

Topology-Aware Line Guidance for Warehouse MAVs: Lightweight Junction-Driven Navigation with Real-Time Path Encoding and Multi-Path Adaptation

Yongmei Dou^{1,2}, Ling Shuang Soh¹, Hann Woei Ho¹✉

(1. School of Aerospace Engineering, Universiti Sains Malaysia, Nibong Tebal, Pulau Pinang 14300, Malaysia; 2. Shanxi Province Intelligent Optoelectronic Sensing Application Technology Innovation Center, Yuncheng University, Yuncheng 044000, Shanxi, China)

Abstract: This paper presents a vision-based navigation framework for micro air vehicles (MAVs) operating in confined warehouse environments. To address the trade-off between low localization accuracy in mapless methods and high computational demands in map-based approaches, the proposed system leverages topology-aware path guidance using monocular vision. Navigation is driven by a digital instruction format (DIF) that encodes both the path index and target junction, enabling autonomous navigation without environmental modifications. The framework comprises a cascaded perception–encoding–control pipeline. For structured paths, foreground pixel density trend analysis with sliding window smoothing for robust junction recognition, while lateral proportional-integral-derivative (PID) control ensures accurate path tracking. For geometric trajectories, the control logic incorporates L-junction triggers, fixed-angle turns, and spatial yaw correction to accommodate sharp corners and curved segments. ROS-Gazebo simulations validate the method’s effectiveness, achieving up to 94.40% junction recognition accuracy (92.01% on average), trajectory tracking errors below 0.1 m, and terminal localization deviations under 0.2 m. These results validate the method’s accuracy, stability, and suitability for computationally constrained MAV platforms in warehouse automation.

Keywords: lightweight navigation technique; junction recognition; multi-path adaptation strategy; topology-aware line guidance; warehouse logistics automation

1 Introduction

Micro air vehicles (MAVs), owing to their high maneuverability and compact size, have attracted significant attention in indoor automation tasks [1,2]. Their autonomous navigation capability in warehouse environments has become a key

research focus in smart logistics and warehouse management [3]. However, achieving high-precision localization and stable navigation control on platforms with limited computational resources remains challenging, particularly under global positioning system (GPS)-denied and complex confined conditions [1,2].

Vision-based navigation has emerged as a key approach to enhancing environmental perception and navigation accuracy for MAVs, owing to its lightweight design, low-cost sensors, and flexible algorithms [4,5]. Existing solutions can be categorized into mapless and map-based strate-

Manuscript received Sep. 14, 2025; revised Nov. 15, 2025; accepted Nov. 20, 2025. The associate editor coordinating the review of this manuscript was Dr. Lijuan Jia. This work was supported by the Fundamental Research Grant Scheme (FRGS) (No. FRGS/1/2024/TK04/USM/02/3) funded by the Malaysian Ministry of Higher Education (MOHE).

✉ Corresponding author. Email: aehannwoei@usm.my
DOI: [10.15918/j.jbit1004-0579.2025.063](https://doi.org/10.15918/j.jbit1004-0579.2025.063)

gies: the former offers deployment flexibility and requires no environmental dependence but suffers from limited localization accuracy and robustness; the latter achieves higher accuracy but imposes significant computational load, making it unsuitable for resource constrained platforms [5].

Mapless navigation strategies generate control commands directly from sensor data by extracting visual features [6,7], yet they face several significant limitations. Supervised learning methods, such as DroNet, exhibit strong dependency on training data and show poor generalization in environments with glass surfaces or textureless regions. Moreover, the absence of global constraints often leads to accumulated errors over long trajectories [8,9]. Although deep reinforcement learning performs well in simulation, its deployment in real-world scenarios suffers from reduced tracking accuracy due to domain shift and dynamic environmental changes [10,11]. Optical flow-based methods frequently fail in low-texture or low-light conditions due to unreliable feature tracking, while inertial measurement unit (IMU) fusion performance is constrained by sensor precision [12]. In addition, limited field of view and resolution restrict accurate localization over long distances [13].

Map-based navigation strategies can be classified into two categories: prior map-dependent methods and real-time simultaneous localization and mapping (SLAM)-based mapping [14]. In the former, visual odometry (VO) estimates motion by comparing consecutive image frames without requiring a global map [15,16]. Landmark-based localization relies on pre-deployed visual patterns, such as markers for pose estimation [17], but it requires full environmental coverage and involves high system complexity [18]. In the latter, visual SLAM performs simultaneous localization and mapping. Representative methods include monocular SLAM, such as oriented features from accelerated segment test (FAST)

and rotated BRIEF (ORB)-SLAM, which builds sparse maps via feature matching [19], and large-scale direct (LSD)-SLAM, which constructs semi-dense maps through photometric optimization [20]. Red-green-blue and depth (RGB-D) cameras enable dense mapping [21], while visual-inertial SLAM (V-ISLAM) integrates IMU data to improve accuracy and robustness [22]. However, high computational cost and strong dependency on visual features remain key bottlenecks. Dense mapping often requires hardware acceleration to achieve real-time performance. Feature-based sparse maps tend to miss structural details [19], while direct methods suffer from significant performance degradation in textureless areas [20]. As a result, most systems are highly reliant on texture-rich environments to maintain stability.

In the context of warehouse navigation, several studies have explored path tracking using line detection from ground textures via downward-looking cameras [6,23,24]. Control strategies often rely on proportional-integral-derivative (PID) controllers [25,26], with some adopting advanced techniques like non-singular terminal sliding mode control (NTSMC) for robustness and fast convergence. These systems typically focus on path following without precise localization or environmental context awareness.

Among these, the method in [23] stands out for using Canny-Hough transforms to extract ground path centroids, followed by Kalman filtering for smoothing. However, it lacks the ability to localize the MAV within the structured layout of the environment and is not designed to generalize across various structured and geometric path topologies.

These limitations motivate a lightweight, structure-aware navigation strategy that operates without maps or dense visual features and remains compatible with constrained onboard processors.

To address this, we propose a novel topology-based navigation framework that leverages floor

markings in warehouses as navigational references. Our method uses classical vision techniques for efficient line segmentation and introduces a novel junction recognition algorithm based on the trend of segmented pixel density. This allows the MAV not only to follow paths, but also to recognize its position at key junctions and navigate toward target nodes using a compact numeric encoding referred to as the digital instruction format (DIF).

As shown in Fig.1, warehouse layouts are categorized into two main types: static storage

zones (Fig.1 (a)) and dynamic storage zones (Fig.1(b)–(d)). Static storage zones feature fixed shelving units arranged in regular grids, with straight yellow guidelines forming structured junctions, such as L-, T-, and cross-shaped nodes. These configurations support reliable routing among inventory locations. In contrast, dynamic storage zones accommodate temporary or overflow storage and exhibit less regular layouts, such as circular, square, or octagonal paths surrounding irregular shelf clusters.

