RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohamed Khider – BISKRA

Faculté des Sciences et de la Technologie

Département de Génie Électrique



Arrêté N° : ...../../2026

Thèse de Doctorat

En vue de l'obtention du diplôme de docteur LMD en Automatique et Systèmes

Navigation visuelle d'un drone de type Quadrotor

Présentée et soutenue publiquement par : **Oumaima Gharsa**

Soutenue le 13/09/2026

Devant le jury composé de :

Président du Jury:	Hatem GHODBANE	Professeur	Université de Biskra
Directeur de thèse :	Mostefa Mohamed TOUBA	Professeur	Université de Biskra
Co-directeur de thèse:	Mohamed BOUMEHRAZ	Professeur	Université de Biskra
Examinateur :	Achour BETKA	Professeur	Université de Biskra
Examinateur :	Abdelmalik TALEB-AHMED	Professeur	Université de Valenciennes
Examinateur :	Yahia KOURD	Professeur	Université de Souk Ahras

Année universitaire 2026-2027

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

Ministry of Higher Education and Scientific Research

Mohamed Kheider University – BISKRA

Faculty of Science and Technology

Department of Electrical Engineering



Order No: ……../../2026

Doctoral Thesis

For the purpose of obtaining a Doctor of Science degree in Control Engineering and Systems

Visual navigation of a quadrotor drone

Presented and publicly defended by: **Oumaima Gharsa**

Defended on 09/13/2026

Members of the jury:

President:	Hatem GHODBANE	Professor	University of Biskra
Supervisor:	Mostefa Mohamed TOUBA	Professor	University of Biskra
Co-Supervisor:	Mohamed BOUMEHRAZ	Professor	University of Biskra
Examiner:	Achour BETKA	Professor	University of Biskra
Examiner:	Abdelmalik TALEB-AHMED	Professor	University of Valenciennes
Examiner:	Yahia KOURD	Professor	University of souk Ahras

Academic Year 2026-2027

Acknowledgements

All praise and gratitude are due to Allah, the Most Merciful and Most Compassionate, for His endless blessings, guidance, and strength. It is by His will that this work has been completed. I am deeply thankful for the patience, knowledge, and perseverance He has granted me throughout this journey. May all praise and glory be to Him alone.

I extend my deepest gratitude to my supervisor, Professor Mostefa Mohamed TOUBA, whose guidance and knowledge were invaluable throughout this research. He was a great teacher both in knowledge and in character. May Allah reward him abundantly.

Thanks also to my co-supervisor Professor Mohamed Boumehraz for his continuous support and valuable advice. He was a great teacher both in knowledge and in character. May Allah reward him abundantly.

And in this special occasion I would like to express my sincere appreciation and deepest gratitude to Professor Abdelmalik Taleb-Ahmed for his precious help and valuable advice on my thesis. May Allah reward him abundantly.

Special thanks to Professors Nacira Agram, who contributed to facilitating my path to success, and all committee members for their time and effort in reviewing this thesis.

Finally, I would like to thank my colleagues and friends who shared this journey with me, offering support, motivation, and camaraderie through both challenges and successes. With heartfelt appreciation and gratitude to all who have accompanied me on this path.

Dedication

First and foremost, I dedicate this work to myself, for the patience, perseverance, and strength that carried me throughout this journey.

To my dear parents, the pillars of my life, who have always been a source of love, security, and unwavering support. Today, from the depths of my heart, I say: Thank you, my beloved parents. You have been my greatest source of courage, inspiration, and strength.

To my beloved husband, for your unwavering love, patience, encouragement, and support throughout this journey. Your belief in me gave me the strength to overcome every challenge and persevere until the completion of this work.

To my beloved sister and brothers, who have been my greatest supporters and a constant source of strength. Your love, encouragement, and confidence in me have been reflected in every achievement. I am profoundly grateful to share this success with you.

To my family-in-law, for your kindness, understanding, and constant support. Your encouragement and goodwill have meant a great deal to me throughout this journey.

To everyone who knows me, cares for me, and wishes me well, and to all those whose kind words and support have left a lasting impact on my life, you are all cherished in my heart.

To my dear friends and classmates, thank you for every kind word, every moment of support, and every shared memory. Your companionship has made this journey more meaningful and enjoyable.

This achievement belongs, in one way or another, to all of you.

Contents

List of Figures vi
List of Tables ix
List of Acronyms x
Abstract xiii
Résumé xiv
Abstract in Arabic xvi
List of Publications xvii

1 Introduction **1**
1.1 Context and Motivation 1
1.2 Thesis Research Goal 2
1.3 Thesis Contribution 3
1.4 Thesis Structure 4

2 State of the Art in Vision-Based Control for Quadrotor UAVs **6**
2.1 Introduction 6
2.2 Trade-offs Between Autonomy and Reliability in UAV Operations 7
2.3 Autonomous Drone Navigation in Unknown Environments 8
2.3.1 Simultaneous Localization and Mapping 9
2.3.2 Path Planning and Trajectory Generation 10
2.3.3 Obstacle Detection and Collision Avoidance 11
2.3.4 Autonomous Landing for the UAV 11
2.4 Sensing and Perception for Quadrotor Autonomy 14
2.4.1 Sensing (Data Acquisition) 14
2.4.2 Perception (Data Interpretation) 15

2.5 Reactive Vision-Based Approaches . . . 15
2.5.1 Classical vision-based reactive navigation techniques . . . 16
2.5.2 Learning-based visual approaches . . . 17
2.6 Quadrotor control . . . 18
2.7 State estimation . . . 20
2.8 Vision-based control system . . . 21
2.9 Unsupervised and Weakly-Supervised Learning for UAV Vision . . . 22
2.10 Simulation . . . 24
2.11 Conclusion . . . 25

3 Vision-Based Autonomous Landing of Quadrotor UAVs 26
3.1 Introduction . . . 26
3.2 Image Features and Assumptions for 2D Vision-Based Landing System Requirements . . . 27
3.3 Vision-Based Landing Target Detection and Monitoring . . . 28
3.3.1 Static Landing Target Detection . . . 28
3.3.2 Dynamic Landing Target Detection and Monitoring . . . 29
3.4 Vision-based Quadrotor Control . . . 32
3.5 Simulation Results . . . 35
3.5.1 Landing on a Static Target . . . 35
3.5.2 Landing on a Moving Target . . . 38
3.6 Conclusion . . . 41

4 Vision-Based Autonomous Object Detection and Tracking in Quadrotor UAVs: Classical Methods and Experimental Validation 43
4.1 Introduction . . . 43
4.2 System Overview . . . 44
4.3 Vision-Based Detection and Tracking Pipeline . . . 46
4.3.1 Object Detection Algorithms . . . 46
4.3.2 Object Tracking Algorithms . . . 47
4.4 Vision-Based PID Control for Guidance . . . 50
4.5 Simulation and Experimental Results . . . 53
4.5.1 Simulation Results . . . 53

4.5.2 Experimental Results . . . 57
4.6 Conclusion . . . 62

5 An Advanced YOLO-Based Framework for Real-Time Autonomous Object Detection in UAVs 63
5.1 Introduction . . . 63
5.2 UAV Platform and Vision System Implementation . . . 64
5.2.1 The UAV Platform (DJI Tello) . . . 64
5.2.2 The Moving Target . . . 65
5.3 Software Setup and YOLOv8 Integration . . . 66
5.4 Target Detection Framework . . . 68
5.5 Implementation of the Autonomous Tracking System . . . 69
5.6 Flight Control Setup . . . 72
5.7 Experimental Results . . . 73
5.8 Conclusion . . . 78

6 Autonomous Obstacle Avoidance for UAVs in Unknown Environments 79
6.1 Introduction . . . 79
6.2 Proposed System Framework . . . 80
6.3 Obstacle Avoidance by Optical Flow Method . . . 81
6.4 Proposed Control System for Obstacle Avoidance . . . 84
6.5 Simulation Results . . . 88
6.6 Discussion . . . 99
6.7 conclusion . . . 100

7 Conclusions and Future Perspectives 101
7.1 Summary of Contributions . . . 101
7.2 Future Perspectives . . . 102

Bibliography 104

List of Figures

2.1 Trade-off between autonomy and reliability in UAV control strategies (Source: [1]). 7
2.2 Proposed overview of the system. 8
2.3 Systems for localization and mapping using visual data. 10
2.4 (Left): A quadcopter illustrating the propeller rotation directions. (Right): The six degrees of freedom (6-DOF) of our drone. 19
2.5 Classical moving-object detection methods typically use superpixels and color constancy to maintain object associations across time (left) [2]. Using deep learning, a 3D depth representation is inferred from a monocular image for obstacle avoidance, employing a hybrid direct–indirect approach (right) [3]. . 23
2.6 Examples of Quadrotor Simulation Environments. 25

3.1 Vision-based system in a virtual environment. 28
3.2 Extraction of the circular target using HSV-based color segmentation. 29
3.3 Visual representation of the Canny edge detection process. 29
3.4 (a) Raw RGB input frame; (b) Conversion to HSV color space; (c) Binary segmentation mask of the target; (d) Geometric representation of the target states in image coordinates. 30
3.5 Visual state definition relative to the image frame. 31
3.6 A proportional control loop. 33
3.7 A proportional-Integral-Derivative control loop. 34
3.8 Detected landing-zone contour with centroid coordinates (x_c, y_c). 35
3.9 The UAV's 3D trajectories across different scenarios during the search for the landing platform. 36

3.10 Successful detection of the yellow circular target in the Gazebo simulator. . . 37
3.11 The altitude of the UAV relative to the moving platform. 38
3.12 Autonomous UAV Landing on a Moving Target Following a Linear Path. . . 39
3.13 Representative UAV trajectories during tracking and autonomous landing on a mobile target following a partial square path. 39
3.14 Visual Sequence of the Autonomous UAV Target Following and Landing Experiment Conducted in the Gazebo Simulator. 40
3.15 Relative Distance Between the UAV and the Landing Platform During Autonomous Landing. 41

4.1 Proposed overview of the system. 44
4.2 Architecture of the vision and control system. 45
4.3 The system measurements are encoded in an observation vector comprising the ball's image-plane centroid (x_c, y_c) and its estimated radius r_c in the current frame. 48
4.4 Schematic representation of the PID feedback loop used for UAV control. . . 50
4.5 Simulated-environment object tracking model. 54
4.6 Deviation between estimated and actual velocity. 55
4.7 Tracking performance for targets following square trajectories. 56
4.8 Tracking performance for targets following circular trajectories. 56
4.9 Sequential snapshots of a UAV tracking a target moving along a straight-line trajectory: (**a**) takeoff from the ground, (**b**) approach phase, (**c**) initiation of target following, (**d**) continuous tracking during motion, and (**e**) successful mission completion demonstrating stable tracking. 58
4.10 Autonomous tracking of a target executing a circular maneuver. 59
4.11 UAV tracking a target executing complex, non-deterministic maneuvers. . . 61

5.1 UAV and Ball Object. (a) DJI Tello quadrotor equipped with an onboard camera; (b) spherical ball target. 66
5.2 General communication architecture between a Tello UAV and a PC. 67
5.3 Block diagram of the YOLOv8 architecture. 68
5.4 Coordinate system definition for ball tracking. 71
5.5 PID Controller for UAV. 72

5.6 Fundamental UAV Maneuvers. 73
5.7 Real-time autonomous ball detection and tracking using onboard vision. . . 74
5.8 UAV vertical (up/down) motion during ball tracking. 75
5.9 Tracking error along the $X-$, $Y-$, and $Z-$axes versus time. 77

6.1 Integrated Simulation Framework and Vision-Based Perception Pipeline. . . 81
6.2 Implementation of the Obstacle Detection and Avoidance (ODA) system. (a) No obstacle detected in the flight path. (b) Active obstacle detection with visual warning and vector analysis. 83
6.3 Block Diagram of the Integrated Control Strategy. 85
6.4 Simulation environment for autonomous navigation. 89
6.5 2D autonomous MAV flight path: Scenario 1. 90
6.6 3D autonomous flight trajectory: Scenario 1. 92
6.7 2D autonomous MAV flight path: Scenario 2. 94
6.8 3D autonomous flight trajectory: Scenario 2. 96
6.9 Sequential frames illustrating the MAV obstacle avoidance maneuver. . . . 98

List of Tables

5.1 Materials and computer specifications used in this work. 65
5.2 Materials used in this study. 67
5.3 A Comparative Analysis of YOLO Model Performance. 69
5.4 Comparison of YOLOv8-Based and YOLOv5-Based Methodologies for UAV Object Tracking. 70

List of Acronyms

UAV	Unmanned Aerial Vehicle
PID	Proportional-Integral-Derivative
EKF	Extended Kalman Filter
GPS	Global Positioning System
HSV	Hue, Saturation, Value
IMU	Inertial Measurement Unit
CNN	Convolutional Neural Network
MAV	Micro Aerial Vehicle
ROS	Robot Operating System
RGB	Red, Green, Blue
SDK	Software Development Kit
ROI	Region of Interest
SITL	Software in the Loop
FCU	Flight Control Unit
PC	Personal Computer
CPU	Central Processing Unit
GPU	Graphics Processing Unit
DRL	Deep Reinforcement Learning

OS	Operating System
KF	Kalman Filter
SSD	Single Shot Detector
mAP	mean Average Precision
EIS	Electronic Image Stabilization
SVM	Support Vector Machine
SLAM	Simultaneous Localization and Mapping
INS	Inertial Navigation System
MPC	Model Predictive Control
USV	Unmanned Surface Vehicle
AUV	Autonomous Underwater Vehicle
ILS	Instrument Landing System
DGPS	Differential Global Positioning System
SOTA	State of the Art
PnP	Perspective-n-Point
6-DoF	Six Degrees of Freedom
OF	Optical Flow
SIFT	Scale-Invariant Feature Transform
SURF	Speeded-Up Robust Features
LQR	Linear Quadratic Regulator
UKF	Unscented Kalman Filter
CW	Clockwise
CCW	Counterclockwise
VIO	Visual-Inertial Odometry
DNN	Deep Neural Network

IBVS Image-Based Visual Servoing

Abstract

This thesis presents a novel vision-based framework for Unmanned Aerial Vehicles (UAVs) operating in both indoor and outdoor environments, with a focus on autonomous navigation in previously unknown environments using the UAV's onboard camera for visual sensing, without relying on GPS. Visual processing and state estimation are performed on an external PC, which communicates with the UAV via Wi-Fi. The framework relies on an image-based visual approach that integrates HSV filtering and shape detection for robust target detection, while target states are estimated using an enhanced Extended Kalman Filter (EKF) for accurate and reliable state prediction. For navigation in complex environments, an optical flow-based method is adopted for obstacle avoidance, and a closed-loop Proportional-Integral-Derivative (PID) controller is designed to enhance flight stability, tracking accuracy, and overall performance. Furthermore, the thesis introduces a sensing and action abstraction framework to facilitate knowledge transfer across domains, particularly from simulation to real-world deployment. It also proposes novel approaches for predictive uncertainty estimation, enabling the safe, reliable, and accountable integration of vision-based perception with control systems in aerial robotics. The experimental results confirm that the communication delay and model performance are compatible with the requirements of real-time tracking tasks, with an object-tracking success rate of up to 90% under the best-case conditions. Finally, the modular structure of the proposed pipeline offers significant potential for future enhancement of its individual components and supports the realization of a fully onboard navigation system for real-world applications, thereby pushing the boundaries of autonomous quadrotor exploration.

Keywords: Quadrotor, computer vision, vision-based control, unmanned aerial vehicles, Visual Navigation, Moving object, tracking, Autonomous landing, PID controller, Extended Kalman Filter (EKF), Optical Flow.

Résumé

Cette thèse présente un nouveau cadre basé sur la vision pour les véhicules aériens sans pilote (UAV) opérant aussi bien en intérieur qu'en extérieur, en mettant l'accent sur la navigation autonome dans des environnements inconnus en utilisant la caméra embarquée du drone pour la perception visuelle, tandis que le traitement des images et l'estimation d'état sont réalisés sur un PC externe. Les commandes de contrôle sont ensuite transmises au drone via Wi-Fi, sans recours au GPS. Ce cadre s'appuie sur une approche visuelle par imagerie qui intègre un filtrage HSV et une détection de forme pour une détection robuste des cibles, tandis que les états des cibles sont estimés à l'aide d'un filtre de Kalman étendu (EKF) amélioré pour une prédiction précise et fiable de l'état. Pour la navigation dans des environnements complexes, une méthode basée sur le flux optique est adoptée pour l'évitement d'obstacles, et un contrôleur PID en boucle fermée est conçu pour améliorer la stabilité de vol, la précision de suivi et les performances globales. De plus, la thèse présente un cadre d'abstraction de la détection et de l'action afin de faciliter le transfert de connaissances entre les domaines, en particulier de la simulation au déploiement en conditions réelles. Elle propose également des approches novatrices pour l'estimation prédictive de l'incertitude, permettant l'intégration sûre, fiable et responsable de la perception visuelle avec les systèmes de contrôle en robotique aérienne. Les résultats expérimentaux confirment que le délai de communication et les performances du modèle sont compatibles avec les exigences des tâches de suivi en temps réel, avec un taux de réussite du suivi d'objets pouvant atteindre 90% dans les meilleures conditions. Enfin, la structure modulaire du pipeline proposé offre un potentiel significatif pour l'amélioration future de ses composants individuels et permet la réalisation d'un système de navigation entièrement embarqué pour des applications concrètes, repoussant ainsi les limites de l'exploration autonome par quadcopter.

Mots-clés: Quadcopter,vision par ordinateur, commande basée sur la vision, drones, navi-

gation visuelle, objet en mouvement, suivi, atterrissage autonome, régulateur PID, filtre de Kalman étendu (EKF), flux optique.

الملخص

تقدم هذه الأطروحة إطار عمل مبتكرًا قائمًا على الرؤية للطائرات بدون طيار (UAVs) التي تعمل في البيئات الداخلية والخارجية على حد سواء، مع التركيز على الملاحة الذاتية في بيئات غير معروفة، بالاعتماد على الاستشعار البصري بواسطة الكاميرا المثبتة على متن الطائرة، في حين يتم تنفيذ معالجة الصور وتقدير الحالة على حاسوب خارجي. ويتم إرسال أوامر التحكم إلى الطائرة عبر اتصال Wi-Fi ، دون الاعتماد على نظام GPS. يعتمد الإطار على نهج بصري قائم على الصور يدمج تصفية HSV وكشف الشكل من أجل الكشف القوي عن الأهداف، بينما يتم تقدير حالات الأهداف باستخدام مرشح كالمان الموسع المحسّن (EKF) من أجل التنبؤ الدقيق والموثوق بالحالة. للملاحة في البيئات المعقدة، يتم اعتماد طريقة قائمة على التدفق البصري لتجنب العوائق، ويتم تصميم وحدة تحكم ذات حلقة مغلقة من نوع التناسب والتكامل والتفاضل (PID) لتعزيز استقرار الطيران ودقة التتبع والأداء العام. علاوة على ذلك، تقدم الأطروحة إطار عمل لتجريد الاستشعار والعمل لتسهيل نقل المعرفة عبر المجالات، لا سيما من المحاكاة إلى النشر في العالم الحقيقي. كما تقترح طرقًا جديدة لتقدير عدم اليقين التنبئي، مما يتيح التكامل الآمن والموثوق والمسؤول للإدراك القائم على الرؤية مع أنظمة التحكم في الروبوتات الجوية. تؤكد النتائج التجريبية أن تأخير الاتصال وأداء النموذج متوافقان مع متطلبات مهام التتبع في الوقت الفعلي، مع معدل نجاح في تتبع الأجسام يصل إلى 90٪ في أفضل الظروف. أخيرًا، توفر البنية المعيارية للخط المقترح إمكانات كبيرة للتحسين المستقبلي لمكوناته الفردية وتدعم تحقيق نظام ملاحة متكامل على متن المركبة للتطبيقات الواقعية، مما يوسع حدود استكشاف المروحيات الرباعية ذاتية القيادة.

الكلمات المفتاحية: المروحية الرباعية المراوح، الرؤية الحاسوبية، التحكم القائم على الرؤية، المركبات الجوية غير المأهولة، الملاحة البصرية، الأجسام المتحركة، التتبع، الهبوط الذاتي، وحدة التحكم PID، مرشح كالمان الموسع (EKF)، التدفق البصري.

List of Publications

- O. Gharsa, M. M. Touba, M. Boumehraz, N. Abderrahmani, I. B. Ghiloubi, "MAV Obstacle Avoidance System based on optical flow Method in unknown environments," Electrotehnica, Electronica, Automatica (EEA), ISSN 1582-5175. (Under Review)
- O. Gharsa, M. M. Touba, M. Boumehraz, and N. Agram, "Autonomous vision-based object detection and tracking system for quadrotor unmanned aerial vehicles," Sensors, vol. 25, no. 20, p. 6403, 2025. DOI: https://doi.org/10.3390/s25206403
- O. Gharsa, M. M. Touba, M. Boumehraz, N. Abderrahmani, I. B. Ghiloubi, and A. H.Drid, "Yolov8-based autonomous ball detection and tracking for rotorcraft uavs using onboard vision sensors," Engineering Proceedings, vol. 118, no. 1, p. 8, 2025. DOI: https://doi.org/10.3390/ECSA-12-26570
- O. Gharsa, M. M. Touba, M. Boumehraz, N. Abderrahman, S. Bellili, and A. Titaouine, "Autonomous landing system for a quadrotor using a vision-based approach," in 2024 8th International Conference on Image and Signal Processing and their Applications (ISPA). IEEE, 2024, pp. 1–5.
- O. Gharsa, M. M. Touba, M. Boumehraz, N. Abderrahmani and I. B. Ghiloubi, "Autonomous Landing of a Quadrotor on a Moving Target Using a Vision-Based Approach," 2025 5th International Conference on Applied Automation and Industrial Diagnostics (ICAAID), Ghardaia, Algeria, 2025, pp. 1-6.

Chapter 1

Introduction

1.1 Context and Motivation

In recent years, significant advancements have been achieved in unmanned aerial vehicle (UAV) technology [4], Automated technologies are increasingly important for simplifying UAV applications in modern society, especially in situations that are hazardous, complex, or impractical to manage manually. In particular, UAVs are designed to operate without an onboard pilot and are deployed in contexts where direct human intervention is not feasible. Although quadrotors typically have lower flight efficiency than fixed-wing aircraft, today's low-cost platforms still offer sufficient endurance and payload capacity for many applications. These quadrotors offer valuable capabilities, including safe and rapid data collection and access to remote or hard to reach areas, creating new opportunities for indoor and outdoor missions such as post disaster emergency response, search-and-rescue operations, commercial and aerospace applications [5]. Across these domains, a common goal is to employ highly autonomous aircraft that can reduce, or even eliminate, the need for human involvement. High-performance autonomous navigation capabilities are essential for successfully carrying out these missions. In general, UAV navigation can be described as the process of planning a safe and feasible path that guides the vehicle from a starting location to a target destination. Traditional navigation approaches include satellite-based navigation, inertial navigation, radar-based navigation, and sonar-based navigation, among others. However, cameras are powerful vision sensors widely used in robotics. By capturing visual information such as color, shape, and texture they provide a machine analogue of biological vision. Computer vision then enables UAVs to interpret these images to measure

and detect elements of the environment without physical contact. Even so, building fully autonomous quadrotors that can match or even surpass the agility of birds or expert human pilots in unknown, cluttered environments remains extremely challenging and largely unresolved. During UAV flight in unstructured settings, perception systems must cope with changing illumination, perceptual aliasing (i.e., different places appearing highly similar), and motion blur issues that are well known yet still difficult to solve with current SOTA methods [6, 7, 8]. Moreover, uncertainty stemming from imperfect models of the environment, sensors, and actuators together with the need for closed-loop safety and a low latency perception–action loop in dynamic environments adds substantial complexity to high-speed navigation [9]. Limitations in autonomous drone navigation have motivated our research toward a vision-based approach. While vision-based navigation offers clear advantages over traditional methods, it introduces practical challenges. In particular, many high performing techniques especially deep-learning approaches [10], are computationally expensive, limiting their deployment on low-cost UAVs relying solely on an onboard RGB camera. Although lightweight alternatives have been proposed, achieving an optimal trade-off between real-time efficiency and detection accuracy in unknown environments remains difficult. These challenges underscore the need for vision-based navigation methods that are both robust and computationally efficient.

1.2 Thesis Research Goal

The principal objectives of this thesis are as follows:

- To critically review the state of the art in UAV visual navigation, emphasizing the technical challenges that limit robustness and real-time performance.
- To investigate the impact of UAV specific image characteristics such as motion blur, vibration, and rapid changes in lighting conditions on the accuracy and latency of real-time vision processing.
- To develop a lightweight, real-time navigation framework for low-cost UAVs that balances autonomous exploration with safe obstacle avoidance in unstructured environments.

- To develop a computationally efficient AI-based object detection framework that allows low-cost UAVs to explore unknown environments in real time while detecting and tracking targets.
- To assess the efficacy of optical flow algorithms in providing accurate state estimation and reliable collision detection for quadrotors in unknown environments.

1.3 Thesis Contribution

More precisely, this research addresses the challenge of autonomous navigation in unstructured environments by developing real-time trajectory generation frameworks that leverage visual perception to ensure safe, collision-free paths. State estimation in quadrotors often depends on keeping specific visual targets in view during motion. To mitigate the limitations this places on navigating large environments, this work proposes efficient vision-based control algorithms. These methods are applied to moving target tracking, navigation, and autonomous landing. The contributions of this thesis are as follows:

- **Autonomous Landing via Visual Servoing:** This research contributes a complete visual navigation pipeline for autonomous landing. Validated through ROS and Gazebo simulations, the proposed method integrates a visual detection algorithm (targeting specific geometric shapes, such as a yellow circular landing zone) with a control scheme that allows the UAV to independently search for, intercept, and descend onto a specific landing platform [11]. This approach effectively relaxes the need for manual intervention or external positioning systems during the critical landing phase.
- **Vision-Based Navigation in GPS-Denied Environments:** This study contributes a robust object detection and tracking framework designed specifically for GPS-denied environments. The method focuses on guiding an autonomous quadrotor toward a specific target (e.g., a yellow sphere) using visual data alone [12]. Key contributions include the system's ability to maintain high tracking accuracy under varying lighting conditions and its power-efficient design, which ensures enhanced real-time operational efficacy for continuous target following.

- **Real-Time Autonomous Tracking using YOLOv8:** This work introduces an autonomous tracking system that leverages the YOLOv8 deep learning algorithm combined with a re-detection mechanism. The contribution lies in enabling a rotorcraft UAV to effectively detect and track moving objects (specifically dynamic targets like balls) using solely onboard vision sensors. The system employs a PID controller to regulate the drone's motion based on the target's position within the image frame, demonstrating robustness against complex scenes, scale changes, and occlusions without relying on external infrastructure like GPS [13].
- **Development of a Vision-Based Collision Avoidance Strategy:** A key contribution is the implementation of a perception-and-control loop that leverages Optical Flow for depth abstraction and obstacle detection. By coupling flow divergence measurements with a PID control scheme, the system ensures smooth, reactive evasion maneuvers. The robustness of this method was demonstrated in complex, unstructured Gazebo scenarios, proving the quadrotor's ability to safely navigate environments with high obstacle density.

1.4 Thesis Structure

This section describes the organizational structure of the thesis and presents a short overview of each chapter. The remainder of this document is organized as follows:

Chapter 2 provides a comprehensive review of the SOTA in autonomous quadrotor navigation, establishing the theoretical background for the proposed methods. It details the fundamental concepts of visual perception, ranging from classical object detection techniques to advanced deep learning algorithms such as YOLO. The chapter also addresses optical flow methods for motion estimation and reviews essential flight control strategies. Finally, it presents an overview of the simulation tools and environments commonly used to test and validate UAV performance.

Chapter 3 presents the first contribution. It proposes a complete vision-based framework that allows the quadrotor to identify a designated landing zone and execute a precision descent. The chapter details the image processing algorithms we used for target recognition

(Specifically, we perform geometric shape detection using HSV-based segmentation) and describes the control strategy that aligns the UAV with the landing pad. The performance of this system is validated through simulations in the ROS/Gazebo environment, demonstrating the ability of the drone to land safely in static or dynamic platforms without external positioning systems.

Chapter 4 describes the second contribution, a vision-based navigation system designed for dynamic object tracking, specifically a ball, within GPS-denied environments. It details the implementation of an Extended Kalman Filter (EKF) to estimate the target's state (position and velocity) from noisy visual measurements. This chapter explains how the EKF predicts the ball's trajectory, improving performance during fast motion and partial occlusions. It also describes how this estimate is integrated with the PID flight controller to enable smooth and accurate real-time autonomous following using a DJI Tello drone.

Chapter 5 highlights the third contribution, which presents the use of advanced object detection methods (YOLOv8) to address the task of tracking a moving target using monocular vision. The experimental results demonstrate that the proposed system can maintain stable tracking performance even under challenging conditions, including target scale variations and rapid motion.

Chapter 6 details the fourth contribution, which addresses the challenge of navigating unstructured and complex environments. It introduces a reactive obstacle avoidance approach based on Optical Flow. The chapter details the implementation of the perception-control loop in the Gazebo simulator, demonstrating how the system detects obstacles and adjusts trajectory to prevent collision. [Under Review]

Finally,

chapter 7 concludes the thesis with a summary of the main contributions and synthesizes the principal findings of this work. Presents a critical discussion of the experimental results, highlighting the successes of the proposed vision-based systems. In addition, we identify potential directions for future research, outlining several strategies to further improve the developed methodologies and enhance their robustness in diverse environments.

Chapter 2

State of the Art in Vision-Based Control for Quadrotor UAVs

2.1 Introduction

This chapter lays the groundwork for the thesis by presenting essential background. Autonomous drones have surged in popularity due to their diverse applications across many domains. Early research focused on enabling operation in environments where human presence is impossible, extremely risky, or otherwise impractical. This evolution is driven by progress across the autonomy stack especially vision-based perception, real time motion planning, and robust control enabling deployments in complex and GPS-denied environments. Recent reviews highlight that autonomy performance is increasingly defined not only by accurate localization and stable control, but also by safe navigation among obstacles, resilience to uncertainty, and the ability to operate reliably in real-world conditions [14].

Autonomy, in particular, has attracted significant attention and lies at the heart of many emerging innovations. Enhanced capabilities for safe, autonomous flight in unknown and cluttered environments combined with progress in connected and distributed edge intelligence and the rapid growth of AI are actively transforming the UAV sector. This chapter establishes the foundational knowledge required to understand the research presented in this thesis. State of the art approaches to agile autonomous quadrotor navigation typically decompose the problem into a sequence of interrelated subtasks, including mapping, path planning, and obstacle avoidance [15]. Decomposing autonomous quadrotor navigation into sequential subtasks is appealing from an engineering perspective, as it allows independent

progress on each module and yields an overall system that is easier to interpret and debug.

Finally, the chapter concludes by

2.2 Trade-offs Between Autonomy and Reliability in UAV Operations

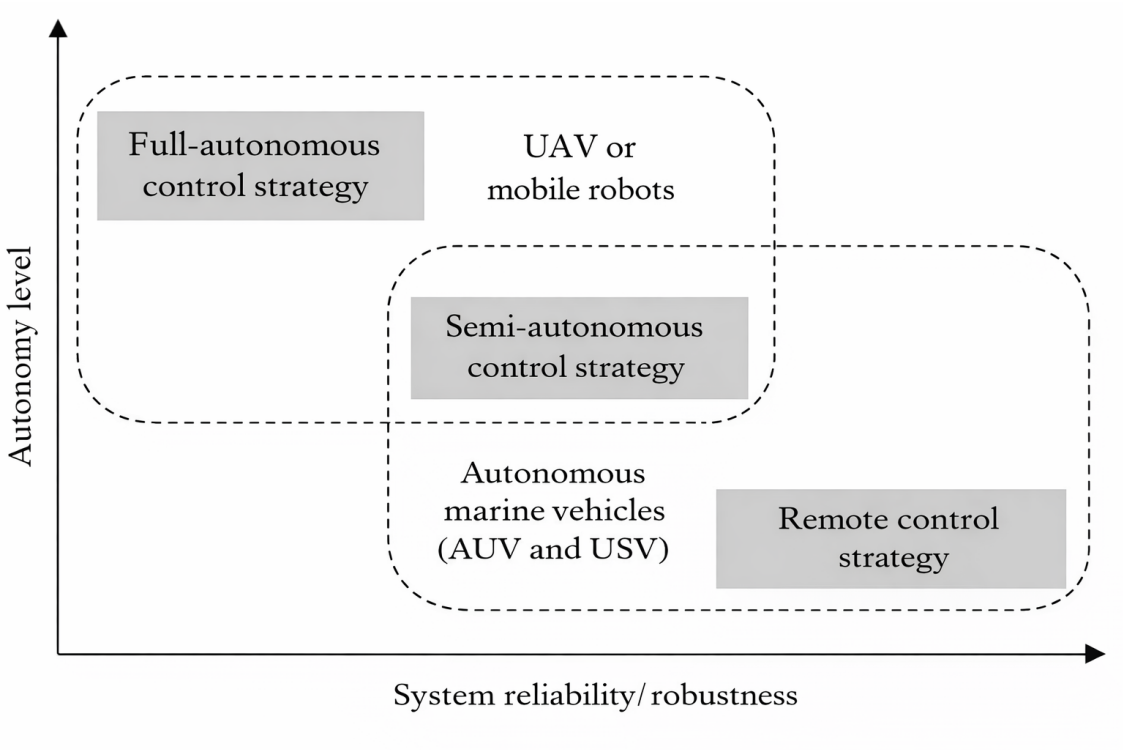


Figure 2.1: Trade-off between autonomy and reliability in UAV control strategies (Source: [1]).

autonomy level and system reliability (robustness) are evaluated across three control strategies, As shown in Figure 2.1. Owing to continuous human involvement, remote control offers the lowest degree of autonomy but typically the highest operational reliability. In remote control operation, a human operator directly supervises perception, decision making, and corrective actions. This human in the loop structure results in the lowest autonomy, but it generally provides the highest operational reliability, since the operator can compensate

for sensing errors, unexpected disturbances, and unmodeled environmental changes in real time [16]. In contrast, fully autonomous control aims to execute the mission with minimal or no human intervention, leading to the highest autonomy level. However, this increased autonomy often comes with a reduction in reliability and robustness, because the system must handle uncertainty independently. Performance becomes strongly dependent on the accuracy of onboard perception, state estimation, and prediction, as well as the planner's ability to generate safe trajectories under dynamic constraints. Any weakness in these modules such as perception failures due to poor lighting, occlusions, motion blur, communication loss, or incorrect environment modeling can propagate through the autonomy stack and degrade overall system safety and mission success.

