performance of the integrated formation control and real-time obstacle avoidance functions.

In this experiment, the flight path also followed the autonomous mission route shown in Fig. 13; however, this route was uploaded only to the leader UAV. The wingman UAVs entirely relied on the leader for formation flight, with their positions and attitudes dynamically calculated and controlled on the basis of the real-time position data transmitted from the leader's edge computing module according to the predefined formation settings. To ensure that the entire UAV formation can safely and synchronously avoid the no-fly zone during obstacle avoidance maneuvers—and to prevent situations where only the leader successfully avoids while the wingmen inadvertently enter the restricted area—the formation offset parameters were specially optimized and enlarged in this study to ensure that each wingman remains within a safe range after offsetting.

As shown in Fig. 17, the UAV formation enters the formation flight mode immediately after takeoff, with the wingmen dynamically adjusting their positions on the basis of real-time flight information transmitted by the leader. During flight, the ground control station uploads new no-fly zone information in real time and sends update commands to the leader's edge computing module via the MAVLink routing mechanism, instructing it to update the no-fly zone data list promptly.

Upon receiving the updated information, the leader's edge computing module recalculates the flight path and simultaneously computes the relative formation positions, then transmits the new path and formation navigation parameters to the wingmen. This enables the entire formation to safely and consistently complete the obstacle avoidance maneuver, thereby verifying the real-time performance, stability, and reliability of the integrated system in multi-UAV cooperative missions.

Figure 18 shows the mission waypoint data retrieved from the UAV by the ground control station after the leader UAV completed obstacle avoidance. From the figure, it is clearly evident that the leader successfully reprogrammed an alternative route to bypass the no-fly zone and promptly updated the new waypoints into the mission list, thereby validating the effective real-time path reprogramming capability of the edge computing module during actual flight.

The corresponding flight trajectories are illustrated in Fig. 19. In this figure, the orange segments represent the flight path executed by the leader after deviating from the original route

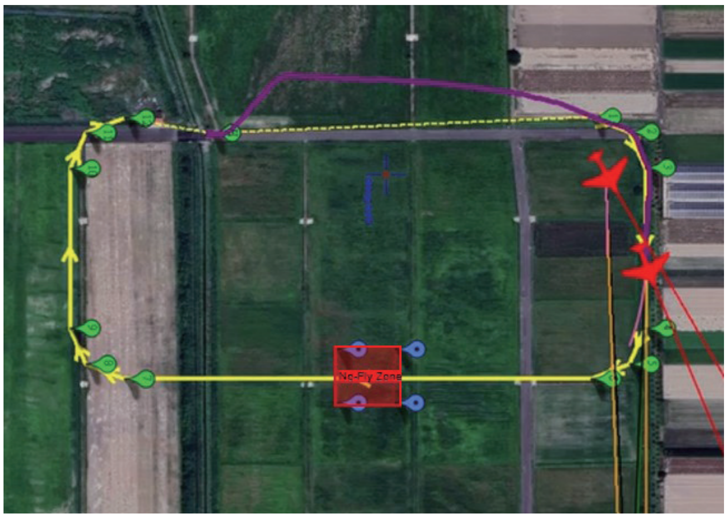

Fig. 17. (Color online) Uploading of new no-fly zone to formation UAVs.

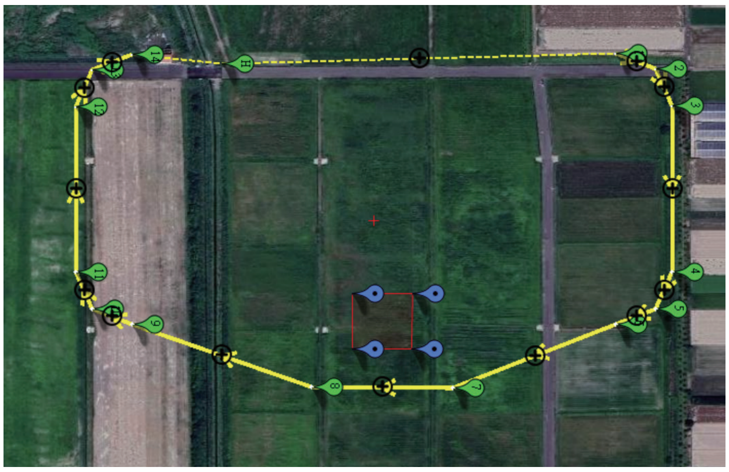

Fig. 18. (Color online) New path of leader after avoidance of no-fly zone.