The DIF encodes navigation targets in a

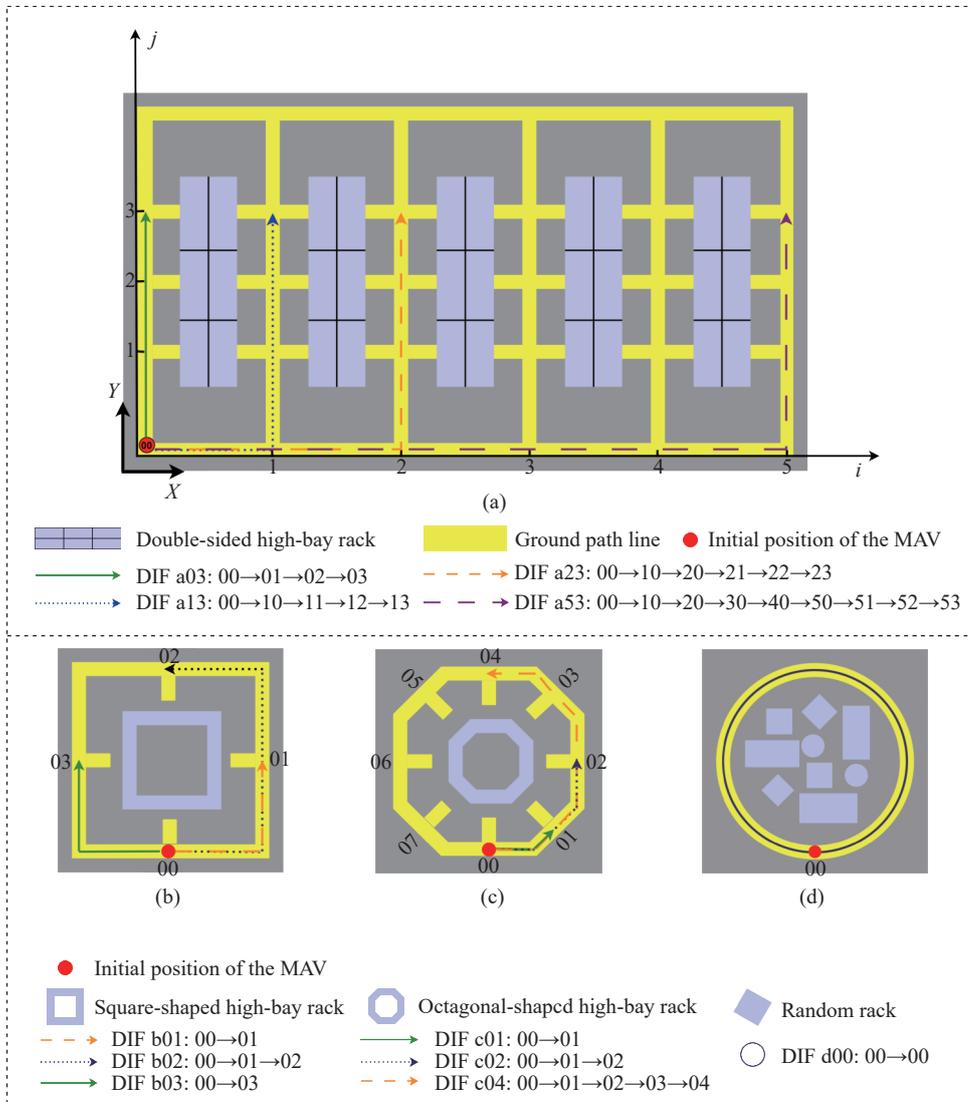


Fig. 1 Example warehouse layouts and corresponding MAV flight paths for static and dynamic storage zones: (a)ground trajectory consisting of yellow lines featuring L-shaped, T-shaped, and cross-shaped junctions;(b)ground trajectory following a square path featuring T-junctions;(c)ground trajectory following a octagonal path featuring T-junctions;(d)ground trajectory following a circular path around a temporary storage area

Tab. 1 DIF encoding rules

Zone	Range of coordinate (ij)
a static	Any whole number
b square	$i=0$ $j=0,1,2,3$
c octagonal	$i=0$ $j=0,1,2,3,4,5,6,7$
d circular	$i=0, j=0$

three-digit format, as presented in Tab. 1. The first digit indicates the zone type (e.g., a for static, b–d for dynamic), while the second and third digits represent the (ij) grid coordinates of the target junction, with (00) denoting the take-off point. This format allows the MAV to interpret task goals efficiently without prior mapping, supporting lightweight, real-time navigation across varying warehouse topologies.

This paper makes the following key contributions:

1) A novel topology-based visual navigation method that performs map-free autonomous navigation using junction recognition derived from ground-line topology.

2) A robust yet efficient visual encoding approach for junction recognition, which uses foreground pixel density trends in segmented images to identify various types of path junctions and enable early prediction of upcoming junctions even under partial visibility.

3) A unified navigation strategy supporting both structured grid paths and irregular geometric layouts through a shared perception–encoding–control pipeline.

2 Topology-Aware Line-Guided MAV Navigation Framework

This section introduces our topology-based visual navigation framework for autonomous MAV flight in warehouse environments using only a downward-facing monocular camera. Fig. 2 presents an overview of the proposed autonomous navigation framework for structured paths. The

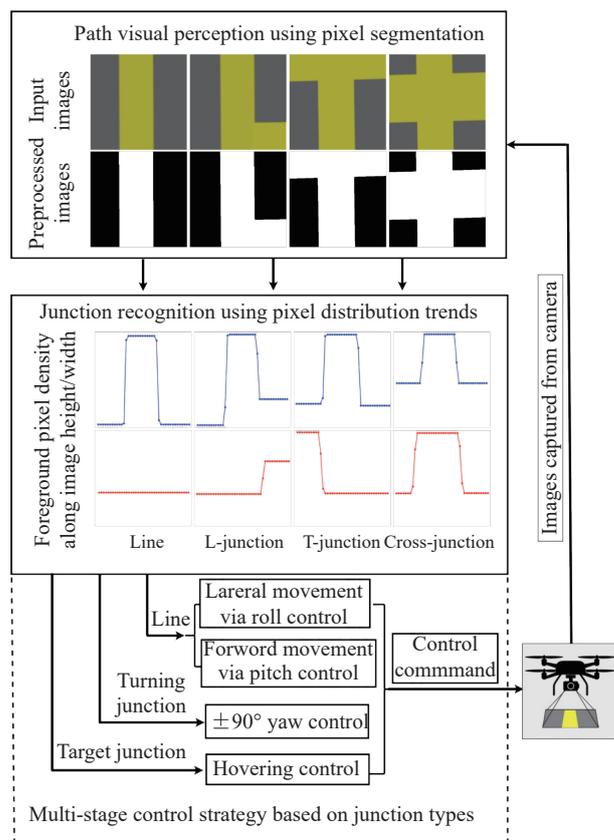


Fig. 2 Overview of the proposed topology-based visual navigation framework using DIF and foreground pixel density analysis

system integrates real-time visual perception, junction recognition, and a multi-stage motion control strategy to guide an MAV along floor-marked paths in indoor environments. The pipeline consists of image preprocessing, junction recognition, and DIF-driven control actions for structured and geometric paths.

The following subsections detail each component of the methodology: Section 2.1 outlines the visual perception process for extracting path features; Section 2.2 introduces the junction recognition algorithm and instruction mapping; Section 2.3 describes the multi-stage control strategy; and Section 2.4 discusses adaptations for irregular geometric layouts.