2.3 Autonomous Drone Navigation in Unknown Environments

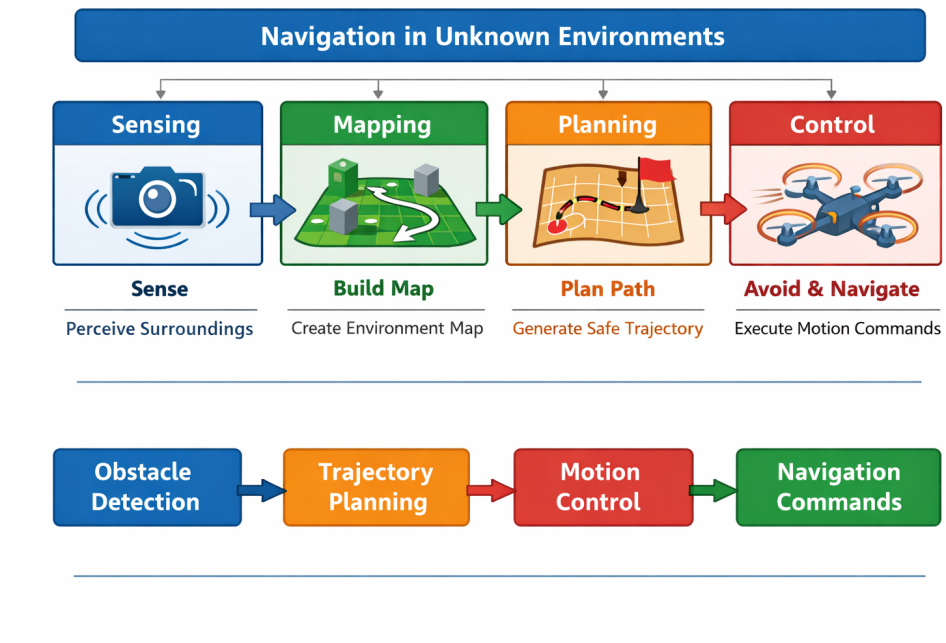


Figure 2.2: Proposed overview of the system.

Many approaches to navigation in previously unknown environments decompose the task into a sequence of subtasks. Although the modular architecture shown in Figure 2.2, is attractive from an engineering standpoint and improves interpretability, it can overlook the coupling between modules, allowing errors to cascade and accumulate across the pipeline.

2.3.1 Simultaneous Localization and Mapping

Simultaneous Localization and Mapping (SLAM) refers to the capability of a moving platform to estimate its position and orientation while simultaneously building a map of the surrounding environment using observations of environmental features gathered by onboard sensors. Since the seminal work of Smith et al. [4], SLAM has developed into a foundational technique in mobile robotics, with numerous practical implementations demonstrated in the literature. In UAV applications, localization and mapping are typically performed concurrently to enable reliable autonomous navigation, particularly in unknown or GPS-denied environments [17]. In this context, SLAM allows a UAV to continuously infer its pose and incrementally construct a representation of the environment from onboard sensory measurements, most commonly vision and inertial data, forming the basis for downstream tasks such as obstacle avoidance and trajectory planning [18, 19].

- **Visual Localization and Mapping**

  Depending on the operating environment and the amount of prior information available, visual localization and mapping systems can be broadly categorized into three classes: mapless, map-based, and map-building approaches [20]. Figure 2.3 illustrates this taxonomy and summarizes the main approaches within each category of visual localization and mapping [21].

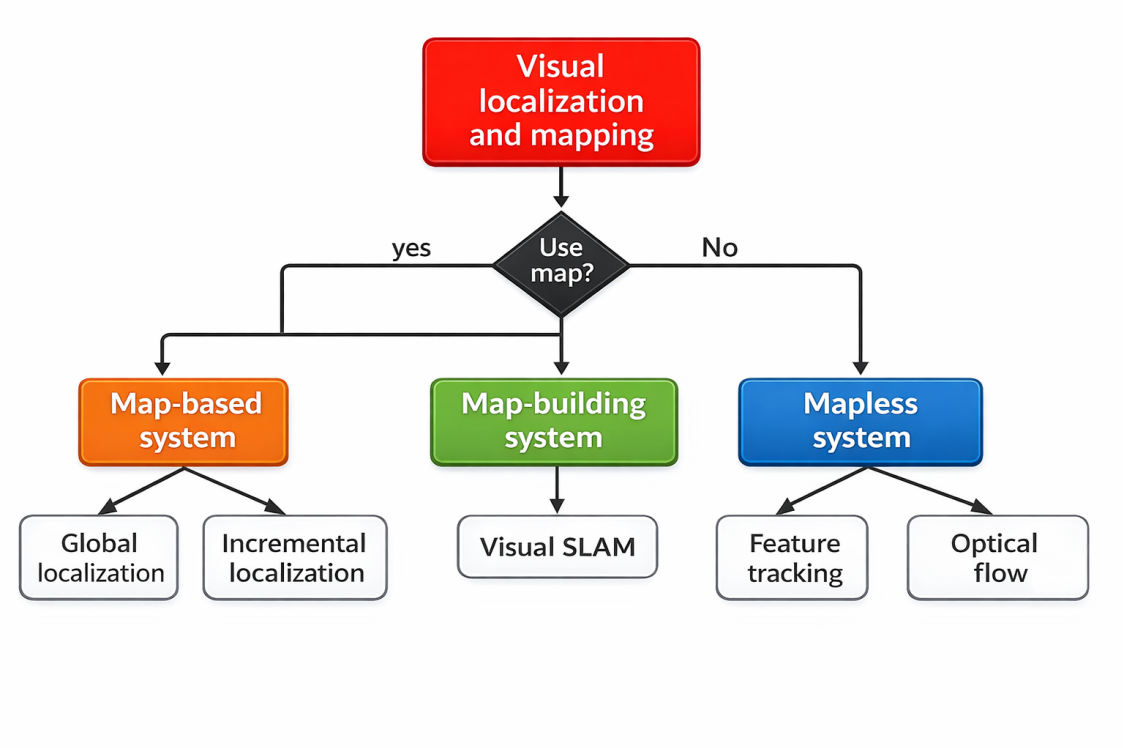


Figure 2.3: Systems for localization and mapping using visual data.

2.3.2 Path Planning and Trajectory Generation

Path planning is frequently treated as a waypoint following problem, requiring the UAV to correct its pose toward each waypoint in sequence [22]. Yet, most methods focus primarily on navigation, with limited attention to smooth, dynamically feasible trajectory generation [23]. This modular approach offers a key benefit: it reduces the number of design parameters while still producing paths that are short, safe, and dynamically feasible. As a result, development is faster and the resulting algorithms are often more lightweight and efficient. However, a major limitation especially in cooperative planning is that these planners depend on underlying system models. In practice, such models are only approximations of the real UAV dynamics and often require simplifying assumptions, which can be restrictive when the true system behavior is complex and highly nonlinear [4].

2.3.3 Obstacle Detection and Collision Avoidance

Obstacle detection and collision avoidance enable a UAV to operate safely by identifying obstacles in its surroundings and adapting its motion to prevent collisions. Obstacles may be physical objects sensed onboard (e.g., trees, buildings, power lines) or abstract constraints specified externally, such as no fly zones or geofenced regions [24]. Dynamic obstacle avoidance for quadrotors typically couples online perception of moving agents (detection and tracking or prediction) with safety-aware planning or control. Early vision-guided MAV systems demonstrated fully onboard stereo-based obstacle avoidance and autonomous maneuvers in unknown, dynamic environments, motivating tight perception control loops for real-time collision avoidance [25]. More recent pipelines explicitly estimate obstacle motion states and uncertainty: [26] estimate moving obstacles position, velocity, and size from depth imagery and enforce probabilistic safety using a chance constrained MPC that accounts for both sensing and state estimation uncertainty. Complementary vision first approaches detect dynamic obstacles with lightweight object detectors and use Kalman filtering to track or estimate motion, then integrate predicted obstacle trajectories into trajectory generation (e.g., spline-based search and optimization) to improve safety in real time [27]. On the control side, safety guarantees in dynamic settings are often expressed through constraint based formulations such as collision cone control barrier functions solved via real-time QPs [28]. Finally, uncertainty aware collision avoidance has expanded beyond deterministic replanning, including chance constrained formulations that incorporate obstacle velocity uncertainty [29] and sampling-based MPC variants with probabilistic collision checking for reliable navigation in uncertain dynamic environments [30].

2.3.4 Autonomous Landing for the UAV

Recently, there has been an increasing number of studies on autonomous landing for UAVs, and many surveys have been conducted on related issues such as general target detection, UAV-specific target detection, and end-to-end landing approaches [31]. Reliable autonomous landing systems are still a vital research area and have attracted increasing attention due to the fast growth of unmanned aerial vehicles. The key issue in developing an efficient landing system is how to estimate the aircraft's position with respect to the runway or landing area. Although most civilian airports have instrument landing systems

(ILS), the automatic landing system for UAVs typically employs high-precision differential GPS (DGPS) and tactical-grade inertial navigation systems (INS) [32]. However, such systems are expensive and may involve comprehensive airport surveys, especially for smaller recreational airports and low-cost UAV systems. Furthermore, GPS-based navigation can be interfered with by both passive and active jamming, which may cause signal disruption [33]. Autonomous UAV landing must account for both the surrounding environment and the vehicle's own state in order to detect the landing target and estimate the relative pose in real time, even under unstable platforms and challenging landing conditions. In this process, the large volume of sensory data, the computational complexity, and the stringent requirements for low latency, robustness, and stability make reliable autonomous landing particularly difficult.

Building on prior work, cameras offer a low-cost and dependable sensing solution that can be fused with inertial measurement unit (IMU) data to estimate the vehicle's motion relative to its environment [5, 34]. Traditionally, object detection on resource-limited UAV platforms has relied on classical computer vision methods based on handcrafted features. These approaches often require carefully designed descriptors or geometric primitives tailored to specific operating conditions [35]. As a result, their performance and portability are limited: features that work well in one scenario frequently generalize poorly to others. This motivates the adoption of more robust techniques that can automatically learn discriminative features from data rather than depending on manual feature engineering [31]. After reviewing existing work, we propose an updated vision-based taxonomy for UAV autonomous landing, distinguishing between static scene and dynamic scene scenarios.

- **Autonomous Landing of UAVs in Static Scenes**

  Vision-based methods for autonomous UAV landing have been studied extensively. Many studies describe vision-guided landing as a three stage process: (i) target detection and tracking, where visual features are extracted and matched to estimate the relative pose; (ii) flight guidance, in which the vision derived error is used to correct the UAV's position and attitude; and (iii) high-rate touchdown refinement to achieve a precise and stable landing [31]. We further categorize static-scene landing into two primary approaches: **cooperative target-based** and **natural scenario based** landing. In cooperative target methods, the UAV detects artificial landmarks—such as "T" or "H" symbols [36], circular [11, 37] or rectangular shapes [38], composite patterns,

or AprilTag like fiducials and estimates its relative pose from the marker's geometric properties [39]. These cooperative target systems typically follow two main detection lines: classical feature-based pipelines (e.g., edge/shape extraction, Hough transforms, invariant moments) [40] and learning-based approaches [41]. The latter are driven by the limited robustness and weak transferability of handcrafted features, spanning traditional classifiers (e.g., sliding window methods with SVM or AdaBoost) as well as modern deep-learning detectors. A persistent challenge for learning-based methods is achieving reliable performance under real flight degradations such as motion blur often mitigated by integrating image enhancement or deblurring with detection [42]. For natural scenario based landing, the focus is on cases where deploying artificial markers is not feasible (e.g., search and rescue missions). In such settings, UAVs rely on the natural appearance of the environment and its 3D geometric structure to localize and select a landing site [43]. Two common directions are scene matching approaches, which register onboard images to stored references and often fuse visual cues with inertial sensing (e.g., Kalman filtering) for navigation [44, 45], and near-field 3D reconstruction/SLAM-based methods, where the UAV is first guided close to the target using GPS/INS and then employs visual SLAM typically fused with IMU data and sometimes enhanced with learning-based depth estimation for precise local pose estimation and mapping [46, 47]. For example, [48] highlights that combining SLAM with depth sensing and 3D modeling can improve terrain analysis and enable more reliable landing site selection in static environments.

- **Autonomous Landing of UAVs in Dynamic Scenes** Dynamic-scene landing refers to landing on a moving platform and is considerably more challenging than static landing, as the UAV must track the platform's motion while preserving accurate navigation and stable control [34]. The literature also highlights that advances in dynamic landing are an essential step toward enabling even more demanding landing scenarios [43]. Existing approaches can be broadly grouped into four main directions: **external-aided guidance**, **vision-based navigation**, **multi-sensor fusion**, and **learning-based control**. Since robustly detecting and tracking the full platform structure remains difficult, many methods equip the platform with distinctive cooperative markers and estimate the UAV's relative pose from these visual targets [31].

Many state of the art works further categorize dynamic scene landing according to the type of moving carrier [49], including **vehicle-based** and **ship-based** platforms [50]. For vehicle-based platforms, the motion is often approximated as largely planar (2D) compared with maritime scenarios. Typical solutions rely on cooperative visual markers (e.g., circular patterns or AprilTags) [51] to enable vision-based pose estimation, and then perform closed-loop landing using controllers such as PID or MPC [52, 53], often combined with Kalman filtering for state estimation and measurement smoothing. These approaches also face practical limitations, such as wireless image transmission latency and noise, which can reduce reliability during high-speed landings [54]. In contrast, shipboard landing is generally more difficult because the deck can exhibit full 3D motion, requiring highly accurate real-time 6-DoF relative pose estimation [55]. The maritime setting further challenges perception due to factors like sea spray, mist, and adverse weather, which can degrade image quality. Reported solutions include using stereo or infrared sensing for low-light or harsh conditions, enhancing cooperative targets (e.g., heated "T" markers) [56], and combining vision-based detection with filtering and PnP-style pose estimation, while noting common failure cases such as false detections caused by reflections [57, 58].

2.4 Sensing and Perception for Quadrotor Autonomy

Autonomous drone navigation based on vision introduces fundamental challenges in both perception and control. Although alternative sensing modalities such as LiDAR or event-based cameras can mitigate some of these issues [59, 60], they are often too heavy or too costly for small quadrotors. From a perception standpoint, motion blur, difficult illumination, and aliasing can significantly degrade visual state estimation and lead to severe drift [6, 61, 62]. In addition, many SOTA estimation approaches implicitly assume a largely static environment, and their performance can deteriorate when the scene contains dynamic changes.

2.4.1 Sensing (Data Acquisition)

Quadrotor autonomy relies on onboard sensors to capture information about the environment and the vehicle state. Common sensing modalities include:

- **RGB cameras:** Provide high-resolution 2D visual information for detection, tracking, and mapping.
- **Depth sensors (stereo/RGB-D/ToF):** Measure scene geometry to support obstacle detection and range estimation.
- **LiDAR:** Produces accurate 3D point clouds for mapping and navigation, often at the cost of added weight and power.
- **Thermal cameras:** Capture heat signatures, useful in low-light conditions and applications such as search-and-rescue or inspection.

2.4.2 Perception (Data Interpretation)

Perception transforms raw sensor data into representations that support decision-making and control, including [63]:

- **Low-level vision processing:** Feature extraction, filtering, and optical flow to enhance image quality and motion cues.
- **Object detection and recognition:** Identifying relevant objects (e.g., people, vehicles, obstacles) for tracking and avoidance.
- **Localization:** Estimating the UAV pose (position and orientation), typically via visual–inertial odometry and/or SLAM.
- **Scene understanding:** Interpreting the structure and semantics of the environment (e.g., traversable free space, dynamic agents).
- **Deep learning:** CNN-based models (and related architectures) for robust perception tasks such as detection, segmentation, and depth prediction.

2.5 Reactive Vision-Based Approaches

Vision-based navigation has emerged as a central direction for UAV autonomy, driven largely by rapid advances in computer vision. Visual sensors offer several practical advantages for autonomous flight: they provide rich environmental information, remain effective

in dynamic scenes, and are typically compact, lightweight, and low-cost [21]. In parallel, the foundations of reactive navigation can be traced back to Brooks' influential work in the mid-1980s. In his 1986 paper, Brooks challenged the need for explicit world models and representations in autonomous robots by proposing a robust layered control architecture [64]. He later expanded this idea into a behavior-based framework in which complex robot actions emerge from the combination of simple stimulus–response reflexes [65]. Together, these developments motivate modern reactive, vision-based UAV navigation approaches that aim to map visual observations directly to control actions without relying on detailed global maps.

2.5.1 Classical vision-based reactive navigation techniques

The two primary classical techniques for vision-based reactive navigation are feature tracking [66] and optical flow (OF) methods [67].

One of the first feature-tracking methods detects objects by extracting visual features from images and matching them across consecutive frames. Tracking these features enables the estimation of both the robot's ego-motion and the motion of surrounding obstacles, providing informative descriptors that support tasks such as detection, matching, and classification. A vision-based reactive approach is presented in [44], in which the drone must detect the landing pad and align itself with it during descent. Two widely used methods for detecting salient image points and extracting features are the Scale-Invariant Feature Transform (SIFT) [68] and Speeded-Up Robust Features (SURF) [69]. Both provide descriptors that are robust to changes in scale, rotation, and, to a large extent, illumination and viewpoint, making them suitable for feature tracking in navigation and localization tasks [70]. Notably, SIFT underpins the region-based expansion-rate approaches proposed in [71] and [72]. Objects or obstacles are identified by matching feature points across successive images. The change in their apparent size, especially the size ratio between frames, is then used to estimate distance, enabling object tracking and the planning of avoidance maneuvers. In [11], the authors propose a vision-based autonomous landing system for a quadrotor UAV designed for scenarios where GPS is unavailable. The method targets landing on a predetermined pad and is evaluated in a Gazebo-based simulation environment, which is used to validate the approach and support safe parameter tuning. The reported simulation results indicate robust and precise landing performance. A closely related approach is presented in [52], which addresses autonomous UAV landing on a moving predefined platform.

on the other hand OF–based navigation, inspired by insect flight behavior [73], estimates scene motion from the robot's camera stream. In [74], OF values (computed with the Horn–Schunck method) feed a Takagi–Sugeno fuzzy controller for obstacle avoidance, while a second fuzzy controller handles goal seeking and steers the robot toward the destination; this works because closer obstacles produce higher optical flow than distant ones. Recent work has explored using optical flow (OF) not only as a handcrafted navigation cue but also as a learned signal: for example, [75] reports a simulation study that evaluates a "balance" strategy while estimating OF with a convolutional neural network (CNN). In a closely related direction, [76] learns obstacle avoidance directly from optical flow with one camera. However, OF-based control remains challenging due to the difficulty of obtaining accurate flow at high speeds, sensitivity to noise, and the need for robustness in complex environments.

2.5.2 Learning-based visual approaches

For many years, convolutional neural networks (CNNs) have served as the standard backbone for object detection models [77]. Inspired by the human visual system, CNNs extract hierarchical visual features by applying learnable convolutional filters to an image, producing feature maps that capture its most salient information. This design enables strong recognition capability while keeping computational requirements relatively efficient [78]. R-CNN [79] leverages high-capacity CNNs by running them on bottom-up region proposals to perform object localization and segmentation [13]. More recently, vision transformers (ViTs) [80] initially introduced for image classification have been successfully adapted to object detection [81]. By leveraging self-attention to model long-range, non-local interactions in images [82], these architectures often achieve improved accuracy. Nevertheless, ViTs typically require substantially more computation, which can hinder real-time inference, especially on resource-constrained drone platforms. Many object detectors, particularly SSD [83], often struggle with small-object detection. In contrast, the YOLO family (YOLO [84], YOLOv2 [85], YOLOv3 [86], and later versions such as YOLOv4 [87] through YOLOv8 [88]) represents a well-known and widely used line of one-stage detection models.

2.6 Quadrotor control

A flight control system is necessary to keep the quadrotor stable and to achieve the required performance, such as accurately tracking a desired trajectory despite external disturbances. As a result, UAV flight control is typically safety and mission critical, as a poor design, insufficient robustness to model uncertainties, or component failures can degrade performance (e.g., payload pointing accuracy) or even lead to loss of the vehicle. Linear control techniques for UAVs are well established and widely adopted because of their simplicity, ease of implementation, and well-defined stability and performance metrics. Among these, the most commonly used methods for quadrotor control are PID [89] and LQR [90]. In general, the main control design challenges involve achieving sufficiently high low-frequency gain for good tracking performance, sufficiently low high-frequency gain to reduce the influence of sensor noise, and adequate stability margins in the intermediate frequency range.

Regarding the control of quadrotors, several nonlinear techniques have been developed, such as backstepping control [91], integral backstepping control [92], sliding control [93], feedback linearization control [94], and their combinations [95]. Nevertheless, owing to the complexity of the highly nonlinear dynamics, a common practice in the literature is to linearize the system model around the hover point [96]. Although this simplification makes the analysis of stability and convergence mathematically tractable by ignoring the complete underactuated dynamics, it is obvious that this approach is inherently restricted to small roll and pitch angles during flight [97]. In addition to these common regulation approaches, the literature has extensively discussed approaches based on optimal control theory. These include the Linear Quadratic Regulator (LQR) [98], which aims to optimize a quadratic cost function for linearized systems, and Model Predictive Control (MPC) [99], which is generally preferred for its capability to address system constraints and trajectory optimization.

In this research, we employ the DJI Tello drone. As shown in Figure 2.4(right), the vehicle is represented as a rigid body with 6 degrees of freedom (6-DOF), which include three translational and three rotational motions. Despite having 6-DOF, the quadrotor is an underactuated system because its dynamics are controlled by only four independent motor commands. The flight dynamics are determined by the differential speeds of the four rotors:

Pitch () and Forward/Backward Motion: A speed differential between the front and rear rotor pairs results in a torque around the body -axis. This causes a horizontal force

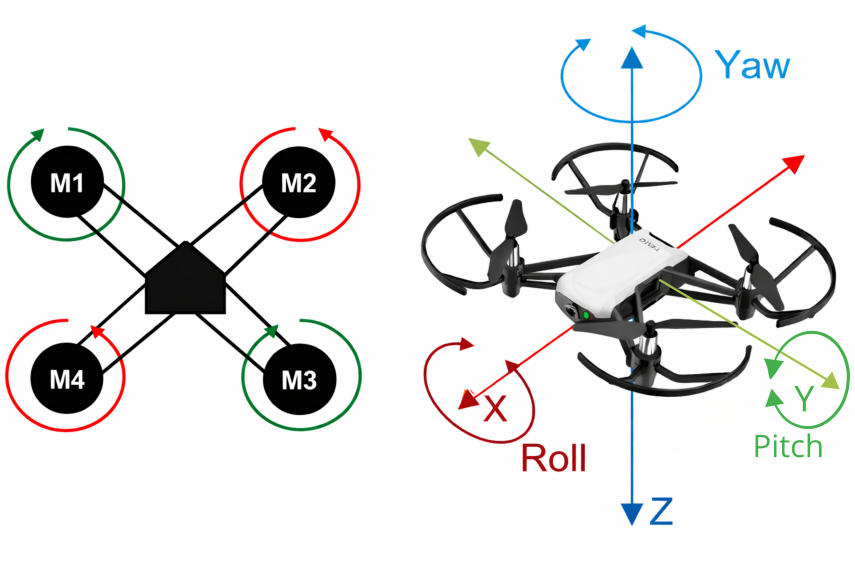


Figure 2.4: (Left): A quadcopter illustrating the propeller rotation directions. (Right): The six degrees of freedom (6-DOF) of our drone.

component, which propels the drone forward or backward.

Roll () and Lateral Motion: A speed differential between the left and right motor pairs results in a rotation around the -axis. This lateral force component directs the thrust vector sideways, translating the drone to the left or right. The translational positions () are directly related to the attitude angles ().

Yaw () and Heading: A quadrotor uses the reaction torque generated by the drag of the propellers. As depicted in Figure 2.4(left), conventionally, two propellers rotate clockwise (CW) and two counter-clockwise (CCW). A speed differential between the diagonal pairs (e.g., increasing CW rotors while decreasing CCW rotors) results in a non-zero reaction torque, which causes the drone to rotate around its vertical -axis.

Altitude () and Vertical Motion: Collective control is used for motion along the -axis. Increasing or decreasing the speed of all four rotors simultaneously changes the total force of the thrust vector against gravity, enabling the drone to ascend, descend, or hover.

2.7 State estimation

The achievement of high-performance control in Unmanned Aerial Vehicles (UAVs) is essentially contingent on the availability of accurate and reliable state estimation [100]. Although many state estimation tasks are accomplished using external centralized localization systems such as the Global Positioning System (GPS) in outdoor settings or motion capture systems such as Vicon and OptiTrack in indoor settings [101], these systems are often not reliable enough to be used for interaction with the environment or navigation in cluttered, GPS-denied regions [102, 103]. To address these infrastructure-related limitations, reliable state estimation needs to be obtained from local sensor data, combining sources such as LiDAR and monocular RGB images to estimate the position and velocity of the vehicle [104]. This is usually accomplished using recursive Bayesian filters or Gaussian mixtures [105], which, however, involve complex manipulations of likelihood models and data association. In model-based state estimation, the state estimator is designed as an observer of deterministic systems or a Kalman filter for stochastic models. Apart from navigation, these state estimators play a pivotal role in system health monitoring, where the estimation residual, defined as the difference between the estimated and actual outputs, ideally needs to be small or resemble a white noise process. Any departure from this or a non-white noise process facilitates real-time fault detection and parameter validation, ensuring that the estimated system parameters are consistent with the nominal values during flight [106, 107].

There exist several Kalman filter algorithms designed for fault detection and state estimation in aerial systems. In the past, the Extended Kalman Filter (EKF) has been the most popular algorithm for nonlinear problems, including the estimation of aerodynamic properties and wind velocity [108, 109], mainly because of its simplicity. However, the EKF algorithm is based on linearization through Jacobian matrices and does not provide any theoretical guarantees for convergence or stability. To overcome these drawbacks, recent advances prefer the Unscented Kalman Filter (UKF), also referred to as the Sigma Point Filter (SPF). Unlike the EKF, the UKF represents probability distributions by a finite number of sigma points [110]. The UKF has been shown to provide better performance in highly nonlinear real-time problems with increased robustness against initial uncertainties and tuning parameters, while requiring the same computational complexity as the EKF [111, 112].

2.8 Vision-based control system

For reactivity to environmental changes to be ensured, the integration of onboard cameras has become a basic requirement in robotics. As passive and low-power sensors, they make possible a broad spectrum of computer vision tasks such as object tracking, mapping, obstacle detection, and nonlinear state estimation. As a result, the integration of visual feedback into the control loop, also known as visual servoing, has become a common practice for robust positioning [113], grasping [114], and navigation in cluttered environments [115]. Nevertheless, vision-based control is faced with a number of challenges, including high computational complexity that can result in low sampling rates and system latencies caused by image acquisition and processing. Additionally, monocular vision systems are faced with the problem of uncertain scale, where the observer cannot specifically separate the motion of the camera and the scale of the scene [116]. Despite these challenges, advances in processing hardware are currently addressing these issues, and there has been rapid development in autonomous vision-based systems [117, 118, 119].

Although robotic perception relies on distance sensors like LiDAR or Radar in many cases, these alternatives are usually large, costly, and prone to environmental interference such as reflective surfaces. In contrast, RGB cameras provide much higher spatial resolution and adaptability to the environment, making them the most popular sensors for interaction and scene understanding. Nevertheless, traditional frame-based cameras have some physical constraints, such as being prone to motion blur, having a low dynamic range, and a low temporal resolution. These drawbacks make them less reliable in high-speed navigation or in environments with extreme lighting changes (for example, when transitioning from an indoor to an outdoor setting). To overcome these limitations, bio-inspired vision sensors, namely event cameras, have been proposed as a reliable solution that provides the ability to detect the environment in such extreme conditions [9, 15]. Event-camera research remains less mature than established traditional frame-based computer vision approaches [120].

2.9 Unsupervised and Weakly-Supervised Learning for UAV Vision

Perception is a necessary step for UAVs to have a good understanding of and be able to interact with their environment. Although most current algorithms, as well as some of the methodologies described in this thesis, are based on [supervised learning using labeled datasets], natural systems learn from direct interaction with their environment. This section reviews previous work that attempts to turn this paradigm on its head in an attempt to have a more natural and feasible learning process. It was demonstrated that this method, over two different tasks—one low-level **Depth estimation** and one high-level **moving object detection**, was effective.

- **depth estimation** Depth estimation is a key component of vision-based UAV perception, complementing object detection by enabling the recovery of three-dimensional information about surrounding obstacles and free space. In agile flight conditions, visual perception can be severely degraded by motion blur, low-texture scenes, and challenging illumination, which in turn compromises feature tracking in Visual–Inertial Odometry (VIO) and reduces detection reliability [121]. To obtain accurate 3D positioning under these constraints, researchers widely adopt depth estimation approaches based on either monocular vision [122] or stereo vision [123], with Convolutional Neural Networks (CNNs) commonly serving as the backbone of modern depth inference pipelines [77]. Furthermore, fusing depth estimates with active range measurements from sensors such as LiDAR allows the construction of dense 3D maps that encode distance information for both environmental structures and detected objects. Such volumetric reconstructions are crucial for safe navigation and are particularly important for identifying planar, obstacle-free regions suitable for autonomous landing [78].

- **Moving Object Detection** Moving object detection is a fundamental capability for both ground and aerial robots, and it has therefore been widely studied through motion-based segmentation methods that separate dynamic elements from the static scene (Figure 2.5 (left)). Early work explicitly modeled occlusions using layered representations with piecewise-affine regions [124], while later approaches improved efficiency and accuracy through graph-based formulations [125] and variational/proba-

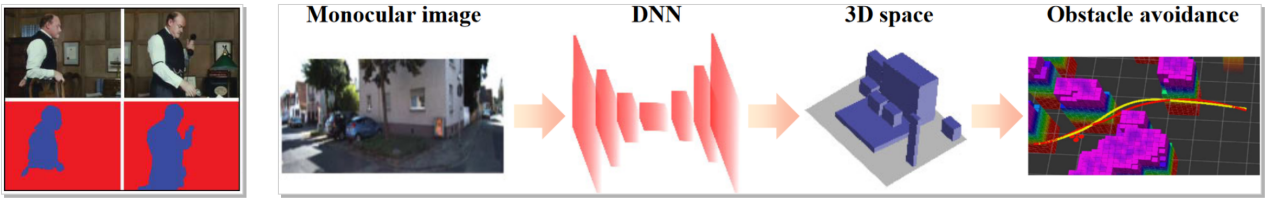


Figure 2.5: Classical moving-object detection methods typically use superpixels and color constancy to maintain object associations across time (left) [2]. Using deep learning, a 3D depth representation is inferred from a monocular image for obstacle avoidance, employing a hybrid direct–indirect approach (right) [3].

bilistic inference techniques that jointly estimate motion and segmentation [126, 127]. Extensions that enforce long-term temporal consistency and color constancy can further enhance robustness, though they often increase optimization complexity and sensitivity to parameter tuning [2]. In modern robotic perception, these motion cues are frequently complemented by computer vision pipelines that extract either handcrafted features or learned semantic information using pre-trained detectors such as YOLO, enabling more reliable scene understanding for downstream mapping and planning, albeit at the cost of requiring carefully prepared and labeled training data. Addressing high-speed navigation scenarios, Michele et al. [3] proposed a monocular perception system that detects obstacles at long range and high speed (Figure 2.5 (right)) without assuming a specific motion model; trained on a mixture of real and synthetic images, their DNN produces robust and consistent depth estimates from monocular inputs to support agile UAV obstacle avoidance.

In our paper [12], we present a moving object detection framework that, similar to classical computer vision literature, operates without the need for annotated training data (unsupervised). However, to achieve robustness comparable to modern learning-based methods, our approach incorporates temporal filtering cues to enhance prediction accuracy and stability.