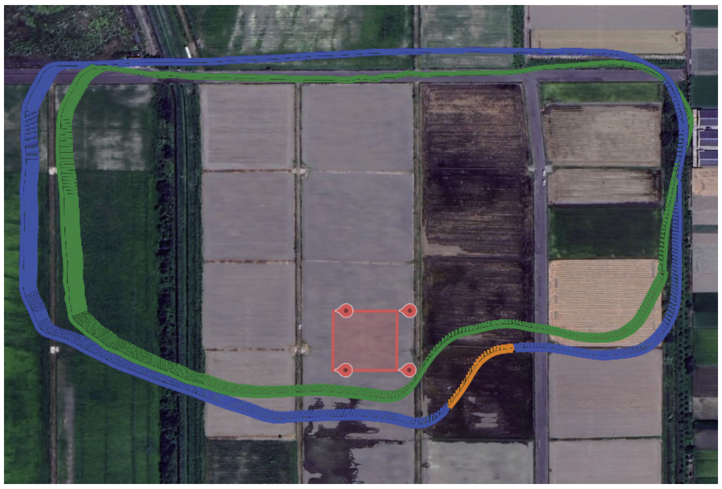

Fig. 19. (Color online) Track of formation experiment.

and following the newly programmed path for obstacle avoidance, successfully bypassing the no-fly zone and returning to the subsequent mission route. The green segments represent the flight trajectory of the wingman, demonstrating that the wingman can promptly receive the formation information transmitted by the leader and dynamically adjust its position in real time, safely avoiding the no-fly zone together with the leader. The experiment's actual flight photo is shown in Fig. 20.

The results indicate that even after the leader reprogrammed its path, the entire formation was able to maintain a good structure and coordinated flight. This fully verifies that the proposed integrated edge computing architecture and formation control strategy exhibit excellent real-time performance, stability, and reliability in dynamic obstacle avoidance and multi-UAV cooperative scenarios.

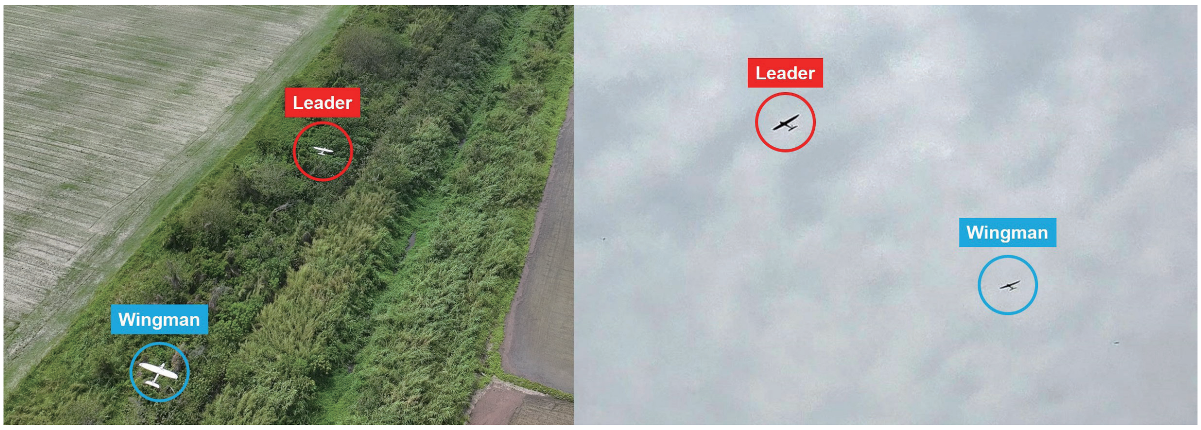

Fig. 20. (Color online) Photo of formation experiment.

