

*Research Article*

Integration of Kalman State Prediction and Outer-Loop Velocity PID Control for Robust Autonomous Monocular Unmanned Aerial Vehicle Landing

Nada Syifa Qolbiyah^{1*}, Muhammad Fahmi Khoirurrijal², Ferry Faizal², Irwan Purnama^{1,3}

¹School of Electrical Engineering, Telkom University, Bandung, 40257, Indonesia

²Department of Physics, Padjadjaran University, Jatinangor, 45363, Indonesia

³National Research and Innovation Agency, Bandung, 40135, Indonesia

*Corresponding author: nadasyifaqolbiyah@telkomuniversity.ac.id; Tel.: +6287716782600

Abstract: Precise autonomous landing of unmanned aerial vehicles in global positioning system (GPS)-denied environments remains a challenging control problem owing to image-plane measurement noise, sensor latency, and aerodynamic disturbances. This study proposes a prediction-informed landing framework that integrates a discrete-time Kalman filter tuned for ArUco marker motion in the monocular image plane with an outer-loop velocity proportional-integral-derivative controller that references the predicted state rather than the raw visual measurement. The filter operates on a state vector comprising the image-plane marker features and their temporal derivatives, supplying temporally consistent, noise-attenuated estimates throughout the landing sequence. Performance is evaluated through comparative numerical simulation and real-flight experiments under uncontrolled outdoor disturbances and benchmarked against a conventional raw-measurement pipeline. The proposed integration reduces the closed-loop steady-state positional error by 86.7% and reduces the total three-dimensional root-mean-square error relative to the baseline, with improvements of 20.8% and 21.3% in the longitudinal and lateral channels, respectively. The closed-loop settling time is preserved across all conditions, confirming that the gains primarily arise from the enhanced state estimation rather than from the control law. Real-flight results demonstrate operation under uncontrolled outdoor disturbances, with horizontal velocity tracking root-mean-square errors of 0.162 and 0.190 m/s, without requiring an additional landing sensor beyond the downward-facing monocular camera.

Keywords: ArUco marker detection; Kalman state prediction; Monocular vision; Proportional-integral-derivative control; Unmanned aerial vehicle

1. Introduction

The advanced growth in popularity of unmanned aerial vehicles (UAVs), particularly quadcopters, has garnered significant attention from various sectors (Laghari et al., 2023; Mohsan et al., 2023). Various disciplines within military and civilian services use them for various applications. The cutting-edge technologies serve a wide range of military purposes at tactical, operational, and strategic levels worldwide (Alqudsi & Makaraci, 2025; Criollo et al., 2024; Paucar et al., 2018). In civilian services, UAVs have been used for disaster management (Mohd Daud et al., 2022), remote sensing (Deng et al., 2018; Li et al., 2019), search and rescue operations (Alotaibi et al., 2019; Rahman et al., 2018), precision agriculture (Murugan et al., 2017), emergency medical services (Nenni et al., 2020), and environmental surveillance, such as real-time peatland fire detection using infrared imaging (Qolbiyah et al., 2024), among numerous other applications (Adel et al., 2024; Ghamari et al., 2022).

Most UAV applications require the ability to precisely land on constrained platforms, such as construction and infrastructure inspection sites, maritime vessels, or designated charging

pads (Santos et al., 2024; Xin et al., 2022). In other words, a solid landing capability that does not require human intervention is essential. Another high demand from such complex tasks is an extensive range of sensors mounted on the quadcopter (Tsouros et al., 2019). However, there is a growing demand for computational power and energy with the increasing use of sensors, which further limits the quadcopter's flight time and its relatively small dimensions (Anderson & Gaston, 2013; Sonugür, 2023). The theoretical foundation of vision-based methods is the visual servoing paradigm, in which the control error is defined directly on image features rather than on reconstructed Cartesian pose (Chaumette & Hutchinson, 2006). The need for heavy-duty sensors, such as LiDAR or differential GPS equipment, can be avoided. Among the existing methods, the fiducial marker-based monocular landing method is at the top of the list, particularly in the use of ArUco and AprilTag (Kalaitzakis et al., 2021). Both methods detect a high-contrast border, decode an inner bit field, and infer the pose from the corner correspondence.

The comparative evaluation of these tags reveal complementary strengths. ArUco has a stronger localization accuracy, whereas AprilTag is more robust in detection. Both remain the actual choices for UAV landing stacks. Marker-design extensions, including embedded ArUco variants and customized fiducial markers, have been developed to improve the reliability of pose estimation under challenging conditions (Jurado-Rodriguez et al., 2021). Alternative square-binary designs, such as Stag refine detection stability (Heselden et al., 2025), have been adopted in landing stacks (Jones & Hauert, 2023; Schempp & Schulz, 2022). Monocular marker detection has been extensively used for pad localization and pose estimation, followed by the application of simple descent controllers (Aikins et al., 2024; Byun et al., 2025; Gaigalas et al., 2025; Guo et al., 2025). The differences are mainly in marker design, pad motion, and camera configurations. Although effective, these approaches are often sensitive to disturbances in pose estimation, such as motion blur, occlusion, or wind gusts that cause disturbances in pose estimation (Jung, 2024; Zhang et al., 2024). Any degradation in detection quality therefore surfaces in the control loop as a temporal defect: corrupted or missed detections delay the arrival of usable pose updates, so the controller acts on information that lags the vehicle's true state.

Collectively, these studies indicate that the dominant failure mode lies less in marker detection than in the temporal quality of the pose signal delivered to the controller. Most pipelines integrate the classical linear Kalman filter (Simon, 2006) or its extended variants (EKF) (Jarayaya et al., 2025; Nabavi et al., 2022; Semerikov et al., 2025). Guo et al. (2025) implemented an EKF to reduce descent instabilities in the terminal phase of an ArUco-based landing pipeline, maintaining the pose when the marker is intermittently invisible. Park et al. (2024) demonstrated that combining image filtering with a Kalman filter in a fiducial-based landing pipeline substantially improves pose estimation accuracy and noise rejection during the descent phase. In a closed-loop flight, Gamagedara et al. (2024) applied a Kalman filter to compensate for sensing or processing lags during autonomous take-off or landing in challenging environments, further emphasizing the improvement of robustness. In GNSS-degraded or denied conditions, Kalman filter fusion is the backbone of a dependable UAV localization, commonly coupling fiducial vision with inertial sensing (Chang et al., 2023; Fu et al., 2025). A recent Kalman filter using standard avionics achieved a sub-1 m/s wind speed error in flight tests, illustrating the direct improvement in navigation quality resulting from estimator design (Lee et al., 2022). This result is also a product of filtered fusion, enabling auxiliary estimands that are important for guidance.

Pose estimation propagates to the control mechanism. In landing and trajectory-tracking tasks, Kalman filters are often coupled with PID loops to enhance control against sensor noise. A comprehensive survey of PID control strategies for quadrotor UAVs confirms that cascade PID structures remain a dominant and effective choice for autonomous flight applications (Lopez-Sanchez & Moreno-Valenzuela, 2023). Adaptive PID and sliding mode variants have been shown to robustly handle variable payloads and external disturbances robustly (Eltayeb et al., 2022). Learning-based alternatives, such as neural network-based direct inverse control of quadcopter attitude and altitude, have also been demonstrated (Heryanto & Kusumoputro, 2021). However,

their training demands contrast with the transparency of fixed-gain designs. Kalman-based estimation enhances the accuracy and outlier tolerance of low-cost MEMS IMUs, producing cleaner inputs for PID or MPC control (Gamagedara et al., 2024). Zhou et al. (2025) estimated position with a Kalman filter and then drove cascade PID inner loops for thrust in PX4 autopilots, delivering a stable flight and noiseless feedback in indoor and outdoor environments. Referring to previous research on autonomous landing on moving or GPS-denied targets, the implementation of the Kalman filter with PID inner loops demonstrates a reliable performance. The practicality of the Kalman-PID pattern for monocular guided operations is underscored. However, in all of these pipelines, the filter output is consumed as a smoothed surrogate of the current measurement, so the controller continues to act on information that is at least one detection and processing cycle old. Smoothing attenuates the measurement noise but does not remove this cycle-old delay. For a descending quadrotor, the residual latency accumulates directly as image-plane error at touchdown.

Nonetheless, the integration of these established elements leaves a specific architectural gap: the Kalman filter serves exclusively as a measurement conditioner, and the controller references the filtered estimate of the current state. Therefore, the interval between marker detection and control actuation, which spans image acquisition, detection, estimation, and command transport, remains uncompensated, producing overshoot and oscillation that can precipitate landing failure, particularly under wind disturbance. This research bridges that gap by promoting the one-step-ahead Kalman prediction from an internal filtering byproduct to the outer-loop controller. The contributions are threefold. First, a prediction-informed control architecture is formulated in which the outer-loop velocity PID controller references the predicted image-plane state, providing structural compensation of one sampling interval of pipeline latency without requiring an additional landing sensor or modification of the existing outer-loop control law. Second, a ten-state constant-velocity Kalman design operating directly on the monocular image-plane features, including an occlusion-handling mode that propagates the prediction whenever detection is lost. Third, a three-condition evaluation protocol—real flight, simulation with prediction, and simulation without—isolates the estimator’s contribution from that of the controller, quantifying an 86.7% reduction in steady-state positional error and a 15.1% reduction in total three-dimensional RMSE, with operation demonstrated in real flight under uncontrolled outdoor disturbances. In contrast to prior Kalman-PID integrations, which attribute improvement to smoothing alone, this framework establishes prediction-as-reference as a distinct and separable mechanism for latency mitigation in monocular landing.

2. Methods

This section describes the proposed prediction-informed landing framework for autonomous quadrotor UAV landing. The framework adopts a two-tier control scheme that separates high-level perception and navigation from low-level attitude and motor control, enabling the onboard autopilot to execute the outer-loop velocity commands derived from monocular visual feedback to be executed independently of the inner-loop stabilization dynamics managed by the onboard autopilot.