2.1 Path Visual Perception Using Pixel Segmentation

The visual perception module is responsible for processing raw camera input into a binary representation that reveals the structural layout of the

path. A downward-facing monocular camera mounted on the MAV continuously captures images during flight. These images are first standardized by converting them into a format compatible with the onboard computer vision stack, and resized to a fixed resolution $R \times R$, which provides a consistent trade-off between processing speed and spatial resolution.

To mitigate the effects of varying ambient lighting, each image is transformed from the RGB to hue-saturation-value (HSV) color space [27]. HSV's decoupling of chromaticity from intensity enables more robust segmentation of the yellow path features. A thresholding operation on the HSV channels produces a binary mask highlighting the path regions of interest.

However, due to environmental noise and sensor limitations, the segmented image may contain fragmented regions and high-frequency artifacts. To address this, a two-step enhancement process is applied. First, a median filter with a window of size $k_r \times k_r$ removes salt-and-pepper noise while preserving edge details [28]. This is

followed by a morphological closing operation using a structuring element of the same size to restore continuity and fill gaps in the segmented foreground [29].

The resulting binary image captures a clean and coherent structure of the navigation path. To extract semantic information, the system analyzes the density of foreground pixels along both the horizontal and vertical axes. The trends of these densities are encoded as directional variation curves, which serve as input to the junction recognition module described in the next section.

2.2 Junction Recognition using Pixel Distribution Trends

To achieve lightweight and accurate path junction recognition, we introduce a classification method based on the spatial distribution of foreground pixels along each axis, as illustrated in Fig. 3. Following image preprocessing and binarization, the image is uniformly divided into N_b bins along both horizontal and vertical image axes. In each bin, the foreground pixel density (FPD), D_k , in the k -th bin is defined as

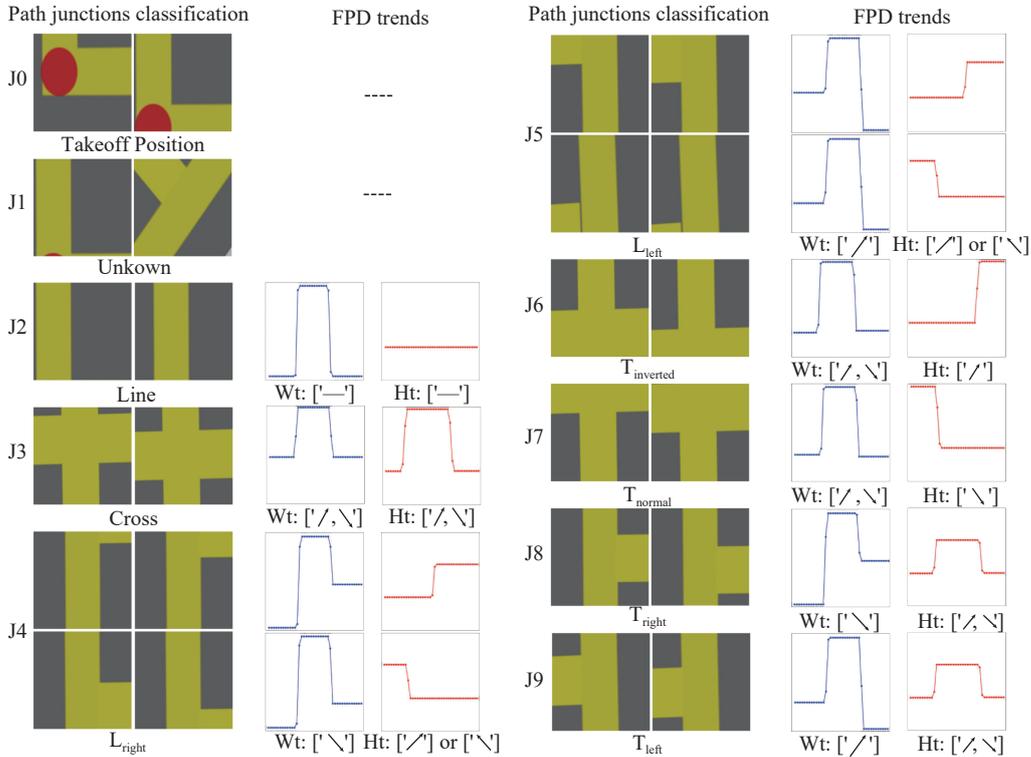


Fig. 3 Junction types (J0–J9) classified using foreground pixel density trends along horizontal and vertical axes

$$D_k = \frac{1}{|B_k|} \sum_{(x,y) \in B_k} F(x,y) \quad (1)$$

where B_k is the set of pixel coordinates in the k -th bin, $|B_k|$ is the total number of pixels in bin B_k , and $F(x,y) \in \{0,1\}$ denotes the binary image, with 1 representing a foreground pixel and 0 a background pixel.

After computing the FPD along both the horizontal and vertical axes, their distribution trends are used to classify junctions labeled from J0 to J9, as illustrated in Fig. 3. To reduce noise, bins with near-zero density, which represents background, are excluded. Adjacent bins with minimal deviation of FPD ($< \varepsilon_g$) are grouped and averaged, producing smoother trend curves. Each axis is then analyzed for directional trends categorized as increasing (\nearrow), decreasing (\searrow), or constant ($-$). The combined horizontal and vertical trends form a unique signature that reflects the topological features of the environment, which enable robust identification of straight paths, L-shaped, T-shaped, and cross junctions.

Start-point detection is handled separately by identifying red regions in the HSV color space; once detected, the system assigns a predefined code J0 and bypasses further junction classification. If no start point is found, trend analysis proceeds to determine the junction type. For example, an inverted T-junction may show an “increase–decrease” pattern horizontally and “increase” vertically. A symmetric peak across both axes indicates a cross junction, while flat trends correspond to straight paths. Unrecognized patterns are assigned a default label J1, which ensures that only valid topological configurations trigger control decisions. Each junction code directly links to the DIF-based control logic, which enables real-time transitions between flight modes, such as forward motion, turning, or hovering.

2.3 Multi-Stage Control Strategy Based on DIF

To translate high-level instructions into autonomous navigation, the system leverages the

DIF, which encodes a flight path (as defined in Section 1) and its target destination as a sequence of expected junction codes. Each instruction guides the MAV from the takeoff point through a set of predefined junctions to a target location. These junctions are identified in real time through the junction recognition method described in Section 2.2 and matched against the expected code sequence defined in the DIF.