2.10 Simulation

For UAV object goal navigation, simulation software must accurately reproduce real-world physics and vehicle dynamics. It should also provide tools for acquiring simulated sensor data and interfacing with the simulator for control and evaluation. Using competition-style benchmarks helps standardize scenarios by specifying the available information, system dynamics, and evaluation criteria. Consequently, a key requirement of any simulation platform is its ability to faithfully represent real-world operating conditions. Simulation software enables the creation of controlled replicas of real-world conditions, while the auxiliary software used in many studies provides the interface between the proposed algorithms and the simulated environment. A widely used tool is MATLAB, a numerical computing platform with an integrated development environment, often employed for prototyping and control. The choice of programming language typically depends on the simulator itself, as it must support reading simulated sensor outputs and commanding virtual actuators. Another common framework is the Robot Operating System (ROS), a robotics middleware that facilitates communication between heterogeneous software components. ROS's main advantage is its modular architecture, which allows nodes written in different languages to interact through a unified interface and even supports parallel development across simulated and real systems [19]. The ecosystem of quadrotor control simulators is fragmented (Figure 2.6): some prioritize computational speed, such as MuJoCo [128], while others emphasize highly accurate dynamics [129] or photorealistic rendering [130]. RotorS [131] and Hector [132] are widely used Micro Aerial Vehicle (MAV) simulators built on Gazebo [133], a general-purpose robotics simulator commonly integrated with the Robot Operating System (ROS). The Iris quadrotor model is also frequently used within Gazebo-based simulation setups. These Gazebo-based environments can leverage multiple high-performance physics engines and simulate a wide range of sensors, from laser range finders to RGB cameras.

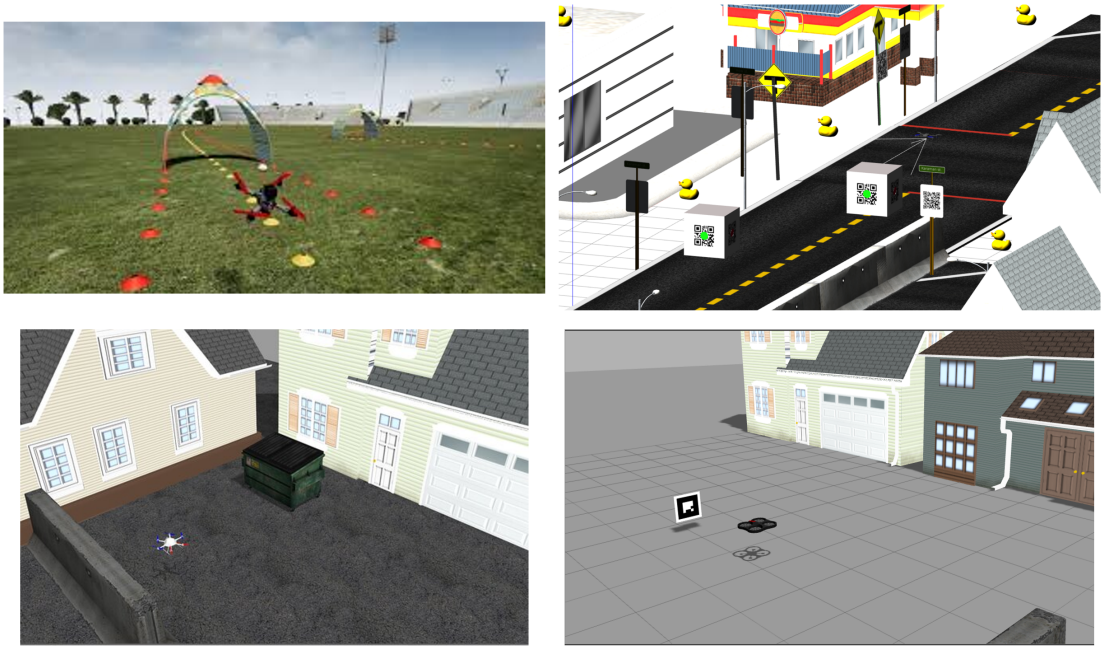


Figure 2.6: Examples of Quadrotor Simulation Environments.

2.11 Conclusion

In this chapter, we provided an overview of aerial robotics with a focus on quadrotors, introducing UAV autonomous navigation and the main sensing modalities used for localization and tracking. We then examined visual navigation for quadrotors in both indoor and outdoor settings, outlining different vision-based control strategies and emphasizing the classical vision-based approach adopted in this work. Building on this background, the next chapter presents a broader review of vision-based systems for autonomous UAV landing, along with the theoretical foundations of the two key components of our method: landing-pad detection and quadrotor control.

Chapter 3

Vision-Based Autonomous Landing of Quadrotor UAVs

3.1 Introduction

Landing remains one of the most critical and hazardous phases of flight. Together with the approach phase, it accounts for a large proportion of aircraft accidents, as reported in [134]. With the rapid growth of unmanned aerial vehicles (UAVs), the development of reliable autonomous landing capabilities has therefore become an increasingly important research topic, particularly for operations in cluttered environments and in situations where conventional navigation aids may be unavailable [11].

In this chapter, we present a 2D image-based feedback control approach for autonomously guiding a quadrotor UAV during the landing mission. In contrast to methods that depend on external localization infrastructure such as motion capture systems or the Global Positioning System (GPS), the proposed solution operates using onboard sensing only and does not require prior knowledge of the landing target location [135]. As highlighted in Chapter 2, cameras offer a lightweight, low-cost, and reliable sensing modality, making them well suited for small UAV platforms. The proposed method leverages visual measurements throughout the maneuver to estimate the relative alignment between the UAV and the landing platform and to generate appropriate guidance commands.

Although visual features are directly used in the feedback loop, the proposed approach differs from classical image-based visual servoing (IBVS). Instead of deriving the control law from the guidance level based on the vehicle dynamics, we design the controller at the guid-

ance level based on the image-feature based dynamics, while exploiting image measurements as (i) a velocity cue and (ii) geometric features encoding the relative position with respect to the landing pad. This design improves robustness to camera and target calibration errors and avoids key limitations of conventional IBVS, most notably the explicit computation of the associated requirement to estimate feature depth [52].

The remainder of this chapter is organized as follows. Section 3.2 introduces the image features and modeling assumptions adopted for the 2D vision-based landing system. Section 3.3 describes the vision-based landing target detection and monitoring pipeline. Section 3.4 presents the proposed control strategy, while Section 3.5 reports and discusses the simulation results. Finally, Section 3.6 concludes the chapter and outlines final remarks.

3.2 Image Features and Assumptions for 2D Vision-Based Landing System Requirements

To execute a landing maneuver using a 2D image-based controller, suitable visual features must first be defined. The UAV is equipped with a single, downward-facing camera that provides a 640×480 pixel video stream. This vision stream is used to detect a predefined yellow, circular ground platform in a Gazebo-based environment (Figure 3.1). This sensing configuration is adopted because it is cost-effective, lightweight, and compact, making it well suited for small UAV platforms [136].

In our approach, the edges of the landing platform's bounding box serve as key cues for aligning the quadrotor with the platform and correcting lateral offsets. In addition, the surrounding ground texture particularly in the regions adjacent to the platform provides motion cues that can be used to estimate the UAV's velocity relative to the ground, helping to ensure a stable and smooth descent [34]. To simplify the problem formulation, we adopt the following assumptions:

- Camera mounting: The camera is rigidly mounted at the UAV's center of mass and aligned with the vehicle body frame, such that the camera reference frame coincides with the body-fixed reference frame.
- Planar landing surface: The landing platform is placed on a flat, horizontal surface, which defines the target plane for landing.

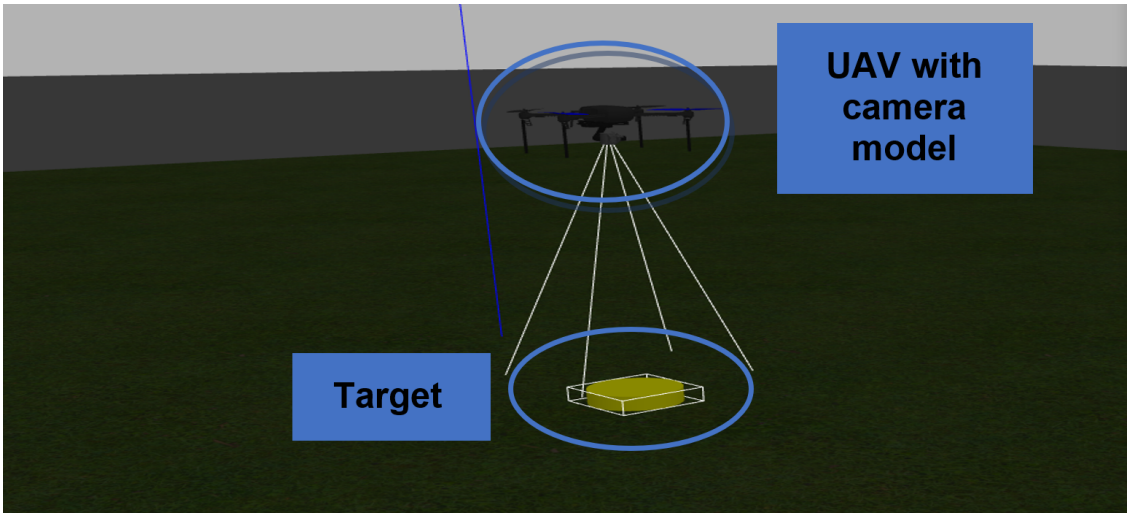


Figure 3.1: Vision-based system in a virtual environment.

3.3 Vision-Based Landing Target Detection and Monitoring

3.3.1 Static Landing Target Detection

In the static-landing scenario, the UAV detects a predefined yellow circular platform using an HSV color–segmentation pipeline [11]. Each 640×480 RGB frame from the onboard camera is first converted to HSV, after which lower and upper yellow thresholds are applied to isolate the landing pad from the background (see Figure 3.2). To improve detection quality, Gaussian blurring is used to suppress noise and small artifacts before applying the Canny edge detector, which produces a clean outline of the circle with minimal internal clutter (see Figure 3.3). A dilation operation is then performed to strengthen and enlarge the detected edges, making the target boundary more prominent and easier to extract. The resulting contour of the landing zone is used to estimate the platform centroid (xc, yc) (see Figure 3.4), which provides the key image features required to align the UAV with the pad and initiate descent when the target is centered.

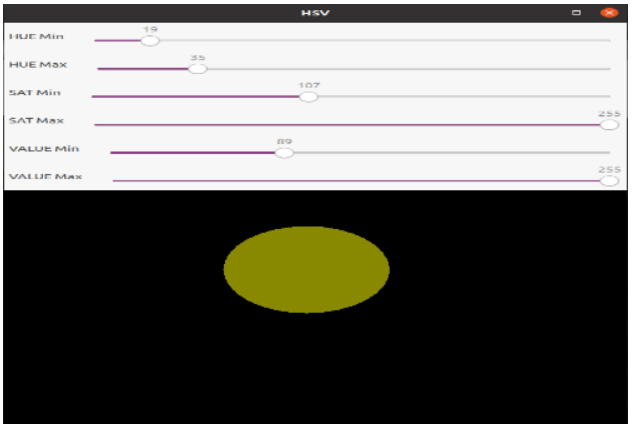


Figure 3.2: Extraction of the circular target using HSV-based color segmentation.

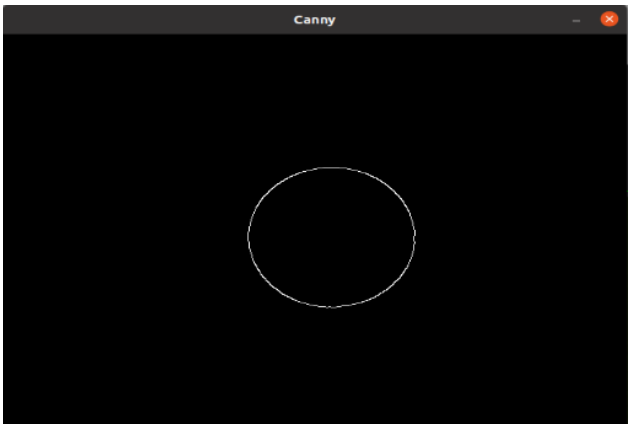


Figure 3.3: Visual representation of the Canny edge detection process.

3.3.2 Dynamic Landing Target Detection and Monitoring

- **Landing Target Detection**

  The object recognition module uses images from the UAV's onboard camera to autonomously detect the landing target in the environment [52]. In particular, the yellow circular platform is identified using an HSV-based color segmentation approach. Since color-based segmentation is most effective when there is a clear contrast between the target and the background, the HSV color space is adopted, as it typically provides more reliable segmentation performance than RGB [137].

After applying HSV thresholding to isolate the yellow region and suppress irrelevant scene elements, a Gaussian blur filter is used as a preprocessing step to reduce image noise and remove small artifacts [138]. To further improve robustness, the Canny edge detector is then applied to the blurred image. This enhances the system's resilience to noise and produces a clean outline of the circular target, visible as thin white edges [139]. Finally, a dilation operation is performed to thicken the detected boundaries by adding pixels along the edges, effectively enlarging the circular region and making it more prominent for subsequent contour extraction, as shown in Figure 3.4. In cluttered Gazebo environments where similarly colored objects may be present, combining both shape cues and color information can further improve reliable identification of the yellow ground circle [136].

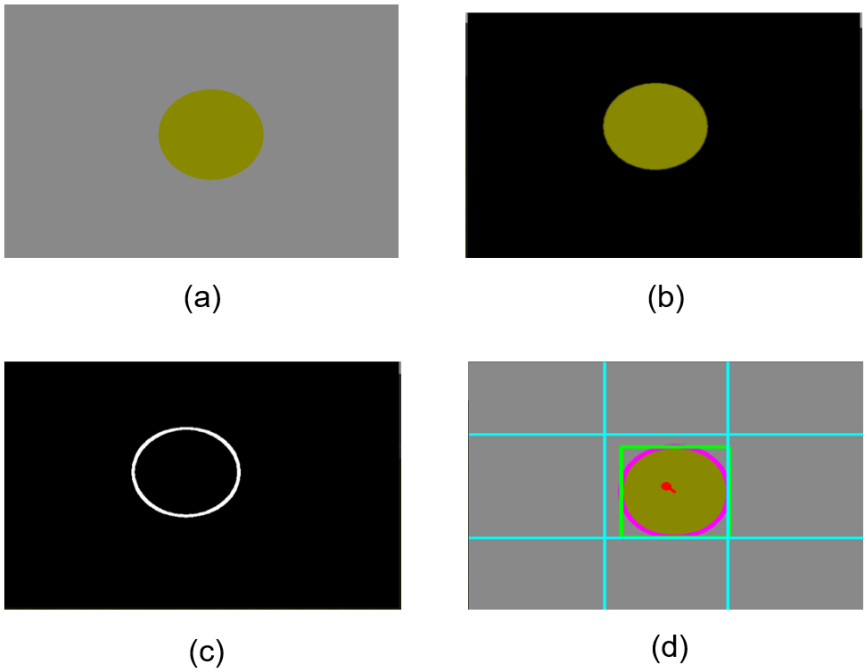


Figure 3.4: (a) Raw RGB input frame; (b) Conversion to HSV color space; (c) Binary segmentation mask of the target; (d) Geometric representation of the target states in image coordinates.

- **Landing Target Monitoring**

  This approach relies on a single onboard video sensor to localize the landing target in

the scene and track it across various trajectory scenarios [52]. As outlined in Algorithm 1, the system detects and monitors the contour of the yellow circular platform, allowing the vision pipeline to run reliably at 30 Hz. Using the extracted contour, the algorithm computes the platform's centroid (x_c, y_c) in the image frame (Figure 3.5). These image-plane coordinates are then used as guidance inputs to steer the UAV and achieve accurate landing.

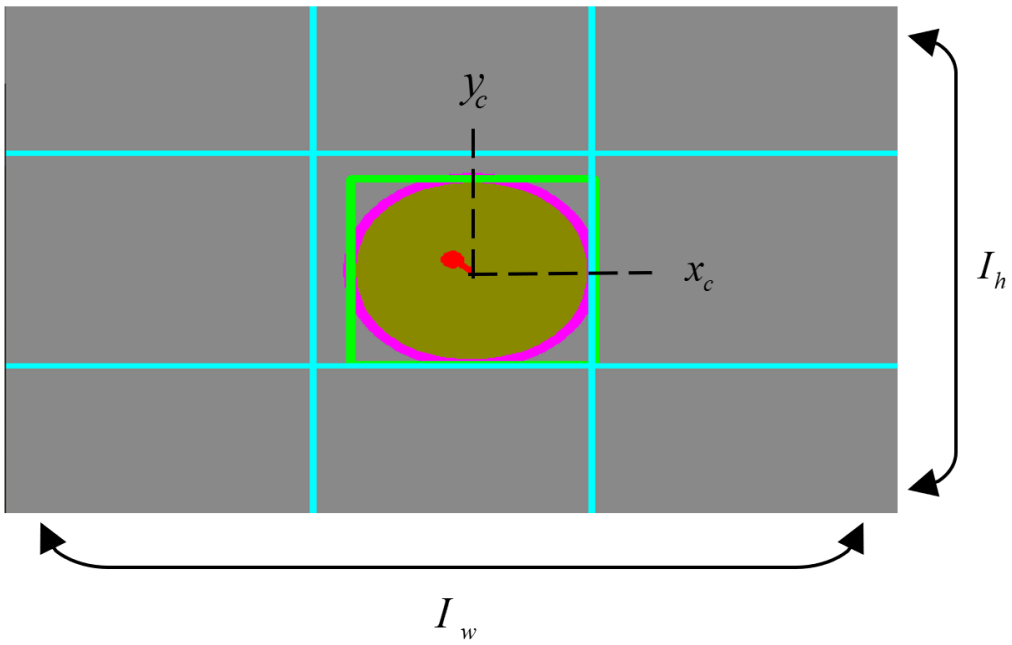


Figure 3.5: Visual state definition relative to the image frame.

Algorithm 1 Identification and Monitoring of the Landing Platform

Input: The onboard camera provides an image for each frame
Output: Position of the landing platform
I_w: Image Width, I_h: Image Height
(x_c, y_c): center coordinates of the landing platform
for each frame **do**
    $x_c \leftarrow \text{computePoint}(x_c)$
    $y_c \leftarrow \text{computePoint}(y_c)$
    if $x_c < I_w/2 - 100$ **then**
        Moved $\leftarrow$ Left
    else if $x_c > I_w/2 + 100$ **then**
        Moved $\leftarrow$ Right
    else if $y_c > I_h/2 - 100$ **then**
        Moved $\leftarrow$ Forward
    else if $y_c < I_h/2 - 100$ **then**
        Moved $\leftarrow$ Backward
    else if $|x_c - I_w/2| < 10$ **and** $|y_c - I_h/2| < 10$ **then**
        Moved $\leftarrow$ Land
    else
        moved $\leftarrow$ No Movement
    end if
    return Moved
end for

3.4 Vision-based Quadrotor Control

The proposed control strategy uses visual feedback to robustly guide the UAV toward the landing platform. Figure 3.6 illustrates the proposed control architecture. In particular, the vision module extracts the contour of the yellow circular pad and estimates its centroid (x_c, y_c), which is used as the reference for control. This contour-based reference improves robustness to sudden image variations and background clutter, and it supports continuous target tracking as long as the platform remains within the camera field of view. Using the

centroid location in the image plane, the controller computes the error signal, i.e.,

$$\begin{aligned} e_x &= x_c - x, \\ e_y &= y_c - y, \end{aligned} \tag{3.1}$$

The error signals are then mapped to velocity commands that drive the UAV laterally and longitudinally (left, right, forward, backward) and trigger landing once the target is sufficiently centered.

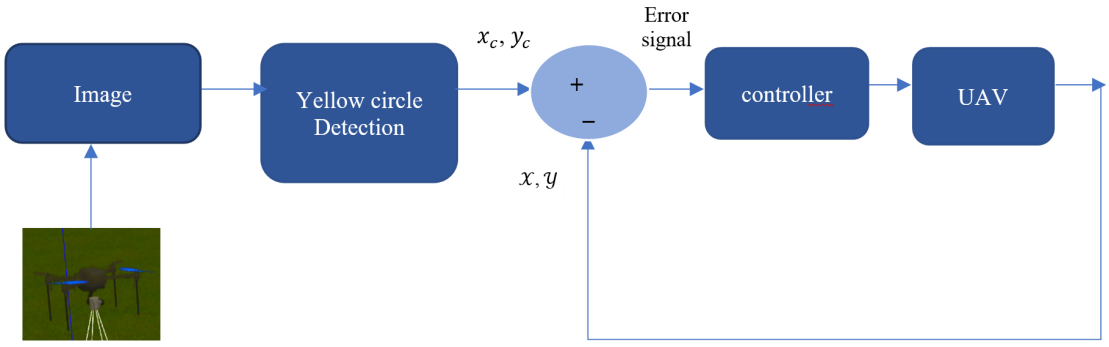


Figure 3.6: A proportional control loop.

As a baseline, a proportional controller is first implemented to generate a velocity command directly from the instantaneous image error,

$$u(t) = k_p\, e(t), \tag{3.2}$$

The proportional gain k_p is tuned empirically in the Gazebo simulation to minimize alignment error and avoid unwanted oscillations in altitude and motion. A proportional (P) controller produces a control action that is directly proportional to the instantaneous tracking error, in our case, the image-plane error is scaled by K_p to generate the velocity command $u(t)$. While this simple feedback yields a rapid response, using only the P term can result in a steady-state offset. As the UAV approaches the target, the error becomes small and the corresponding control output may no longer be sufficient to overcome disturbances, actuator dead zones, or unmodeled effects, causing the drone to settle before reaching the desired reference.

To enhance tracking smoothness and ensure a stable landing, we extend the baseline proportional controller to a full Proportional-Integral-Derivative (PID) scheme. The PID controller preserves the rapid response of the proportional term, while addressing the main limitations of P-only control. In particular, the integral term (I) accumulates the error over time, so that even small residual offsets gradually increase the control effort until the UAV continues moving and the error converges to zero. However, increasing K_p to obtain faster convergence typically amplifies overshoot and may induce oscillations. To counteract this behavior, the derivative term (D) is introduced to provide damping by responding to the error's rate of change. As the UAV approaches the target and the error decreases quickly, the derivative action produces an opposing command that effectively 'brakes' the motion, reducing overshoot and improving settling. By combining these three effects, the PID controller achieves fast correction (P), zero steady-state error (I), and improved stability with minimal overshoot (D), resulting in the final velocity command:

$$u(t) = K_p\, e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de(t)}{dt}. \tag{3.3}$$

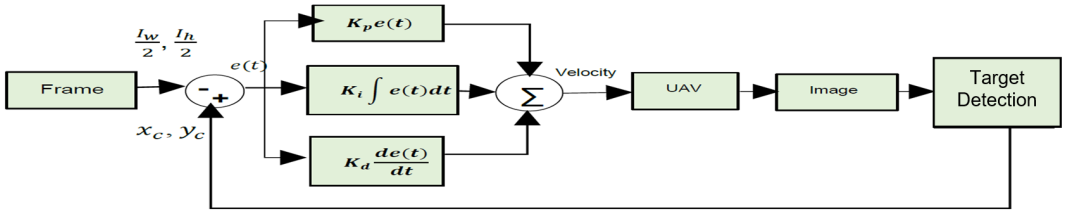


Figure 3.7: A proportional-Integral-Derivative control loop.

As shown in Figure 3.7 using the centroid location of the landing target's (x_c, y_c) in the image plane, the controller computes the tracking error e(t) with respect to the image center $(I_w/2,\, I_h/2)$, i.e.,

$$e_x = x_c - \frac{I_w}{2}, \qquad e_y = y_c - \frac{I_h}{2}, \qquad e_z = 0, \tag{3.4}$$

where I_w and I_h are the image width and height, respectively. Overall, this vision-driven control loop enables the UAV to continuously correct its relative position with respect to the

landing pad and execute a safe, autonomous landing on the designated target.

3.5 Simulation Results

3.5.1 Landing on a Static Target

This section presents the proposed autonomous landing system, evaluated in a Gazebo-based simulation environment using ROS Noetic. The architecture is implemented as two ROS nodes. The first node acquires RGB images from the downward-facing camera and detects the yellow circular landing pad by extracting contours using the method described in Section III (see Figure 3.8). The second node implements the control logic: it receives the detection flag and associated target information, then continuously guides the UAV until a valid landing zone is confirmed and a landing command is executed.

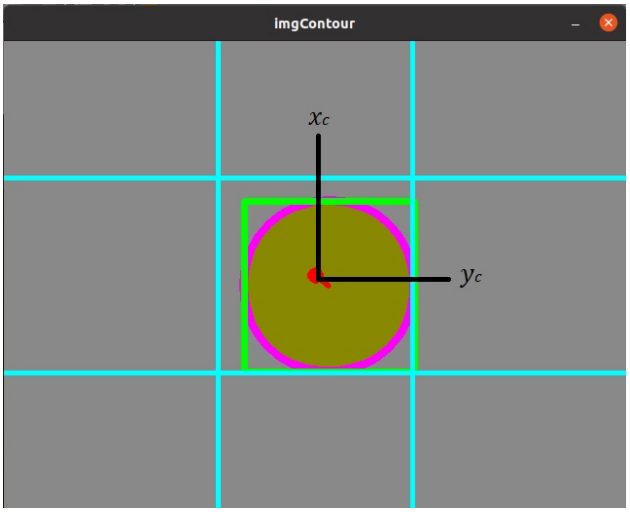


Figure 3.8: Detected landing-zone contour with centroid coordinates (x_c, y_c).

In each simulation, the UAV is initialized at a fixed altitude. After extensive testing, an altitude of 1.7 m was found to provide the best trade-off between reliable detection and stable control. Once the mission starts, the UAV autonomously moves forward to search for the landing platform. Importantly, the pad is not always located directly beneath the UAV; it may appear laterally offset within the camera frame. As a result, the drone adjusts its position at constant altitude, computes the shortest feasible path toward the detected

target, and then performs the landing maneuver. Several representative landing trajectories are shown in Figure 3.9, demonstrating that the UAV can detect the platform, re-align its position, and land autonomously across different approach configurations without human intervention.

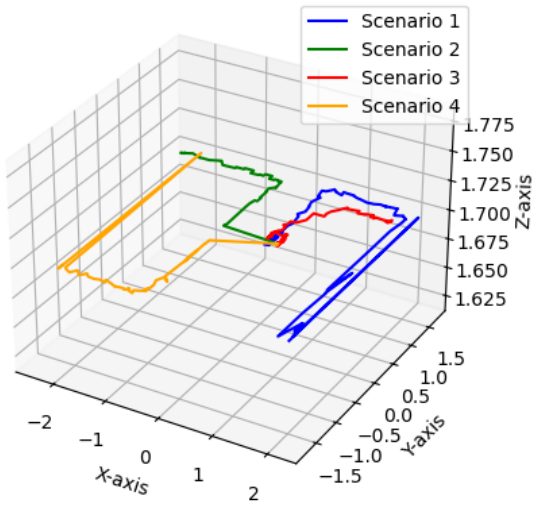


Figure 3.9: The UAV's 3D trajectories across different scenarios during the search for the landing platform.

An illustrative example of the executed autonomous trajectory is shown in Figure 3.10 (red path), guided by the proposed vision-based method.

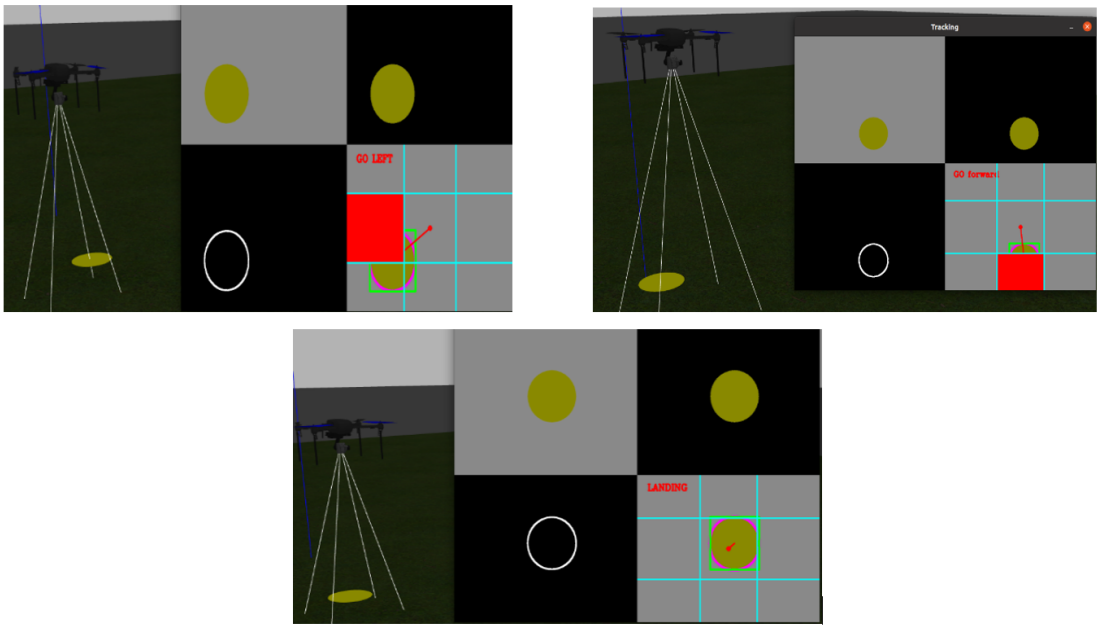


Figure 3.10: Successful detection of the yellow circular target in the Gazebo simulator.

Overall, the simulations confirm that visual navigation enables consistent landing performance. Figure 3.11 reports the distance between the UAV and the landing pad during the procedure; the vertical distance steadily converges to zero, verifying successful touchdown on the platform.

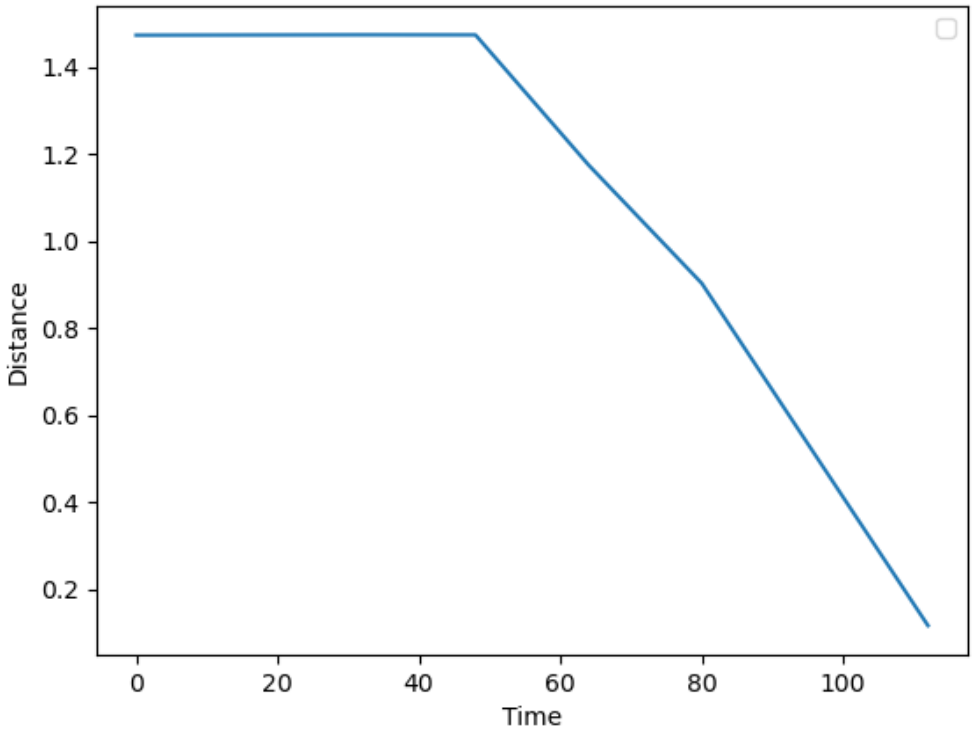


Figure 3.11: The altitude of the UAV relative to the moving platform.

3.5.2 Landing on a Moving Target

This section evaluates a vision-based guidance strategy combined with a PID controller for autonomous landing on a moving platform, simulated in Gazebo with ROS. To validate the proposed approach, we conducted a large set of simulation trials to assess both performance and robustness, with a particular focus on repeatable and accurate landings under different target motions. In these experiments, the landing platform moves at a constant speed of 1 m/s along predefined trajectories, including straight-line paths and partial square segments. Each trial begins with the UAV taking off to an altitude of 1.2 m, after which it searches for the target, detects it using the onboard camera, and transitions to tracking before initiating the landing maneuver.