2.1 System Architecture

Figure 1 shows the overall system architecture as the end-to-end processing pipeline from camera acquisition to vehicle actuation, which is implemented on a quadrotor platform using a two-layer control scheme that separates high-level perception and navigation from low-level attitude and motor control. The airframe is based on the Holybro X500 development kit, configured as an “X”-type quadrotor with four 2216 KV880 brushless motors driven by BLHeli-S electronic speed controllers rated at 20 A. A pixhawk-4 autopilot executes the inner-loop control, stabilizing the vehicle by modulating motor speeds and fusing onboard inertial and barometric measurements. In this study, the Pixhawk also provides local pose estimates by publishing the

IMU-based state to the `mavros/local_position/pose` topic and receives GPS data for global navigation toward the approximate landing area from an external module connected to the GPS port for global navigation toward the approximate landing area. A 4-cell LiPo battery (4200 mAh) powers the propulsion and flight-control electronics. In contrast, the companion computer and camera use an independent 5 V/3 A power bank to prevent brownouts or resets of the high-level processing during flight.

High-level computation is performed on a Raspberry Pi 4 Model B, which serves as the navigation and perception. The Ubuntu 20.04 server with ROS Noetic runs on the Raspberry Pi to reduce the processing overhead. Meanwhile, a USB camera with a 1280×720 resolution at 25 frames per second and a 140° field of view is mounted on the airframe, providing monocular images to a ROS package responsible for ArUco marker detection, pose estimation, and data logging. When the ArUco marker is within the field of view of the camera, it approximates the corresponding pose between the landing pad and the drone. Subsequently, the calculation process begins when the marker ID matches, and a Kalman filter filters it to mitigate image noise and intermittent detections. The filter state used by a PID controller, which is implemented as an ROS node, generates the position setpoints. These setpoints are transmitted to the Pixhawk 4 via a UART serial link using MAVLink through MAVROS in offboard mode. The hardware partition is reported in this level of detail to support reproducibility; methodologically, its significance lies in the bounded perception-to-actuation latency it imposes, which the prediction layer of Section 2.4 is designed to compensate.

The autonomous mission is initiated by commanding the drone to navigate using GPS toward a candidate landing platform that stores the coordinates in a local database on the Raspberry Pi. At time t , the current GPS position is compared with the database entries, and the target waypoint is selected as the target waypoint. After the vehicle reaches this waypoint, the system transitions from GPS-only guidance to combined GPS-vision guidance and finally to purely vision-based landing as soon as the correct ArUco marker is detected and tracked. The gradual transition from coarse GPS navigation to fine visual servoing enables the system to achieve marker-relative landings while maintaining robustness to temporary loss of visual detection through Kalman filtering.

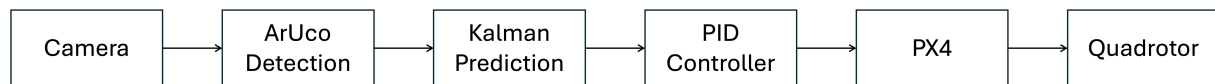


Figure 1 System architecture of the autonomous landing platform

The ground control station (GCS) runs Ubuntu 20.04 with ROS Noetic and QGroundControl. The GCS is connected to the Pixhawk 4 using a telemetry radio pair via the TELEM1 port. It enables independent bidirectional exchange between the flight data and mission commands. QGroundControl also establishes the telemetry link, monitors the drone's state, records flight logs, and performs the initial GPS calibration. Because the Raspberry Pi exposes an SSH interface, the operator can remotely configure ROS parameters remotely. This includes starting or stopping the autonomous landing nodes and retrieving experimental data without physical access to the drone. For additional safety, a standard RC transmitter-receiver pair is connected to the Pixhawk, allowing for manual takeover from offboard control in the event of software, communication, or hardware failure, and supporting initial tuning and validation flights in manual or assisted modes.

For preliminary testing, the same architecture is reproduced in a software-in-the-loop (SITL) environment for preliminary testing. The GCS operates the Gazebo 3D simulator, which features a quadrotor model equipped with simulated sensors and ArUco markers. Simultaneously, PX4 SITL replaces the physical Pixhawk 4 and exchanges data with ROS and Gazebo over MAVLink. This setup enables the evaluation and debugging of perception, Kalman filtering, and PID landing controller implementations on ROS using the same software stack as in the real-flight experiments.

2.2 Design of the Autonomous Landing System

The block diagram in Figure 2 summarizes the proposed autonomous landing system. Once the drone takes off, the ground control station issues a mission command to the drone once it takes off. Navigating to the corresponding marker target. Throughout this phase, the ArUco detection module, which is equipped on the Raspberry Pi, continuously processes the images continuously captured by the onboard camera. The system then transitions to a vision-based landing mode from pure GPS navigation once the landing marker is within the camera's field of view and the correct marker ID.

The ArUco detection module returns the pixel coordinates of the marker center (x_c, y_c) , the width and height of the observed marker (w, h) , and the angle of the camera's pose relative to the x-axis of the detected marker in the image frame (θ) for each incoming frame. These quantities form a measurement vector at time t , defined as follows:

$$z_t = [x_c, y_c, w, h, \theta]^T \quad (1)$$

To obtain a smooth estimate of the drone state $(\hat{x}_t)$ relative to the landing marker, the observed vector is passed to a discrete-time Kalman filter that combines the successive measurements. The system represents the target pose for a centered and aligned touchdown by comparing the estimated state with the desired reference (r_t) . Subsequently, the resulting error becomes the value supplied to a PID controller. In the marker coordinate frame, the PID controller generates velocity or position commands (u_t) , which are transmitted in offboard mode to the autonomous Pixhawk. The autopilot closes the inner attitude and motor speed loops, driving the drone along a descending trajectory until it satisfies the landing condition, and disarms the motors.

The segmentation shown in Figure 2 reflects the hardware architecture and clarifies the information flow from camera detection to low-level motor commands during the autonomous landing. The flowchart is organized into four color-coded layers. The blue perception layer covers image acquisition, ArUco marker detection, and extraction of the marker-feature measurement vector; the green estimation layer contains the ten-state constant-velocity Kalman filter and its one-step-ahead prediction; the orange control layer computes the tracking error against the predicted reference and generates velocity and yaw commands through the outer-loop PID; and the gray actuation layer conveys these commands via MAVROS/MAVLink offboard mode to the PX4 inner attitude and rate loops driving the motors.

2.3 Visual Marker Detection and Pose Observation

ArUco fiducial marker detection serves as the basis for visual landing. This subsystem executes the ArUco OpenCV module and invokes the *detectMarkers()* function, which forms the core of the marker detection workflow. This function takes a grayscale image from the camera as input and returns two primary data structures: *markerCorners* and *markerIds*. While the *markerIds* vector contains integer identifiers of all detected markers in the image, *markerCorners* stores the pixel coordinates of its four corner points for each detected marker. The corners return clockwise, starting from the top left corner. That is, (p_1, p_2, p_3, p_4) correspond to the top left, top right, bottom right, and bottom left corners, respectively. Only markers with a predefined target ID are retained for the landing task. If multiple markers are visible, discard the others.

The basic geometric quantities of the marker in the image plane are computed according to the corner coordinates. The Euclidean distance between any two points $\vec{p} = (p_x, p_y)$ and $\vec{q} = (q_x, q_y)$ is defined as follows:

$$d(\vec{p}, \vec{q}) = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2} \quad (2)$$

which is a direct consequence of the Pythagorean theorem. The width w and height h of the marker in pixels are obtained by applying Equation 2 to the corresponding pairs of opposite

corners, which are averaged to mitigate slight detection noise. The marker centroid (x_c, y_c) is computed as the arithmetic mean of the four corner coordinates as follows: The in-plane orientation (θ) of the marker with respect to the image x-axis is determined from the angle of the line segment joining the top left and top right corners, again using the corner coordinates returned by *detectMarkers()*.

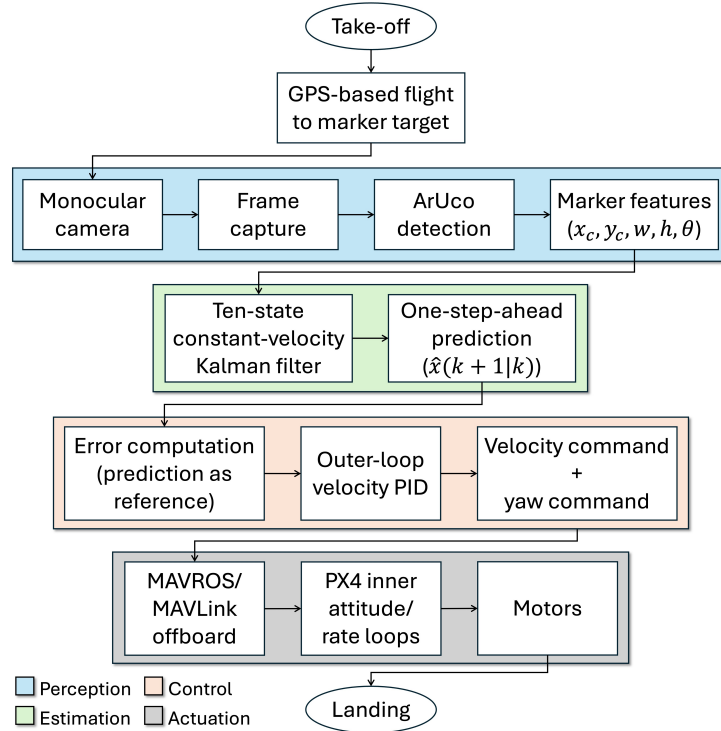


Figure 2 Block diagram of the autonomous landing system block diagram

These quantities are used to construct the observation vector z_t introduced in Equation 1. The center point (x_c, y_c) encodes the lateral displacement of the landing pad in the image plane. Conversely, the distance between the drone and the marker is indirectly related to the width and height of the observed marker (w, h) . Meanwhile, θ captures the misalignment between the drone and the pad. Therefore, the vector z_t provides a solid representation of the appearance of the marker in the image and serves as the measurement input to the Kalman filter state.