Each DIF instruction corresponds to a specific path and target junction, guiding the MAV through a sequence of visually recognized junction codes. As illustrated in Fig. 4, several representative trajectories are extracted from the static storage layout shown in Fig. 1(a), denoted as paths ①–④. These examples demonstrate how the system interprets consecutive junction detections—identified by codes J0–J9—to determine navigation decisions, such as forward movement, turning, or hovering. For example, along path ① (DIF a03), the recognition sequence J4→J8 indicates progression toward a right-facing T-junction. The MAV moves forward after the first two recognitions and initiates a hover when the third confirms arrival at the target a03. In another case, path ② (DIF a13) involves a sequence of J5→J9, signaling a left-facing T-junction that triggers a 90° counterclockwise turn followed by continued forward motion. Upon the

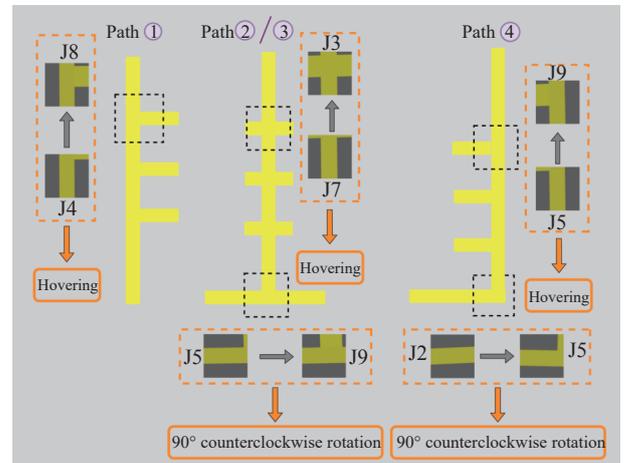


Fig. 4 Sample trajectories showing DIF-guided navigation using junction recognition in structured storage zones

third recognition of a cross-junction (J7→J3), the system confirms arrival at the target a13. These examples illustrate how the DIF, in conjunction with real-time junction recognition, enables perception and control without requiring an explicit map.

This navigation logic is underpinned by a junction class transition strategy, in which decisions are not based on isolated detections but on the transitions between recognized junction types. By interpreting recognition trends over time, the vehicle can anticipate upcoming maneuvers, such as turns or hovers, even when junctions are only partially visible or occluded. This predictive capability improves responsiveness and continuity, especially in complex layouts where early context aids decision-making.

To further enhance robustness, a temporal consistency filter that validates each recognition is employed using a sliding window across multiple frames. This suppresses transient misclassifications caused by visual noise, motion blur, or jitter. Only when a consistent match is observed does the system confirm a junction type and proceed with the corresponding control action.

Once a recognition is validated, the DIF-guided sequence activates a multi-stage navigation strategy composed of modular controllers tailored to specific junction types and flight behaviors. The control flow is shown in Fig. 5. The MAV continuously perceives the floor layout using its downward-facing camera and matches recognized junctions against the instruction sequence in a temporally smoothed manner

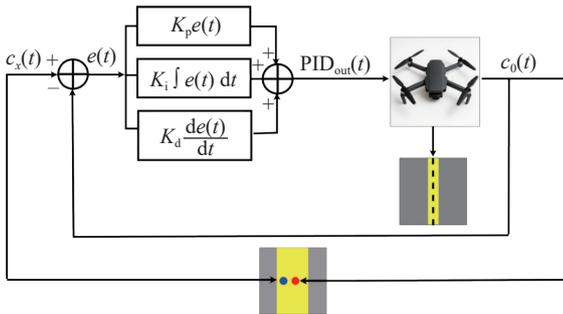


Fig. 5 Lateral PID control loop for path tracking based on centroid deviation

to ensure robustness.

For lane following along straight segments (e.g., J2), the system employs a lateral PID controller using the centroid of the yellow guide line in the image. This centroid $c_x(t)$ is computed via image moments, defined as

$$c_x(t) = \frac{m_{10}}{m_{00}} \quad (2)$$

where m_{00} represents the total number of pixels in the yellow region, and m_{10} denotes the weighted pixel sum along the width direction. The horizontal deviation between $c_x(t)$ and the image center $c_0(t)$ is normalized as $e_n(t)$, as shown

$$e_n(t) = \frac{c_x(t) - c_0(t)}{c_0(t)} \quad (3)$$

A dead-zone filter is first applied to suppress minor oscillations in the error signal $e_n(t)$. The filtered error $e(t)$ is then used as input to the PID controller, which generates the lateral velocity command $v_{yb} = \text{PID}_{\text{out}}(t)$. Meanwhile, the forward velocity command V_{xb} is set to a constant value to maintain steady forward motion.

This code-driven control architecture offers a lightweight, real-time alternative to map-based or learning-intensive methods, while maintaining high flexibility and robustness in diverse indoor layouts.

2.4 Adaptation for Irregular Geometric Paths in Dynamic Storage Zones

The navigation framework described so far is designed to handle regular indoor layouts in static storage zones, as illustrated in Fig. 1(a), where path networks consist of common junction types, such as L-, T-, and cross-shaped junctions. However, real-world warehouse environments often include irregular zones, such as temporary storage areas or multi-level shelf arrangements, where paths follow non-standard geometric patterns. These may include square, octagonal, or circular layouts, requiring enhanced adaptability in perception and control.

To support navigation in such dynamic stor-

age zones, the proposed system is extended to three geometric path scenarios, as illustrated in Figs. 1(b-d). These paths are encoded using the DIF similar for regular layouts. For instance, DIF b02 refers to the second key location on the square layout, and DIF c04 corresponds to the fourth location on the octagonal layout. For the circular route, DIF d00 represents a complete loop from start to end.

The same junction recognition and centroid-based lateral control strategy are applied to these paths, with geometric-specific adjustments. For paths *b* and *c*, the classification logic for junction types remains consistent with that used in regular layouts. However, corner handling differs: in path *b*, detecting an L-type turn triggers a fixed 90° yaw adjustment. In contrast, path *c* forms an octagon with 135° internal corners, prompting a 45° rotation at each turn.

For the circular path *d*, a model-based yaw correction mechanism is introduced to continuously adjust heading based on path curvature. The pixel offset $\Delta x(t)$ between the yellow line centroid and the image center is first mapped to a spatial deviation Δx_m as follows

$$\Delta x_m(t) = \frac{\Delta x(t) h}{f} \quad (4)$$

The horizontal focal length f is obtained from the horizontal field of view (HFOV) θ_h and the raw image width W

$$f = \frac{W}{2 \tan(\theta_h/2)} \quad (5)$$

The resulting spatial deviation is then converted into a yaw command ψ through the geometric projection model:

$$\psi = \arctan\left(\frac{\Delta x_m(t)}{h}\right) = \arctan\left(\frac{\Delta x(t)}{f}\right) \quad (6)$$

The yaw control in both paths *c* and *d* are activated when the image centroid deviation exceeds a threshold $\text{th}_d = \Gamma/2h$, formulated for adapting navigation at various heights h . Γ is selected or tuned by compromising both the forward velocity and the turning rate of the vehicle.

2.5 Experiment Setup

To evaluate the performance of the proposed navigation algorithm, an MAV simulation test platform was developed based on Ubuntu 20.04, integrating ROS Noetic and Gazebo11 (Experiment videos: https://youtu.be/ji_vz1R-Vgw). The flight control systems adopts the Software-In-The-Loop (SITL) architecture provided by ArduPilot. The control logic combines high-level task interfaces from DroneKit-Python with custom modules developed using the ROS messaging mechanism. These components communicate bidirectionally via the MAVLink protocol, enabling a closed-loop control system for command transmission and state feedback.

The simulation platform operates within a VMware virtual environment configured with a dual-core CPU and 8 GB of RAM, hosted on a machine equipped with an i9-13900H processor, 32 GB of RAM, and an RTX 3050 GPU. The experimental design includes four representative flight paths: a structured path for static storage zones and three geometric closed-loop paths for dynamic storage zones composed of square, octagonal, and circular shapes. Fig. 6 illustrates the simulated flight environment for the structured path, where yellow lines define the floor paths over which the MAV autonomously tracks. The subfigure in the lower-left corner shows real-time images captured by the downward-looking camera.