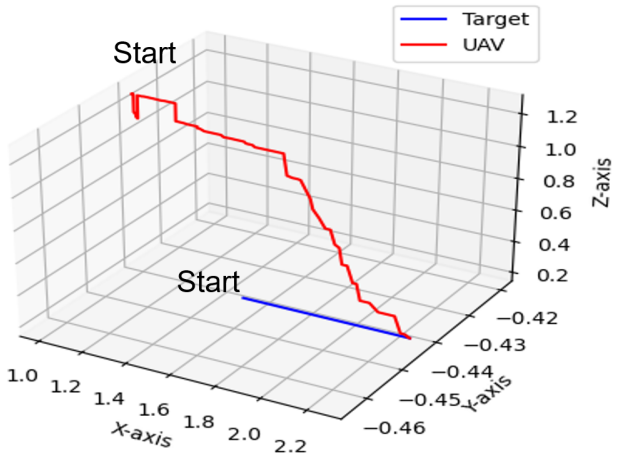


Figure 3.12: Autonomous UAV Landing on a Moving Target Following a Linear Path.

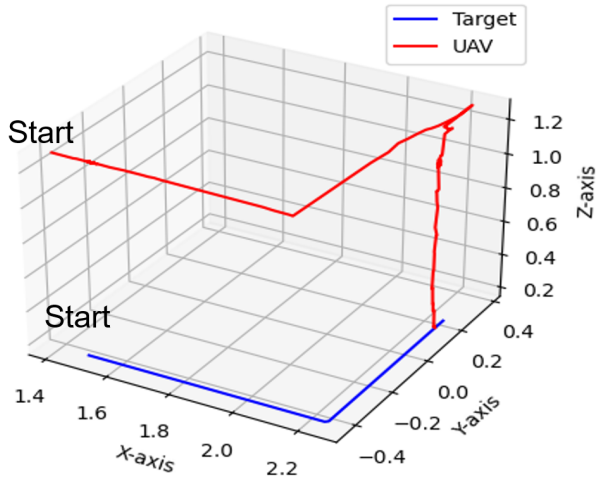


Figure 3.13: Representative UAV trajectories during tracking and autonomous landing on a mobile target following a partial square path.

Figure 3.12 shows a representative successful landing on a target moving along a straight trajectory: the UAV first explores to acquire the platform, then smoothly follows its motion, and finally lands precisely on the moving pad. Figure 3.13 further illustrates the 3D trajectories of both the quadrotor and the platform along a partial square path, demonstrating that the UAV can maintain the commanded altitude while tracking the target and starting the descent at an appropriate point along the path. A sample autonomous landing trajectory is depicted in Figure 3.14, where the UAV, after switching from guided takeoff to autonomous

mode, continuously adjusts its velocity to preserve alignment with the moving platform until landing conditions are satisfied.

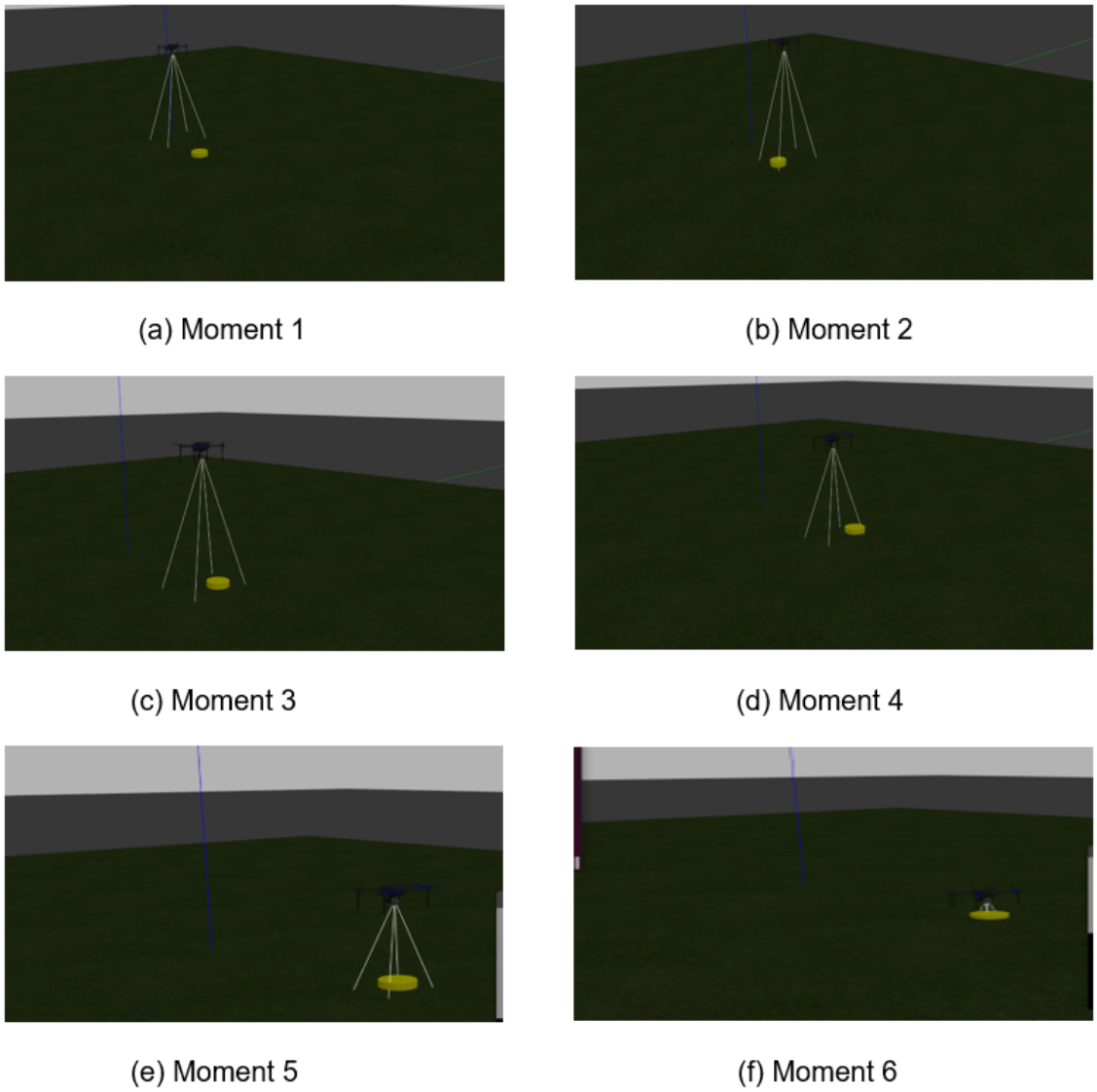


Figure 3.14: Visual Sequence of the Autonomous UAV Target Following and Landing Experiment Conducted in the Gazebo Simulator.

Overall, the results confirm that the relative distance between the UAV and the platform decreases and approaches zero (Figure 3.15), indicating successful interception and touchdown. As shown in Figure 3.12 and 3.13, the PID controllers provide stable and consistent

responses across multiple scenarios, enabling reliable tracking and reducing the time required to complete the landing. Importantly, these simulations demonstrate that accurate landings can be achieved using vision alone, without relying on odometry or additional sensors such as GPS or LiDAR: the UAV continuously estimates the target motion from visual cues, adapts its trajectory accordingly, and performs a fully autonomous landing without human intervention.

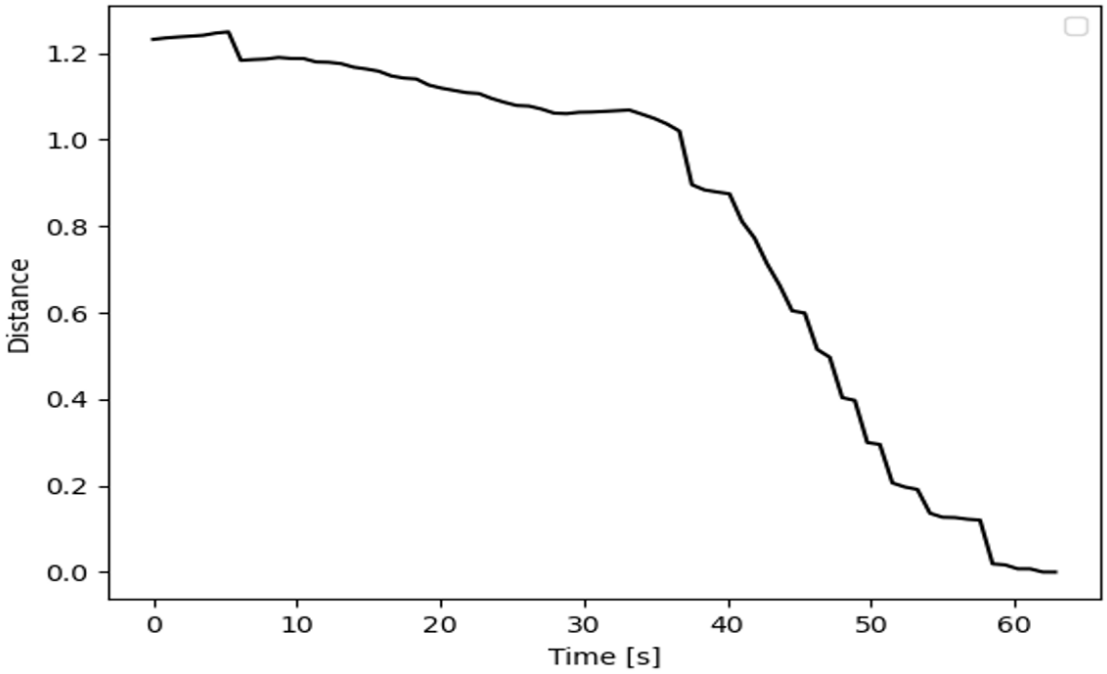


Figure 3.15: Relative Distance Between the UAV and the Landing Platform During Autonomous Landing.

3.6 Conclusion

This chapter presented a vision-based control strategy for autonomous quadrotor landing. By extracting geometric features of the landing pad in the image plane, the controller ensures accurate horizontal alignment and facilitates a smooth descent through the alignment, glideslope, and flare phases. The use of HSV-based segmentation effectively mitigates sensitivity to lighting variations, obviating the need for complex illumination estimation algorithms and simplifying the control design. Finally, the stability of the proposed approach was validated

through theoretical convergence analysis and simulation results.

Chapter 4

Vision-Based Autonomous Object Detection and Tracking in Quadrotor UAVs: Classical Methods and Experimental Validation

4.1 Introduction

The aerial robotics community is progressively advancing toward increasingly demanding operational capabilities, such as tracking high-speed targets, avoiding dynamic obstacles, and navigating complex waypoint sequences, as further detailed in the state of the art review in Chapter 2. However, developing a vision-based controller capable of executing these objectives in the real world is inherently challenging. Factors such as sensor noise and environmental unpredictability often lead to suboptimal performance when compared to controlled laboratory settings. Despite these challenges, vision-based control offers a flexible alternative to external infrastructure (e.g., GPS or Motion Capture), as it can autonomously handle environmental constraints like variable lighting and target occlusion within a unified framework. Furthermore, as onboard computing power continues to improve, solving complex estimation and control algorithms in real time has become increasingly feasible. This allows vision-based methods to complement or even replace traditional external sensors in many applications. Consequently, to be effective, the proposed system must generate tracking solutions fast enough for real-time operation while simultaneously satisfying mission

objectives, system dynamics, and actuator constraints.

The rest of this chapter is organized as follows. Section 4.2 presents the overall system architecture and the methodology of the proposed vision-based approach. Section 4.3 describes the object detection and target tracking algorithms, while Section 4.4 explains the adopted control strategy. Section 4.5 provides the simulation outcomes and their discussion, and also reports the experimental validation of the vision-based tracking system in real-world tests. Finally, Section 4.6 concludes the chapter and highlights potential directions for future work.

4.2 System Overview

The vision-based target tracking setup employs a DJI Tello drone as the UAV platform and communicates with a laptop via a Wi-Fi link. A system overview is presented in Figure 4.1.

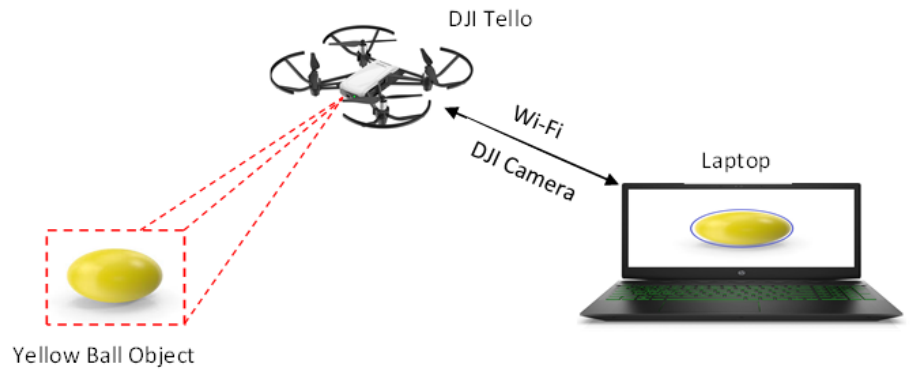


Figure 4.1: Proposed overview of the system.

The workflow of the object detection and tracking system is illustrated in Figure 4.2. Both the computer-vision pipeline and the UAV flight controller run in real time on a laptop [12].

- **Step 1:** The DJI Tello's onboard camera captures images and streams them to the laptop, where the yellow spherical target is detected. Its estimated location is then passed to the tracking module as the initial pixel position.

- **Step 2:** Based on the state estimated in Step 1, a switching tracking strategy is applied to compute the target's current pixel coordinates in each video frame.
- **Step 3:** Using the pixel position obtained in Step 2, the control module computes the UAV's next attitude commands and velocity setpoints, updating the flight-control inputs so the drone follows the target and keeps it within the camera's field of view. This closed-loop interaction improves the overall tracking accuracy.

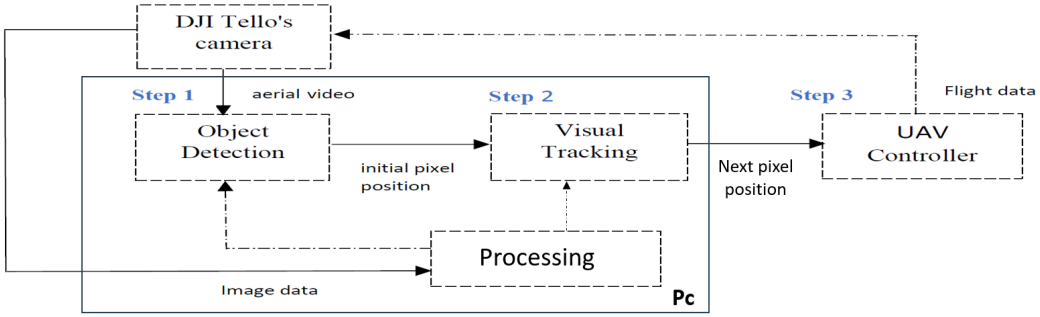


Figure 4.2: Architecture of the vision and control system.

We provide a brief overview of the state transitions. After issuing a state command, the PC typically waits until the drone has either reached the requested state or otherwise responded appropriately before continuing. The system begins in the Ground state. Here, the PC initiates the setup procedure and, once the communication link is established, informs the drone that task initialization has started. When initialization finishes, the PC commands the UAV to take off and waits until the drone is airborne before switching to the Hovering state. The PC then triggers the detection task. It streams images over the Wi-Fi link, performs object detection, and transmits the detection outputs to the drone. Upon receiving these messages, the drone runs the tracking module, converts the tracking output into command references, and forwards them to the low-level controller. Throughout this phase, the drone periodically reports its status to the PC, indicating that it is operational and continuing to track the yellow ball. This loop continues until the UAV reaches the ball waypoint (i.e., successfully "catches" the target). At that point, the drone signals task completion to the PC, which subsequently commands the UAV to land. After touchdown, the system returns to the Ground state.

4.3 Vision-Based Detection and Tracking Pipeline

4.3.1 Object Detection Algorithms

Object detection remains one of the most compelling and challenging problems in computer vision and is a core component of robotic perception. In our approach, the first step is to estimate the ball's location in the image plane using HSV-based segmentation, and then to cover the detected region with a 2D bounding box. The HSV color space is characterized by three components H (hue), S (saturation), and V (value), which better matches human color perception. For segmentation tasks, HSV is often more effective than RGB, particularly under changing illumination conditions [137]. A key advantage is that the hue channel makes the detection pipeline more robust to lighting variations. In general, HSV encodes color through the combination of hue, saturation, and value [140]:

- **Hue** denotes the dominant color component (e.g., red, blue, yellow) and is typically expressed as an angle in the range 0° to 360°.
- **Saturation** represents the color purity or intensity, ranging from 0 to 100.
- **Value** describes the brightness (lightness) of the color and also lies within the range 0 to 100.

The conversion of color images from RGB to the HSV color space is carried out using Equations (4.1)–(4.3) as follows [137, 12]:

$$H = \arccos\left(\frac{\frac{1}{2}(2R - G - B)}{\sqrt{(R-G)^2 - (R-B)(G-B)}}\right) \tag{4.1}$$

$$S = \frac{\max(R,G,B) - \min(R,G,B)}{\max(R,G,B)} \tag{4.2}$$

$$V = \max(R,G,B) \tag{4.3}$$

After extracting the largest contour in the image, we estimate the target's geometric parameters by fitting the minimum enclosing circle. This step yields the circle center (x_c, y_c) and radius r_c, which are subsequently used for tracking and control. The required computations are given by the following equations:

$$C_{\text{largest}} = \arg \max_{C_i \in \text{Contours}} A(C_i) \tag{4.4}$$

$$(x' - x)^2 + (y' - y)^2 = r^2 \quad \forall (x', y') \in C_{\text{largest}} \tag{4.5}$$

This approach facilitates training-free, adaptive color localization by dynamically deriving HSV thresholds from a designated reference ROI. By integrating this calibration with selective region processing and similarity-based aggregation, the system achieves significant robustness against the variable lighting conditions typical of real-world settings [137].

4.3.2 Object Tracking Algorithms

Observation Model (Measurement Model)

The object detection module extracts the ball's geometric features from each incoming image frame by estimating its centroid and apparent size (radius) [136]. Specifically, after segmenting the target and selecting the corresponding contour, the algorithm computes the centroid coordinates (x_c, y_c) in the image plane and the radius r_c of the fitted enclosing circle. These quantities provide a compact measurement of the target's location and scale, and they serve as the sensor observations supplied to the Extended Kalman Filter (EKF) for recursive state estimation [12].

Accordingly, the observation (measurement) vector at time t is defined as

$$O_t = [x_c,\ y_c,\ r_c], \tag{4.6}$$

where x_c and y_c denote the pixel coordinates of the detected ball center, and r_c represents the ball radius in pixels (see Figure 4.3). In practice, (x_c, y_c) captures the target displacement relative to the camera's optical axis, while r_c provides an indirect cue of range (the ball appears larger as it gets closer). By filtering these observations over time, the EKF reduces measurement noise and improves the stability of the tracking estimates[141].

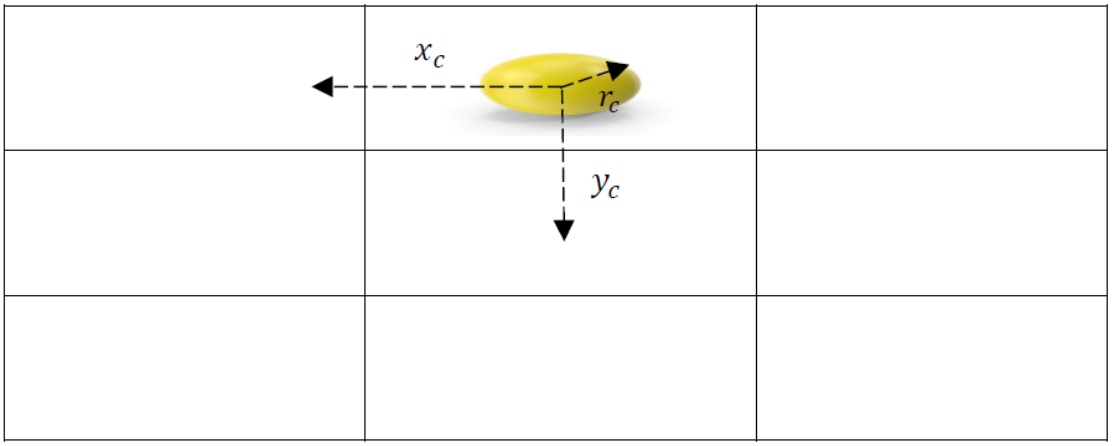


Figure 4.3: The system measurements are encoded in an observation vector comprising the ball's image-plane centroid (x_c, y_c) and its estimated radius r_c in the current frame.

EKF-Based Visual Tracking (State Estimation)

The Extended Kalman Filter (EKF) is a state-estimation method that reconstructs unobserved system states from indirect measurements corrupted by noise and uncertainty. It does so by approximating nonlinear process and measurement models with locally linear representations, obtained via a first-order Taylor expansion about the current estimate at each time step. As a result, the EKF can filter the noisy outputs of the detection module and provide continuous estimates of the target state, such as its position, over time [44]. Consider the following nonlinear system, described by the state-transition and measurement models:

$$x_k = f(x_{k-1}, u_{k-1}) + w_{k-1} \tag{4.7}$$

$$z_k = h(x_k) + v_k \tag{4.8}$$

Here, $f(\cdot)$ denotes the nonlinear state-transition function, while $h(\cdot)$ represents the nonlinear measurement (observation) function. The vectors w_k and v_k correspond to process and measurement noise, respectively, and are modeled as zero-mean multivariate Gaussian disturbances. In this formulation, the control input u_k is set to zero, reflecting the assumption that no external acceleration input is applied.

Given the functions f and h, the target pose estimation proceeds in two stages. First, a predicted (a priori) state $\breve{x}_k$ is propagated from the previous estimate. Next, the filter

incorporates the new measurement to compute the corrected state $\hat{x}_k$, refining the prediction using the measurement residual.

- The prediction step is as follows:

$$\breve{x}_k = f(x_{k-1}) \tag{4.9}$$

$$\breve{P}_k = F_k P_{k-1} F_k^T + Q_{k-1} \tag{4.10}$$

- The correction step is as follows:

$$S_k = H_k P_{k-1} H_k^T + R_k \tag{4.11}$$

$$K_k = P_{k-1} H_k^T S_k^{-1} \tag{4.12}$$

$$\hat{x}_k = x_{k-1} + K_k \left(z_k - h(x_{k-1})\right) \tag{4.13}$$

$$\hat{P}_k = (I - K_k H_k) P_{k-1} \tag{4.14}$$

In the Extended Kalman Filter (EKF), the estimation procedure is summarized by Equations (4.9)–(4.14). The matrices Q_{k-1} and R_k denote the process-noise and measurement-noise covariance matrices, respectively. The term P_k represents the *a posteriori* error covariance, K_k is the Kalman gain, and S_k is the innovation (residual) covariance. Since the EKF operates on nonlinear models, the functions $f(x_k)$ and $h(x_k)$ are locally linearized at each time step, and the resulting Jacobian matrices are given by F_k and H_k, respectively [142].

By selecting the ball's image-plane location and its translational velocity as the quantities to be estimated, we model the target dynamics with a state vector that jointly captures both position and motion. In particular, the position components describe the ball center in the camera image coordinates, while the velocity components represent the rate of change of this position over time. This choice enables the filter to predict the target's future location between frames and to smooth noisy detections by enforcing a consistent motion model. Accordingly, the state vector of the ball-motion model is defined as follows:

$$\begin{bmatrix} X_c \\ Y_c \\ \dot{X}_c \\ \dot{Y}_c \end{bmatrix} = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_{c-1} \\ Y_{c-1} \\ \dot{X}_{c-1} \\ \dot{Y}_{c-1} \end{bmatrix} + \begin{bmatrix} \frac{T^2}{2} \\ \frac{T^2}{2} \\ T \\ T \end{bmatrix} w_k \tag{4.15}$$

4.4 Vision-Based PID Control for Guidance

The proposed vision-based control architecture is implemented on a quadrotor UAV equipped with an onboard 640 × 480 pixel camera, enabling accurate real-time tracking of a moving aerial target. To keep the target centered in the image plane, a PID control scheme is used to regulate the UAV's translational velocities and yaw rate [143]. The PID controllers take as input the image-plane tracking error $e(t)$, computed at time t as:

$$e_x = \text{prd}_x - \text{cen}_x \tag{4.16}$$

$$e_y = \text{prd}_y - \text{cen}_y \tag{4.17}$$

prd_x and prd_y denote the EKF-predicted target position in the image plane after the measurement update, whereas cen_x and cen_y correspond to the coordinates of the image-frame center (see Figure 4.4).

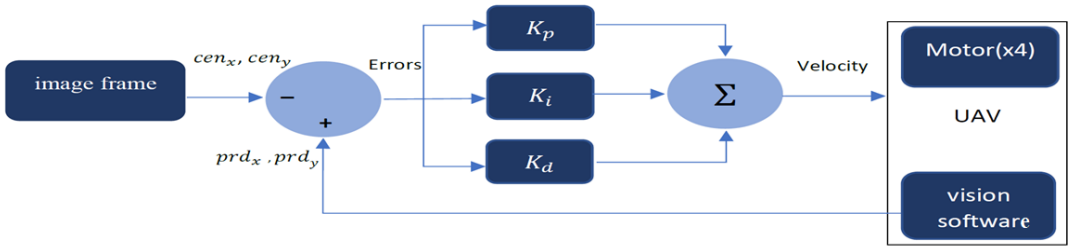


Figure 4.4: Schematic representation of the PID feedback loop used for UAV control.

This class of controller is commonly adopted in many applications because of its simple structure and reliable performance, and because its gains can be tuned with relative

ease [19, 144]. The control output is mapped to the UAV velocity command, where the proportional, integral, and derivative gains (K_p, K_i, K_d) are selected to achieve the desired response. Accordingly, the final velocity command $u(t)$ is computed as follows [52]:

$$u(t) = K_p\, e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de(t)}{dt} \tag{4.18}$$

- The associated transfer function is expressed in Equation (4.19):

$$u(s) = K_p + \frac{K_i}{s} + K_d s \tag{4.19}$$

Algorithm 2 outlines the implementation of the quadrotor's PID control pipeline.

Algorithm 2 PID Control

$prev_error \leftarrow 0$
$I_error \leftarrow 0$
while true **do**
    $cur_error \leftarrow predicted_state - center_of_image$
    $P_error \leftarrow cur_error$
    $I_error \leftarrow I_error + cur_error \times dt$
    $D_error \leftarrow \dfrac{cur_error - prev_error}{dt}$
    $output \leftarrow k_p \times P_error + k_i \times I_error + k_d \times D_error$
    send_command$(output)$
    $prev_error \leftarrow cur_error$
    sleep(dt)
end while

At each time step, the target's position is extracted from the current and previous video frames to estimate its speed. Using these measurements, the system computes the required flight-control commands and transmits them to the flight controller, allowing the UAV to track the yellow ball reliably. Target centering is achieved directly from the ball's image coordinates: the Y-coordinate regulates vertical motion so the drone ascends or descends to align the ball vertically, while the X-coordinate drives yaw control so the UAV rotates about its Z-axis to center the ball horizontally. Thus, when the ball drifts left or right in the frame, the drone adjusts its yaw to re-align the target. Finally, forward motion (pitch)

Algorithm 3 Tracking Controller

Inputs: Width and Height of the onboard camera's image, prd_x, prd_y
Initialization:
    $\text{cen}_x \leftarrow$ Image Width / 2
    $\text{cen}_y \leftarrow$ Image Height / 2
    $PID \in \mathbb{R}^3$: PID parameters (K_p, K_i, and K_d)
Outputs: $u(t) \in \mathbb{R}^3$
while true **do**
    $e_x \leftarrow \text{prd}_x - \text{cen}_x$
    $e_y \leftarrow \text{prd}_y - \text{cen}_y$
    if $|e_x| > 30$ **then**
        if $e_x > 0$ **then**
            $u(t) \leftarrow$ left
        else
            $u(t) \leftarrow$ right
        end if
    else if $|e_x| \geq 7$ **then**
        $u(t) \leftarrow$ forward
    end if
    if $|e_y| > 30$ **then**
        if $e_y > 0$ **then**
            $u(t) \leftarrow$ up
        else
            $u(t) \leftarrow$ down
        end if
    else if $|e_y| \geq 7$ **then**
        $u(t) \leftarrow$ forward
    else
        $u(t) \leftarrow 0$
    end if
end while

is commanded according to the magnitude of the positional error, enabling the UAV to smoothly reduce the distance to the ball [145]. The complete tracking procedure is outlined in Algorithm 3, and the corresponding pseudo-code is presented above.

4.5 Simulation and Experimental Results

To demonstrate the feasibility and robustness of our approach, we conducted a two-phase evaluation. First, numerical simulations are presented to analyze the state estimation and control logic under controlled, yet realistic conditions. Second, field experiments are detailed to confirm the system's performance in dynamic real-world settings, specifically focusing on GPS-denied tracking of a dynamic target.

4.5.1 Simulation Results

This subsection presents our tracking system, which was first validated in a Gazebo–ROS simulation. Figure 4.5 illustrates the drone tracking a ball target within the simulated environment. In this setup, the UAV is equipped with an onboard camera that continuously observes the scene and estimates the ball's position. Using these visual measurements, the drone adjusts its orientation and relative position to remain aligned with the target and stay on course. The results highlight the effectiveness of visual perception for real-time object tracking: the UAV continually corrects its trajectory to follow the ball's motion, including when the target moves along a circular path.

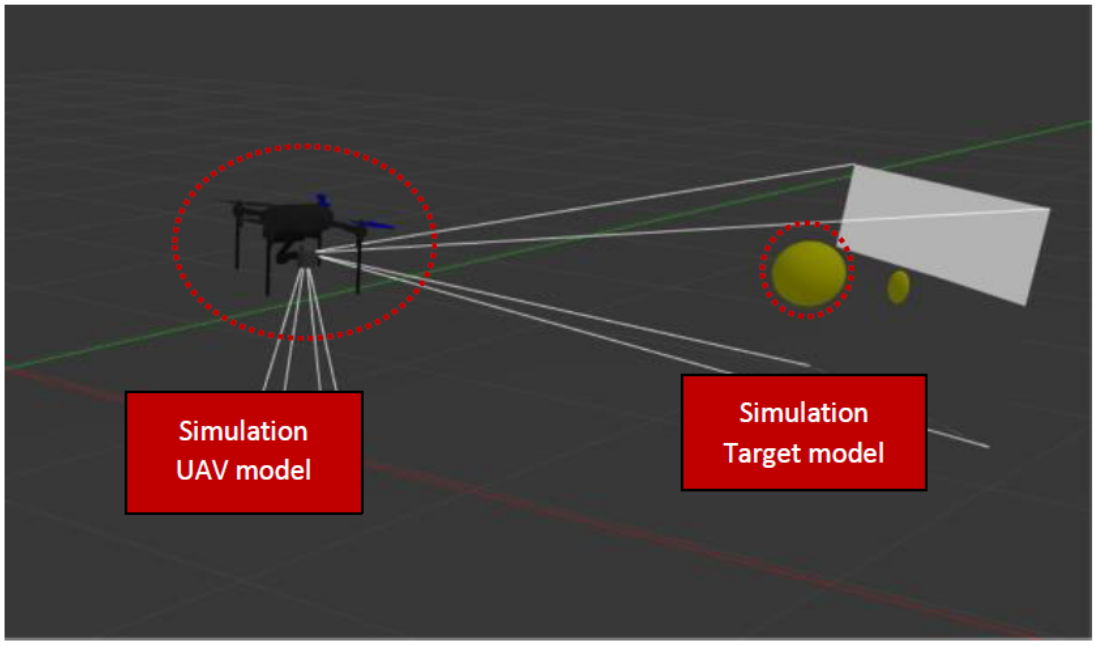


Figure 4.5: Simulated-environment object tracking model.

Figure 4.6 presents the velocity estimation error of the aerial target. While the target remains stationary between 40 and 60 s, the error is essentially negligible. At $t = 60\,\mathrm{s}$, the target begins moving at a constant speed of $v = 1\,\mathrm{m/s}$, which produces a slight increase in estimation error. Nevertheless, the error consistently stays below $0.01\,\mathrm{m/s}$, confirming that the proposed state-estimation algorithm maintains high accuracy and robust performance during both rest and steady-motion phases.