2.4 Kalman Filter State Estimation

An ArUco marker detection-based approach often exhibits stochastic jitter owing to corner localization errors, motion blur, illumination changes, and occasional missed detections. However, algorithms that use a series of measurements to estimate the unknown stochastic jitter sources are required. These algorithms are more accurate than those based on a single measurement. In addition, directly feeding these raw measurements to the incoming controller can produce high-frequency oscillations. Therefore, a discrete-time linear Kalman filter is employed to obtain a smooth and temporally consistent estimate of the landing-related features used for landing. Furthermore, it is ideal for continuously changing systems and requires only the storage of the current state and covariance, with no need to retain measurement history. The fixed-dimension linear formulation was successfully executed within the demonstrated 10 Hz onboard processing loop.

The constant-velocity model is preferred over higher-order alternatives on both observability and variance grounds. During the terminal descent, the marker's image-plane motion is dominated by slow relative drift, whereas the accelerations induced by gusts or corrective maneuvers are short-lived and effectively zero-mean at the 10 Hz observation rate; therefore, such transients

are therefore more appropriately absorbed into the process noise covariance than modeled as additional states. A constant-acceleration formulation would double the number of derivative states. However, at this rate, image-plane acceleration is only weakly observable from pixel measurements at this rate, so the added states inflate estimator variance without a commensurate reduction in bias. Consequently, the constant-velocity structure consequently offers the most favourable bias-variance trade-off for this observation geometry.

The landing marker features observed in the image plane are assumed to follow a constant-velocity model. For a point pixel detected in the image frame p_t , the discrete-time dynamics are described as follows:

$$p_t = p_{t-1} + \Delta t \dot{p}_{t-1} \quad (3)$$

$$\dot{p}_t = \dot{p}_{t-1} \quad (4)$$

where Δt is the sampling interval between successive observations. In the implementation, the visual observation stream is published at 10 Hz, hence $\Delta t = 0.1$ s. Based on the observation features (x_c, y_c, w, h, θ) , the Kalman filter state is defined as the observation vector augmented with its first-order derivatives:

$$x = [x_c, y_c, w, h, \theta, \dot{x}_c, \dot{y}_c, \dot{w}, \dot{h}, \dot{\theta}]^T \quad (5)$$

In its general discrete-time linear form, the Kalman filter uses a state transition model and a measurement model, as shown below:

$$x_t = A x_{t-1} + B u_t + w_t \quad (6)$$

$$z_t = H x_t + v_t \quad (7)$$

where u_t is the control input, B is the input matrix, $w_t \sim N(0, Q)$ is the zero-mean Gaussian process noise with covariance Q , and $v_t \sim N(0, R)$ is the measurement noise. Under the constant-velocity assumption, the state transition matrix takes the form of a block matrix as follows:

$$A = \begin{bmatrix} I_{5 \times 5} & \Delta t I_{5 \times 5} \\ 0_{5 \times 5} & I_{5 \times 5} \end{bmatrix} \quad (8)$$

It can be written explicitly in a 10×10 matrix form as

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (9)$$

Because the measurement vector z_t contains only the first five components (x_c, y_c, w, h, θ) , the observation matrix that selects those states is as follows:

$$H = [I_{5 \times 5} \quad 0_{5 \times 5}] \quad (10)$$

In this work, the estimator is based solely on camera observations, which permits a standard linear Kalman filter implementation without requiring Jacobian computation. Accordingly, the input term $B u_t$ in Equation 6 is omitted from the estimator, yielding

$$x_t = A x_{t-1} + w_t \quad (11)$$

At each time step, the filter performs a prediction phase followed by a correction phase when a valid observation is available. The prediction equations are:

$$\hat{x}_t^- = A \hat{x}_{t-1} \quad (12)$$

and the predicted error covariance is as follows:

$$P_t^- = A P_{t-1} A^T + Q \quad (13)$$

When an observation z_t is obtained, the following correction step is applied through the innovation residual and innovation covariance:

$$y_t = z_t - H \hat{x}_t^- \quad (14)$$

$$S_t = H P_t^- H^T + R \quad (15)$$

The Kalman gain, corrected state, and corrected covariance are then calculated as follows:

$$K_t = P_t^- H^T S_t^{-1} \quad (16)$$

$$\hat{x}_t = \hat{x}_t^- + K_t y_t \quad (17)$$

$$P_t = (I - K_t H) P_t^- \quad (18)$$

If the marker or the ID is undetected, the correction step is skipped and the predicted estimate $\hat{x}_t^-$ is propagated forward, maintaining the state used by the landing controller. Finally, the filtered estimate $\hat{x}_t$, particularly $\hat{x}_c, \hat{y}_c, \hat{w}, \hat{h}, \hat{\theta}$, is used as the reference signal for the PID controller to reduce jitter in the commanded lateral motion, yaw response, and descent behavior during the terminal landing phase.

The noise covariances governing the filter were configured as follows: The measurement covariance, R , was set from the empirical variance of the detected image-plane features recorded while the vehicle hovered above a stationary marker. Thus, R reflects the actual pixel-level noise of the detection stage rather than a nominal assumption. The process covariance Q was then selected to bound the model mismatch of the constant-velocity assumption, and the ratio between Q and R , which sets the filter bandwidth and hence the trade-off between responsiveness and smoothing, was refined in the software-in-the-loop environment prior to field deployment. Larger Q/R ratios tracked abrupt marker excursions more closely at the cost of reinjecting measurement jitter into the reference signal, whereas overly small ratios introduced estimator lag that negated the prediction's latency benefit. The deployed setting was selected at the knee of this trade-off. The deployed configuration used $R = \text{diag}(\sigma_{x_c}^2, \sigma_{y_c}^2, \sigma_w^2, \sigma_h^2, \sigma_\theta^2) = \text{diag}(10^{-4}, 10^{-4}, 10^{-2}, 10^{-2}, 10^{-2})$, with the centroid channels weighted two orders of magnitude tighter than the bounding-box and heading channels in line with the hover-measured detection variance, and $Q = 10^{-4} \times I_{10}$ over the $\Delta t = 0.1$ s sampling interval. This setting was held constant across all simulations and flight experiments.

2.5 PID Controller

During the terminal landing phase, the responsibility segmentation between the Raspberry Pi, which handles closed-loop visual servoing, and the PX4 autopilot, which handles the inner loop, is essential. The Kalman-filtered state is used to generate flight commands that drive the quadrotor to center the landing marker in the image plane. It also aligns the yaw of the vehicle with the marker orientation. Conversely, the inner loop ensures attitude stability and safe handling under real-time flight constraints.

Figure 3 depicts a cascade architecture from the inertial frame, including position and velocity control to body-frame attitude and angular-rate control, followed by motor mixing, is depicted in Figure 3 as the PX4 internal control structure. The outer loop generates intermediate setpoints, such as velocity and attitude, are generated by the outer loop. The inner loops track

these setpoints at higher frequencies. The proposed landing controller does not replace these inner loops; instead, it leverages them by providing setpoints via OFFBOARD mode, retaining PX4 as the low-level stabilization controller during landing maneuvers.

Figure S1 shows the complete closed-loop landing controller implementation is illustrated in Figure S1, where the tracking error is calculated from the Kalman-filtered visual state relative to the desired marker alignment. Let Sp_t denote the reference setpoint vector for the desired centered marker and yaw alignment and let $\hat{X}_t^{i=1:3}$ denote the corresponding controlled components of the Kalman-filtered estimate. The error vector at time t is defined as follows:

$$e_t = Sp_t - \hat{X}_t^{i=1:3} \quad (19)$$

where and in a consistent notation, $Sp_t = [x_c^*, y_c^*, \psi^*]^T$ and $\hat{X}_t^{i=1:3} = [\hat{x}_c, \hat{y}_c, \hat{\psi}]^T$. Here (x_c, y_c) represents the marker centroid in the image plane, and ψ represents yaw alignment referenced to the detected marker orientation θ . The PID controller produces the command vector expressed as follows:

$$U_t = [\dot{x}_c, \dot{y}_c, \dot{\psi}]^T \quad (20)$$

which corresponds to the commanded lateral velocity components and yaw rate transmitted as OFFBOARD setpoints via MAVROS. Conceptually, $\dot{x}_c$ and $\dot{y}_c$ are the velocity commands that reduce the centroid error in the horizontal plane, while $\dot{\psi}$ denotes the yaw command that nullifies the yaw misalignment relative to the marker. These setpoints are then tracked by PX4's internal loops as summarized in Figure 3. For each controlled channel (lateral x , lateral y , and yaw), the continuous-time PID law is

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau) d\tau + K_d \frac{de(t)}{dt} \quad (21)$$

and its discrete-time implementation at the sampling interval Δt is

$$u_t = K_p e_t + K_i \sum_{i=0}^t e_i \Delta t + K_d \frac{e_t - e_{t-1}}{\Delta t} \quad (22)$$

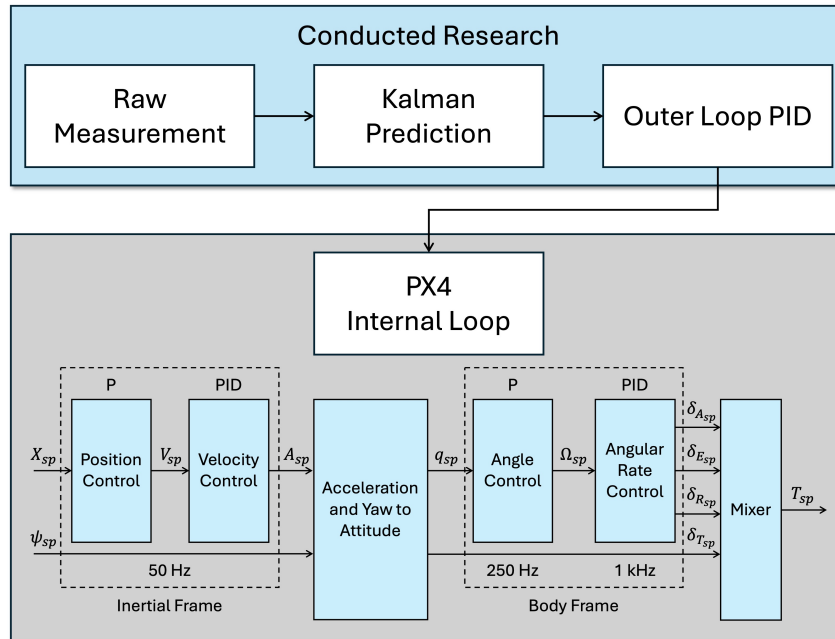


Figure 3 Control architecture of the proposed system: the outer-loop contribution of this research commanding the PX4 internal cascade controller diagram

The resulting command vector U_t is bounded to remain within safe operational limits before being published to PX4, ensuring that the high-level visual servoing remains compatible with the autopilot tracking capability. Finally, PX4 converts the received setpoints into motor thrust commands via its cascaded control pipeline and motor mixer, enabling stable tracking during the descent and alignment maneuvers until touchdown.