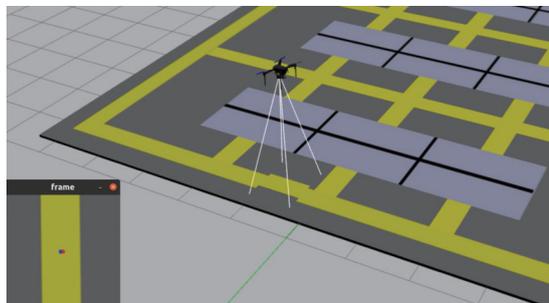


Fig. 6 Simulation environment of structured grid paths with MAV flight and downward camera view

Upon receiving a predefined DIF, the MAV ascends from the takeoff point to an altitude of $h = 0.5$ m and navigates along the corresponding

yellow path to the target location. During flight, the downward-looking camera continuously captures image data, which is converted to an OpenCV-compatible data structure using a bridging tool. The image is then uniformly resized to a resolution of $R \times R$ pixels, with $R = 400$, for subsequent processing.

At the takeoff point, red marker detection is performed based on two HSV ranges: $H \in [0, 10] \cup [170, 180]$, $S \in [100, 255]$, and $V \in [100, 255]$. A pixel count threshold of 350 is set to trigger structure code J0. If no red region is detected, the system proceeds to extract the yellow path within the HSV range $H \in [18, 175]$, $S \in [84, 255]$, and $V \in [140, 255]$. The resulting binary mask image is first processed with a $k_r \times k_r$ median filter, where $k_r = 5$, followed by a morphological closing operation using a 5×5 convolution kernel to restore path connectivity.

During the junction recognition stage, the image is divided into $N_b = 40$ regions along both the horizontal and vertical directions, with each region having a width or height of 10 pixels. The FPD is computed for each region. Regions with zero FPD are first discarded. Clusters of adjacent regions with FPD differences smaller than $\varepsilon_g = 0.01$ and a minimum length of two are identified and merged. Each cluster is then assigned the mean FPD value in place of the original values. Based on the trend of the cluster mean values, each is classified as “increasing” (\nearrow), “decreasing” (\searrow), or “constant” ($—$). These trends are matched jointly across horizontal and vertical directions to recognize 10 junction patterns, which are encoded as codes J0 to J9. If the current frame is recognized as J1, the result from the previous frame is retained to enhance temporal stability. The final code output is smoothed using a sliding window of size 5, as determined from the analysis in Section 3.1, to enhance system robustness.

When the recognition result is J2, the system calculates the centroid of the yellow region using image moments and determines its pixel

deviation from the image center. This deviation is normalized and compared with a predefined dead zone threshold $\varepsilon_{\text{threshold}} = 0.05$. If the threshold is exceeded, a PID controller is triggered. The controller parameters are set to $K_p = 0.5$, $K_i = 0.01$, and $K_d = 0.25$, with the integral term constrained within ± 50 to prevent integral windup. The controller output is the MAV’s lateral velocity command V_{yb} , while maintaining a constant forward velocity of $V_{xb} = 0.1$ m/s.

If a special structure code is detected, the system triggers a predefined yaw adjustment of $\pm 90^\circ$ or a hovering operation. All actions, including takeoff, forward motion, turning, and hovering, are executed via application programming interface (API) interfaces provided by DroneKit-Python.

For the octagonal path, a fixed yaw rotation of $\pm 45^\circ$ is required at each corner. For the circular path, the MAV achieves stable path tracking through continuous yaw adjustment. The core principle lies in mapping the lateral pixel deviation between the yellow line centroid and the image center to an actual yaw angle. The horizontal focal length f of the camera is computed using the pinhole camera model, where the required horizontal field of view (HFOV) θ_h and the raw image width W are predefined parameters obtained from the SDF file of the IRIS MAV model used in the simulation. Specifically, the HFOV is set to $\theta_h = 1.57$ radians and the raw image width is $W = 640$ pixels, resulting in a computed focal length of $f = 320$ pixels. The yaw control strategy incorporates a dynamic threshold th_a , which is adaptively determined according to the flight height h , to activate the control, where Γ is tuned to be 45. This enables a more robust response for navigating at various heights.

3 Results and Discussion

This section presents the evaluation of the proposed vision-based navigation framework across different path scenarios. The analysis is organized into three parts: Section 3.1 assesses the

junction recognition performance based on foreground pixel density trends and examines the effect of temporal smoothing using a sliding window; Section 3.2 quantifies the MAV’s tracking accuracy and target localization on static storage zones; and Section 3.3 demonstrates the system’s navigation performance in dynamic storage zones, including square, octagonal, and circular trajectories.

3.1 Junction Recognition Accuracy and Sliding Window Smoothing Analysis

To evaluate the performance of the proposed junction recognition method, frame-by-frame comparison are conducted between the recognition results and manually annotated ground truth junction types. Accuracies are computed from raw, unsmoothed predictions to assess the core recognizer’s performance objectively.

Across all evaluated paths ① to ④, the system consistently achieves recognition accuracies exceeding 90%, despite variations in task complexity, trajectory geometry, and visual conditions. Path ① records the highest recognition accuracy at 94.40%, as summarized in Tab. 2.

Tab. 2 Accuracy of junction recognition under different flight paths.

Flight path	Duration	Captured frames	Misclass frames	Accuracy
①	76.87s	696	39	94.40%
②	117.25s	1031	75	92.73%
③	153.48s	1416	137	90.34%
④	312.69s	2509	237	90.55%
Mean	165.07s	1413	122	92.01%

Note: Duration: total flight time; Misclass frames: number of misclassified frames.

Analysis of misclassified frames reveals that most errors stem from transient, frame-level inconsistencies—typically caused by motion blur, partial occlusions, or viewpoint changes (often resulting from MAV rotations will be further discussed in Section 3.2). These transient fluctuations can lead to premature transitions if not smoothed.

To address this issue, a temporal sliding window filter is applied to smooth recognition results

over multiple consecutive frames. This approach reduces spurious transitions by enforcing temporal consistency before control decisions are made. As shown in Fig. 7, without smoothing (Fig.7(a)), the recognition output fluctuates frequently. A 3-frame window (Fig.7(b)) partially suppresses noise, while a 5-frame window (Fig.7(c)) significantly suppresses unstable outputs while introducing minimal delay—only 2 frames exhibit switching lag. Although the 9-frame configuration (Fig.7(d)) achieves greater smoothness, the response latency becomes noticeable, which introduces a maximum delay of approximately 4 frames. A 5-frame window provides optimal stability–latency trade-off and is therefore adopted.