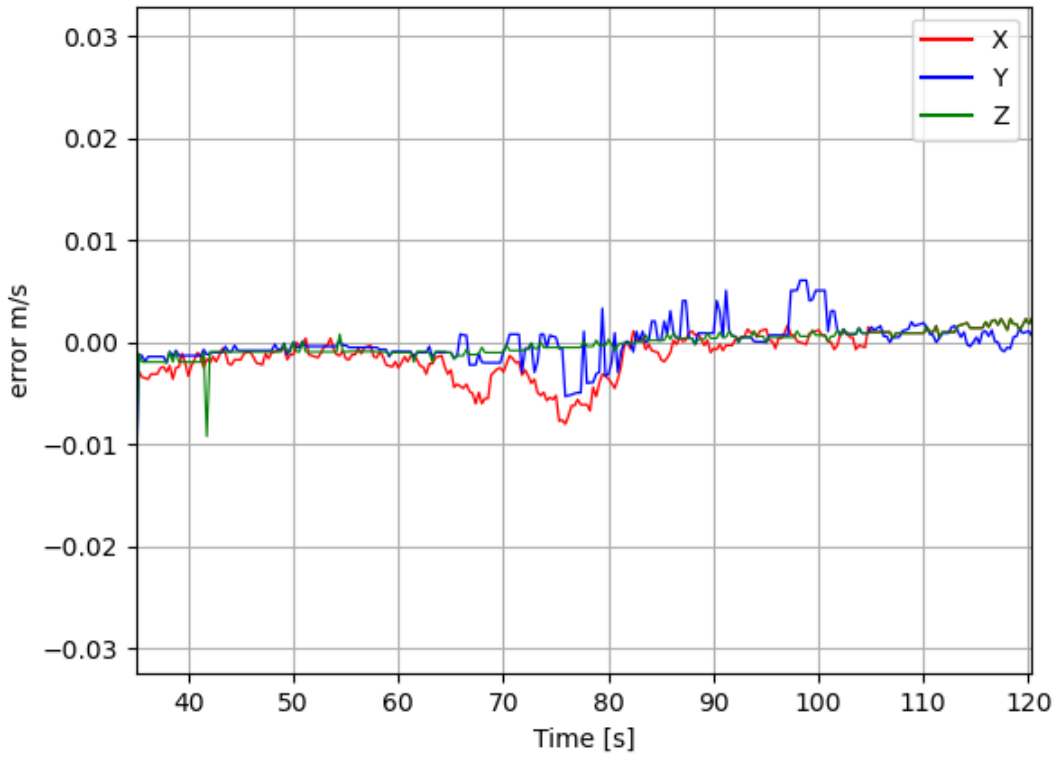


Figure 4.6: Deviation between estimated and actual velocity.

Figures 4.7 and 4.8 illustrate the 3D trajectories of the target object and the quadrotor, highlighting the UAV's ability to navigate autonomously while accurately tracking the moving ball. In the Gazebo–ROS simulation, the ball follows two motion patterns: a square path (Figure 4.7) and a circular loop (Figure 4.8). Throughout both scenarios, the drone uses onboard perception to estimate the ball's position and continuously adapts its flight path to remain aligned with the target.

During the square trajectory (Figure 4.7), the UAV repeatedly updates its state estimates to maintain smooth and consistent tracking along the full path. When the target switches to the circular motion (Figure 4.8), the controller further adjusts its commands to reduce tracking deviations and preserve close following performance. Overall, these results demonstrate the UAV's adaptability to different target dynamics and emphasize the robustness of integrating vision-based tracking within the Gazebo–ROS framework for real-time operation.

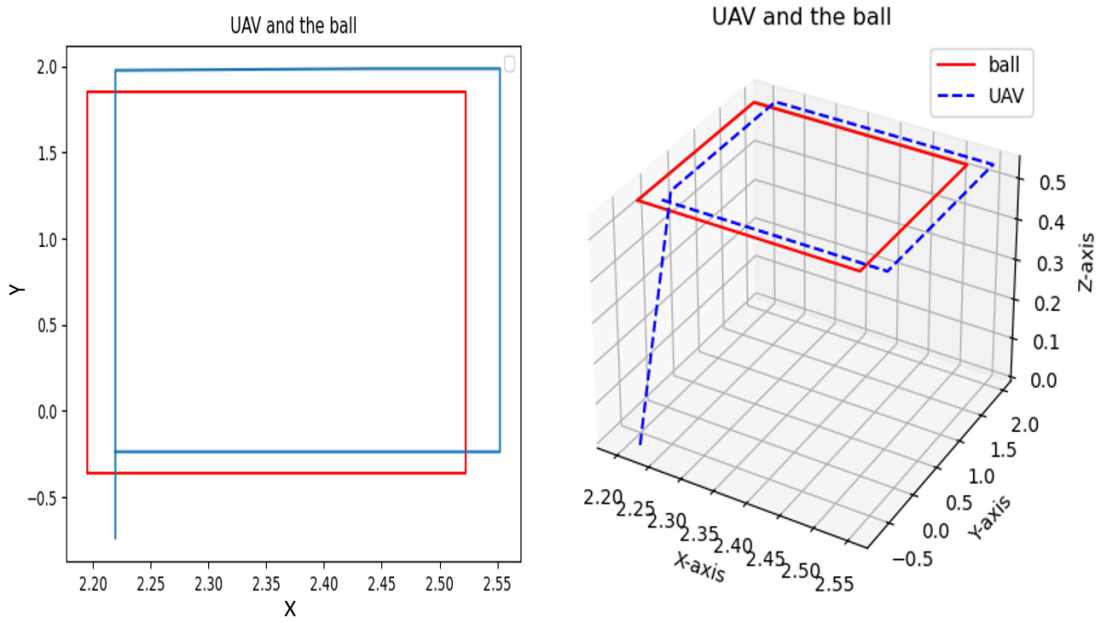


Figure 4.7: Tracking performance for targets following square trajectories.

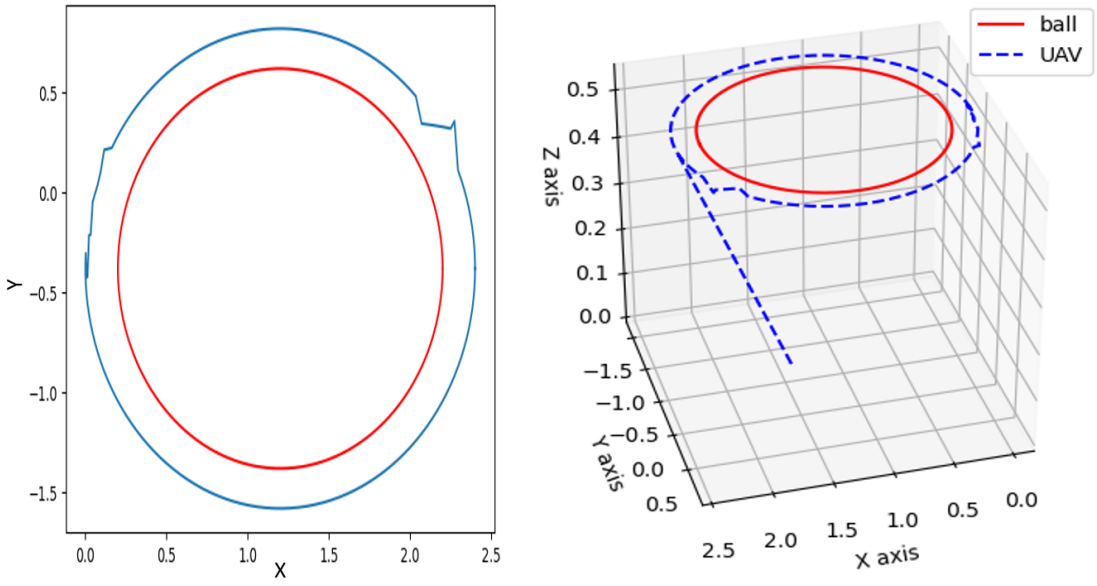


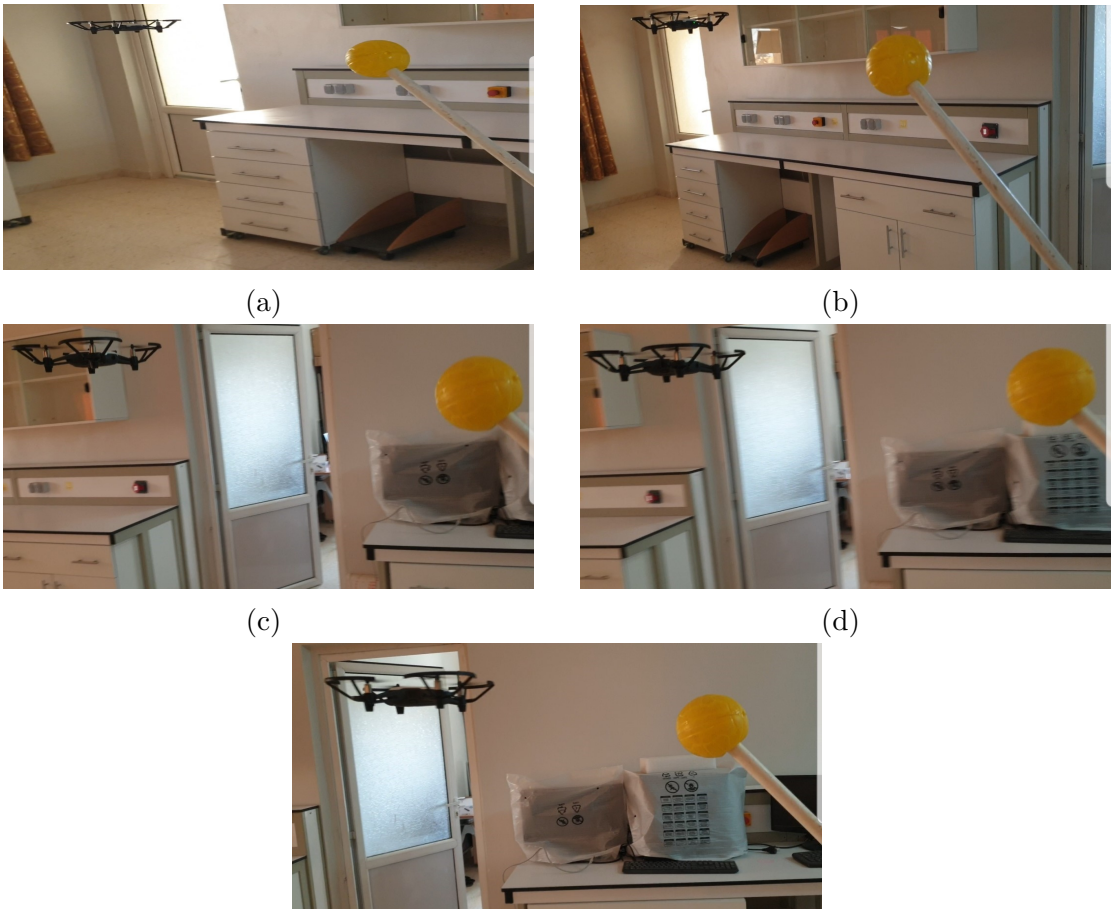
Figure 4.8: Tracking performance for targets following circular trajectories.

The following results are derived from a simulation environment designed to mimic real-world limitations. By restricting the quadrotor to onboard sensors for state estimation and

immediate control generation, we effectively evaluate the complete pipeline. This approach ensures that the interplay between detection, estimation, and control is tested against diverse target maneuvers.

4.5.2 Experimental Results

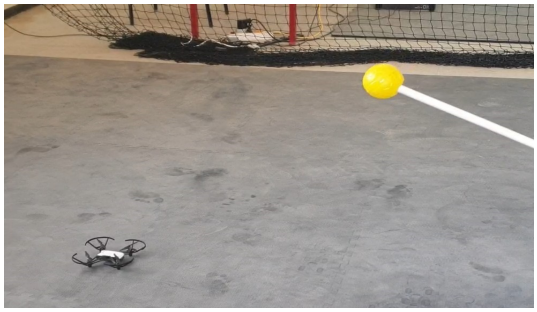
In this subsection, we report results from real-world experiments in which a DJI Tello drone executes an autonomous target-tracking sequence. The ground station is an HP Pavilion Gaming 15 laptop connected to the Tello over a Wi-Fi link, with an effective communication range of up to 100 m. The laptop streams the onboard video, processes it using a vision-based algorithm, and transmits control commands back to the drone through the flight-controller interface. The DJI Tello carries an RGB camera, and during the experiments the video stream was recorded at 30 fps with a resolution of 640×480 pixels. As an initial step, a yellow ball was selected as the aerial target. The UAV took off to an altitude of 1 m, stabilized at the desired position, and detected the ball while maintaining an expected separation of approximately 5 m. The target was then moved along a straight-line trajectory. Throughout this test, the UAV reliably detected and accurately tracked the moving ball, demonstrating the robustness of the proposed tracking system. Figure 4.9 shows the UAV following the yellow ball as it moves at a constant speed along the straight path.



(a) (b) (c) (d) (e)

Figure 4.9: Sequential snapshots of a UAV tracking a target moving along a straight-line trajectory: (**a**) takeoff from the ground, (**b**) approach phase, (**c**) initiation of target following, (**d**) continuous tracking during motion, and (**e**) successful mission completion demonstrating stable tracking.

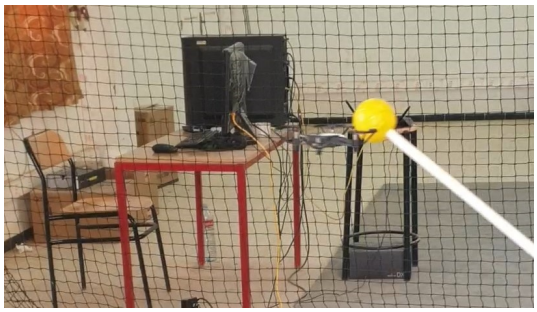
Figure 4.10 illustrates the system's detection and tracking performance as the yellow ball is moved along a circular trajectory. The DJI Tello drone is configured to lock onto and follow the target, providing a controlled testbed for assessing the proposed algorithms. The results confirm the effectiveness and robustness of the detection and tracking pipeline, while also demonstrating the reliability of the target state-estimation method under changing illumination. Overall, the experiment verifies the stability of the approach and supports its suitability for operation in dynamic environments.



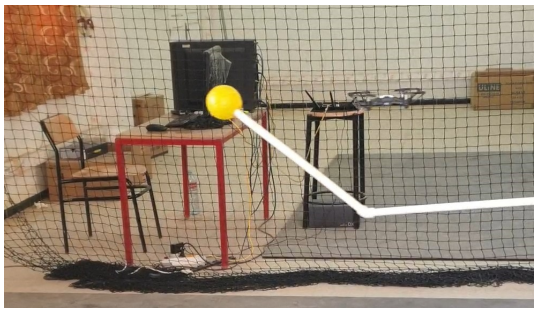
$t = 0s$



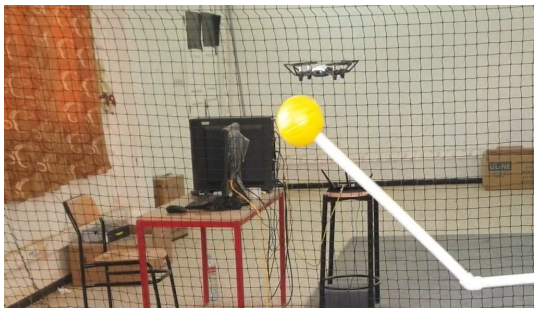
$t = 3s$



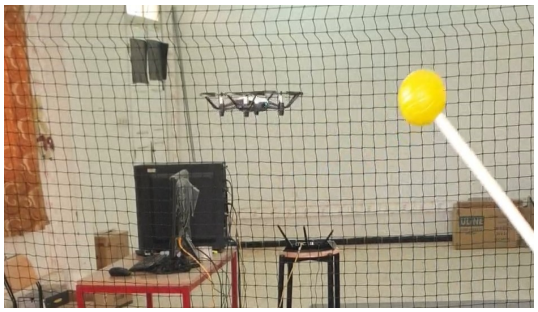
$t = 6s$



$t = 9s$



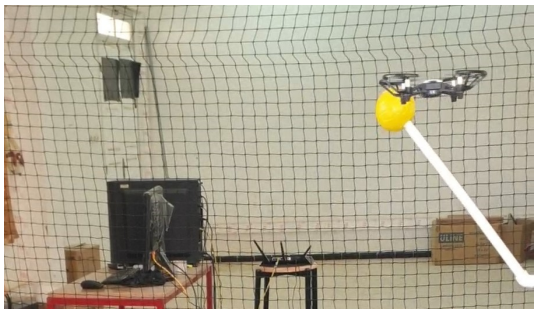
$t = 12s$



$t = 15s$



$t = 18s$



$t = 21s$

Figure 4.10: Autonomous tracking of a target executing a circular maneuver.

Despite the presence of numerous objects in the scene, the proposed approach remains robust under changing environmental conditions and consistently maintains reliable tracking of the yellow ball. Figure 4.11 shows the UAV and the target moving along random trajectories, with the ball's speed varying over time. In this experiment, the DJI Tello successfully follows the target, although a small tracking lag appears during intervals of rapid ball motion.

Overall, the proposed system achieved a 90% success rate in tracking the yellow ball. The remaining 10% of trials failed primarily when the target moved outside the DJI Tello's camera field of view. This work aims to guide an autonomous drone toward a yellow ball using visual information only, thereby mitigating issues related to GPS inaccuracies and enabling operation in GPS-denied environments. The experimental results show that the proposed detection algorithm consistently identifies the target under varying illumination and adapts rapidly to changes in the surrounding scene. To assess performance and robustness, a total of ten flight tests were performed.



$t = 0s$



$t = 3s$



$t = 6s$



$t = 0s$



$t = 3s$



$t = 6s$

Figure 4.11: UAV tracking a target executing complex, non-deterministic maneuvers.

4.6 Conclusion

This chapter presented a vision-based autonomous object detection and tracking framework for quadrotor UAVs, offering an efficient and lightweight solution for recognizing and following dynamic targets in real time. The proposed pipeline is training-free and relies on adaptive color segmentation by automatically deriving HSV thresholds, followed by an Extended Kalman Filter (EKF) to estimate the target state over time. The filtered estimates are then used as inputs to a PID controller to generate flight commands that maintain stable tracking performance.

The approach was validated through a two stage evaluation. First, a Gazebo–ROS simulation environment was used to assess the full tracking loop under controlled conditions. Second, real world experiments were conducted to test robustness under practical challenges such as background clutter, motion variations, and illumination changes. Across both phases, the results confirmed that the system achieves reliable detection and tracking accuracy while maintaining low computational cost an essential requirement for resource constrained platforms such as low-cost UAVs equipped with onboard cameras.

Building on the findings of this chapter and Chapter 3, which demonstrated that vision-based control can guide UAV motion using visual input alone for both static and dynamic targets, the next chapter extends the study to deep-learning based detection and tracking. In particular, we investigate how integrating a YOLOv8 detector with Kalman filtering can improve object detection and classification performance in unknown and unstructured environments.

Chapter 5

An Advanced YOLO-Based Framework for Real-Time Autonomous Object Detection in UAVs

5.1 Introduction

In recent years, the field of robotics has witnessed a surge in vision-based methodologies that enable remarkably agile and dynamic maneuvers. These breakthroughs range from high-speed translations [146] and the reactive precision required for ball-catching [147] to the high-stakes navigation of autonomous drones through narrow apertures [148]. Traditionally, engineering a controller for such sophisticated behaviors is a Herculean task; manual design often struggles to account for the sheer complexity of high-speed flight dynamics, frequently resulting in significant performance sub-optimality.

Vision-based controllers, however, offer a much more flexible architecture, allowing researchers to seamlessly integrate diverse operational objectives and complex environmental constraints into a single framework. As computational hardware continues to evolve, the prospect of robots planning their own trajectories has shifted from a theoretical ambition to a practical reality. The current frontier of the field is the realization of "on-the-fly" optimal trajectory generation. The ultimate goal is to develop systems capable of synthesizing real-time solutions that maximize efficiency and objective achievement while strictly adhering to the robot's inherent system dynamics, physical state limits, and hardware actuation boundaries. In the study presented in [12], a quadrotor achieves fully autonomous ball tracking by

relying strictly on onboard camera sensing and external computation (external PC for image processing). The system demonstrated its robustness in real-world experiments, maintaining successful tracking even while executing maneuvers with high angular rates. However, the research highlights a critical bottleneck: the aggressiveness of the flight is primarily constrained by the fidelity of the visual data, with motion blur serving as the leading factor that limits higher-speed performance. Building on the precedents set by these earlier studies, this chapter details the development of an accurate autonomous tracking system for UAVs in indoor environments using a single monocular RGB camera. Our framework integrates a YOLOv8 object detector with a Kalman filter (KF)-based tracker to facilitate robust, real-time target acquisition. While the system is designed with modularity in mind, this specific study focuses on a ball as the primary target. The architecture comprises of three integrated modules—object detection, target tracking, and Proportional-Integral-Derivative (PID) control—and was implemented on a DJI Tello drone to evaluate end-to-end performance.

The rest of this chapter is structured as follows. Section 5.2 describes the UAV platform and the implementation of the vision system. Section 5.3 outlines the software setup and the integration of YOLOv8. Section 5.4 presents the target detection framework, while Section 5.5 details the implementation of the autonomous tracking system. Section 5.6 provides an in-depth explanation of the control strategy. Section 5.7 discusses and analyzes the experimental results. Finally, Section 5.8 concludes the chapter with final remarks.

5.2 UAV Platform and Vision System Implementation

5.2.1 The UAV Platform (DJI Tello)

In this work, the aerial platform selected for validation is the DJI Tello quadrotor, shown in Figure 5.1(a). Manufactured by Ryze Robotics and powered by DJI flight-control technology, the Tello is a lightweight and compact UAV designed for safe and accessible experimentation [149]. It is equipped with a front-facing monocular RGB camera capable of capturing 720p video at 30 frames per second. To improve the stability of the visual stream under airframe vibrations, the camera supports Electronic Image Stabilization (EIS). From a safety and operational perspective, the Tello provides several built-in mechanisms that make it suitable for experimental validation. The maximum flight time is approximately 13

Table 5.1: Materials and computer specifications used in this work.

Material	**Specification**
CPU	Intel(R) Core(TM) i5-10300H (2.50 GHz)
RAM	16 GB (8 GB + 8 GB)
Graphics Card	NVIDIA GeForce GTX 1650 (4 GB); Intel(R) UHD Graphics (128 MB)
Connectivity	Wi-Fi 2.4 GHz
Drone	DJI Tello (Mini Tello)
Webcam	720p
OS	Windows 10

minutes, and a failsafe protection mode triggers an automatic safe landing in the event of communication loss. In addition, the motors are designed to stop if the drone experiences a significant collision, reducing the risk of damage during testing. The materials used in this work and their key specifications are summarized in Table 5.1.

Although the proposed method is platform agnostic, the Tello was chosen due to its programmability and the availability of a robust software development kit (SDK), which facilitates the integration of custom computer-vision modules [150]. The system follows a closed-loop architecture in which the drone communicates with a ground station (laptop) over Wi-Fi link with an effective range of approximately 100 m. The ground station receives the video stream, resizes incoming frames to 640×480 for efficient processing, executes the detection and tracking pipeline, and transmits control commands back to the UAV in real time.

5.2.2 The Moving Target

The experimental scenario considers a single UAV tasked with detecting and tracking a moving spherical target using onboard visual feedback only. As shown in Figure 5.1(b), a blue ball is used as the dynamic target. The objective is for the UAV to autonomously follow the ball along a predefined trajectory by estimating its relative position from the monocular camera stream and maintaining a stable visual lock throughout the sequence.



(a)



(b)

Figure 5.1: UAV and Ball Object. (a) DJI Tello quadrotor equipped with an onboard camera; (b) spherical ball target.

5.3 Software Setup and YOLOv8 Integration

The software pipeline is developed in Python and communicates with the DJI Tello via the `DJITelloPy` library, which provides access to the live video stream, basic telemetry, and high-level flight control commands. The UAV streams onboard video to a ground station over a Wi-Fi connection, where each frame is resized to 640×480 and processed by a YOLOv8 detector to estimate a 2D bounding box around the target. The bounding-box centroid is then used as the measurement input to a Kalman filter, which attenuates detection noise and maintains target-state prediction during short detection dropouts. The filtered target position is converted into image-plane tracking errors with respect to the frame center and supplied to a PID controller that outputs velocity and yaw-rate commands. These commands are sent back to the drone in real time, thereby closing the perception estimation control loop. To ensure low-latency inference, all experiments are executed on a laptop equipped with an NVIDIA GeForce GTX 1650 GPU. The DJI Tello also supports Python-based programmability: through `DJITelloPy`, sensor data can be accessed and control instructions can be issued directly. The Software materials used in this study are summarized in Table 5.2.

The `DJITelloPy` library enables rapid prototyping of a wide range of computer-vision applications, such as face tracking, object detection, and hand-gesture recognition. In this

Table 5.2: Materials used in this study.

Material	Name	Version
Programming language	Python	3.9
Library	DJITelloPy	2.5.0
IDE	PyCharm	22

work, however, we focus specifically on implementing real-time object detection and tracking.

To establish communication between the UAV and the application, we rely on the DJI Tello SDK, which provides a programming interface for automating the flight of the Tello EDU through Python callable commands that offer direct access to the drone. In this work, the SDK instructions are wrapped into Python functions, forming a higher level control layer that can be seamlessly integrated into standard program logic such as loops, conditional statements, and common data structures (e.g., lists and dictionaries). The resulting communication scheme between the Tello EDU and the application is illustrated in Fig. 5.2.

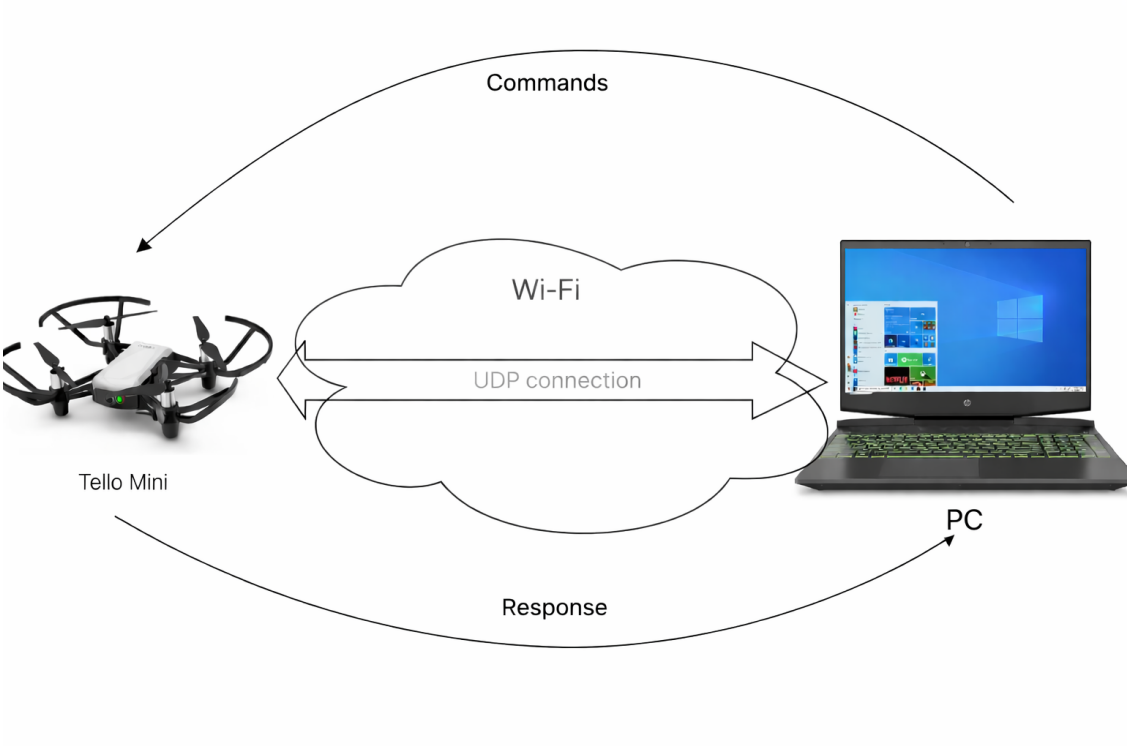


Figure 5.2: General communication architecture between a Tello UAV and a PC.

5.4 Target Detection Framework

Object detection is a fundamental computer vision task that aims to identify and localize objects within digital images or video streams, enabling the recognition of diverse categories (e.g., laptops, keyboards, cups, and balls). A variety of detection frameworks have been proposed, including R-CNN, RetinaNet, and the Single Shot MultiBox Detector (SSD). Although these methods address important challenges such as limited data and effective model design, they often rely on multi-stage pipelines rather than a single end-to-end inference pass. In contrast, the YOLO (You Only Look Once) family has become widely adopted due to its real-time capability and strong accuracy, frequently outperforming earlier approaches in both speed and detection performance [151]. Building on these advantages, we employ YOLOv8 to enable UAV target tracking. The detector outputs a 2D bounding box in the image plane, which is then used as the measurement input for state estimation [13]. In our experiments, the ball is detected in aerial imagery using YOLOv8, whose architecture is illustrated in Figure 5.3 [152]. Prior to inference, all incoming frames are resized to 640×480, regardless of their original resolution. YOLOv8 was selected for this task because it offers an excellent trade-off between accuracy and computational efficiency, achieving high mean average precision (mAP) and low inference latency on the COCO benchmark [153].

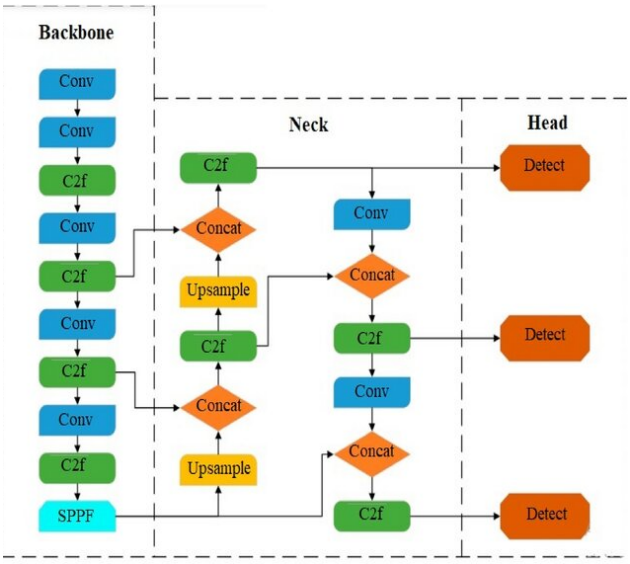


Figure 5.3: Block diagram of the YOLOv8 architecture.

Table 5.3: A Comparative Analysis of YOLO Model Performance.

Model	**mAP**	**FPS**	**Dataset**
YOLOv1	63.4	45	VOC
YOLOv1-Tiny	57.1	155	VOC
YOLOv2	78.6	40	VOC
YOLOv2-Tiny	57.1	244	VOC
YOLOv3	55.3	20	COCO
YOLOv3-Tiny	33.1	220	COCO
YOLOv4	65.7	60	COCO
YOLOv4-Tiny	40.2	400	COCO
YOLOv8n	37.3	500	COCO
YOLOv8x	53.9	100	COCO

Table 5.3 compares the inference speed and detection performance of different YOLO variants. The results highlight the expected trade-off between accuracy and runtime, lighter models offer lower latency, whereas larger models generally achieve higher precision. In particular, YOLOv8n ranks among the fastest real-time detectors and still provides strong accuracy relative to its compact size. By contrast, the larger YOLOv8x model attains substantially higher accuracy, with an approximately 16 mAP points higher, making it a compelling choice for applications where detection precision is critical while maintaining real-time operation.

5.5 Implementation of the Autonomous Tracking System

In general, target tracking systems are designed to balance simplicity and robustness. In this work, the initial pixel coordinates of the target ball, obtained by the detection algorithm from the drone's aerial camera, together with the flight motion parameters, are provided as inputs to the visual tracking algorithm. This approach simplifies the overall pipeline while preserving robust tracking performance. Table 5.4 summarizes the key dif-

ferences between YOLOv8-based and YOLOv5-based UAV object-tracking methodologies, highlighting variations in detection design, tracking integration, computational demands, and control coupling.

Table 5.4: Comparison of YOLOv8-Based and YOLOv5-Based Methodologies for UAV Object Tracking.

Aspect	**YOLOv8-Based Methodology**	**YOLOv5-Based Methodology**
Detection framework	YOLOv8, one-stage detector with anchor-free design (Gharsa et al.) [13]	YOLOv5, one-stage detector with anchor-based design (Khanam & Hussain.) [154]
Anchor mechanism	Anchor-free bounding box regression	Predefined anchor boxes requiring dataset-specific tuning
Target type	Single moving object (ball)	Single or multiple objects, depending on tracker integration
Tracking strategy	Detection with appearance-based re-detection with Kalman filter (no external tracker)	Detection combined with Kalman filter, SORT, or DeepSORT [155]
Position estimation	Image-plane error (2D pixel displacement)	Image-plane or state-estimated position
Control integration	Direct coupling with image-based PID controller (Gharsa et al.)	Often indirect, via estimated states or predicted trajectories
External sensors	None (no GPS, no motion capture)	Often combined with GPS or additional estimators
Computational complexity	Low–moderate (single detector, no tracker)	Moderate–high (detector + tracker + filters)
Embedded suitability	Optimized for low-cost onboard hardware	Efficient but limited by additional tracking modules
Robustness to scale changes	High (anchor-free detection)	Moderate (anchor sensitivity to scale variation)

Aspect	YOLOv8-Based Methodology	YOLOv5-Based Methodology
Robustness to illumination	High, learned feature representation	Moderate, depends on training data
System complexity	Simple, tightly integrated pipeline	More modular but architecturally complex
Typical use cases	Autonomous UAV tracking, robotics, real-time control	Multi-object tracking, surveillance, analytics

The ground station laptop performs real-time target detection and maintains continuous tracking until the mission is completed. The system relies on a single onboard camera to infer the target's position and follow it along its trajectory, the UAV streams the live video feed to the laptop, where YOLOv8x detects the target and encloses it within a 2D bounding box parameterized by (b_x, b_y, b_w, b_h). Here, b_x and b_y denote the top-left corner of the bounding box, while b_w and b_h represent its width and height, respectively. These bounding-box measurements are then forwarded to the tracking module, where a Kalman filter (KF) [156] is used for state estimation based tracking, refining noisy detections and predicting the target's future location [13]. From each bounding box, the ball's image-plane centroid $(\hat{x}, \hat{y})$ is computed and filtered (see Fig. 5.4), providing the primary reference for guidance. Accurate tracking is achieved by minimizing the alignment error between the estimated centroid $(\hat{x}, \hat{y})$ and the center of the camera frame (x_c, y_c). Moreover, the KF prediction step improves robustness during brief detection dropouts by continuing to provide a plausible target location, enabling the UAV to maintain stable tracking behavior.