The architectural placement of the estimator warrants explicit control-theoretic comment. The end-to-end loop delay comprises image exposure and acquisition, marker detection, state estimation, and command transport to the autopilot, amounting to approximately one sampling interval of the 10 Hz perception loop. Since the outer-loop PID references the one-step-ahead prediction rather than the filtered estimate of the current state, the reference presented to the controller is advanced by exactly this interval; therefore, the predictor provides structural compensation of the dominant measurement path delay, which is functionally analogous to a Smith-predictor arrangement restricted to the feedback signal path. The effective prediction horizon is deliberately limited to a single interval ($\Delta t = 0.1$ s): within this horizon, the constant-velocity assumption holds to first order, whereas longer extrapolation would accumulate model mismatch faster than it removes delay.

The prediction layer does not alter the nominal closed-loop characteristic equation. The Kalman filter is located in the reference path and receives no feedback from the control input; thus, it does not alter the PX4 cascade's loop transfer function. The inner attitude and rate loops execute in PX4 at approximately 250 Hz and 1 kHz, respectively, placing their bandwidths more than an order of magnitude above the 10 Hz outer loop and satisfying the time-scale separation required for cascaded designs. Consequently, the stability margins established by the tuned outer-loop gains of Section 3.1 are unchanged by the introduction of prediction—a property empirically corroborated by the near-identical settling times of the predicted and baseline configurations reported in Section 3.5.

This architectural argument can be formally stated. Let

$$P(z) = \frac{\lambda \Delta t}{z - 1} \quad (23)$$

denote the discrete-time response of an image-plane feature to the commanded velocity over one $\Delta t = 0.1$ s cycle, where λ (px·m⁻¹) is the image-scale gain—measurable as the detected marker width divided by the 0.18 m physical marker side—and let the one-cycle computation-and-transport latency is represented by z^{-1} . With the outer-loop PID $C(z)$ of Table S1 and the predictor $F(z)$, the closed-loop transfer function from the reference to the feature is given by

$$\frac{y}{r} = \frac{F(z) C(z) P(z) z^{-1}}{1 + C(z) P(z) z^{-1}} \quad (24)$$

The predictor appears only in the numerator, whereas the characteristic equation

$$1 + C(z) P(z) z^{-1} = 0 \quad (25)$$

and hence every closed-loop pole is independent of the prediction mechanism. Retaining the dominant proportional term (integral action is two orders of magnitude smaller, and derivative action contributes to the phase lead, so its neglect is conservative), the characteristic equation reduces to the following:

$$z^2 - z + K = 0 \quad (26)$$

With $K = K_p \lambda \Delta t$, which satisfies the Jury stability conditions for $0 < K < 1$. With $K_p = 0.005$ – 0.006 this bound admits image-scale gains up to $\lambda \approx 1.7 \times 10^3$ px·m⁻¹, whereas the deployed geometry spans $\lambda \approx 2.5 \times 10^2$ px·m⁻¹ at detection onset (marker width ≈ 45 px) to $\approx 1.4 \times 10^3$ px·m⁻¹ at the 0.198 m touchdown-registration range. The loop operates inside the stability region throughout the descent and disarms before approaching the bound. At the representative approach scale, the loop crossover lies near 1.5 rad·s⁻¹, corresponding to a

phase margin of approximately 77° and a delay margin of approximately 0.9 s—9 control cycles beyond the one-cycle latency already compensated by the predictor. Finally, an erroneous prediction perturbs only the reference channel and reaches the output through the complementary sensitivity.

$$T(z) = \frac{L(z)}{1 + L(z)} \quad (27)$$

it can therefore degrade tracking accuracy but does not alter the nominal linear closed-loop pole locations under the adopted model; nonlinear effects, such as actuator saturation, mode switching, or estimator divergence, lie outside this linearized analysis.

2.6 Simulation and Experimental Setup

The simulation environment is constructed within Gazebo 11, where the ArUco landing marker is rendered as a textured flat plane at world coordinates (7.5, 5.5, 0) m, consistent with the real-flight target position. The Gazebo wind plugin introduces wind disturbance, which applies a persistent lateral body force to the UAV model to replicate the aerodynamic loading encountered during outdoor trials. As illustrated in Figure 4, the PX4 firmware operates in software-in-the-loop mode, exchanging MAVLink messages with the Gazebo physics engine, relaying simulated vehicle states from Gazebo to PX4 and actuator inputs from PX4 back to Gazebo while the ROS Noetic computation graph hosts the *marker_detector*, *drone_controller*, and *drone_offboard* nodes over the same topic interfaces employed in real flight. This architecture ensures that the full software stack is executed without modification across both simulation and real-flight deployments.

Real-flight experiments were performed on an open grass field under uncontrolled outdoor conditions (Figure S2) under uncontrolled outdoor disturbance conditions throughout all trials. The landing target was an ArUco marker of approximately 0.18 m per side, which was printed on A4 paper and mounted on a rigid white backing board. The touchdown was registered when the vehicle descended to 0.198 m above the pad plane and disarmed. Five independent flight trials—presented here as a proof-of-concept field validation rather than a statistical demonstration of robustness—are conducted under identical initial conditions, with the UAV departing from a fixed start position of (0, 0, 0) m and executing the complete mission, loiter, and landing sequence autonomously without operator intervention. All five trials terminated in autonomous touchdown without manual takeover. Section 3 and Supplementary Table S6 quantitatively analyze the three trials with complete onboard position logs. Post-flight trajectory data were extracted from the MAVROS log and processed in Python for statistical error analysis. In the simulation campaign, wind disturbance is injected explicitly and varied from 0 to 30 m/s, and the resulting image-plane state trajectories obtained with and without the Kalman filter, together with the corresponding position-error traces across the wind settings, are provided as supplementary material. Wind speed is not directly instrumented in real-flight trials; its aerodynamic effect is instead captured implicitly through the trajectory deviations quantified in Section 3.

3. Results and Discussion

This section presents the simulation and experimental results of the proposed prediction-informed landing framework, which integrates Kalman state prediction with an outer-loop velocity PID controller referenced directly to the predicted image-plane states. The evaluation encompasses three comparative conditions: real-flight (RF) experiments, numerical simulation with the Kalman Filter (Sim KF), and numerical simulation without the Kalman Filter (Sim No KF), designated hereafter as the baseline. The analysis is structured sequentially to address PID controller parameterization, three-dimensional flight trajectory reconstruction, per-axis translational tracking, image-plane state evolution, closed-loop transient performance, positional root mean square error (RMSE) statistics, and outer-loop velocity tracking fidelity.

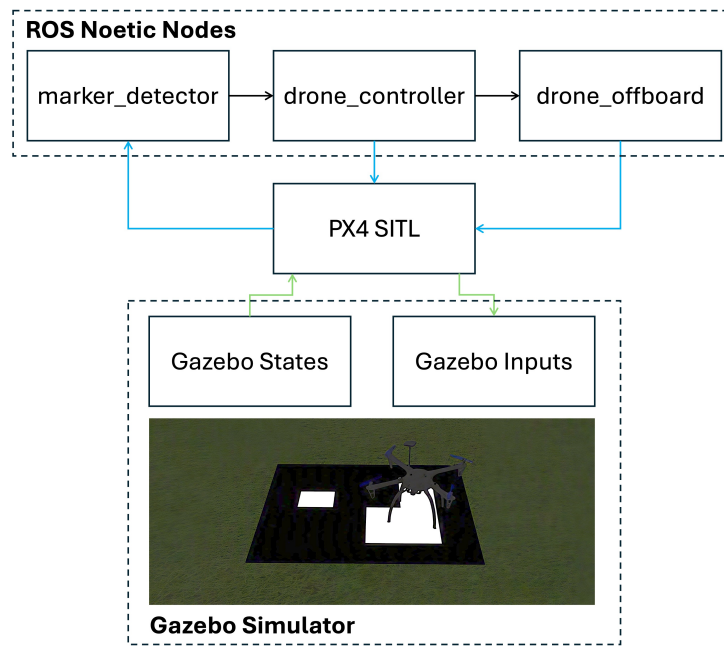


Figure 4 In-the-loop simulation framework

3.1 PID Controller Parameterization

Table S1 summarizes the outer-loop velocity PID gains adopted in this work are summarized in Table S1. Three independent control loops regulate longitudinal velocity (v_x), lateral velocity (v_y), and yaw rate ($\dot{\psi}$), with error signals derived from the image-plane centroid displacement in the y- and x-axes and from the angular heading residual (θ). For the three loops, the proportional gains were set to 0.00600, 0.00500, and 0.00400 for the three loops, while the derivative gains were assigned as 0.00085, 0.00080, and 0.00050, and the integral gains were maintained at subordinate magnitudes of 0.00006, 0.00005, and 0.00001 to suppress wind-up accumulation. The deliberate asymmetry between longitudinal and lateral gains reflects the differential sensitivity of the monocular image plane to forward versus sideward motion.