5. Conclusions

On the basis of the results of the two flight experiments described above, it can be confirmed that in this study, we successfully developed a low-power edge computing module and effectively integrated it into a fixed-wing UAV system. This module not only enables real-time communication management with the flight controller but also possesses the autonomous capability to assess route risks. By leveraging real-time UAV position, heading, and waypoint information, it can determine whether the UAV is at risk of entering a no-fly zone and rapidly perform path reprogramming before the UAV reaches the restricted area. The module consumes only about 1–2 W, accounting for roughly 1% of the total UAV power, yet it remains a critical factor affecting endurance in multi-UAV cooperative missions.

The newly generated path is immediately transmitted to the flight controller, ensuring that the UAV can safely avoid no-fly zones while maintaining stable flight and formation, thereby enhancing the overall autonomy and reliability of mission execution.

While the present study focuses on two-dimensional obstacle avoidance in horizontal flight, we acknowledge that three-dimensional obstacle avoidance, including vertical maneuvers, is essential for UAV operations in complex terrains. As part of our future work, we plan to extend the proposed framework by integrating additional sensors such as Light Detection and Rangingor ultrasonic altimeters to enable real-time 3D obstacle detection and avoidance in fixed-wing UAV formations.

Despite the promising results of this study, several challenges remain for achieving fully autonomous UAV operations. These include real-time 3D obstacle detection and avoidance in complex environments, scalability to larger multi-UAV cooperative missions, effective energy management, and rigorous reliability validation under unpredictable outdoor conditions. Addressing these challenges will be the focus of our future work toward building fully intelligent and distributed UAV systems.

Furthermore, future research will focus on incorporating more complex and diverse no-fly zone shapes, as well as introducing multiple no-fly zones within a single mission, to further validate the system's adaptability and robustness in highly complex mission environments. In addition, the framework can be extended to larger multi-UAV cooperative scenarios, thereby strengthening the capability for autonomous collaborative obstacle avoidance and moving toward fully intelligent and distributed UAV operational systems.

Acknowledgments

This work was supported by the National Science and Technology Council of the Republic of China (Taiwan) under grant number NSTC 113-2637-E-150-004.

References

1 R. W. Beard and T. W. McLain: Small Unmanned Aircraft: Theory and Practice. (Princeton University Press, Princeton, NJ, 2012) Chap.1.
2 X. Wang and W. Ren: IEEE Trans. Cybern. **50** (2020) 3875.
3 K.-K. Oh, M.-C. Park, and H.-S. Ahn: Automatica **53** (2015) 424 https://doi.org/10.1016/j.automatica.2014.10.022
4 Naufaldo, H. M. Wu, and M. Q. Zaman: J. Chin. Inst. Eng. **1** (2025) 1. https://doi.org/10.1080/02533839.2025.2503867
5 R. Olfati-Saber, J. A. Fax, and R. M. Murray: Proc. IEEE. **95** (2007) 215. https://doi.org/10.1109/JPROC.2006.887293
6 P. E. Hart, N. J. Nilsson, and B. Raphael: IEEE Trans. Syst. Sci. Cybern. **4** (1968) 100. https://doi.org/10.1109/TSSC.1968.300136
7 C. Zhang, W. Liu, J. Chen, H. Li, M. Sun, L. Wang, N. Zhou, R. Xu, and P. Yang: IEEE Access **9** (2021) 16223.
8 Z. Chen, H. Li, Y. Zhang, and J. Sun: IEEE Access **10** (2022) 12345.
9 R. W. Beard and T. W. McLain: J. Guid. Control Dyn. **26** (2003) 896.
10 Y. Liu, X. Zhang, and P. Wang: Rob. Auton. Syst. **142** (2021) 103812.
11 Y. Zhang, J. Liu, X. Zhou, and L. Sun: IEEE Trans. Aerosp. Electron. Syst. **58** (2022) 1884.
12 Y. Liu, X. Zhang, P. Wang, and L. Sun: IEEE Trans. Ind. Electron. **67** (2020) 10438.
13 M. Zuluaga, H. Chen, L. Zhang, P. Lu, X. Fan, B. Xu, and Z. Yan: IEEE Trans. Rob. **39** (2022) 3585.
14 M. Lu, X. Fan, B. Xu, Z. Yan, R. Peng, H. Chen, L. Zhang, and P. Lu: IEEE Trans. Veh. Technol. **70** (2025) 10242.