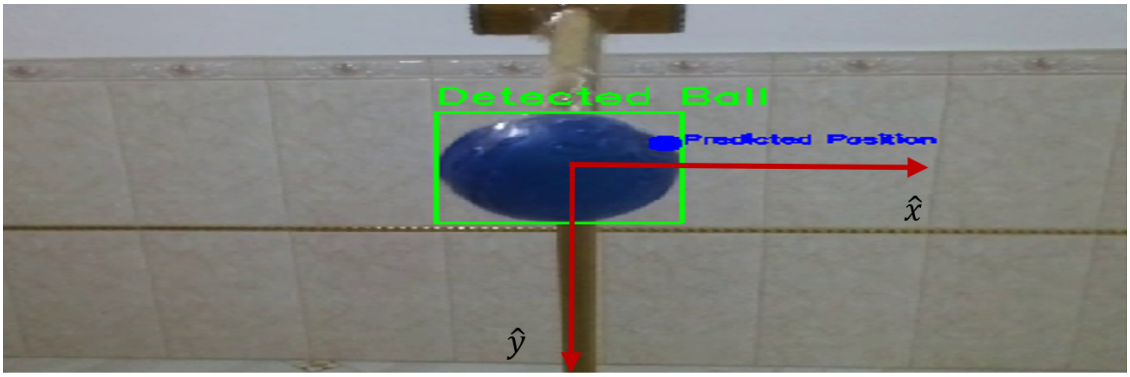


Figure 5.4: Coordinate system definition for ball tracking.

5.6 Flight Control Setup

This section presents the Proportional-Integral-Derivative (PID) controller used to steer the UAV, enabling smooth and safe ball tracking behavior. Using the state estimates provided by the Kalman filter, the drone autonomously adjusts its motion to follow the ball's trajectory. An overview of the control architecture is shown in Fig. 5.5.

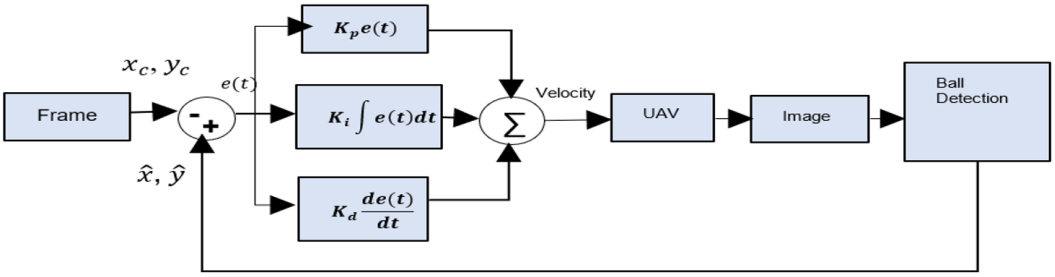


Figure 5.5: PID Controller for UAV.

From the camera stream, the Kalman filter provides the ball's estimated image-plane center $(\hat{x}, \hat{y})$. The tracking errors relative to the image center (x_c, y_c) are computed as

$$e_x = \hat{x} - x_c \tag{5.1}$$

$$e_y = \hat{y} - y_c \tag{5.2}$$

For effective tracking, e_x and e_y should remain at or near zero. These errors are then used as inputs to the PID control law

$$u(t) = K_p\, e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de(t)}{dt}, \tag{5.3}$$

where K_p, K_i, and K_d are the proportional, integral, and derivative gains of the PID controller, respectively. In this work, the drone uses three control parameters: roll, altitude, and pitch. Roll governs the drone's side-to-side motion, altitude regulates its vertical movement, and pitch controls its forward and backward motion. Figure 5.6 shows the basic flight maneuvers of the UAV.

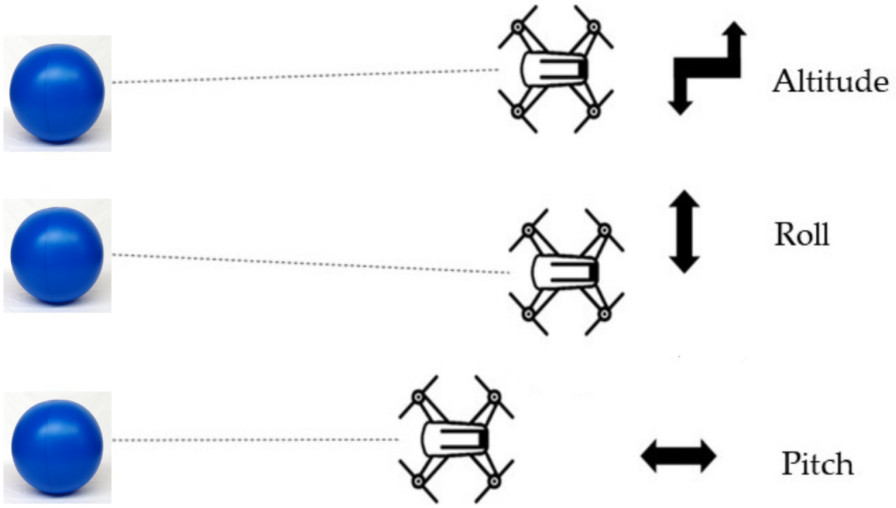
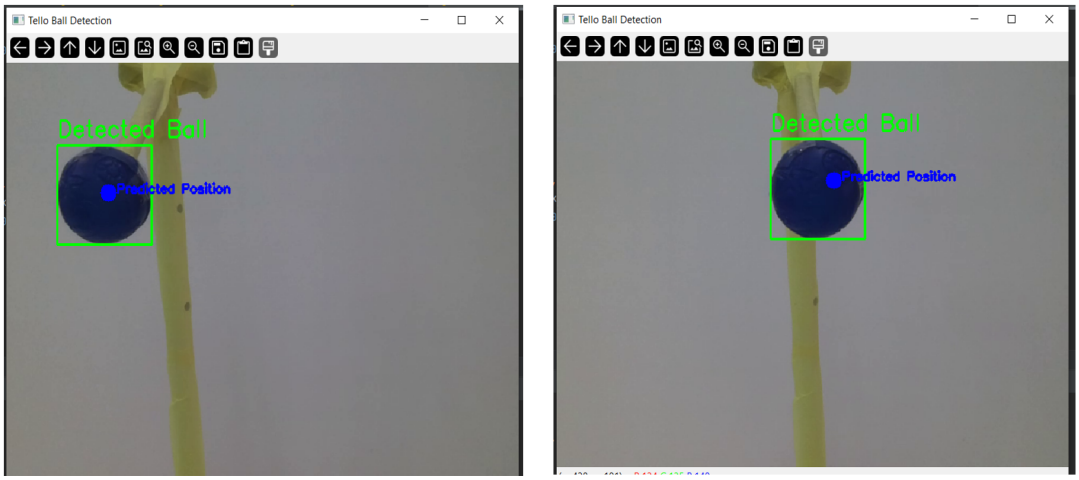


Figure 5.6: Fundamental UAV Maneuvers.

The error along the X-axis is used to control the drone's **roll**, which governs lateral motion. When the tracked object shifts to the left or right, the drone adjusts its roll to maintain alignment with the target. Similarly, **pitch** controls forward and backward motion. By comparing the actual distance between the drone and the target with the desired distance, a distance error can be computed and used to regulate the drone's forward or backward movement. Finally, the error along the Y-axis is obtained by subtracting the tracked object's Y-coordinate from the Y-coordinate of the screen center. This error is then used to control the drone's **altitude**, enabling vertical ascent or descent as needed [79].

5.7 Experimental Results

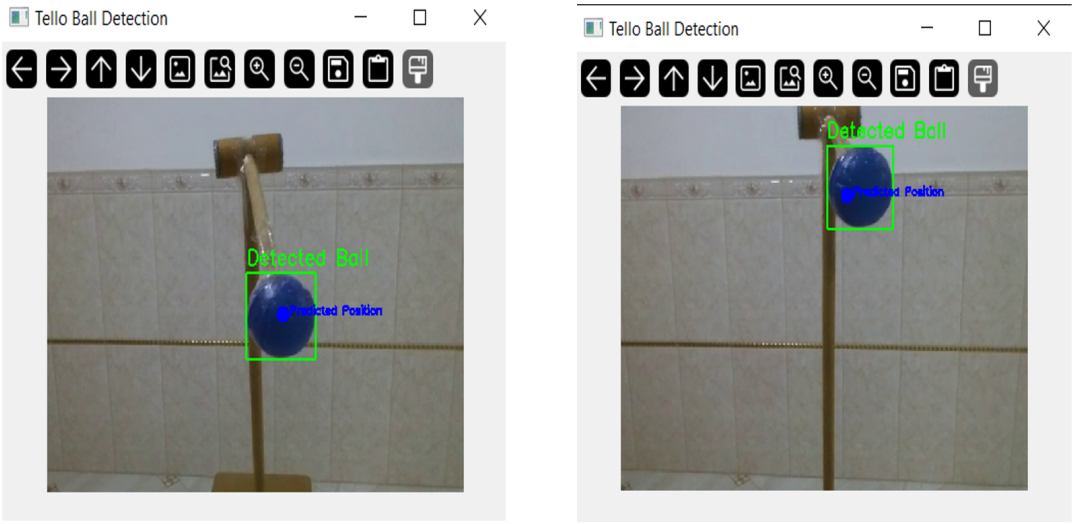
This section presents a vision-based guidance strategy combined with a PID controller for target detection and tracking. The approach is validated on a DJI Tello drone linked to a laptop equipped with an NVIDIA GeForce GTX 1650 GPU, where robustness and reliability are assessed. Experimental results confirm stable performance and successful tracking.



(a) **(b)**

Figure 5.7: Real-time autonomous ball detection and tracking using onboard vision.

Following the initial detection of the target ball, the UAV enters an active tracking phase in which it continuously adjusts its flight path to maintain the target at the center of the camera's field of view while simultaneously advancing toward it. As shown in Figure 5.7a, the ball initially appears in the upper-left quadrant of the image, indicating a positional error along both the horizontal and vertical visual axes. Once the autonomous tracking sequence is activated, the control system computes the offset between the target's detected pixel coordinates and the desired image center, and then processes this error in real-time to generate corrective roll, pitch, and altitude commands. As a result, the UAV reorients itself and refines its trajectory to shift the ball from the image periphery to a stable centered position, as illustrated in Figure 5.7b. This centering action is performed concurrently with the drone's forward motion, allowing the UAV to approach the target with high spatial precision while reducing the likelihood of losing visual contact during rapid motion. The transition from Figure 5.7a to Figure 5.7b therefore provides clear visual evidence of the effectiveness of the proposed end-to-end system, demonstrating not only its ability to detect the blue ball in a complex environment but also to convert visual information into accurate physical movement, thereby keeping the target within the optimal sensing region required for the autonomous tracking maneuvers.



(a) **(b)**

Figure 5.8: UAV vertical (up/down) motion during ball tracking.

Figure 5.8 illustrates the robustness of the proposed system in vertical state estimation and tracking by highlighting the UAV's ability to maintain a stable visual lock during changes in the height of the target ball. In this experiment, the vertical component of the control loop is emphasized to show how the drone responds to target motion along the image-plane Y-axis. As shown in Figure 5.8a, the ball initially appears near the lower region of the camera frame, producing a noticeable vertical displacement error with respect to the image center. This pixel-based error is detected by the vision system and transmitted in real time to the PID controller, which adjusts the UAV's throttle to initiate a controlled ascent. As the ball moves upward, the UAV follows this variation and continuously corrects its altitude to maintain alignment with the target. The outcome of this process is shown in Figure 5.8b, where the ball is successfully re-centered in the image. This transition demonstrates that the UAV can effectively track vertical target motion, preserve spatial alignment during rapid altitude changes, and reduce the risk of target loss, which is essential for ensuring reliable autonomous detection and tracking performance. Figure 5.9 quantitatively evaluates the performance of the developed three-axis visual servoing loop by presenting the positional tracking errors along the X, Y, and Z axes, expressed in pixels, over time during key dynamic maneuvers. A detailed examination of the three subplots clearly demonstrates the effectiveness of the proposed controller. As observed in the figure, each maneuver initiation

or external disturbance produces a brief increase in tracking error across all axes; however, the controller responds rapidly and efficiently to suppress these deviations. In the X-axis plot, for instance, the error reaches a peak of approximately -0.01 pixels before being quickly driven back toward zero, indicating a fast recovery of tracking stability. A similarly strong response is evident in the Y- and Z-axis plots, where the error magnitudes remain confined within very small bounds, specifically below 0.005 pixels for the Y-axis and under 0.003 pixels for the Z-axis, and are rapidly attenuated after each disturbance. The short duration of these transient responses is particularly important, as it confirms the controller's ability to preserve accurate target alignment under dynamic conditions. Overall, the results shown in Figure 5.9 provide strong visual and quantitative evidence that the proposed control strategy achieves stable and high-precision visual tracking while effectively compensating for disturbances encountered during autonomous UAV operation. Our approach is designed to achieve high accuracy while maintaining real-time performance. We employ the relatively large YOLOv8x model for the computationally demanding initial detection stage, ensuring the blue ball is localized with high precision. For subsequent frame-to-frame tracking, we use a Kalman filter, which is well suited for computationally efficient real-time state estimation. When integrated with the DJI Tello's onboard camera, this hybrid pipeline forms an effective and engaging tracking system that demonstrates strong performance on a low-cost UAV.

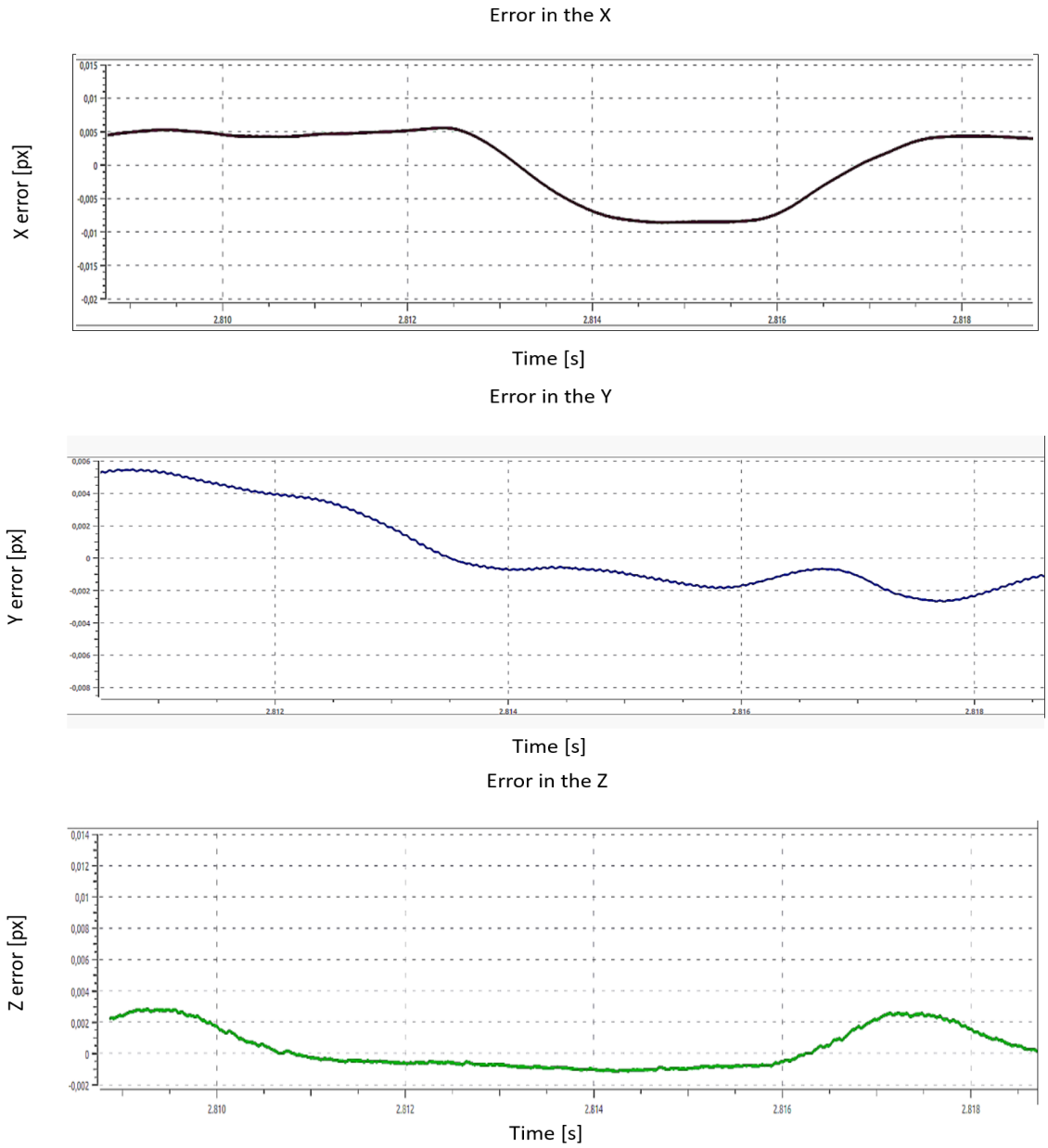


Figure 5.9: Tracking error along the $X-$, $Y-$, and $Z-$axes versus time.

5.8 Conclusion

This chapter has detailed the comprehensive design and successful implementation of an autonomous, vision-based aerial tracking system integrated into a UAV platform. By leveraging a synergistic pipeline of detection, estimation, and control, the system achieves robust performance in dynamic indoor environments. The core of the proposed architecture relies on the YOLOv8 framework, which proved highly effective for real-time target acquisition, maintaining high detection confidence even when the target ball occupied a minimal pixel area. To bridge the gap between raw detection and smooth navigation, a Kalman Filter (KF)-based tracking algorithm was utilized to provide continuous state estimation and trajectory prediction, ensuring tracking persistence during rapid maneuvers. Finally, the transition from visual perception to physical action was managed by a Proportional-Integral-Derivative (PID) control module. By processing the spatial error between the target's estimated coordinates and the image center, the PID controller effectively modulated the drone's attitude governing roll, pitch, and altitude to maintain a stable visual lock. Extensive flight trials have empirically validated the system's efficacy, demonstrating that the combination of high-speed YOLOv8 inference, predictive Kalman filtering, and reactive PID control results in a resilient, end-to-end autonomous tracking solution capable of mitigating external disturbances and maintaining stable flight.

Chapter 6

Autonomous Obstacle Avoidance for UAVs in Unknown Environments

6.1 Introduction

Achieving robust and reliable outdoor navigation for autonomous platforms, such as mini unmanned aerial vehicles (MAVs), remains a significant and unresolved challenge within the field of robotics. In unknown and complex environments, robots must be equipped with powerful control systems and accurate perception sensors [157]. The autonomous navigation of aerial vehicle platforms in both static and dynamic landscapes constitutes a cornerstone objective in the field of robotics [158]. Successfully executing such tasks requires the seamless integration of a multi-faceted architectural triad encompassing environmental perception, robust control laws, and precise localization. The complexity of these requirements is significantly amplified when operating within unstructured and previously unmapped environments, with dense urban settings presenting the most rigorous test of system resilience. In these high-stakes scenarios, the autonomous agent must transcend basic path following; it is tasked with the simultaneous optimization of mission efficiency and reactive collision avoidance, ensuring reliable adaptation to environmental shifts while adhering to strict safety constraints. Successfully navigating while avoiding obstacles is a fundamental requirement for enabling many robotic applications, including surveillance, construction monitoring, delivery, and emergency response [159, 72, 160]. To perform the autonomous navigation task with high performance, we employ an efficient optical flow based method.

The primary objective of this work is to realize fully autonomous obstacle avoidance

within unmapped and unstructured environments using a monocular RGB camera as the sole sensory input. By leveraging optical flow as the foundational visual signal, the proposed system extracts critical environmental cues and estimates obstacle proximity to facilitate real-time, reactive motion control.

The remainder of this chapter is structured as follows: Section 6.2 delineates the proposed system architecture and framework. Section 6.3 details the methodology for optical-flow based obstacle detection, while Section 6.4 outlines the underlying control strategy. simulation results are presented in Section 6.5, followed by a comprehensive discussion of the findings in Section 6.6. Finally, Section 6.7 provides concluding remarks and suggests potential avenues for future work.

6.2 Proposed System Framework

The proposed system framework, illustrated in Figure 6.1, establishes the connection between the simulation environment shown in figure 6.1(a) and the perception action loop presented in figure 6.1(b). It adopts a dual-rate synchronization strategy to balance the high-speed requirements of flight stabilization with the computational demands of visual perception. In this architecture, the high-frequency control loop is responsible for low level motor stabilization, while the lower-frequency perception pipeline processes visual information from the environment. To achieve autonomous environmental awareness without relying on heavy or power intensive active sensors such as LiDAR, the system uses the Lucas-Kanade (LK) algorithm to extract optical flow from monocular RGB image sequences [161]. By tracking the apparent displacement of brightness patterns between consecutive frames, the algorithm generates motion vectors that provide qualitative information about obstacle proximity and relative direction. As the UAV approaches an object, the magnitude of the observed optical flow increases, offering a direct visual cue for potential collision risk. These motion vectors are then processed to compute an average optical flow signal, which serves as an indicator of the threat level in the UAV's forward path. This signal is subsequently fed into a set of PID controllers that translate the perceived visual motion into corrective flight commands, including roll, pitch, and yaw adjustments, enabling the UAV to avoid high-flow regions and maintain safe clearance from obstacles in real time. Overall, Figure 6.1 demonstrates how raw spatial data from the Gazebo simulation environment is transformed into a sparse

feature-based representation that supports reactive obstacle avoidance, allowing the UAV to perceive and respond to its surroundings efficiently without requiring computationally expensive 3D reconstruction or explicit depth estimation [162].

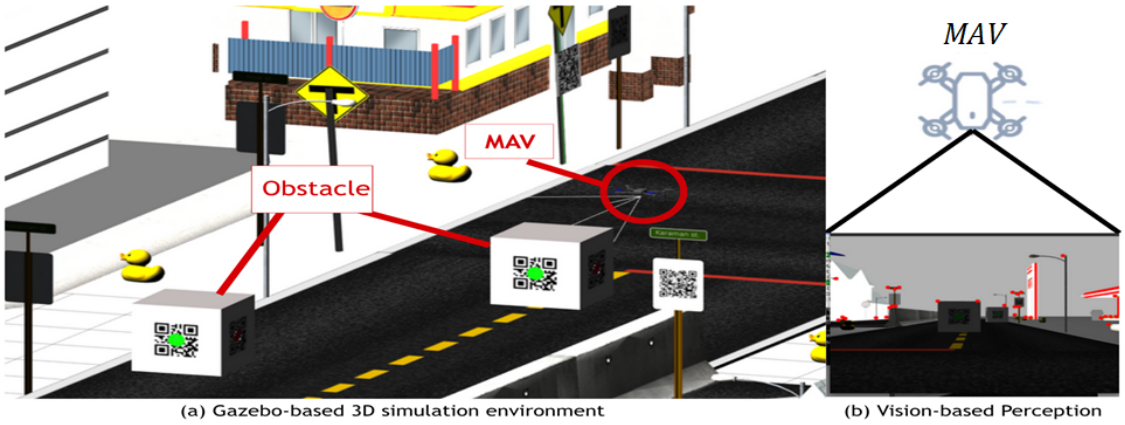


Figure 6.1: Integrated Simulation Framework and Vision-Based Perception Pipeline.

6.3 Obstacle Avoidance by Optical Flow Method

Inspired by the human visual system, the proposed algorithm detects obstacles directly along the MAV flight path by relying on image-motion cues rather than computationally intensive 3D reconstruction or explicit depth estimation. Instead of building a geometric model of the environment, the method estimates the image-plane velocities of pixels through optical flow computation [72]. Because object distance influences perspective projection, the number of visible feature points, their spatial distribution on the image plane, and their apparent size vary as a function of range. To estimate this motion, the Lucas-Kanade (LK) method is adopted [163], as it is less sensitive to noise than pointwise approaches and, by integrating information over a local neighborhood, alleviates the aperture ambiguity commonly associated with optical-flow estimation [9]. Let $I_1, I_2 \in \mathbb{R}^{m_u \times n_v}$ denote two grayscale images acquired at two consecutive time instants. For a pixel $p = (u_i, v_i)^\top$, its gray level intensity is defined as $G_p = I(u_i, v_i)$. Accordingly, for two corresponding pixels in successive frames, the intensities can be written as

$$G_{p_1} = I_1(u_1, v_1), \qquad G_{p_2} = I_2(u_2, v_2) \tag{6.1}$$

Given an image point $p_1 \in I_1$, the goal is to identify the corresponding point $p_2 \in I_2$ under the assumption of approximate brightness constancy, such that

$$G_{p_1} \approx G_{p_2} \tag{6.2}$$

The relationship between the matched pixels p_1 and p_2 is expressed as

$$p_2 = p_1 + \mathbf{r} = \begin{bmatrix} u_1 \\ v_1 \end{bmatrix} + \begin{bmatrix} r_u \\ r_v \end{bmatrix} \tag{6.3}$$

where $\mathbf{r} = [r_u \ r_v]^\top$ represents the image displacement vector, i.e., the optical flow. This displacement is estimated by minimizing the following residual function [164]:

$$\varepsilon(\mathbf{r}) = \sum_{u=u_1-w_u}^{u_1+w_u} \sum_{v=v_1-w_v}^{v_1+w_v} [I_1(u,v) - I_2(u+r_u,\, v+r_v)]^2 \tag{6.4}$$

where $w_u, w_v \in \mathbb{Z}$ define the half window sizes along the u and v directions, respectively. The Lucas-Kanade optical-flow method employs an adaptive integration window, which improves robustness to larger pixel displacements while also acting as a low-pass filter that reduces the effect of image noise [165].

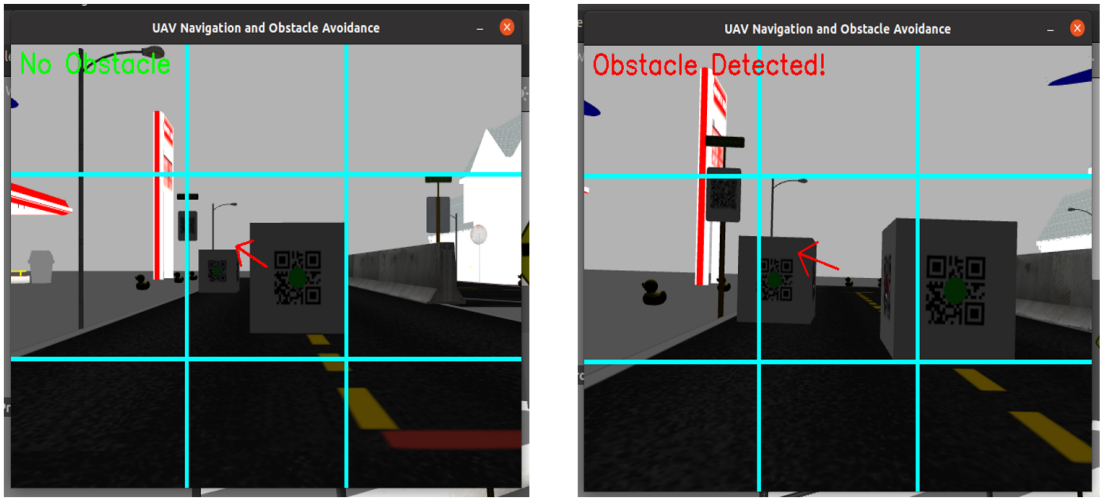


Figure 6.2: Implementation of the Obstacle Detection and Avoidance (ODA) system. (a) No obstacle detected in the flight path. (b) Active obstacle detection with visual warning and vector analysis.

Figure 6.2 illustrates the real-time operation of the vision-based perception and obstacle avoidance system in the Gazebo simulation environment. To provide directional awareness, the monocular RGB image is divided into multiple observation regions corresponding to the main navigation directions, namely left, right, up, down, and forward (center). For each region, the algorithm computes the mean optical flow magnitude, which serves as an indicator of potential obstacle presence and proximity. This spatial partitioning enables the system to localize high-risk areas in the visual field and generate appropriate steering commands. In Figure 6.2(a), the UAV operates in a nominal navigation state, where visible obstacles remain sufficiently distant, resulting in low optical flow values across all regions. Since these regional flow magnitudes stay below the predefined safety threshold, the system reports a safe condition and maintains its current heading and velocity. In contrast, Figure 6.2(b) shows the reactive detection phase as the UAV approaches an obstacle. As the distance to the obstacle decreases, the apparent visual expansion increases, causing the optical flow magnitude in the affected regions to exceed the safety threshold and trigger obstacle detection. The dominant flow vector, visible in the image, indicates both the location and the direction of the perceived threat, allowing the controller to identify safer regions with lower flow intensity. Based on this information, the control system generates corrective maneuvers, such as roll or yaw

adjustments, to steer the UAV away from the obstacle and ensure safe navigation without requiring explicit 3D reconstruction or depth estimation [74].

6.4 Proposed Control System for Obstacle Avoidance

The Proportional-Integral-Derivative (PID) controller is adopted in this work because of its computational efficiency, simple structure, and well established reliability in real-time control applications. By incorporating this classical control strategy, the proposed system effectively links high level visual perception, represented by the optical flow information, with low level flight actuation. A properly tuned PID controller is particularly important in obstacle avoidance scenarios, where both fast response and stable motion are required. In this context, the proportional term generates the immediate corrective action needed to steer the UAV away from potential obstacles, the integral term compensates for accumulated steady-state errors that may lead to drift, and the derivative term predicts the system's rate of change to reduce oscillations and overshoot. The combined action of these three components ensures that, once the Lucas-Kanade algorithm indicates a potential collision, the MAV can perform a rapid, smooth, and accurate avoidance maneuver rather than an unstable or excessive correction [166, 38]. As illustrated in the block diagram in Figure 6.3, the controller operates within a closed loop feedback structure, continuously updating the UAV attitude according to the residual error between the measured optical flow and the predefined safety reference.

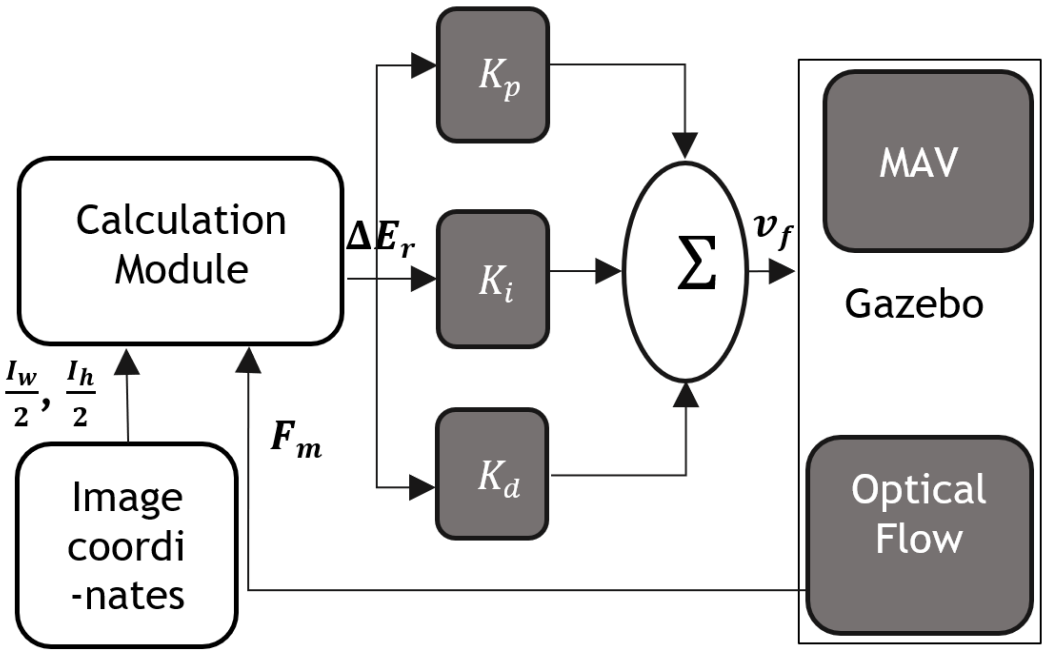


Figure 6.3: Block Diagram of the Integrated Control Strategy.

The proposed control architecture leverages a PID scheme, a selection motivated by its structural simplicity, computational efficiency, and the maturity of its tuning methodologies in real-world robotic applications [19]. In this framework, the controller serves as the primary bridge between the perception layer and the motion actuators, translating processed visual data into actionable velocity commands. By carefully modulating the controller's three fundamental gains K_p, K_i, and K_d the system is optimized to achieve a balance between high-frequency disturbance rejection and stable, overshoot-free avoidance maneuvers [12].