This asymmetry is consistent with the Kalman observation model's pixel-based error normalization employed in the Kalman observation model. The values were obtained through a sequential loop-closure procedure conducted in the software-in-the-loop environment: the proportional gains were increased until the onset of the oscillatory image-plane response and then reduced with a safety margin. Derivative gains were added to damp the residual transient, and the deliberately small integral gains were retained solely to remove residual bias while avoiding integrator wind-up during the terminal phase, in which the error signal is persistently small. The resulting gains were held invariant across both simulation and real-flight conditions to ensure a fair comparative evaluation of the Kalman prediction benefit alone.

3.2 Three-Dimensional Flight Path Analysis

The reconstructed three-dimensional positional trajectories for all three conditions, spanning from the common departure point (0, 0, 0) m to the designated landing target at (7.5, 5.5, 0) m, are presented in Figure 5, decomposed into a top view of the ground plane (Figure 5a) and a side view of the descent profile (Figure 5b), and the three-dimensional view retained in Figure 5c for overall spatial context. The simulated trajectory with Kalman prediction demonstrates the tightest geometric correspondence with the intended terminal coordinates, vertically converged in a near-straight descent profile with minimal lateral excursion during the approach phase. The baseline simulation exhibits a discernibly wider ground-plane excursion (Figure 5a) and a transient altitude exceedance peaking at approximately 5.8 m (Figure 5b) during the

loiter-to-land transition, which can be attributed to the delayed error feedback inherent in raw visual measurement pipelines. In contrast, the real-flight trajectory follows a globally consistent path. However, it deviates in the altitude envelope. It exhibits oscillatory lateral displacement during the mid-flight segment, which is consistent with the absence of aerodynamic perturbations, including wind loading, in the idealized simulation environment. Because the wind field was not instrumented, this attribution is inferred from the trajectory record rather than measured. The Kalman-integrated simulation replicates the qualitative descent geometry of the real flight more accurately than the baseline, confirming that the predictive state estimation effectively internalizes a portion of the dynamic uncertainty that would otherwise manifest as measurement noise in the visual feedback loop.

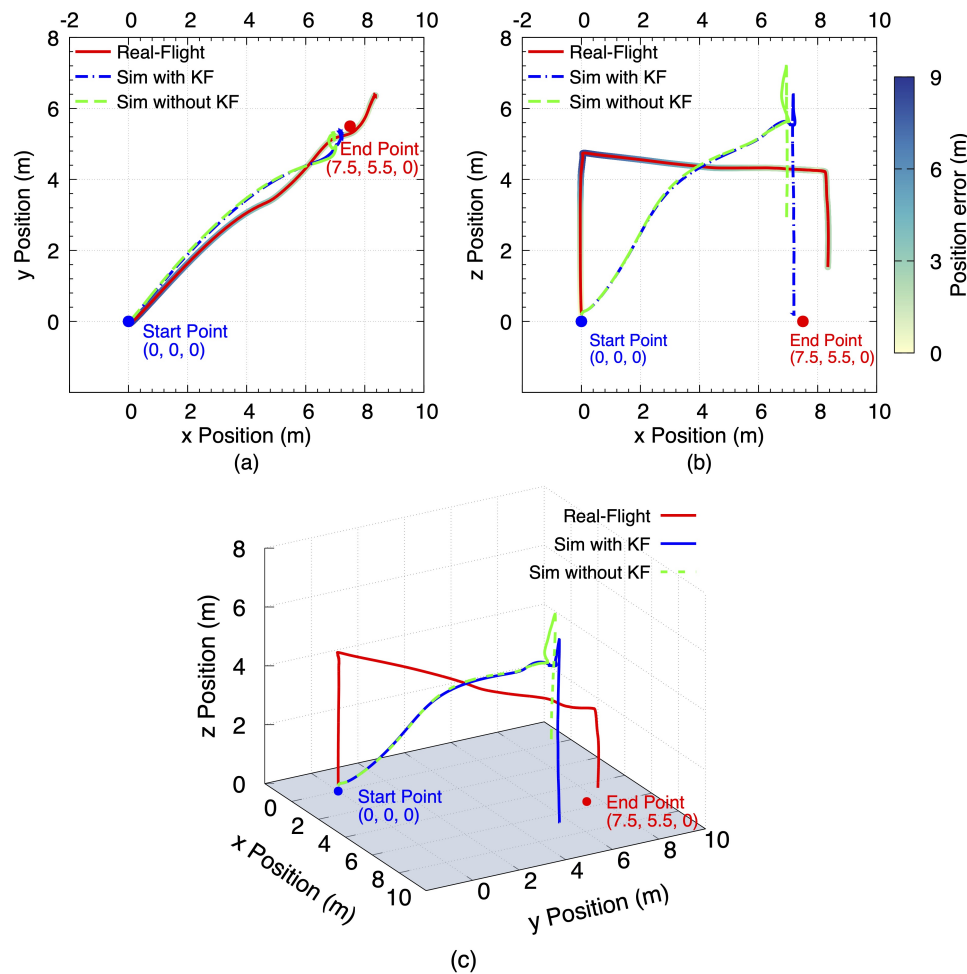


Figure 5 Comparison of the flight path of the UAV between real-flight experiments and numerical simulations with and without Kalman state prediction: (a) top view (x–y plane); (b) side view (x–z plane); (c) three-dimensional trajectory

3.3 Translational Position Tracking

The three-dimensional trajectory is decomposed into independent per-axis position-time histories in Figure 6, providing higher-resolution insight into the temporal dynamics of each translational degree of freedom. In the longitudinal axis (Figure 6a), both simulation conditions converge to the setpoint $x = 7.5$ m within approximately 15 s, with the KF-augmented trajectory reaching the steady state marginally earlier owing to its predictive error correction. The real-flight curve exhibits a characteristic delay phase from 0 to 10 s, during which the UAV holds its initial position while completing the mission planning waypoint, followed by a monotonic rise phase that overshoots the setpoint by approximately 0.5–0.8 m before recovering, consistent

with the onboard velocity controller's finite bandwidth under actual inertial loading. In the lateral axis (Figure 6b), the simulation traces converge smoothly to $y = 5.5$ m, whereas the real-flight trajectory exhibits a stepped progression, reflecting lateral drift corrections consistent with uncontrolled outdoor disturbances, ultimately settling within the acceptable landing envelope. The altitude channel (Figure 6c) reveals the most pronounced inter-condition divergence, where the Kalman filter executes a smooth monotonic descent from the cruise altitude of approximately 5 m to the landing setpoint.

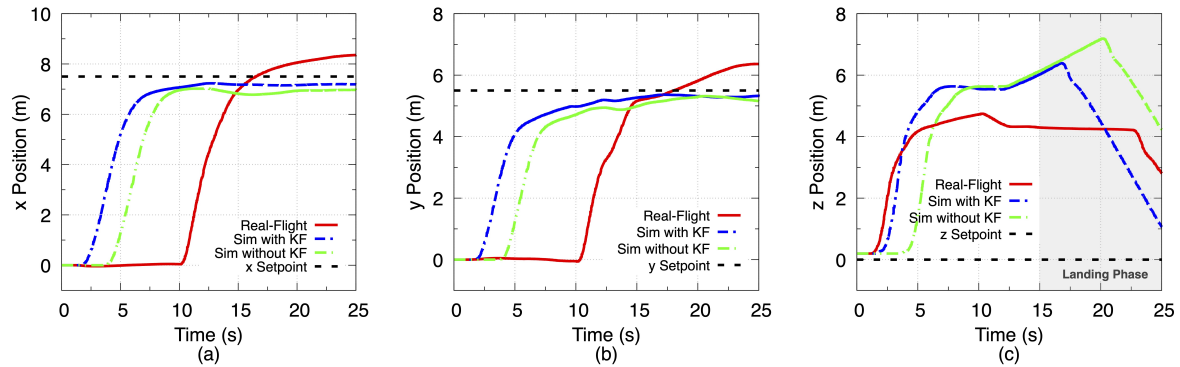


Figure 6 Comparison of translational position tracking in the x-, y-, and z-axes between real-flight experiments and numerical simulations with and without Kalman state prediction for the monocular unmanned aerial vehicle landing task. (a) x-, (b) y-, and (c) z-position trajectories

In contrast, the baseline simulation shows a transient altitude spike before the landing phase. The real-flight altitude trace shows a more complex two-stage descent profile, with a partial descent followed by a re-climb and final approach, attributed to the ArUco marker re-acquisition logic triggered when the marker temporarily exits the camera field of view due to attitude perturbations consistent with uncontrolled outdoor disturbances. Therefore, the Kalman predictor's ability to sustain a valid state estimate during brief marker occlusion events is particularly consequential for maintaining altitude setpoint fidelity in real-world deployments. The repeatability of the real-flight campaign is characterized separately in Supplementary Figure S3, which reports the phase-normalised mean ± 1 SD position trajectories across the trials with complete onboard position logs, together with the corresponding per-trial dispersion statistics (Table S6).

3.4 Image Plane State and Attitude Comparison

Figure 7 shows a comprehensive five-panel comparison of the image-plane controller states and vehicle attitude across all three conditions is provided in Figure 7. The horizontal centroid coordinate x_c (Figure 7a) exhibits a characteristic high-frequency oscillation in the real-flight data, consistent with the combined effects of uncontrolled outdoor aerodynamic disturbances and image processing latency on pixel centroid estimation. The simulation with the Kalman filter trace maintains a substantially smoother progression toward the frame center than the simulation without the Kalman filter curve. The vertical centroid y_c (Figure 7b) shows greater consistency across all three conditions during the initial approach phase, with the Kalman filter simulation tracking the real-flight asymptote most accurately. Divergence occurs beyond 15 s, where the real-flight trajectory reflects the marker re-acquisition event, producing a transient centroid reset not reproduced in either simulation condition.