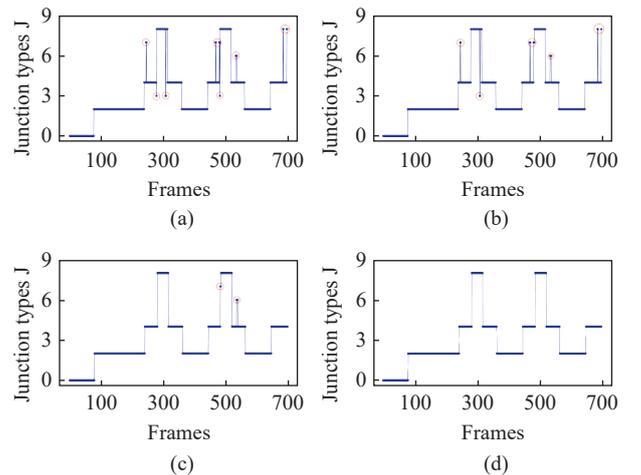


Fig. 7 Effects of different sliding-window sizes on smoothing junction recognition over sequential frames: (a) without sliding window; (b) sliding window size of 3; (c) sliding window size of 5; (d) sliding window size of 9

3.2 Error Analysis of MAV Trajectory Tracking in Static Storage Zones

Fig. 8 compares the actual and desired trajectories of the MAV along four representative flight paths (①–④) in the static storage zone. Gray dashed lines denote desired paths, and colored solid lines represent the actual flight trajectories. Key positions, such as start points, turning corners, and endpoints are marked for clarity—gray circles indicate target locations, and colored circles show the actual arrival points. The results demonstrate close alignment between the desired

and actual paths, with minimal deviation along straight segments. Slight offsets occur near the starting areas and at 90° turns, but overall tracking consistency remains high.

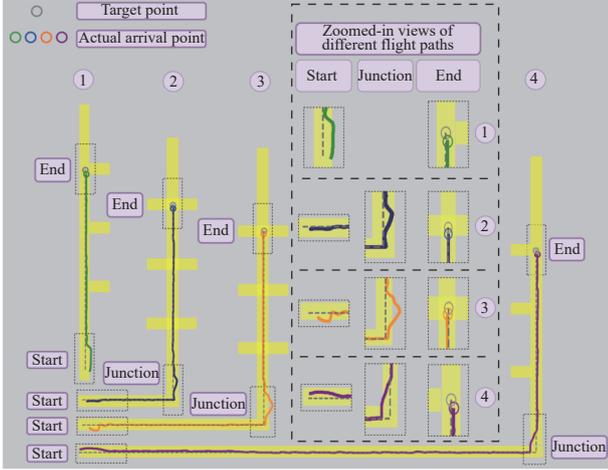


Fig. 8 Comparative analysis of actual and desired trajectories of the MAV under different flight paths, with zoomed-in views of key positional deviations

To quantify tracking accuracy, Tab. 3 reports four error metrics—root mean square error (RMSE), maximum error (ME), mean absolute error (MAE), and standard deviation (STD)—based on lateral coordinate deviations. For all paths, RMSE, MAE, and STD are each maintained below 0.1 m, while the higher ME values stem from isolated peaks during turns, as evident in the time-varying deviations shown in Fig. 9.

Fig. 9 presents the temporal evolution of lateral deviation for each path. For path ①, which follows the Y-axis, deviations in the X direction are plotted. For paths ②–④, the MAV initially tracks the X-axis (blue curve) and then switches to Y-axis (red curve) after turning, with vertical dashed lines indicating turn moment. The spikes observed at these transitions correspond to

the turning phases and align with the ME values reported in Tab. 3, highlighting that most deviations arise during cornering maneuvers rather than straight segments. Therefore, ME alone is not a reliable indicator of overall accuracy.

Terminal accuracy is further evaluated using the Euclidean distance D between the actual and target end points, shown as

$$D = \sqrt{(x_{\text{meas}} - x_{\text{ref}})^2 + (y_{\text{meas}} - y_{\text{ref}})^2} \quad (7)$$

where $(x_{\text{ref}}, y_{\text{ref}})$ is the target point, and $(x_{\text{meas}}, y_{\text{meas}})$ is the actual arrival point.

Tab. 4 shows that endpoint deviations remain under 0.2 m for all paths. Path ③ achieves the best result (0.14 m), while path ④ exhibits the largest deviation (0.19 m). These results verify that the proposed vision-based navigation framework reliably guides the MAV to the designated endpoints with high spatial precision, even in dynamic flight conditions.

3.3 MAV Trajectory Tracking Results in Dynamic Storage Zones

Unlike the structured paths evaluated earlier, the dynamic storage zone introduces irregular geometric layouts and frequent direction changes, making precise coordinate alignment and frame-by-frame error computation difficult. Due to this variability, direct quantitative comparison of tracking errors (e.g., RMSE or MAE) between the desired and actual trajectories is less informative. Instead, this section provides a qualitative assessment based on visual alignment between the planned and executed paths.

Fig. 10 illustrates the MAV’s actual trajectories compared with the desired paths for square and octagonal layouts. Figs. 10 (a)–(c) correspond to square trajectories (DIFs b01, b02, b03), while (d)–(f) show octagonal trajectories

Tab. 3 Error analysis of trajectory tracking in X- and Y- directions for flight paths ①–④

Flight path	RMSE _y	ME _y	MAE _y	STD _y	RMSE _x	ME _x	MAE _x	STD _x
①	–	–	–	–	0.09 m	0.23 m	0.07 m	0.06 m
②	0.01 m	0.05 m	0.003 m	0.01 m	0.03 m	0.16 m	0.01 m	0.03 m
③	0.03 m	0.19 m	0.008 m	0.03 m	0.06 m	0.35 m	0.03 m	0.06 m
④	0.04 m	0.18 m	0.020 m	0.03 m	0.04 m	0.26 m	0.02 m	0.04 m

Note: “–” indicates no deviation data in that direction due to one-dimensional movement.

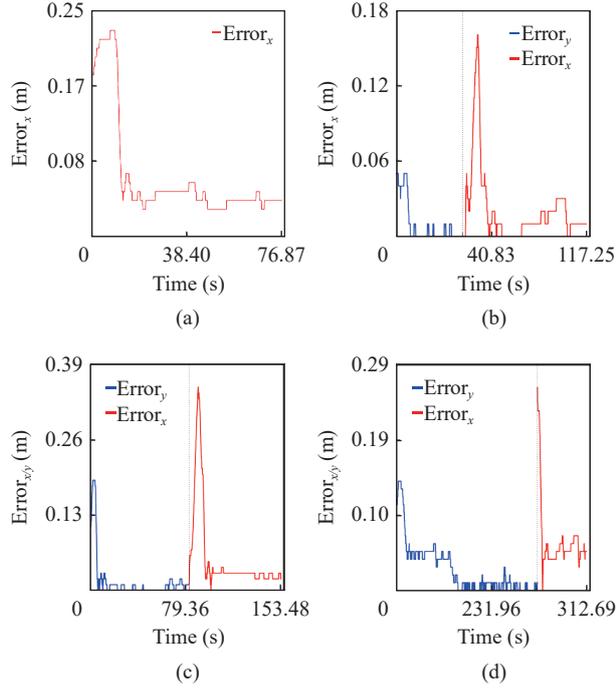


Fig. 9 Time-varying trajectory deviation of the MAV along flight paths ①–④ (each subplot shows lateral deviation over time, relative to the X - or Y -axis): (a) flight path ①; (b) flight path ②; (c) flight path ③; (d) flight path ④

Tab. 4 Target and actual arrival coordinates of the MAV for flight paths ①–④ and the corresponding euclidean distances

Flight path	Target point	Actual arrival point	Euclidean distance
①	(0.6 m, 7.4 m)	(0.64 m, 7.24 m)	0.16 m
②	(4.2 m, 7.4 m)	(4.21 m, 7.24 m)	0.16 m
③	(7.8 m, 7.4 m)	(7.78 m, 7.26 m)	0.14 m
④	(18.6 m, 7.4 m)	(18.62 m, 7.21 m)	0.19 m

(DIFs c01, c02, c04). In each plot, the gray dashed lines represent the desired path, and colored solid lines indicate the actual MAV trajectory.