Mathematically, the control signal $u(t)$, which defines the commanded forward velocity v_f of the Micro Aerial Vehicle (MAV), is formulated as a function of the time varying error signal $e(t)$:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de(t)}{dt} \tag{6.5}$$

The operational logic of the controller is driven by two primary visual inputs: the image center reference, representing the desired setpoint, and the mean optical-flow value, which serves as the feedback signal. By calculating the discrepancy between these two values,

the system evaluates the instantaneous navigation error and its rate of change, denoted as ΔE_r. Unlike traditional mapping approaches that require a static coordinate system or global localization, this controller assumes a constant nominal cruise speed for the MAV. It then imposes a corrective velocity component based on the perceived visual approach effect of obstacles. This end-to-end procedure, summarized in Algorithm 4, represents a purely reactive navigation strategy. By directly mapping pixel level motion information to velocity commands, the system effectively bypasses the need for high-latency 3D environment reconstruction or explicit path planning. The efficacy of this strategy was rigorously validated within a ROS-Gazebo simulation environment, confirming that the MAV can maintain high-speed flight while executing precise evasive maneuvers in unknown, unstructured spaces.

Algorithm 4 Optical Flow Guided Reactive Obstacle Avoidance for MAV

Require: RGB image stream I_t; Lucas-Kanade optical flow f_t; thresholds T_x, T_y, T_{det}

Initialize $\mathbf{v} \leftarrow (0, 0, 0)$
while ROS is running **do**
    Acquire I_t and compute optical flow f_t using Lucas-Kanade
    Compute regional mean flow (d_x, d_y) in the forward/center ROI
    $M \leftarrow \sqrt{d_x^2 + d_y^2}$
    obstacle_detected $\leftarrow (M > T_{\text{det}})$
    if *obstacle_detected* **then**
        $v_x \leftarrow 0$, $v_y \leftarrow 0$, $v_z \leftarrow 0$
        if $d_x > +T_x$ **then**
            $v_x \leftarrow$ *evade left*
        end if
        if $d_x < -T_x$ **then**
            $v_x \leftarrow$ *evade right*
        end if
        if $d_y > +T_y$ **then**
            $v_z \leftarrow$ *move down*
        end if
        if $d_y < -T_y$ **then**
            $v_z \leftarrow$ *move up*
        end if
    else
        $v_y \leftarrow$ *cruise forward*
    end if
    Saturate (v_x, v_y, v_z) to safety limits
    Publish $\mathbf{v} = (v_x, v_y, v_z)$
end while

The PID controller offers several important advantages, including structural simplicity, robustness in the presence of noisy measurements, and effective regulation of the MAV velocity. These characteristics make it particularly suitable for managing discontinuities in

optical-flow based pose estimates, which may arise from bundle adjustment inaccuracies or camera data association errors. When properly tuned, for example through suitable gain selection and derivative filtering, the PID controller can also accommodate moderate time delays while preserving a fast and responsive control action. As illustrated in Figure 6.3, the adopted loop shaping design provides a practical compromise between control performance and system robustness [167].

6.5 Simulation Results

This section presents the simulation-based validation carried out in the Gazebo environment to assess the effectiveness of the proposed navigation framework. To provide a rigorous evaluation of system robustness, the virtual environment was intentionally designed with a high level of structural complexity, including various architectural components such as buildings, vertical panels, and cubic obstacles. The simulation framework, implemented through the integration of ROS (Robot Operating System) and Gazebo, allows the navigation space to be represented in both two dimensional occupancy maps and three dimensional volumetric models, as shown in Figure 6.4. The test environment contains cubic obstacles of different sizes, ranging from 0.4 m to 2.0 m, thereby creating a challenging set of visual looming cues for the UAV. This diverse configuration makes it possible to evaluate both the perception capability of the optical-flow based approach and the responsiveness of the PID controller under complex navigation conditions.



Figure 6.4: Simulation environment for autonomous navigation.

This limitation reduced the sensitivity of the obstacle detector to small relative motions, especially when obstacles were located at greater distances. For this reason, all detections corresponding to camera obstacle distances greater than 1.5 m were excluded from the analysis.

The simulated platform consists of a micro aerial vehicle (MAV) equipped with four motorized propellers and an onboard virtual camera to support autonomous navigation. Based on the visual information captured by this camera, the proposed obstacle avoidance system generates steering commands that enable the MAV to avoid collisions effectively. In all evaluated scenarios, the optical-flow based PID controller demonstrated strong performance, allowing the MAV to navigate freely and autonomously while maintaining safe maneuvering behavior.

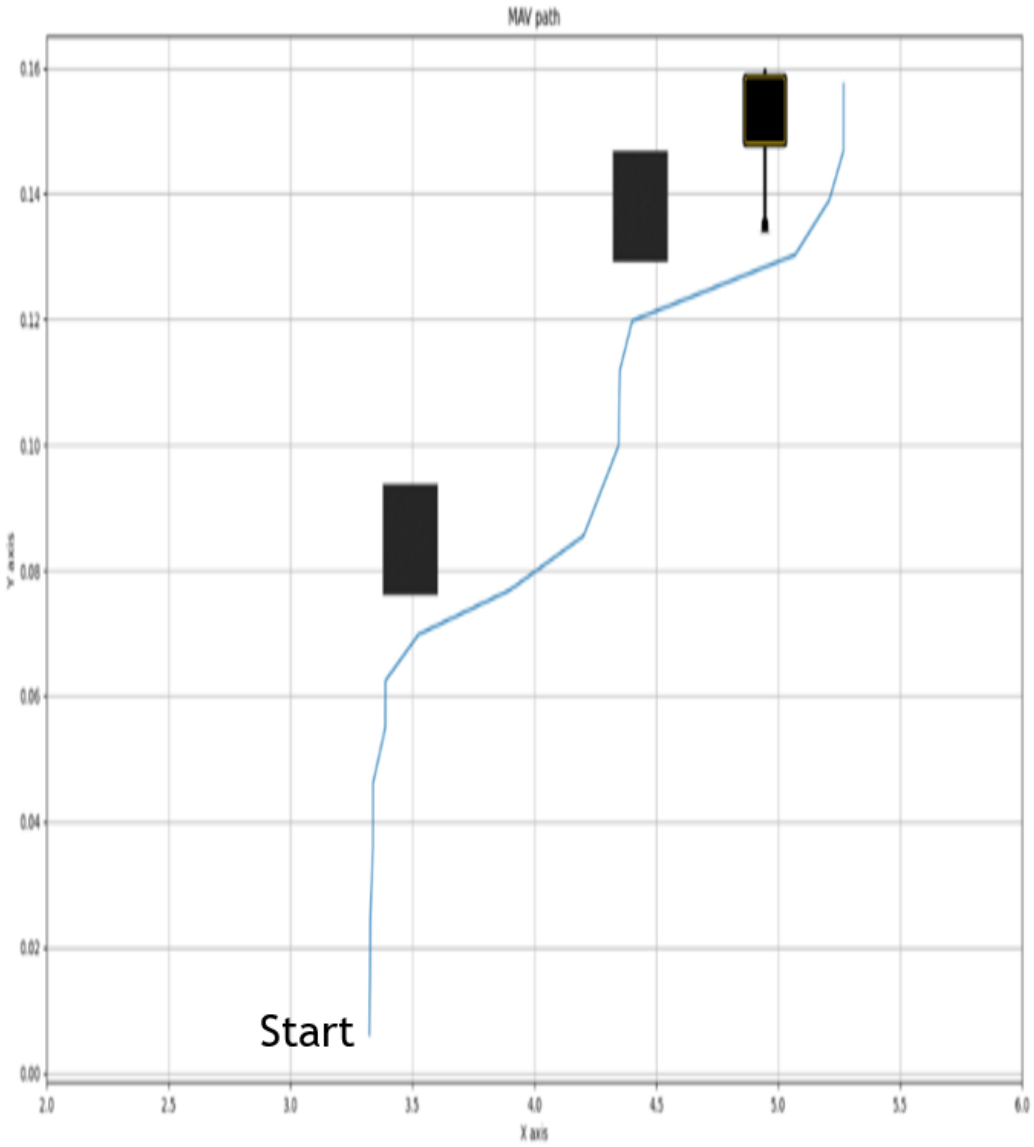


Figure 6.5: 2D autonomous MAV flight path: Scenario 1.

The two dimensional trajectory of the MAV for the first test case is presented in Figure 6.5, which illustrates the vehicle path from takeoff to mission completion. A critical phase of the flight occurs during the initial hovering stage, during which the optical-flow perception module stabilizes the observed obstacle position within the image plane. By analyzing the average motion vector during this phase, the system obtains an initial visual

reference before starting forward motion. It should be noted that vision-based localization and feature tracking are generally less reliable when the MAV is very close to the ground, mainly because the camera may observe fewer distinguishable features and may be affected by strong perspective distortion. For this reason, the takeoff sequence is initiated only after the ROS environment and controller are fully launched. The MAV then performs a gradual ascent to a predefined safety altitude of approximately 7 cm. This stage is executed in open-loop to move the vehicle from the ground condition to an altitude where visual features can be detected more reliably. Once sufficient visual information is available, the closed-loop reactive controller is activated to perform autonomous obstacle avoidance.

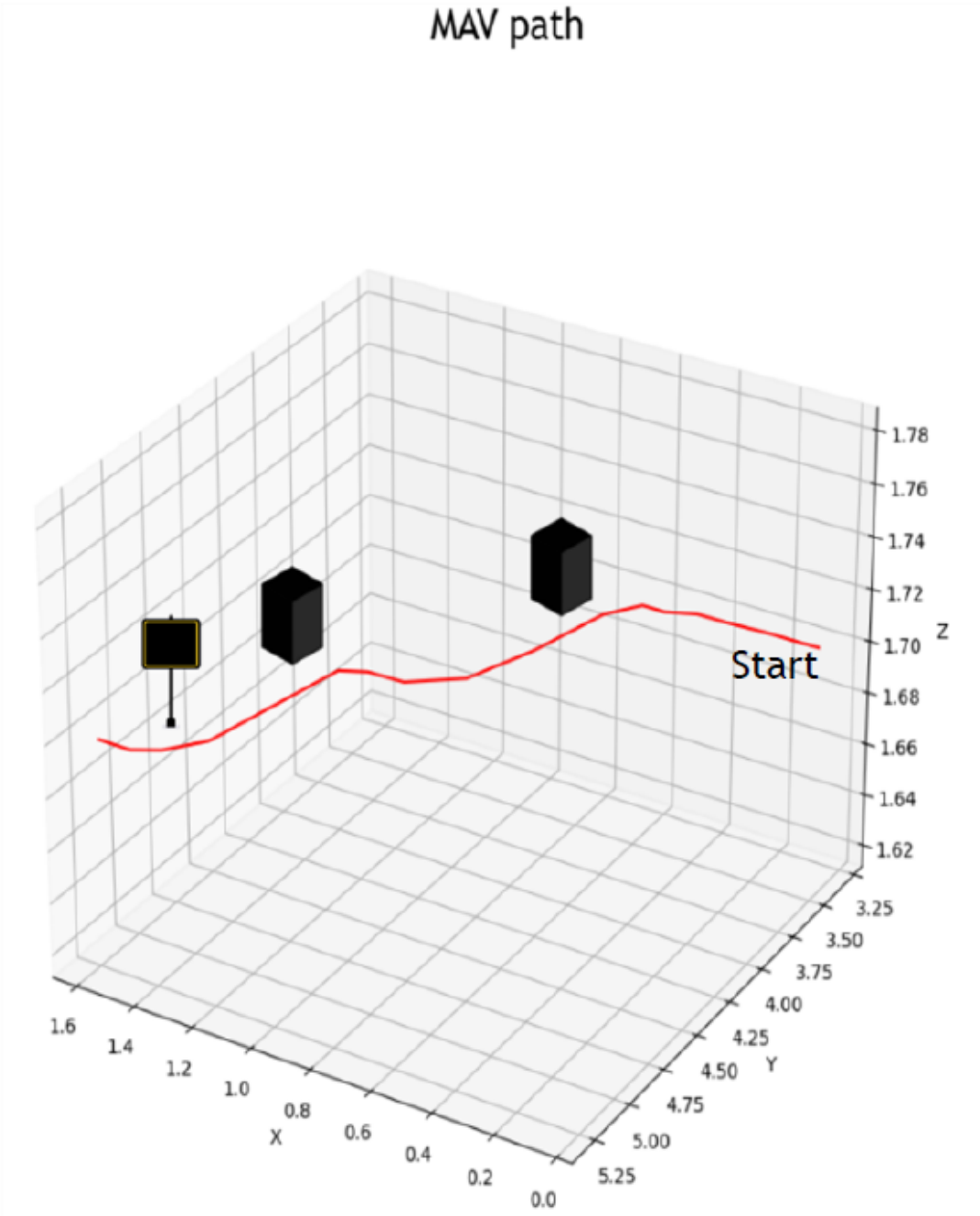


Figure 6.6: 3D autonomous flight trajectory: Scenario 1.

The three dimensional results for Scenario 1 are shown in Figure 6.6. In this case, the

MAV moves at a constant cruise velocity while navigating through an unknown environment. When a cubic obstacle is detected through visual expansion cues, the PID controller generates an appropriate evasive maneuver, allowing the MAV to avoid the obstacle and complete its mission successfully.

Figure 6.7 presents the results for Scenario 2 in the Gazebo simulator. Starting from the designated "Start" position, the MAV navigates through a complex environment containing different types of obstacles, including cubes, vertical panels, and duck models. In this scenario, the controller converts the continuous optical-flow measurements into a discrete set of motion actions, namely Left, Right, Up, Down, and Forward, in order to determine a safe navigation direction. This strategy enables collision-free navigation along a computationally efficient trajectory and demonstrates the adaptability of the proposed framework to different environmental configurations.

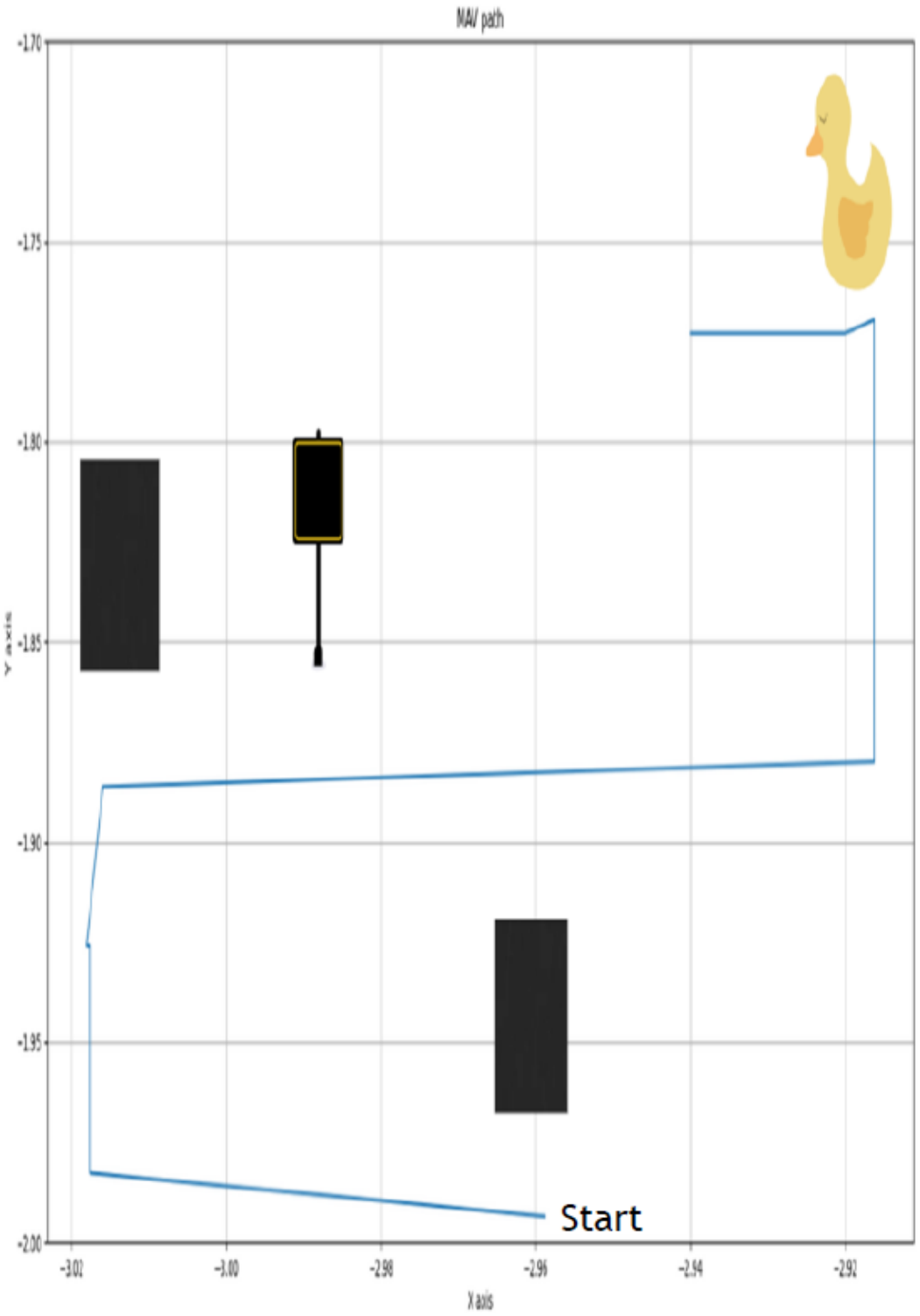


Figure 6.7: 2D autonomous MAV flight path: Scenario 2.

To facilitate the transition from the simulation developed algorithms to the flight control framework, a software-in-the-loop (SITL) configuration was integrated into the system. In this setup, the flight control unit (FCU) was interfaced with the simulated MAV, and all required ROS nodes were launched to ensure full communication between perception, control, and vehicle dynamics. As a result, the detection and control pipeline operated directly on the simulated platform, enabling the MAV to perform autonomous maneuvers under realistic flight-stack conditions. Furthermore, all control and perception parameters defined during the simulation phase were loaded into the simulated quadcopter to maintain consistency between algorithm development and system validation.

Figure 6.8 presents the 3D results for Scenario 2. After takeoff, the MAV initially detects an obstacle and performs a brief leftward sidestep to avoid it before resuming forward motion. It then encounters both a cube and a panel, prompting the controller to execute a rightward avoidance maneuver, after which the MAV continues along its path. Finally, when the duck obstacle is detected, the MAV shifts left once again to prevent collision and successfully completes the mission. These results demonstrate the effectiveness of the proposed perception and control framework in handling multiple sequential obstacles within a complex simulated environment.

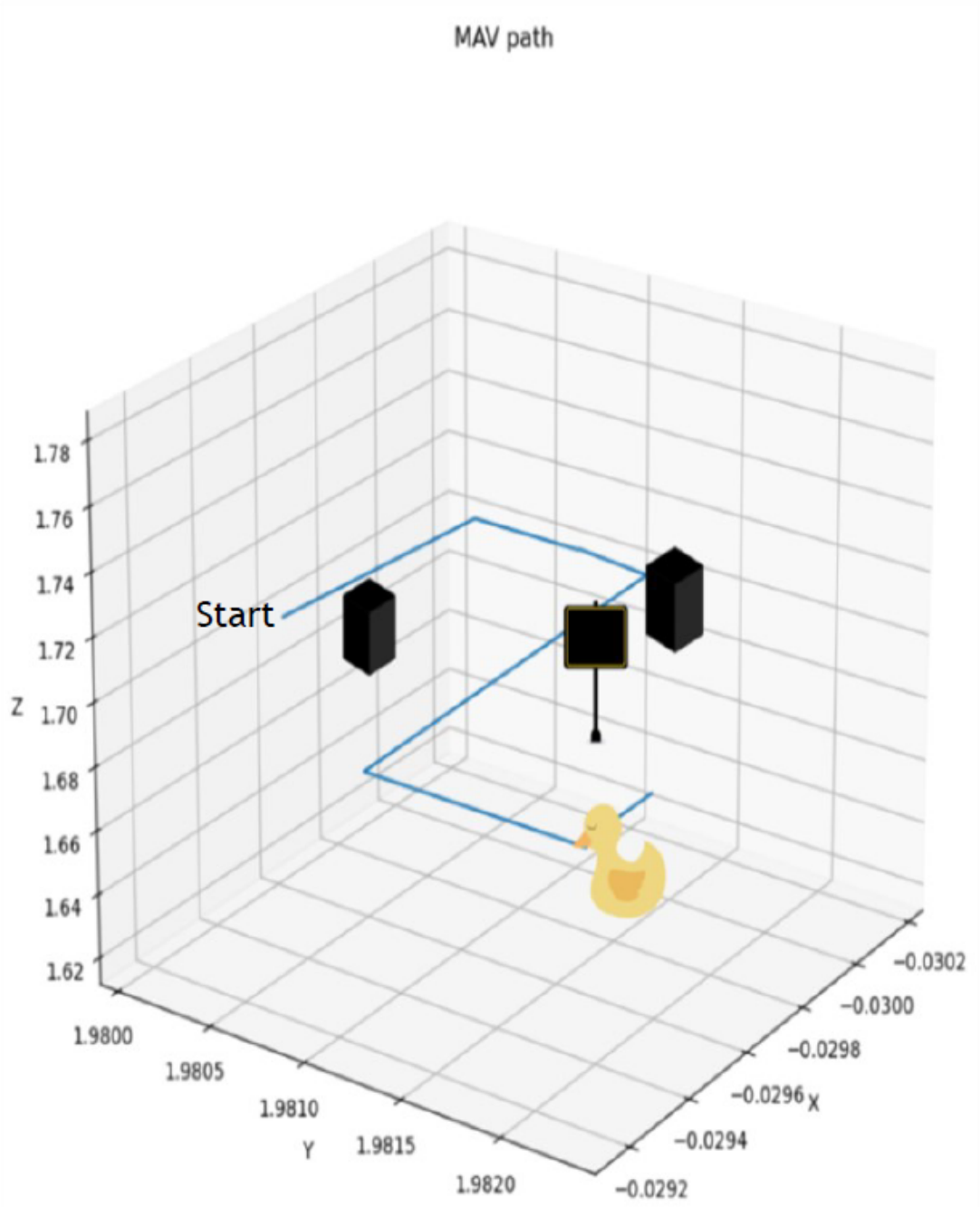


Figure 6.8: 3D autonomous flight trajectory: Scenario 2.

Figures 6.5 – 6.8 present the two dimensional and three-dimensional flight trajectories of the MAV, highlighting its ability to detect and avoid obstacles successfully in different

test scenarios. The observed paths demonstrate that the proposed optical-flow based control strategy provides reliable and effective navigation performance throughout the evaluated experiments. The simulation results obtained using the optical-flow based PID controller are further presented in Figure 6.9. In this case, the MAV moves from an initial position (x_0, y_0) through an unknown environment while maintaining collision-free motion. During navigation, the front-facing camera continuously captures the surrounding scene, and the resulting image sequence is processed to compute optical flow. Based on these motion cues, the system estimates obstacle presence within five predefined image regions, as shown in Figure 6.2, and generates the appropriate avoidance actions.

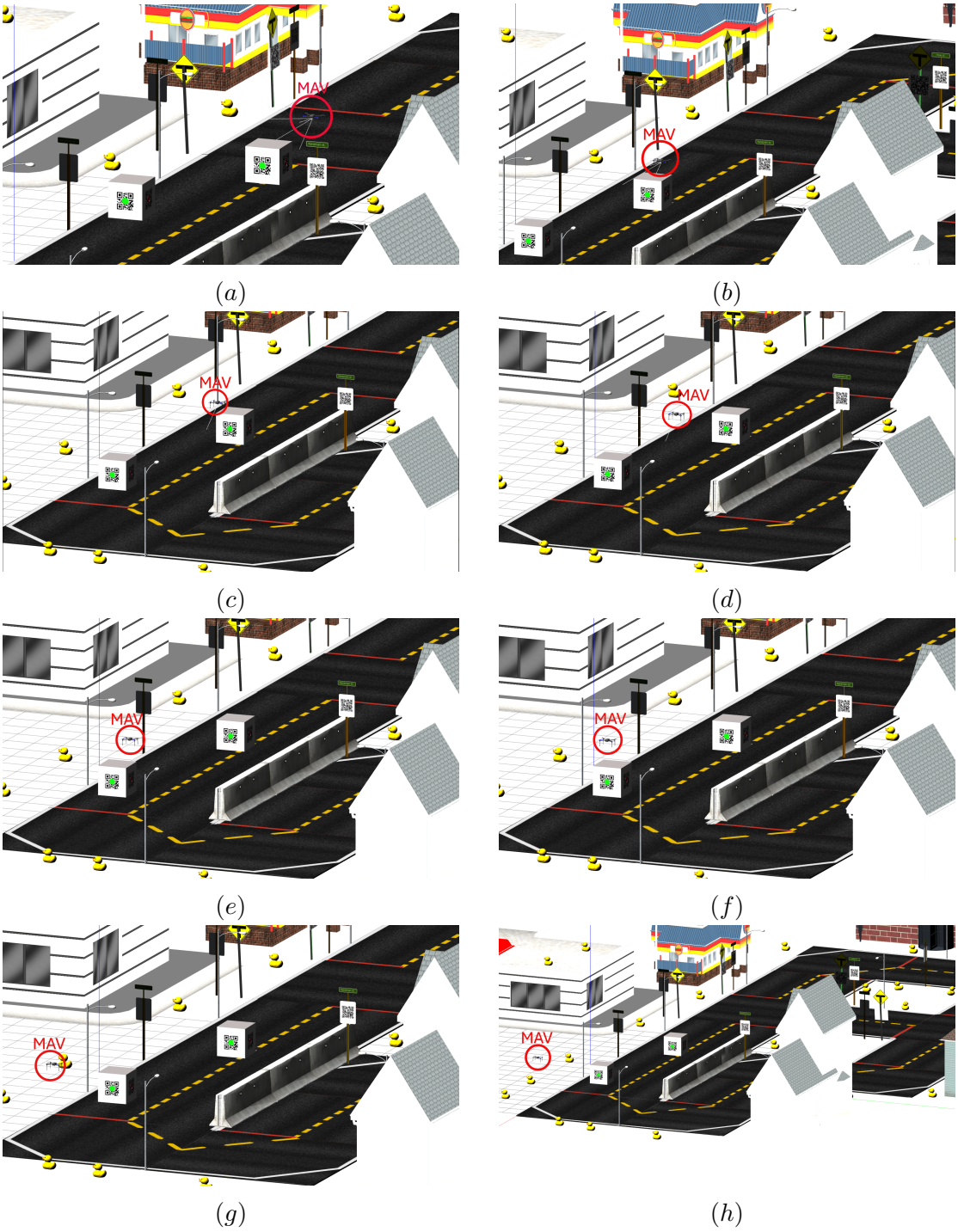


Figure 6.9: Sequential frames illustrating the MAV obstacle avoidance maneuver.

6.6 Discussion

The simulation results were generally encouraging, with most test scenarios demonstrating successful obstacle avoidance. As illustrated in Figure 6.9, the MAV is able to navigate autonomously while avoiding obstacles encountered along its path. The effectiveness of the proposed approach is further supported by the obstacle maps shown in Figures 6.5 and 6.7, where two threatening obstacles are clearly identified on either side of the MAV, while a free navigable space remains available between them. Compared with sonar-based sensing, the vision-based optical-flow approach provides much higher spatial resolution and is capable of detecting obstacles with arbitrary shapes within the image. Although optical flow mainly provides motion information rather than direct absolute range measurements, it offers sufficiently accurate cues for reliable reactive avoidance. However, because range is inferred indirectly from optical-flow measurements, small flow estimation errors may lead to relatively large range uncertainties. This effect becomes more pronounced because of pixel level quantization, especially near the image center, where apparent motion is small and displacement changes occur in coarse steps, resulting in discrete range estimates.

To detect obstacles, the system computes optical-flow values over five predefined image regions. When the nearest obstacle exceeds the detection threshold, the warning message "Obstacle Detected!" is displayed in red at the top-left of the interface, as shown in Figure 6.2(b). Conversely, when no nearby obstacle is present, the message "No Obstacle" appears in green at the top-right, as illustrated in Figure 6.2(a). The simulation experiments indicate that the MAV achieves acceptable performance using the proposed vision-based control framework. The system offers two main advantages: robust collision avoidance in unknown environments and real-time path selection within the simulation environment. Nevertheless, these advantages are only partially realized, since the MAV relies solely on a monocular RGB camera rather than a multi-sensor fusion architecture.

The main limitation of the proposed approach appears when two or more obstacles are located very close to one another within the forward field of view. Under such conditions, distinguishing between nearby obstacles becomes difficult because the system does not use prior knowledge of obstacle shape or size, nor does it exploit additional appearance based information from intensity images to separate objects when motion cues alone are insufficient. In the conducted simulations, however, this issue rarely caused significant practical

difficulties. Once the obstacles became visually separated, the system was able to detect them correctly and treat them as distinct objects.

6.7 conclusion

This chapter introduced a novel monocular vision-based framework for autonomous obstacle avoidance in micro aerial vehicles (MAVs) operating in unknown and unmapped environments. The proposed architecture combines an efficient perception pipeline with a reactive control strategy, enabling reliable navigation without the need for complex and power intensive active sensors. The effectiveness of the approach is mainly based on two key aspects: a compact end-to-end pipeline capable of processing RGB images with low latency, and a PID control scheme that directly converts optical-flow motion cues into stable and accurate avoidance maneuvers. The framework was extensively validated through high-fidelity ROS-Gazebo simulations, where the MAV consistently demonstrated its ability to detect and avoid obstacles in a variety of cluttered and unknown scenarios. By employing the Lucas-Kanade method for optical-flow estimation, the system was able to identify potential threats from visual expansion cues and guide the vehicle toward a safe trajectory in real time. As a continuation of this work, future research will focus on transferring the proposed approach from simulation to real-world flight experiments in order to evaluate its performance under varying lighting and atmospheric conditions. Further investigations will also compare the method with existing state of the art approaches in terms of accuracy and computational cost, while exploring low-cost hardware optimization strategies to facilitate deployment on resource-constrained MAV platforms without compromising flight safety.

Chapter 7

Conclusions and Future Perspectives

7.1 Summary of Contributions

This thesis presented a series of vision-based approaches designed to enhance the performance, robustness, and efficiency of quadrotor navigation in challenging operating conditions, such as lighting variations, dynamic targets, occlusions, and the uncertainty associated with unknown environments. The work initially investigated autonomous landing in both static and dynamic contexts, where reliable visual detection of the landing target plays a critical role in ensuring safe operation. It then expanded to the visual tracking of a moving ball in real-world environments characterized by spontaneous target motion and the absence of predefined paths. In response to these challenges, different solutions were developed, including classical vision-based methods as well as deep-learning-based vision approaches integrated with visual control techniques. The thesis further addressed autonomous navigation in unknown environments, with the aim of controlling the quadrotor from an initial configuration to a desired destination while avoiding collisions, requiring no human intervention, and minimizing tracking error. Altogether, the problems investigated in this thesis resulted in several important contributions, which are summarized below:

- Development of a vision-based autonomous landing framework for quadrotor systems in both static and dynamic scenarios, improving landing safety through accurate and reliable target detection while reducing UAV hardware and computational costs.
- Design and implementation of visual target tracking methods for moving objects, particularly a ball, in unconstrained real-world environments characterized by unpre-

dictable motion and the absence of predefined paths. This contribution includes the investigation of both classical and deep vision-based approaches for perception and control, as well as the integration of vision algorithms with control strategies to achieve effective visual servoing and real-time autonomous response under challenging conditions.

- Proposal of a monocular vision-based obstacle-avoidance framework for autonomous navigation in unknown environments, based on optical-flow information to detect obstacles and guide the quadrotor safely without human intervention. By relying on a front-facing camera, the approach also reduces occlusion effects and provides a lightweight yet effective solution for navigation in complex environments.

In conclusion, the proposed methods were evaluated through simulation, while the target-tracking component was additionally validated through real-world flight experiments demonstrating their effectiveness, robustness, and suitability for quadrotor navigation tasks under lighting variations, occlusions, dynamic targets such as balls and landing platforms, environmental uncertainty, and fully autonomous scenarios.

7.2 Future Perspectives

Building upon the contributions presented in this thesis, several promising avenues for future research have been identified to further enhance the autonomy and resilience of quadrotor systems:

- **Multi-Sensor Fusion and Redundancy:** To overcome the inherent limitations of monocular RGB vision, future efforts will explore the integration of multi-modal sensor fusion. Combining visual data with depth sensors, such as LiDAR or Stereo vision. This would improve the robustness of the landing and navigation frameworks under extreme environmental conditions, such as total darkness or dense fog.

- **Deep Reinforcement Learning (DRL) for Adaptive Control:** While the current PID-based controllers have demonstrated high reliability, future work will investigate the transition toward Deep Reinforcement Learning (DRL) architectures. This approach could allow the UAV to learn complex evasive maneuvers and optimal tracking

strategies directly from high-fidelity simulation data, potentially outperforming traditional control laws in highly dynamic or adversarial environments.