The yaw angle history (Figure 7c) reveals a persistent negative heading bias in the real-flight data, stabilizing near -32 degrees, while both simulation conditions exhibit a converging yaw correction trajectory. The real-flight yaw steady-state offset is attributable to an initial heading misalignment between the UAV body frame and the ArUco marker orientation that was not compensated during pre-flight calibration. Despite this offset, the yaw PID loop suc-

cessfully arrests drift accumulation despite this offset, confirming the effectiveness of the angular error formulation based on the detected marker heading angle θ . Figures 7d and 7e plot the detected bounding box width and height in pixels, respectively, which serve as monotonically increasing proxies for the UAV's altitude proximity to the marker. The real-flight bounding box dimensions exhibit a dramatically steeper rise from approximately 15 s onward, consistent with the observed outdoor aerodynamic disturbances during the final approach. Both simulation conditions lag this growth rate, though the Kalman filter simulation more closely matches the real-flight bounding-box trajectory during the final critical approach window. Table S5 (Supplementary Material) reports the mean, standard deviation, and maximum absolute tracking error of each image-plane state—centroid coordinates, yaw, and bounding-box width and height—for the three experimental conditions, conveying the aggregate contrast that box-plot summaries would provide while preserving the transient structure that constitutes the primary evidence in Figure 7.

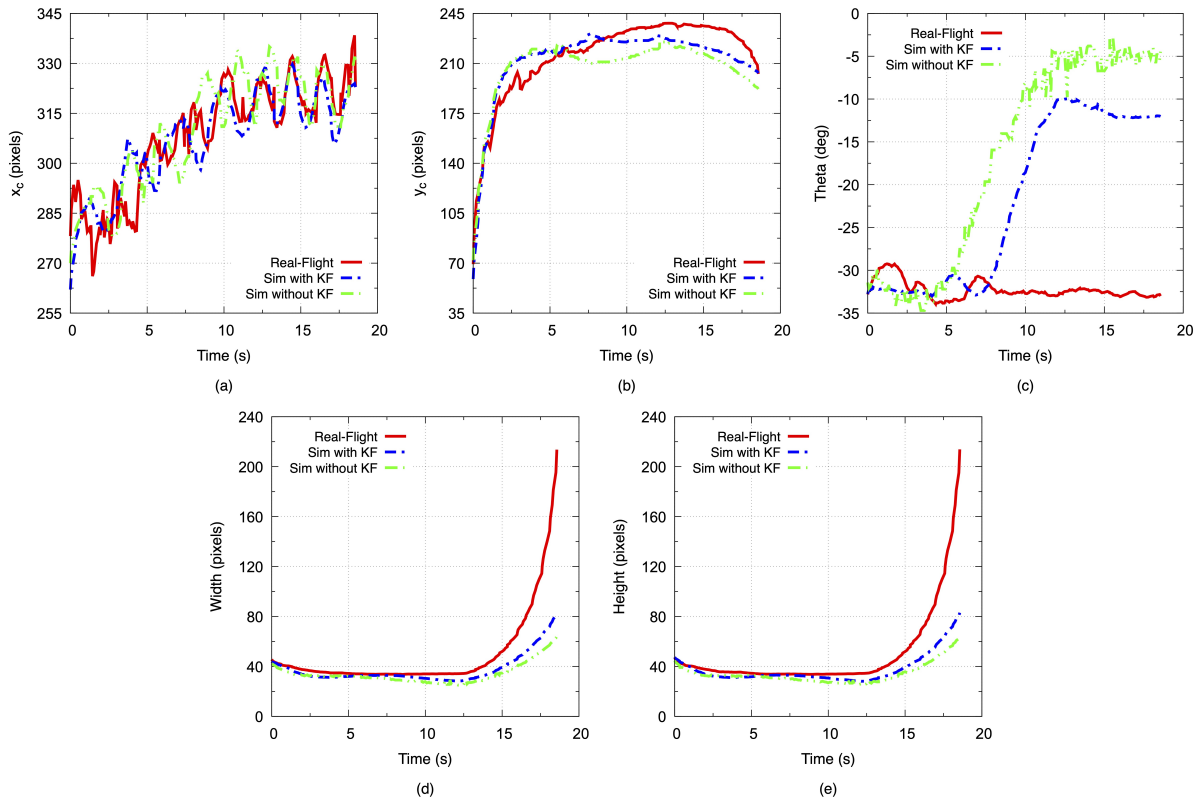


Figure 7 Comparison of the controller state in the image plane and attitude domain between real-flight experiments and numerical simulations with and without Kalman state prediction for the monocular unmanned aerial vehicle landing. (a) Horizontal pixel coordinate of the target centroid x_c ; (b) vertical pixel coordinate of the target centroid y_c ; (c) yaw angle θ of the UAV (deg); (d) detected target width (pixels); (e) detected target height (pixels)

3.5 Closed-Loop Transient Performance Metrics

Table S2 shows the classical closed-loop transient performance metrics for the three experimental conditions. The rise time (t_r) for both simulation conditions is nearly identical (0.0215 s vs. 0.0221 s), with the Kalman-augmented model exhibiting a marginally extended rise time attributable to the predictor's inherent one-step latency introduced by the state transition model. The real-flight rise time of 0.0145 s is considerably shorter, consistent with the higher initial velocity magnitudes during the physical approach and with the direct actuation of the flight controller inner loop, bypassing specific computational latencies in the simulation middleware.

The settling times (t_s) across all three conditions converge to a narrow band of 2.8205–2.8214 s, confirming that the closed-loop bandwidth defined by the outer PID structure is not materially altered by the outer PID structure. This result validates the design decision to confine the Kalman prediction to the observation preprocessing stage rather than embedding it within the PID feedback path, thereby preserving the stability margins and bandwidth characteristics. The maximum overshoot (M_p) is zero in both simulation conditions, as expected given the employed conservative gain selection strategy employed. However, the real-flight experiment shows an overshoot of 12.94%, indicating dynamic coupling between the horizontal and vertical channels under attitude perturbations consistent with uncontrolled outdoor disturbances. The steady-state error (e_{ss}) is the most discriminating metric, where the baseline simulation records a substantially elevated error of 2.9628 m, while the proposed Kalman filter simulation reduces this to 0.3939 m (86.7% reduction) by supplying the PID controller with a predicted centroid position that anticipates the marker's future location in the image plane. As expected for an unstructured outdoor environment with residual aerodynamic disturbances not captured in the numerical model. Collectively, these findings confirm that the primary mechanism by which Kalman prediction enhances landing precision operates through steady-state error attenuation rather than transient response modification.

3.6 Statistical Positional Error Analysis

Table S3 presents the root-mean-square positional error (RMSE) decomposed by coordinate component is presented in Table S3. The RMSE metric provides a statistically robust, trajectory-integrated measure of positional fidelity that complements the transient metrics reported in Table S2. The Kalman-augmented simulation achieves substantially reduced RMSE relative to the baseline across all horizontal components, where the longitudinal RMSE decreases from 3.463 m to 2.742 m (20.8% reduction), the lateral RMSE decreases from 2.495 m to 1.964 m (21.3% reduction), and the total 3D RMSE decreases from 6.645 m to 5.640 m (15.1% reduction). The altitude channel exhibits the smallest relative gain: the KF simulation records 4.520 m against 5.093 m for the baseline, an 11.3% reduction that is roughly half their relative magnitude but consistent in direction with the horizontal channels but roughly half their relative magnitude. This attenuated benefit is a structural consequence of the observation model rather than an anomaly: the altitude is not measured directly but inferred from the bounding-box width and height, whose growth during descent is nonlinear in range and sensitive to perspective distortion. Therefore, the constant-velocity prediction is least representative in the vertical axis. The same mechanism explains why the improvement from prediction concentrates in the longitudinal and lateral channels, where the centroid coordinates constitute direct, approximately linear observations of the tracked state.

The real-flight RMSE values (longitudinal: 4.929 m, lateral: 3.616 m, altitude: 3.893 m, total: 7.248 m) systematically exceed the simulation conditions across the horizontal components, which is the expected consequence of cumulative environmental disturbances acting over the entire flight duration. The altitude RMSE in real flight (3.893 m) is paradoxically lower than both simulation values, an outcome explained by the real autopilot's aggressive altitude correction maneuver executed by the real autopilot in response to the observed altitude offset, which inadvertently reduces the altitude RMSE relative to the nominal setpoint. These results collectively reinforce the improved performance of the KF-integrated pipeline relative to the tested raw-measurement baseline in constrained or structured environments and provide a quantitative basis for expecting further gains in real-flight performance as aerodynamic disturbance rejection and sensor fusion capabilities are improved in future work.

3.7 Outer-Loop Velocity Tracking Performance

Supplementary Figure S6 illustrates the complete velocity-tracking performance of the outer-loop PID controller under the proposed Kalman-integrated framework throughout the

entire flight mission, segmented into mission, loiter, and landing phases, is illustrated in Supplementary Figure S6. In the longitudinal channel (Figure S6a), the measured velocity v_x closely tracks the commanded setpoint profile across all three flight phases, with peak deviations occurring at the mission-to-loiter transition, where the setpoint undergoes a rapid sign reversal. The lateral channel (Figure S6b) exhibited comparable tracking fidelity, with measured v_y following the setpoint with a consistent phase lag of approximately 0.5 s during the high-rate loiter maneuver segment. In the vertical channel (Figure S6c), the tracked v_z exhibits the most complex dynamics, including a transient undershoot at mission initiation (attributable to the altitude hold competition with the outer loop command) and a divergence during the landing phase beyond 28 s, where the real autopilot transitions to an aggressive touchdown mode that overrides the outer velocity setpoint. Figure 8c summarizes the per-channel RMSE of this velocity-tracking performance is summarized in Figure 8c, and the corresponding terminal positional accuracy is presented in Figures 8a and 8b.