Despite the layout complexity and repeated sharp turns, the MAV maintains close adherence to the desired paths, demonstrating stable and responsive tracking. Paths b02 and c04, in particular, involve dense turning sequences yet show strong trajectory conformity. Minor fluctuations occur at corners, but overall deviation remains within an acceptable range, indicating good robustness under geometric complexity.

Fig. 11 presents the MAV’s flight along a

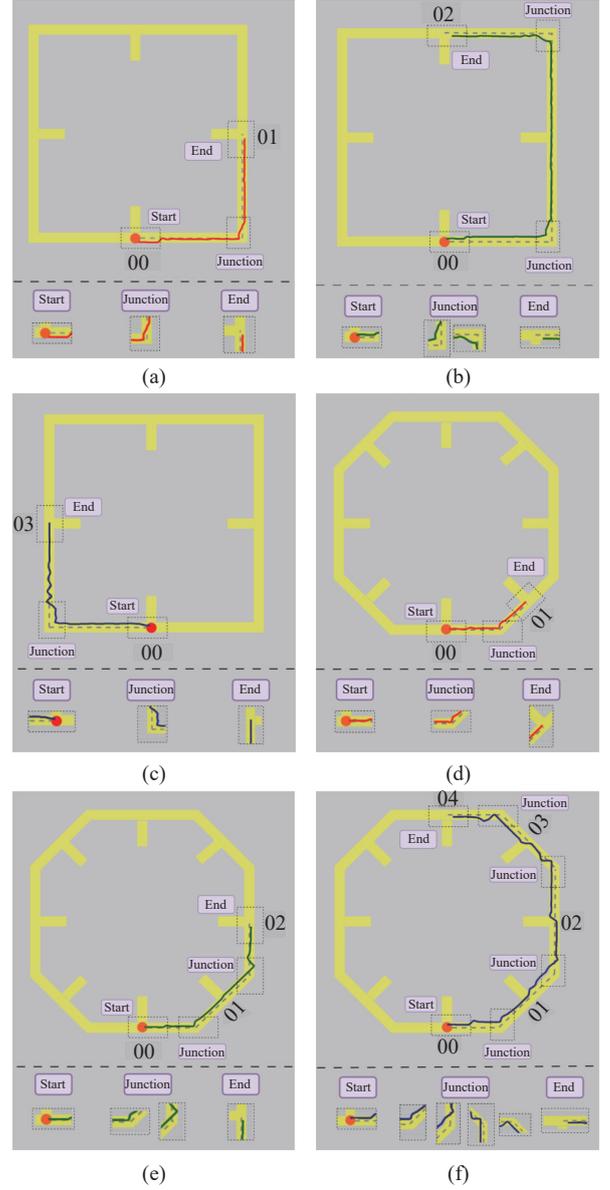


Fig. 10 Comparison of desired and actual MAV trajectories for square and octagonal paths in dynamic storage zones: (a) DIF b01; (b) DIF b02; (c) DIF b03; (d) DIF c01; (e) DIF c02; (f) DIF c03

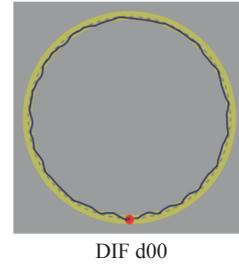


Fig. 11 Comparison of desired and actual MAV flight trajectories along a circular path in the dynamic storage area (the gray dashed line represents the desired trajectory, and the blue solid line corresponds to the actual flight trajectory along the path)

circular path (DIF d00) in the dynamic storage zone. The MAV successfully tracks the continuous curved path, with minimal deviation from the desired ring. An initial offset is observed, but the vehicle quickly realigns, confirming the effectiveness of the yaw control and tracking strategy in non-angular trajectories.

Across both structured and geometric navigation tasks, the proposed system demonstrates robust and consistent performance. In structured storage zones, junction recognition accuracy exceeds 90% on all tested paths, reaching a peak of 94.40%. Lateral trajectory tracking demonstrates high precision, with RMSE, MAE, and STD consistently maintained below 0.1 m for all paths. Terminal localization accuracy remains within 0.2 m, with the smallest endpoint deviation measured at only 0.14 m. In dynamic storage zones, the MAV successfully tracks square, octagonal, and circular paths with minimal drift, maintaining smooth flight even under frequent turning conditions. These results validate the method's high control precision, recognition stability, and adaptability across diverse path structures.

4 Conclusion

This paper presents a vision-based navigation framework that enables a MAV to autonomously navigate structured and geometric warehouse trajectories using a DIF. The navigation relies on a single numerical instruction specifying the path index and target junction. The system integrates a novel junction recognition module based on foreground pixel distribution trend analysis for decision-making, along with a lateral PID control mechanism for path tracking. Simulation experiments validate the method's effectiveness: the average junction recognition accuracy reaches 92.01%, with RMSE, MAE, and STD of trajectory tracking all remaining below 0.1 m, and terminal localization deviation consistently under 0.2 m. These results demonstrate the system's high recognition reliability, trajectory tracking

precision, and suitability for deployment on resource-constrained platforms.

To further evaluate practical adaptability, future work will proceed in two directions. First, the proposed method will be benchmarked against existing path recognition algorithms under unified computational constraints on typical onboard processors, assessing recognition accuracy, trajectory tracking precision, and computational load to evaluate real-time performance. Second, real-world flight validation will be conducted in indoor warehouse environments to examine the system's robustness and applicability in practical autonomous logistics scenarios.

References:

- [1] S. Tavasoli, S. Poorghasem, X. Pan, T. Yang, and Y. Bao, "Autonomous postdisaster indoor navigation and survivor detection using low-cost micro aerial vehicles," *Computer-Aided Civil and Infrastructure Engineering*, vol. 40, no. 1, pp. 130-144, 2025.
- [2] H. Lu, H. Shen, B. Tian, X. Zhang, Z. Yang, and Q. Zong, "Flight in GPS-denied environment: Autonomous navigation system for micro-aerial vehicle," *Aerospace Science and Technology*, vol. 124, pp. 107521, 2022.
- [3] H. Masnavi, J. Shrestha, K. Kruusamäe, and A. K. Singh, "Vacna: Visibility-aware cooperative navigation with application in inventory management," *IEEE Robotics and Automation Letters*, vol. 8, no. 11, pp. 7114-7121, 2023.
- [4] N. Gyagenda, J. V. Hatilima, H. Roth, and V. Zhmud, "A review of GNSS-independent UAV navigation techniques," *Robotics and Autonomous Systems*, vol. 152, pp. 104069, 2022.
- [5] M. Mugnai, M. Teppati Losé, E. P. Herrera-Alarcón, G. Baris, M. Satler, and C. A. Avizzano, "An efficient framework for autonomous UAV missions in partially-unknown GNSS-denied environments," *Drones*, vol. 7, no. 7, pp. 471, 2023.
- [6] N. Maitlo, N. Noonari, K. Arshid, N. Ahmed, and S. Duraisamy, "Ains: Affordable indoor navigation solution via line color identification using monocular camera for autonomous vehicles," in *Proceedings of the 2024 IEEE 9th International Conference for Convergence in Technology (I2CT)*, pp. 1-7, 2024.