- **Neuromorphic Sensing and Event Cameras:** To address the challenges of motion blur and high-latency processing during high-speed tracking and avoidance, we plan to investigate the application of event-based cameras. These bio-inspired sensors could revolutionize the "Sense-and-Avoid" (SAA) capabilities of the MAV by providing microsecond-level temporal resolution, thereby enabling more precise optical-flow estimation during rapid maneuvers.
- **Collaborative Multi-Agent Systems:** Finally, a significant expansion of this work involves scaling the proposed frameworks to multi-UAV swarms. Enabling collaborative obstacle avoidance and coordinated target tracking would allow a team of quadrotors to share environmental maps and tracking data, leading to a more resilient and distributed form of aerial intelligence.

Bibliography

[1] W. Naeem, S. C. de Oliveira Henrique, and M. C. Abu-Tair, "Collision avoidance of maritime vessels," in *Navigation and Control of Autonomous Marine Vehicles*. IET, 2019.

[2] P. Ochs, J. Malik, and T. Brox, "Segmentation of moving objects by long term video analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 6, pp. 1187–1200, 2013.

[3] M. Mancini, G. Costante, P. Valigi, and T. A. Ciarfuglia, "Fast robust monocular depth estimation for obstacle detection with fully convolutional networks," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4296–4303.

[4] K. P. Valavanis and G. J. Vachtsevanos, *Handbook of unmanned aerial vehicles*. Springer Publishing Company, Incorporated, 2014.

[5] D. Falanga, A. Zanchettin, A. Simovic, J. Delmerico, and D. Scaramuzza, "Vision-based autonomous quadrotor landing on a moving platform," in *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. IEEE, 2017, pp. 200–207.

[6] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," *IEEE Transactions on Robotics*, vol. 33, no. 2, pp. 249–265, 2016.

[7] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[8] C. Pfeiffer, S. Wengeler, A. Loquercio, and D. Scaramuzza, "Visual attention prediction improves performance of autonomous drone racing agents," *PLOS One*, vol. 17, no. 3, p. e0264471, 2022.

[9] D. Falanga, K. Kleber, and D. Scaramuzza, "Dynamic obstacle avoidance for quadrotors with event cameras," *Science Robotics*, vol. 5, no. 40, p. eaaz9712, 2020.

[10] A. Singla, S. Padakandla, and S. Bhatnagar, "Memory-based deep reinforcement learning for obstacle avoidance in uav with limited environment knowledge," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 1, pp. 107–118, 2019.

[11] O. Gharsa, M. M. Touba, M. Boumehraz, N. Abderrahman, S. Bellili, and A. Titaouine, "Autonomous landing system for a quadrotor using a vision-based approach," in *2024 8th International Conference on Image and Signal Processing and their Applications (ISPA)*. IEEE, 2024, pp. 1–5.

[12] O. Gharsa, M. M. Touba, M. Boumehraz, and N. Agram, "Autonomous vision-based object detection and tracking system for quadrotor unmanned aerial vehicles," *Sensors*, vol. 25, no. 20, p. 6403, 2025.

[13] O. Gharsa, M. M. Touba, M. Boumehraz, N. Abderrahmani, I. B. Ghiloubi, and A. H. Drid, "Yolov8-based autonomous ball detection and tracking for rotorcraft uavs using onboard vision sensors," *Engineering Proceedings*, vol. 118, no. 1, p. 8, 2025.

[14] G. Sonugür, "A review of quadrotor uav: Control and slam methodologies ranging from conventional to innovative approaches," *Robotics and Autonomous Systems*, vol. 161, p. 104342, 2023.

[15] A. Loquercio, "Agile autonomy: learning tightly-coupled perception-action for high-speed quadrotor flight in the wild," Ph.D. dissertation, University of Zurich, 2021.

[16] M. Kumar and S. Mondal, "Recent developments on target tracking problems: A review," *Ocean Engineering*, vol. 236, p. 109558, 2021.

[17] A. Bachrach, S. Prentice, R. He, and N. Roy, "Range–robust autonomous navigation in gps-denied environments," *Journal of Field Robotics*, vol. 28, no. 5, pp. 644–666, 2011.

[18] X. Zhao, J. Chong, X. Qi, and Z. Yang, "Vision object-oriented augmented sampling-based autonomous navigation for micro aerial vehicles," *Drones*, vol. 5, no. 4, p. 107, 2021.

[19] A. Ayala, L. Portela, F. Buarque, B. J. Fernandes, and F. Cruz, "Uav control in autonomous object-goal navigation: a systematic literature review," *Artificial Intelligence Review*, vol. 57, no. 5, p. 125, 2024.

[20] G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 2, pp. 237–267, 2002.

[21] Y. Lu, Z. Xue, G.-S. Xia, and L. Zhang, "A survey on vision-based uav navigation," *Geo-spatial information science*, vol. 21, no. 1, pp. 21–32, 2018.

[22] F. Cocchioni, V. Pierfelice, A. Benini, A. Mancini, E. Frontoni, P. Zingaretti, G. Ippoliti, and S. Longhi, "Unmanned ground and aerial vehicles in extended range indoor and outdoor missions," in *2014 International conference on unmanned aircraft systems (ICUAS)*. IEEE, 2014, pp. 374–382.

[23] S. Lee and M. Kim, "Analysis of rendezvous guidance laws for autonomous aerial refueling for non-maneuvering and identical speed targets," *Aerospace Science and Technology*, vol. 121, p. 107359, 2022.

[24] D. P. Watson and D. H. Scheidt, "Autonomous systems," *Johns Hopkins APL technical digest*, vol. 26, no. 4, pp. 368–376, 2005.

[25] L. Heng, L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Autonomous obstacle avoidance and maneuvering on a vision-guided mav using on-board processing," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 2472–2477.

[26] J. Lin, H. Zhu, and J. Alonso-Mora, "Robust vision-based obstacle avoidance for micro aerial vehicles in dynamic environments," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2682–2688.

[27] J. Zhong, M. Li, Y. Chen, Z. Wei, F. Yang, and H. Shen, "A safer vision-based autonomous planning system for quadrotor uavs with dynamic obstacle trajectory prediction and its application with llms," in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2024, pp. 920–929.

[28] M. Tayal, R. Singh, J. Keshavan, and S. Kolathaya, "Control barrier functions in dynamic uavs for kinematic obstacle avoidance: A collision cone approach," in *2024 American Control Conference (ACC)*. IEEE, 2024, pp. 3722–3727.

[29] T. Wakabayashi, Y. Suzuki, and S. Suzuki, "Dynamic obstacle avoidance for multirotor uav using chance-constraints based on obstacle velocity," *Robotics and Autonomous Systems*, vol. 160, p. 104320, 2023.

[30] I. S. Mohamed, M. Ali, and L. Liu, "Chance-constrained sampling-based mpc for collision avoidance in uncertain dynamic environments," *IEEE Robotics and Automation Letters*, 2025.

[31] L. Xin, Z. Tang, W. Gai, and H. Liu, "Vision-based autonomous landing for the uav: A review," *Aerospace*, vol. 9, no. 11, p. 634, 2022.

[32] G. Loegering and S. Harris, "Landing dispersion results-global hawk auto-land system," in *1st UAV Conference*, 2002, p. 3457.

[33] J. V. Carroll, "Vulnerability assessment of the us transportation infrastructure that relies on the global positioning system," *The Journal of Navigation*, vol. 56, no. 2, pp. 185–193, 2003.

[34] Z. Tang, "Vision-based control of unmanned aerial vehicles," Ph.D. dissertation, Université Côte d'Azur; Instituto superior técnico (Lisbonne), 2021.

[35] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[36] A. A. Cabrera-Ponce and J. Martinez-Carranza, "A vision-based approach for autonomous landing," in *2017 Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*. IEEE, 2017, pp. 126–131.

[37] D. Lee, W. Park, and W. Nam, "Autonomous landing of micro unmanned aerial vehicles with landing-assistive platform and robust spherical object detection," *Applied Sciences*, vol. 11, no. 18, p. 8555, 2021.

[38] L.-A. Tran, N.-P. Le, T.-D. Do, and M.-H. Le, "A vision-based method for autonomous landing on a target with a quadcopter," in *2018 4th International Conference on green technology and sustainable development (GTSD)*. IEEE, 2018, pp. 601–606.

[39] W. Kong, D. Zhou, D. Zhang, and J. Zhang, "Vision-based autonomous landing system for unmanned aerial vehicle: A survey," in *2014 international conference on multisensor fusion and information integration for intelligent systems (MFI)*. IEEE, 2014, pp. 1–8.

[40] Z. Wu, P. Han, R. Yao, L. Qiao, W. Zhang, T. Shen, M. Sun, Y. Zhu, M. Lin, and R. Fan, "Autonomous uav landing system based on visual navigation," in *2019 IEEE International Conference on Imaging Systems and Techniques (IST)*. IEEE, 2019, pp. 1–6.

[41] S. Lee, D. Jo, and Y. Kwon, "Camera-based automatic landing of drones using artificial intelligence image recognition," *Int. J. Mech. Eng. Robot. Res*, vol. 11, no. 5, pp. 357–364, 2022.

[42] J. Chen, X. Miao, H. Jiang, J. Chen, and X. Liu, "Identification of autonomous landing sign for unmanned aerial vehicle based on faster regions with convolutional neural network," in *2017 Chinese Automation Congress (CAC)*. IEEE, 2017, pp. 2109–2114.

[43] S. O. Semerikov, P. P. Nechypurenko, T. A. Vakaliuk, I. S. Mintii, and A. O. Kolhatin, "Vision-based autonomous uav landing: A comprehensive review of technologies, techniques, and applications," *Journal of Intelligent & Robotic Systems*, vol. 111, no. 4, pp. 1–39, 2025.

[44] M. Saavedra-Ruiz, A. M. Pinto-Vargas, and V. Romero-Cano, "Monocular visual autonomous landing system for quadcopter drones using software in the loop," *IEEE Aerospace and Electronic Systems Magazine*, vol. 37, no. 5, pp. 2–16, 2021.

[45] F. Farahi and H. S. Yazdi, "Probabilistic kalman filter for moving object tracking," *Signal Processing: Image Communication*, vol. 82, p. 115751, 2020.

[46] K. Yang and Q. Quan, "An autonomous intercept drone with image-based visual servo," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2230–2236.

[47] J. Hermansson, A. Gising, M. Skoglund, and T. Schön, *Autonomous landing of an unmanned aerial vehicle*. Linköping University Electronic Press, 2010.

[48] T. Yang, P. Li, H. Zhang, J. Li, and Z. Li, "Monocular vision slam-based uav autonomous landing in emergencies and unknown environments," *Electronics*, vol. 7, no. 5, p. 73, 2018.

[49] A. Rodriguez-Ramos, C. Sampedro, H. Bavle, P. De La Puente, and P. Campoy, "A deep reinforcement learning strategy for uav autonomous landing on a moving platform," *Journal of Intelligent & Robotic Systems*, vol. 93, no. 1, pp. 351–366, 2019.

[50] G. Cho, J. Choi, G. Bae, and H. Oh, "Autonomous ship deck landing of a quadrotor uav using feed-forward image-based visual servoing," *Aerospace Science and Technology*, vol. 130, p. 107869, 2022.

[51] A. Gautam, M. Singh, P. B. Sujit, and S. Saripalli, "Autonomous quadcopter landing on a moving target," *Sensors*, vol. 22, no. 3, p. 1116, 2022.

[52] O. Gharsa, M. M. Touba, M. Boumehraz, N. Abderrahmani, and I. B. Ghiloubi, "Autonomous landing of a quadrotor on a moving target using a vision-based approach," in *2025 5th International Conference on Applied Automation and Industrial Diagnostics (ICAAID)*, 2025, pp. 1–6.

[53] J. A. Macés-Hernández, F. Defaÿ, and C. Chauffaut, "Autonomous landing of an uav on a moving platform using model predictive control," in *2017 11th Asian Control Conference (ASCC)*. IEEE, 2017, pp. 2298–2303.

[54] X. Chen, S. K. Phang, M. Shan, and B. M. Chen, "System integration of a vision-guided uav for autonomous landing on moving platform," in *2016 12th IEEE International Conference on Control and Automation (ICCA)*. IEEE, 2016, pp. 761–766.

[55] J. L. Sanchez-Lopez, J. Pestana, S. Saripalli, and P. Campoy, "An approach toward visual autonomous ship board landing of a vtol uav," *Journal of Intelligent & Robotic Systems*, vol. 74, no. 1, pp. 113–127, 2014.

[56] G. Xu, Y. Zhang, S. Ji, Y. Cheng, and Y. Tian, "Research on computer vision-based for uav autonomous landing on a ship," *Pattern Recognition Letters*, vol. 30, no. 6, pp. 600–605, 2009.

[57] F. Morais, T. Ramalho, P. Sinogas, M. M. Marques, N. P. Santos, and V. Lobo, "Trajectory and guidance mode for autonomously landing an uav on a naval platform using a vision approach," in *OCEANS 2015-Genova*. IEEE, 2015, pp. 1–7.

[58] R. Polvara, S. Sharma, J. Wan, A. Manning, and R. Sutton, "Towards autonomous landing on a moving vessel through fiducial markers," in *2017 European Conference on Mobile Robots (ECMR)*. IEEE, 2017, pp. 1–6.

[59] A. Bry, A. Bachrach, and N. Roy, "State estimation for aggressive flight in gps-denied environments using onboard sensing," in *2012 IEEE international conference on robotics and automation*. IEEE, 2012, pp. 1–8.

[60] A. R. Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Ultimate slam? combining events, images, and imu for robust visual slam in hdr and high-speed scenarios," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 994–1001, 2018.

[61] 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.

[62] S. Lynen, T. Sattler, M. Bosse, J. A. Hesch, M. Pollefeys, and R. Siegwart, "Get out of my lab: Large-scale, real-time visual-inertial localization." in *Robotics: Science and Systems*, vol. 1, no. s 1, 2015.

[63] S. Liu, "Computer vision for perception and localization," in *In Engineering Autonomous Vehicles and Robots: The DragonFly Modular-based Approach*. Wiley-IEEE Press, 2019, pp. 77–96.

[64] R. Brooks, "A robust layered countrol system for a mobil robot," *IEEE Journal of Robotics and Automation*, vol. 2, p. 1, 1986.

[65] R. A. Brooks, "Intelligence without representation," *Artificial intelligence*, vol. 47, no. 1-3, pp. 139–159, 1991.

[66] F. Bonin-Font, A. Ortiz, and G. Oliver, "Visual navigation for mobile robots: A survey," *Journal of intelligent and robotic systems*, vol. 53, no. 3, pp. 263–296, 2008.

[67] S. S. Beauchemin and J. L. Barron, "The computation of optical flow," *ACM computing surveys (CSUR)*, vol. 27, no. 3, pp. 433–466, 1995.

[68] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the seventh IEEE international conference on computer vision*, vol. 2. Ieee, 1999, pp. 1150–1157.

[69] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *European conference on computer vision*. Springer, 2006, pp. 404–417.

[70] K. Choutri, M. Lagha, S. Meshoul, H. Shaiba, A. Chegrani, and M. Yahiaoui, "Vision-based uav detection and localization to indoor positioning system," *Sensors*, vol. 24, no. 13, p. 4121, 2024.

[71] S. Badrloo, M. Varshosaz, S. Pirasteh, and J. Li, "A novel region-based expansion rate obstacle detection method for mavs using a fisheye camera," *International Journal of Applied Earth Observation and Geoinformation*, vol. 108, p. 102739, 2022.

[72] A. Al-Kaff, F. García, D. Martín, A. De La Escalera, and J. M. Armingol, "Obstacle detection and avoidance system based on monocular camera and size expansion algorithm for uavs," *Sensors*, vol. 17, no. 5, p. 1061, 2017.

[73] J. Santos-Victor, G. Sandini, F. Curotto, and S. Garibaldi, "Divergent stereo for robot navigation: Learning from bees," in *Proceedings of IEEE conference on computer vision and pattern recognition*. IEEE, 1993, pp. 434–439.

[74] M. Nadour, M. Boumehraz, L. Cherroun, and V. Puig, "Mobile robot visual navigation based on fuzzy logic and optical flow approaches," *International Journal of System Assurance Engineering and Management*, vol. 10, no. 6, pp. 1654–1667, 2019.

[75] W. Gao, S. Jiang, and Q. Quan, "Multicopters obstacle avoidance by learning optical flow with a balance strategy," in *2023 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2023, pp. 1053–1058.

[76] Y. Hu, Y. Zhang, Y. Song, Y. Deng, F. Yu, L. Zhang, W. Lin, D. Zou, and W. Yu, "Seeing through pixel motion: learning obstacle avoidance from optical flow with one camera," *IEEE Robotics and Automation Letters*, 2025.

[77] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: analysis, applications, and prospects," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 12, pp. 6999–7019, 2021.

[78] J. Xiao, R. Zhang, Y. Zhang, and M. Feroskhan, "Vision-based learning for drones: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, 2025.

[79] S.-Y. Yang, H.-Y. Cheng, and C.-C. Yu, "Real-time object detection and tracking for unmanned aerial vehicles based on convolutional neural networks," *Electronics*, vol. 12, no. 24, p. 4928, 2023.

[80] A. Dosovitskiy, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[81] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 012–10 022.

[82] Y. Liu, Y. Zhang, Y. Wang, F. Hou, J. Yuan, J. Tian, Y. Zhang, Z. Shi, J. Fan, and Z. He, "A survey of visual transformers," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 6, pp. 7478–7498, 2023.

[83] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[84] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[85] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[86] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[87] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[88] R. Varghese and M. Sambath, "Yolov8: A novel object detection algorithm with enhanced performance and robustness," in *2024 International conference on advances in data engineering and intelligent computing systems (ADICS)*. IEEE, 2024, pp. 1–6.

[89] S. Bouabdallah, A. Noth, and R. Siegwart, "Pid vs lq control techniques applied to an indoor micro quadrotor," in *2004 IEEE/RSJ international conference on intelligent robots and systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3. IEEE, 2004, pp. 2451–2456.

[90] I. D. Cowling, J. F. Whidborne, and A. K. Cooke, "Optimal trajectory planning and lqr control for a quadrotor uav," in *International Conference on Control*, 2006.

[91] A. Mokhtari, N. K. M'Sirdi, K. Meghriche, and A. Belaidi, "Feedback linearization and linear observer for a quadrotor unmanned aerial vehicle," *Advanced Robotics*, vol. 20, no. 1, pp. 71–91, 2006.

[92] W. Jasim and D. Gu, "Integral backstepping controller for quadrotor path tracking," in *2015 International Conference on Advanced Robotics (ICAR)*. IEEE, 2015, pp. 593–598.

[93] G. Perozzi, D. Efimov, J.-M. Biannic, L. Planckaert, and P. Coton, "On sliding mode control design for uav using realistic aerodynamic coefficients," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*. IEEE, 2017, pp. 5403–5408.

[94] S. A. Al-Hiddabi, "Quadrotor control using feedback linearization with dynamic extension," in *2009 6th International Symposium on Mechatronics and its Applications*. IEEE, 2009, pp. 1–3.

[95] Z. Jia, J. Yu, Y. Mei, Y. Chen, Y. Shen, and X. Ai, "Integral backstepping sliding mode control for quadrotor helicopter under external uncertain disturbances," *Aerospace Science and Technology*, vol. 68, pp. 299–307, 2017.

[96] J. P. How, B. Behihke, A. Frank, D. Dale, and J. Vian, "Real-time indoor autonomous vehicle test environment," *IEEE Control Systems Magazine*, vol. 28, no. 2, pp. 51–64, 2008.

[97] G. Hoffmann, S. Waslander, and C. Tomlin, "Quadrotor helicopter trajectory tracking control," in *AIAA guidance, navigation and control conference and exhibit*, 2008, p. 7410.

[98] E. Reyes-Valeria, R. Enriquez-Caldera, S. Camacho-Lara, and J. Guichard, "Lqr control for a quadrotor using unit quaternions: Modeling and simulation," in *CONIELECOMP 2013, 23rd International Conference on Electronics, Communications and Computing*. IEEE, 2013, pp. 172–178.

[99] M. Bangura and R. Mahony, "Real-time model predictive control for quadrotors," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 11 773–11 780, 2014.

[100] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1309–1332, 2017.

[101] D. Scaramuzza and F. Fraundorfer, "Visual odometry [tutorial]," *IEEE robotics & automation magazine*, vol. 18, no. 4, pp. 80–92, 2011.

[102] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[103] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "Fast-lio2: Fast direct lidar-inertial odometry," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2053–2073, 2022.

[104] C. Song, Z. Chen, K. Wang, H. Luo, and J. C. Cheng, "Bim-supported scan and flight planning for fully autonomous lidar-carrying uavs," *Automation in Construction*, vol. 142, p. 104533, 2022.

[105] S. Thrun, W. Burgard, and D. Fox, "Probabilistic robotics cambridge," *MA: MIT Press [Google Scholar]*, 2005.

[106] R. Isermann, *Fault-diagnosis systems: an introduction from fault detection to fault tolerance*. Springer Science & Business Media, 2005.

[107] C. Hajiyev and F. Caliskan, *Fault diagnosis and reconfiguration in flight control systems*. Springer Science & Business Media, 2003, vol. 2.

[108] N. Tanaka, S. Suzuki, K. Masui, and H. Tomita, "Restructurable guidance and control for aircraft with failures considering gust effects," *Journal of guidance, control, and dynamics*, vol. 29, no. 3, pp. 671–679, 2006.

[109] G. Ducard and H. P. Geering, "Efficient nonlinear actuator fault detection and isolation system for unmanned aerial vehicles," *Journal of Guidance, Control, and Dynamics*, vol. 31, no. 1, pp. 225–237, 2008.

[110] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, "A new method for the nonlinear transformation of means and covariances in filters and estimators," *IEEE Transactions on Automatic Control*, vol. 45, no. 3, pp. 477–482, 2002.

[111] M. E. Campbell, J.-w. Lee, E. Scholte, and D. Rathbun, "Simulation and flight test of autonomous aircraft estimation, planning, and control algorithms," *Journal of Guidance, Control, and Dynamics*, vol. 30, no. 6, pp. 1597–1609, 2007.

[112] S. Brunke and M. Campbell, "Estimation architecture for future autonomous vehicles," in *Proceedings of the 2002 American Control Conference (IEEE Cat. No. CH37301)*, vol. 2. IEEE, 2002, pp. 1108–1114.

[113] N. Guenard, T. Hamel, and R. Mahony, "A practical visual servo control for an unmanned aerial vehicle," *IEEE Transactions on Robotics*, vol. 24, no. 2, pp. 331–340, 2008.

[114] J. Thomas, G. Loianno, J. Polin, K. Sreenath, and V. Kumar, "Toward autonomous avian-inspired grasping for micro aerial vehicles," *Bioinspiration & biomimetics*, vol. 9, no. 2, p. 025010, 2014.

[115] M. Odelga, P. Stegagno, and H. H. Bülthoff, "Obstacle detection, tracking and avoidance for a teleoperated uav," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 2984–2990.

[116] B. Penin, "Contributions to optimal and reactive vision-based trajectory generation for a quadrotor uav," Ph.D. dissertation, Université de Rennes, 2018.

[117] D. Lee, T. Ryan, and H. J. Kim, "Autonomous landing of a vtol uav on a moving platform using image-based visual servoing," in *2012 IEEE international conference on robotics and automation*. IEEE, 2012, pp. 971–976.

[118] J. Thomas, J. Welde, G. Loianno, K. Daniilidis, and V. Kumar, "Autonomous flight for detection, localization, and tracking of moving targets with a small quadrotor," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1762–1769, 2017.

[119] H. Seo, S. Kim, and H. J. Kim, "Aerial grasping of cylindrical object using visual servoing based on stochastic model predictive control," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 6362–6368.

[120] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis *et al.*, "Event-based vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 1, pp. 154–180, 2020.

[121] P. Pisutsin, J. Xiao, and M. Feroskhan, "Omnidrone-det: Omnidirectional 3d drone detection in flight," in *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*. IEEE, 2024, pp. 2409–2414.

[122] S. F. Bhat, R. Birkl, D. Wofk, P. Wonka, and M. Müller, "Zoedepth: Zero-shot transfer by combining relative and metric depth," *arXiv preprint arXiv:2302.12288*, 2023.

[123] H. Laga, L. V. Jospin, F. Boussaid, and M. Bennamoun, "A survey on deep learning techniques for stereo-based depth estimation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 4, pp. 1738–1764, 2020.

[124] J. Y. Wang and E. H. Adelson, "Representing moving images with layers," *IEEE Transactions on Image Processing*, vol. 3, no. 5, pp. 625–638, 1994.

[125] J. Shi and J. Malik, "Motion segmentation and tracking using normalized cuts," in *Sixth international conference on computer vision (IEEE Cat. No. 98CH36271)*. IEEE, 1998, pp. 1154–1160.

[126] D. Cremers and S. Soatto, "Motion competition: A variational approach to piecewise parametric motion segmentation," *International Journal of Computer Vision*, vol. 62, no. 3, pp. 249–265, 2005.

[127] T. Brox, A. Bruhn, and J. Weickert, "Variational motion segmentation with level sets," in *European Conference on Computer Vision*. Springer, 2006, pp. 471–483.

[128] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 5026–5033.

[129] J. Hwangbo, J. Lee, and M. Hutter, "Per-contact iteration method for solving contact dynamics," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 895–902, 2018.

[130] W. Guerra, E. Tal, V. Murali, G. Ryou, and S. Karaman, "Flightgoggles: Photorealistic sensor simulation for perception-driven robotics using photogrammetry and virtual reality," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 6941–6948.

[131] F. Furrer, M. Burri, M. Achtelik, and R. Siegwart, "Rotors—a modular gazebo mav simulator framework," in *Robot Operating System (ROS) The Complete Reference (Volume 1)*. Springer, 2016, pp. 595–625.

[132] S. Kohlbrecher, J. Meyer, T. Graber, K. Petersen, U. Klingauf, and O. Von Stryk, "Hector open source modules for autonomous mapping and navigation with rescue robots," in *Robot Soccer World Cup*. Springer, 2013, pp. 624–631.

[133] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ international conference on intelligent robots and systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3. Ieee, 2004, pp. 2149–2154.

[134] D. Foyle, A. Goodman, and B. Hooey, "Nasa aviation safety program conference on human performance modeling of approach and landing with augmented displays," *National Aeronautics and Space Administration*, 2003.

[135] B. L. Stevens, F. L. Lewis, and E. N. Johnson, *Aircraft control and simulation: dynamics, controls design, and autonomous systems*. John Wiley & Sons, 2015.

[136] M. Boumehraz, Z. Habba, and R. Hassani, "Vision based tracking and interception of moving target by mobile robot using fuzzy control," *Journal of Applied Engineering Science and Technology*, vol. 4, no. 2, pp. 7–7, 2018.

[137] H.-C. Kang, H.-N. Han, H.-C. Bae, M.-G. Kim, J.-Y. Son, and Y.-K. Kim, "Hsv color-space-based automated object localization for robot grasping without prior knowledge," *Applied Sciences*, vol. 11, no. 16, p. 7593, 2021.

[138] S. Misra and Y. Wu, "Machine learning assisted segmentation of scanning electron microscopy images of organic-rich shales with feature extraction and feature ranking," *Machine learning for subsurface characterization*, vol. 289, no. 3, p. 4, 2019.

[139] F. S. Leira, H. H. Helgesen, T. A. Johansen, and T. I. Fossen, "Object detection, recognition, and tracking from uavs using a thermal camera," *Journal of Field Robotics*, vol. 38, no. 2, pp. 242–267, 2021.

[140] M. H. Malik, T. Zhang, H. Li, M. Zhang, S. Shabbir, and A. Saeed, "Mature tomato fruit detection algorithm based on improved hsv and watershed algorithm," *IFAC-PapersOnLine*, vol. 51, no. 17, pp. 431–436, 2018.

[141] S. Wu, R. Li, Y. Shi, and Q. Liu, "Vision-based target detection and tracking system for a quadcopter," *IEEE Access*, vol. 9, pp. 62 043–62 054, 2021.

[142] P. Foehn, D. Brescianini, E. Kaufmann, T. Cieslewski, M. Gehrig, M. Muglikar, and D. Scaramuzza, "Alphapilot: Autonomous drone racing," *Autonomous Robots*, vol. 46, no. 1, pp. 307–320, 2022.

[143] B. Lei, B. Liu, and C. Wang, "Robust geometric control for a quadrotor uav with extended kalman filter estimation," in *Actuators*, vol. 13, no. 6. MDPI, 2024, p. 205.

[144] D. Hulens, W. Van Ranst, Y. Cao, and T. Goedemé, "Autonomous visual navigation for a flower pollination drone," *Machines*, vol. 10, no. 5, p. 364, 2022.

[145] E. Schnipke, S. Reidling, J. Meiring, W. Jeffers, M. Hashemi, R. Tan, A. Nemati, and M. Kumar, "Autonomous navigation of uav through gps-denied indoor environment with obstacles," in *AIAA Infotech@ Aerospace*, 2015, p. 0715.

[146] D. Mellinger, N. Michael, and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 664–674, 2012.

[147] M. W. Mueller, M. Hehn, and R. D'Andrea, "A computationally efficient motion primitive for quadrocopter trajectory generation," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1294–1310, 2015.

[148] D. Falanga, E. Mueggler, M. Faessler, and D. Scaramuzza, "Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 5774–5781.

[149] Ryze Robotics, "Tello," https://www.ryzerobotics.com/tello, accessed: September 20, 2024.

[150] B. Vergouw, H. Nagel, G. Bondt, and B. Custers, "Drone technology: Types, payloads, applications, frequency spectrum issues and future developments," in *The future of drone use: Opportunities and threats from ethical and legal perspectives*. Springer, 2016, pp. 21–45.

[151] W. Li, Y. Hong, R. Pan, J. Su, L. Wang, and S. Bao, "Enhanced yolo architecture for real-time coal gangue detection with lightweight design," *Signal, Image and Video Processing*, vol. 19, no. 14, p. 1199, 2025.

[152] A. Elhanashi, P. Dini, S. Saponara, and Q. Zheng, "Telestroke: real-time stroke detection with federated learning and yolov8 on edge devices," *Journal of Real-Time Image Processing*, vol. 21, no. 4, p. 121, 2024.

[153] D. Reis, J. Kupec, J. Hong, and A. Daoudi, "Real-time flying object detection with yolov8," *arXiv preprint arXiv:2305.09972*, 2023.

[154] R. Khanam and M. Hussain, "What is yolov5: A deep look into the internal features of the popular object detector," *arXiv preprint arXiv:2407.20892*, 2024.

[155] S. K. Jaiswal and R. Agrawal, "A comprehensive review of yolov5: advances in real-time object detection," *International Journal of Innovative Research in Computer Science and Technology*, vol. 12, no. 3, pp. 75–80, 2024.

[156] R. A. Zitar, M. Al-Betar, M. Ryalat, and S. Kassaymeh, "A review of uav visual detection and tracking methods," *arXiv preprint arXiv:2306.05089*, 2023.

[157] H. Anis, A. H. I. Fadhillah, S. Darma, and S. Soekirno, "Automatic quadcopter control avoiding obstacle using camera with integrated ultrasonic sensor," in *Journal of Physics: Conference Series*, vol. 1011, no. 1. IOP Publishing, 2018, p. 012046.

[158] J. Sagar and A. Visser, "Obstacle avoidance by combining background subtraction, optical flow and proximity estimation," in *IMAV 2014: International Micro Air Vehicle Conference and Competition 2014, Delft, The Netherlands, August 12-15, 2014*. Delft University of Technology, 2014.

[159] S. Scherer, J. Rehder, S. Achar, H. Cover, A. Chambers, S. Nuske, and S. Singh, "River mapping from a flying robot: state estimation, river detection, and obstacle mapping," *Autonomous Robots*, vol. 33, no. 1, pp. 189–214, 2012.

[160] M. Faessler, F. Fontana, C. Forster, E. Mueggler, M. Pizzoli, and D. Scaramuzza, "Autonomous, vision-based flight and live dense 3d mapping with a quadrotor micro aerial vehicle," *Journal of Field Robotics*, vol. 33, no. 4, pp. 431–450, 2016.

[161] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *IJCAI'81: 7th international joint conference on Artificial intelligence*, vol. 2, 1981, pp. 674–679.

[162] T. Low and G. Wyeth, "Learning to avoid indoor obstacles from optical flow," in *Proceedings of the 2007 Australasian Conference on Robotics and Automation*. Australian Robotics and Automation Association, 2007, pp. 1–10.

[163] S. Baker and I. Matthews, "Lucas-kanade 20 years on: A unifying framework," *International journal of computer vision*, vol. 56, no. 3, pp. 221–255, 2004.

[164] J.-E. Gomez-Balderas, G. Flores, L. García Carrillo, and R. Lozano, "Tracking a ground moving target with a quadrotor using switching control: Nonlinear modeling and control," *Journal of Intelligent & Robotic Systems*, vol. 70, no. 1, pp. 65–78, 2013.

[165] J.-Y. Bouguet *et al.*, "Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm," *Intel corporation*, vol. 5, no. 1-10, p. 4, 2001.

[166] D. Howard and T. Merz, "A platform for the direct hardware evolution of quadcopter controllers," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 4614–4619.

[167] M. Blösch, S. Weiss, D. Scaramuzza, and R. Siegwart, "Vision based mav navigation in unknown and unstructured environments," in *2010 IEEE International Conference on Robotics and Automation*. IEEE, 2010, pp. 21–28.