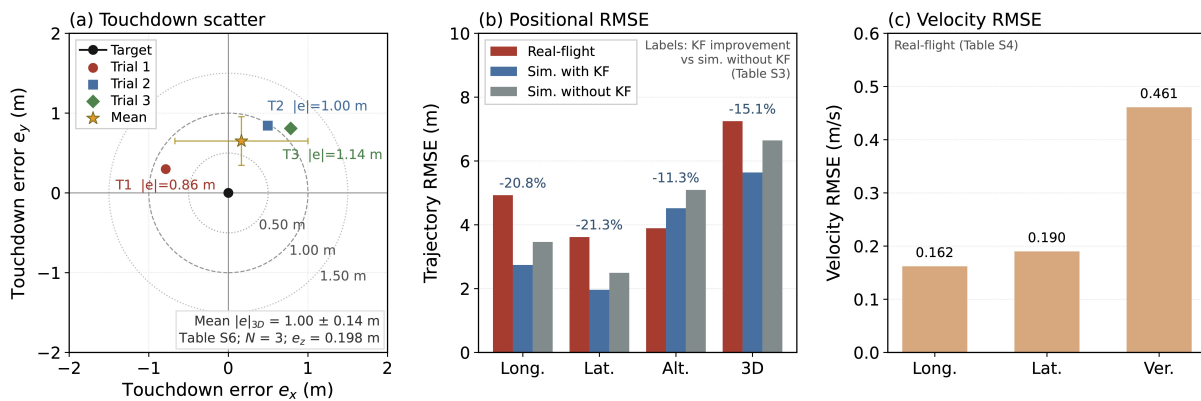


Figure 8 Summary of terminal accuracy and error summary of the proposed framework

Table S4 Quantifies velocity-tracking performance using per-channel RMSE metrics. The longitudinal RMSE of 0.162 m/s represents the highest tracking precision among the three channels, consistent with the greater sensitivity of the monocular system to forward motion in the image y-axis. The lateral RMSE of 0.190 m/s is marginally higher, reflecting the asymmetric pixel-scale resolution of lateral versus longitudinal displacement at typical approach altitudes. The vertical velocity RMSE of 0.461 m/s is substantially higher owing to the vertical channel's reliance on the ArUco bounding box size as a proxy for altitude. This signal exhibits nonlinear scaling and is susceptible to perspective distortion during aggressive pitch maneuvers. Taken together, the RMSE values confirm that the proposed framework delivers effective horizontal velocity regulation for marker-relative terminal guidance under the tested conditions, while identifying vertical velocity regulation as the primary target for performance improvement in subsequent system iterations, potentially through the integration of a dedicated range sensor or barometric altitude fusion within the Kalman observation model.

Beyond the altitude proxy, three further limitations bound the present findings and merit explicit statement. First, the residual real-flight steady-state error of 1.94 m reflects the aggregate of unmeasured wind loading, monocular-scale ambiguity, and re-acquisition transients following momentary detection loss and cannot be attributed to a single mechanism without instrumented disturbance measurement. The ArUco landing marker used in the field campaign measures approximately 0.18 m per side, and the final touchdown offsets of the three fully logged trials were 0.87–1.14 m, with a mean 3D offset of 1.00 ± 0.14 m (Figure 8a; Supplementary Table S6). The identical vertical component across trials (0.198 m) reflects the altitude at which touchdown is registered rather than lateral displacement. Therefore, the 1.94 m figure is therefore the steady-state value of the closed-loop tracking-error metric (Table S2), aggregated over the terminal trajectory segment, and overstates the physical touchdown displacement by nearly a factor of two. Supplementary Table S7 benchmarks the proposed system against recent esti-

mator, guidance, and learning-based landing frameworks in terms of estimator and controller, sensing configuration, real-flight validation, reported landing error, computation platform, disturbance condition, and principal limitation. The comparison indicates that centimeter-level accuracies are reported under short-range, station-type conditions or with dedicated positioning aids, whereas meter-level touchdown offsets are characteristic of fully autonomous outdoor campaigns without motion capture or RTK support, consistent with the 1.00 ± 0.14 m mean offset obtained here.

The positioning that emerges from Table S7 is specific: two of the six comparator systems report simulation-only evidence, so their performance gains remain unvalidated in the field; the centimeter-level result was obtained in short-range, station-type landings with a GPS-aided approach; and the one system operating in a comparably harsh outdoor environment achieves its robustness through RTK-GPS infrastructure and purpose-built hardware rather than through the sensing chain itself. Among the comparator studies included in Table S7, none combines all of the characteristics of the present framework: fully autonomous outdoor flight validation on an infrastructure-free, single-monocular-camera sensing chain, with transparent per-trial touchdown reporting and a controlled three-condition ablation that isolates the estimator's contribution. The meter-level terminal offsets and limited trial count are the explicit cost of operating in this most constrained sensing regime, and they delimit the deployment envelope within which the reported gains are applicable. Second, the evaluation deliberately isolates the prediction mechanism against a PID-only baseline; positioning the framework against alternative estimators (EKF, UKF) and predictive or learning-based controllers requires a common experimental protocol and therefore lies beyond the scope of the present study. Third, the five-flight campaign demonstrates repeated successful autonomous touchdowns under the tested field configuration. A larger campaign must formally quantify repeatability and uncertainty formally. These limitations delimit the claims made here: the demonstrated effect is the separable benefit of prediction-as-reference within a fixed, deliberately simple control structure.

4. Conclusion

This paper presents a prediction-informed autonomous landing framework for monocular quadcopter unmanned aerial vehicles (UAVs). In this framework, a discrete-time Kalman filter tuned for marker motion in the image plane provides predicted centroid states directly to an outer-loop velocity PID controller, replacing the raw or smoothed visual measurements used by conventional pipelines. The framework was evaluated through numerical simulations spanning a systematic 0–30 m/s wind sweep and real-flight experiments under uncontrolled outdoor disturbances, using a three-condition comparison—real flight, simulation with a Kalman filter, and simulation without a Kalman filter—that isolates the contribution of the prediction stage. Integrating the Kalman state prediction into the PID reference pathway reduces the closed-loop steady-state positional error by 86.7% in the simulation, and the total three-dimensional RMSE by 15.1% relative to the raw-measurement baseline, with the largest gains in the horizontal channels (20.8% longitudinal and 21.3% lateral). The Kalman integration preserves the closed-loop bandwidth, with settling times across all three conditions converging within a 9-ms band and maintaining zero overshoot in both simulation conditions. This confirms that the performance gains are primarily attributable to enhanced state estimation rather than changes in the control law. In real-flight experiments, the outer-loop velocity controller achieves longitudinal and lateral RMSE values of 0.162 and 0.190 m/s, demonstrating that the precision of horizontal velocity regulation is sufficient for marker-relative terminal guidance under the tested conditions, and the Kalman predictor sustained valid image-plane state estimates through transient marker occlusions that would otherwise cause abrupt setpoint discontinuities. These results indicate that a fixed-dimension linear image-plane Kalman filter, operating within the demonstrated 10 Hz onboard loop, can deliver meaningful landing-precision gains without requiring an additional landing sensor, stereo vision, depth sensing, or elevated onboard computing. Feasibility is demonstrated on a commodity quadcopter built from standard PX4/ROS components of the

kind operated in GPS-denied or signal-constrained environments where marker-relative landing is operationally critical, such as mobile-platform recovery, parcel delivery onto compact pads, maritime landing aboard moving vessels, and autonomous battery-swap charging. Decoupling state prediction from the PID feedback loop, rather than embedding the filter within the integrator, preserves closed-loop stability margins while supplying a temporally consistent reference robust to camera frame dropout and centroid noise.

Acknowledgements

The authors gratefully acknowledge Telkom University for covering the publication fees of this research article and the National Research and Innovation Agency (BRIN) for the funding support that made this research possible.

Author Contributions

Nada Syifa Qolbiyah: data curation, formal analysis, visualization, writing original manuscript, review and editing. Muhammad Fahmi Khoirurrijal: methodology, investigation, and data collection. Ferry Faizal: conceptualization, methodology, supervision, review and editing. Irwan Purnama: conceptualization, methodology, supervision, funding acquisition, review and editing. All authors have read and agreed to the published version of the manuscript.

Conflict of Interest

The authors declare no conflicts of interest.

References

- Adel, A., Alani, N. H. S., Thompson Whiteside, S., & Jan, T. (2024). Who is watching whom? Military and civilian drone: Vision intelligence investigation and recommendations. *IEEE Access*, 12, 177236–177276. <https://doi.org/10.1109/ACCESS.2024.3505034>
- Aikins, G., Jagtap, S., & Nguyen, K. D. (2024). A robust strategy for UAV autonomous landing on a moving platform under partial observability. *Drones*, 8(6). <https://doi.org/10.3390/drones8060232>
- Alotaibi, E. T., Alqefari, S. S., & Koubaa, A. (2019). LSAR: Multi-UAV collaboration for search and rescue missions. *IEEE Access*, 7, 55817–55832. <https://doi.org/10.1109/ACCESS.2019.2912306>
- Alqudsi, Y., & Makaraci, M. (2025). UAV swarms: Research, challenges, and future directions. *Journal of Engineering and Applied Science*, 72(1), 1–24. <https://doi.org/10.1186/s44147-025-00582-3>
- Anderson, K., & Gaston, K. J. (2013). Lightweight unmanned aerial vehicles will revolutionize spatial ecology. *Frontiers in Ecology and the Environment*, 11(3), 138–146. <https://doi.org/10.1890/120150>
- Byun, W., Huh, S., Jang, H., Yu, S., Lim, S., Lee, S., & Nam, W. (2025). Vector field-based robust quadrotor landing on a moving ground platform. *Aerospace*, 12(7), 1–19. <https://doi.org/10.3390/aerospace12070590>
- Chang, Y., Cheng, Y., Manzoor, U., & Murray, J. (2023). A review of UAV autonomous navigation in GPS-denied environments. *Robotics and Autonomous Systems*, 170, 104533. <https://doi.org/10.1016/j.robot.2023.104533>
- Chaumette, F., & Hutchinson, S. (2006). Visual servo control. I. Basic approaches. *IEEE Robotics and Automation Magazine*, 13(4), 82–90. <https://doi.org/10.1109/MRA.2006.250573>
- Criollo, L., Mena-Arciniega, C., & Xing, S. (2024). Classification, military applications, and opportunities of unmanned aerial vehicles. *Aviation*, 28(2), 115–127. <https://doi.org/10.3846/aviation.2024.21672>