- [7] R. W. Himawan, P. B. A. Baylon, J. Sembiring, and Y. I. Jenie, "Development of an indoor visual-based monocular positioning system for multirotor UAV," in *Proceedings of the 2023 IEEE International Conference on Aerospace Electronics and Remote Sensing Technology (ICARES)*, pp. 1-7, 2023.
- [8] A. Anand, S. Agrawal, S. Agrawal, A. Chandra, and K. Deshmukh, "Grid-based localization stack for inspection drones towards automation of large scale warehouse systems," *arXiv preprint*, arXiv: 1906.01299, 2019.
- [9] G. Loianno, C. Brunner, G. McGrath, and V. Kumar, "Estimation, control, and planning for aggressive flight with a small quadrotor with a single camera and IMU," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 404-411, 2016.
- [10] Y. Zhou, "Efficient online globalized dual heuristic programming with an associated dual network," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 34, no. 12, pp. 10079-10090, 2022.
- [11] J. Ni, Y. Gu, Y. Gu, Y. Zhao, and P. Shi, "UAV coverage path planning with limited battery energy based on improved deep double q-network," *International Journal of Control*, no. 8, pp. 2591-2601, 2024.
- [12] G. Roggi, S. Meraglia, and M. Lovera, "Leonardo drone contest 2021: Politecnico di milano team architecture," in *Proceedings of the 2022 International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 676-685, 2022.
- [13] M. Quigley, K. Mohta, S. S. Shivakumar, M. Watterson, Y. Mulgaonkar, M. Arguedas, K. Sun, S. Liu, B. Pfrommer, and V. Kumar, "The open vision computer: An integrated sensing and compute system for mobile robots," in *Proceedings of the 2019 International Conference on Robotics and Automation (ICRA)*, pp. 1834-1840, 2019.
- [14] Y. Kompis, L. Bartolomei, and M. Chli, "Fully autonomous live 3D reconstruction with an MAV: Hardware-and software setup," in *Proceedings of the 9th International Conference on 3D Vision (3DV 2021)*, 2021.
- [15] K. Mohta, K. Sun, S. Liu, M. Watterson, B. Pfrommer, J. Svacha, Y. Mulgaonkar, C. J. Taylor, and V. Kumar, "Experiments in fast, autonomous, GPS-denied quadrotor flight," in *Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7832-7839, 2018.
- [16] R. Mur-Artal and J. D. Tardós, "Orbslam2: An open-source slam system for monocular, stereo, and RGB-D cameras," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255-1262, 2017.
- [17] H. J. Choi, Y. B. Kim, B. Y. Park, and D. H. Lee, "Robust 6D pose estimation using dual active marker system," *International Journal of Control*, no. 2, pp. 382-391, 2025.
- [18] L. C. Chie and Y. W. Juin, "Artificial landmark-based indoor navigation system for an autonomous unmanned aerial vehicle," in *Proceedings of the 2020 IEEE 7th International Conference on Industrial Engineering and Applications (ICIEA)*, pp. 756-760, 2020.
- [19] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147-1163, 2015.
- [20] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-scale direct monocular slam," in *Proceedings of the European Conference on Computer Vision*, pp. 834-849, 2014.
- [21] J. Mei, T. Zuo, and D. Song, "Highly dynamic visual SLAM dense map construction based on indoor environments," *IEEE Access*, vol. 12, pp. 38717-38731, 2024.
- [22] W. Sun, Y. Zhang, and B. Wei, "A visual-inertial motion prior SLAM for dynamic environments," *arXiv preprint*, arXiv: 2503.23429, 2025.
- [23] L. S. Soh and H. W. Ho, "Autonomous navigation of micro air vehicles in warehouses using vision-based line following," *arXiv preprint*, arXiv: 2310.00950, 2023.
- [24] G. A. Hazareh, K. Azizi, A. M. Soveini, A. Nouri, and M. Norouzi, "A vision-based approach for quadrotor line following with sliding mode controller," in *Proceedings of the 2024 10th International Conference on Artificial Intelligence and Robotics (QICAR)*, pp. 36-42, 2024.
- [25] A. Ma'arif, A. Nuryono, and A. Iswanto, "Vision-based line following robot in webots," in *Proceedings of the 2020 FORTEI-International Conference on Electrical Engineering (FORTEI-ICEE)*, pp. 24-28, 2020.
- [26] A. Sharma and S. G. Kulhalli, "Warehouse automation using line follower robot," in *Proceedings of the 2023 4th IEEE Global Conference for Advancement in Technology (GCAT)*, pp. 1-5, 2023.
- [27] Q. Huang, X. Gao, Y. Huang, Y. Li, Y. Zhai, and

L. Zhang, "Furniture panel image edge extraction method based on HSV color space segmentation," in *Proc. 4th Int. Conf. Mechanical Engineering, Intelligent Manufacturing, and Automation Technology (MEMAT 2023)*, vol. 13082, pp. 29-36, 2024.

[28] A. K. Puli, K. S. Kumar, J. B. Naik, P. J. Saikumar, and B. A. Adugna, "A novel effective edge-based image denoising algorithm," *Scientific Programming*, vol. 2022, no. 1, pp. 9675526, 2022.

[29] S. Fauziah, N. Merlina, N. A. Mayangky, M. Hasan, N. A. Fahrurrozi, Y. Y. Panjaitan, and A. K. Putra, "Improving the image of a banana using the opening and closing method," *Jurnal Pilar Nusa Mandiri*, vol. 21, no. 1, pp. 126-133, 2025.



Yongmei Dou is currently pursuing the Ph.D. degree with the School of Aerospace Engineering, Universiti Sains Malaysia, Malaysia. Her research interests include autonomous navigation of micro air vehicles, deep learning-based path planning, and intelligent control

systems.



Ling Shuang Soh is currently working as an AOG Desk Officer at Airbus Customer Services Sdn Bhd, where she supports time-critical aircraft maintenance operations. Her research interests include autonomous navigation, micro air vehicles, and computer vision.



Hann Woei Ho received the B.Eng. degree in aerospace engineering from Universiti Sains Malaysia, Malaysia, in 2009, the M.S. degree (cum laude) in aerospace engineering, specialized in control and simulation from Delft University of Technology, The Netherlands, in 2012, and the Ph.D. degree in aerospace engineering from the Micro Air Vehicle Laboratory (MAV-Lab), Delft University of Technology, in 2017, with the topic of autonomous landing of micro air vehicles through bio-inspired monocular vision. He is currently a Senior Lecturer and the UAV Laboratory Manager of the School of Aerospace Engineering, Universiti Sains Malaysia, Malaysia. His research interests include bio-inspired vision and control of MAVs, machine learning-based control strategies, state estimation, and MAV design.