- Deng, L., Hao, X., Mao, Z., Yan, Y., Sun, J., & Zhang, A. (2018). A subband radiometric calibration method for UAV-based multispectral remote sensing. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 11(8), 2869–2880. <https://doi.org/10.1109/JSTARS.2018.2842466>
- Eltayeb, A., Rahmat, M. F., Basri, M. A. M., Eltoum, M. A. M., & Mahmoud, M. S. (2022). Integral adaptive sliding mode control for quadcopter UAV under variable payload and disturbance. *IEEE Access*, 10, 94754–94764. <https://doi.org/10.1109/ACCESS.2022.3203058>
- Fu, Y., An, W., Su, X., & Song, B. (2025). Real-time wind estimation for fixed-wing UAVs. *Drones*, 9(8), 1–17. <https://doi.org/10.3390/drones9080563>
- Gaigalas, J., Perkauskas, L., Gricius, H., Kanapickas, T., & Kriščiūnas, A. (2025). A framework for autonomous UAV navigation based on monocular depth estimation. *Drones*, 9(4). <https://doi.org/10.3390/drones9040236>
- Gamagedara, K., Lee, T., & Snyder, M. (2024). Delayed Kalman filter for vision-based autonomous flight in ocean environments. *Control Engineering Practice*. <https://doi.org/10.1016/j.conengprac.2023.105791>
- Ghamari, M., Rangel, P., Mehrubeoglu, M., Tewolde, G. S., & Sherratt, R. S. (2022). Unmanned aerial vehicle communications for civil applications: A review. *IEEE Access*, 10, 102492–102531. <https://doi.org/10.1109/ACCESS.2022.3208571>
- Guo, Z., Wang, J., Zheng, X., Zhou, Y., & Zhang, J. (2025). A visual guidance and control method for autonomous landing of a quadrotor UAV on a small USV. *Drones*, 9(5). <https://doi.org/10.3390/drones9050364>
- Heryanto, M. A., & Kusumoputro, B. (2021). Attitude and altitude control of quadcopter maneuvers using neural network-based direct inverse control. *International Journal of Technology*, 12(4), 843–853. <https://doi.org/10.14716/ijtech.v12i4.3928>
- Heselden, J. R., Paparas, D., Stevenson, R. L., & Das, G. P. (2025). Enhanced STag marker system: Materials and methods for flexible robot localisation. *Machines*, 13(1), 1–36. <https://doi.org/10.3390/machines13010002>
- Jarraya, I., Al-Batati, A., Kadri, M. B., Abdelkader, M., Ammar, A., Boulila, W., & Koubaa, A. (2025). GNSS-denied unmanned aerial vehicle navigation: Analyzing computational complexity, sensor fusion, and localization methodologies. *Satellite Navigation*, 6(1). <https://doi.org/10.1186/s43020-025-00162-z>
- Jones, S., & Hauert, S. (2023). Frappe: Fast fiducial detection on low cost hardware. *Journal of Real-Time Image Processing*, 20(6). <https://doi.org/10.1007/s11554-023-01373-w>
- Jung, S. (2024). Precision landing of unmanned aerial vehicle under wind disturbance using derivative sliding mode nonlinear disturbance observer-based control method. *Aerospace*, 11(4). <https://doi.org/10.3390/aerospace11040265>
- Jurado-Rodriguez, D., Munoz-Salinas, R., Garrido-Jurado, S., & Medina-Carnicer, R. (2021). Design, detection, and tracking of customized fiducial markers. *IEEE Access*, 9, 140066–140078. <https://doi.org/10.1109/ACCESS.2021.3118049>
- Kalaitzakis, M., Cain, B., Carroll, S., Ambrosi, A., Whitehead, C., & Vitzilaios, N. (2021). Fiducial markers for pose estimation. *Journal of Intelligent & Robotic Systems*, 101(4). <https://doi.org/10.1007/s10846-020-01307-9>
- Laghari, A. A., Jumani, A. K., Laghari, R. A., & Nawaz, H. (2023). Unmanned aerial vehicles: A review. *Cognitive Robotics*, 3, 8–22. <https://doi.org/10.1016/j.cogr.2022.12.004>
- Lee, J. C., Chen, C. C., Shen, C. T., & Lai, Y. C. (2022). Landmark-based scale estimation and correction of visual inertial odometry for VTOL UAVs in a GPS-denied environment. *Sensors*, 22(24). <https://doi.org/10.3390/s22249654>
- Li, Y., Peng, B., He, L., Fan, K., & Tong, L. (2019). Road segmentation of unmanned aerial vehicle remote sensing images using adversarial network with multiscale context aggregation. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7), 2279–2287. <https://doi.org/10.1109/JSTARS.2019.2909478>

- Lopez-Sanchez, I., & Moreno-Valenzuela, J. (2023). PID control of quadrotor UAVs: A survey. *Annual Reviews in Control*, 56, 100900. <https://doi.org/10.1016/j.arcontrol.2023.100900>
- Mohd Daud, S. M. S., Mohd Yusof, M. Y. P., Heo, C. C., Khoo, L. S., Chainchel Singh, M. K., Mahmood, M. S., & Nawawi, H. (2022). Applications of drone in disaster management: A scoping review. *Science and Justice*, 62(1), 30–42. <https://doi.org/10.1016/j.scijus.2021.11.002>
- Mohsan, S. A. H., Othman, N. Q. H., Li, Y., Alsharif, M. H., & Khan, M. A. (2023). Unmanned aerial vehicles (UAVs): Practical aspects, applications, open challenges, security issues, and future trends. *Intelligent Service Robotics*, 16(1), 109–137. <https://doi.org/10.1007/s11370-022-00452-4>
- Murugan, D., Garg, A., & Singh, D. (2017). Development of an adaptive approach for precision agriculture monitoring with drone and satellite data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 10(12), 5322–5328. <https://doi.org/10.1109/JSTARS.2017.2746185>
- Nabavi, Y., Asadi, D., & Ahmadi, K. (2022). Automatic landing control of a multi-rotor UAV using a monocular camera. *Journal of Intelligent and Robotic Systems: Theory and Applications*, 105(3). <https://doi.org/10.1007/s10846-022-01655-8>
- Nenni, M. E., Di Pasquale, V., Miranda, S., & Riemma, S. (2020). Development of a drone-supported emergency medical service. *International Journal of Technology*, 11(4), 656–666. <https://doi.org/10.14716/ijtech.v11i4.3951>
- Park, Y., Park, C., Song, W., Lee, C., Kwon, J., Park, J., Noh, G., & Lee, D. (2024). Fiducial marker-based autonomous landing using image filter and Kalman filter. *International Journal of Aeronautical and Space Sciences*, 25(1), 190–199. <https://doi.org/10.1007/s42405-023-00635-y>
- Paucar, C., Morales, L., Pinto, K., Sánchez, M., Rodríguez, R., Gutierrez, M., & Palacios, L. (2018). Use of drones for surveillance and reconnaissance of military areas. *Smart Innovation, Systems and Technologies*, 94, 119–132. https://doi.org/10.1007/978-3-319-78605-6_10
- Qolbiyah, N. S., El Rummy, M. A., Purnama, I., & Faizal, F. (2024). Real-time peatland fire detection using infrared cameras and SVM algorithm. *ICSMeCh 2024 – 1st International Conference on Smart Mechatronics: Transformative Innovations in Smart Mechatronics: Bridging AI, Robotics, and IoT for a Sustainable Future*, 198–201. <https://doi.org/10.1109/ICSMeCh62936.2024.10812237>
- Rahman, M. A., Azad, S., Asyhari, A. T., Bhuiyan, M. Z. A., & Anwar, K. (2018). Collab-SAR: A collaborative avalanche search-and-rescue missions exploiting hostile alpine networks. *IEEE Access*, 6, 42094–42107. <https://doi.org/10.1109/ACCESS.2018.2848366>
- Santos, M. C., Bartlett, B., Schneider, V. E., Brádaigh, F. O., Blanck, B., Santos, P. C., Trslíc, P., Riordan, J., & Dooly, G. (2024). Cooperative unmanned aerial and surface vehicles for extended coverage in maritime environments. *IEEE Access*, 12, 9206–9219. <https://doi.org/10.1109/ACCESS.2024.3353046>
- Schempp, C., & Schulz, S. (2022). High-precision absolute pose sensing for parallel mechanisms. *Sensors*, 22(5), 1–33. <https://doi.org/10.3390/s22051995>
- Semerikov, S. O., Nechypurenko, P. P., Vakaliuk, T. A., Mintii, I. S., & Kolhatin, A. O. (2025). Vision-based autonomous UAV landing: A comprehensive review of technologies, techniques, and applications. *Journal of Intelligent and Robotic Systems: Theory and Applications*. <https://doi.org/10.1007/s10846-025-02314-4>
- Simon, D. (2006). *Optimal state estimation: Kalman, H infinity, and nonlinear approaches*. John Wiley & Sons. <https://doi.org/10.1002/0470045345>
- Sonugür, G. (2023). A review of quadrotor UAV: Control and SLAM methodologies ranging from conventional to innovative approaches. *Robotics and Autonomous Systems*, 161, 104342. <https://doi.org/10.1016/j.robot.2022.104342>

- Tsouros, D. C., Bibi, S., & Sarigiannidis, P. G. (2019). A review on UAV-based applications for precision agriculture. *Information (Switzerland)*, 10(11). <https://doi.org/10.3390/info10110349>
- Xin, L., Tang, Z., Gai, W., & Liu, H. (2022). Vision-based autonomous landing for the UAV: A review. *Aerospace*, 9(11). <https://doi.org/10.3390/aerospace9110634>
- Zhang, Y., Wu, Z., & Wei, T. (2024). Aerodynamic disturbance estimation in quadrotor landing on moving platform via noise reduction extended disturbance observer. *IEEE Sensors Journal*, 24(22), 37566–37574. <https://doi.org/10.1109/JSEN.2024.3472025>
- Zhou, X., Chen, L., Sun, C., Jia, W., Yi, N., & Sun, W. (2025). Highly accurate attitude estimation of unmanned aerial vehicle payloads using low-cost MEMS. *Micromachines*, 16(6), 1–18. <https://doi.org/10.3390/mi16060632>