

Kader Monhamady Kabore

Ph.D. Thesis

AGU 2024

DESIGN AND REALIZATION OF LOCALIZATION ALGORITHMS FOR MULTI-ROBOT SYSTEMS

Ph.D. THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL
AND COMPUTER ENGINEERING
AND THE GRADUATE SCHOOL OF ENGINEERING AND
SCIENCE OF ABDULLAH GUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Ph.D.

By

Kader Monhamady KABORE

December 2024

DESIGN AND REALIZATION OF LOCALIZATION ALGORITHMS FOR MULTI-ROBOT SYSTEMS

A THESIS

SUBMITTED TO THE DEPARTMENT OF ELECTRICAL
AND COMPUTER ENGINEERING
AND THE GRADUATE SCHOOL OF ENGINEERING AND SCIENCE
OF ABDULLAH GUL UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
Ph.D.

By

Kader Monhamady KABORE

December 2024

SCIENTIFIC ETHICS COMPLIANCE

I hereby declare that all information in this document has been obtained in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

Kader Monhamady KABORE



REGULATORY COMPLIANCE

Ph.D. thesis titled Design and Realization of Localization Algorithms for Multi-Robot Systems has been prepared in accordance with the Thesis Writing Guidelines of the Abdullah Gül University, Graduate School of Engineering & Science.

Prepared By

Kader Monhamady KABORE

Signature

Advisor

Assoc. Prof. Dr. Samet GÜLER

Signature

Head of the Electrical and Computer Engineering Program

Assoc. Prof. Dr. Samet GÜLER

Signature

ACCEPTANCE AND APPROVAL

Ph.D. thesis titled Design and Realization of Localization Algorithms for Multi-Robot Systems and prepared by Kader Monhamady KABORE has been accepted by the jury in the Electrical and Computer Engineering Graduate Program at Abdullah Gül University, Graduate School of Engineering & Science.

30/12 /2024

JURY :

Advisor : Assoc. Prof. Dr. Samet GÜLER
Member : Assoc. Prof. Dr. Rifat KURBAN
Member : Assoc. Prof. Dr. Haluk BAYRAM
Member : Assoc. Prof. Dr. Eyüp ÇINAR
Member : Assist. Prof. Dr. Abdulkadir KÖSE

APPROVAL:

The acceptance of this Ph.D. thesis has been approved by the decision of the Abdullah Gül University, Graduate School of Engineering & Science, Executive Board dated /..... / and numbered

..... /..... /

Graduate School Dean

Prof. İrfan ALAN

ABSTRACT

DESIGN AND REALIZATION OF LOCALIZATION ALGORITHMS FOR MULTI-ROBOT SYSTEMS

Kader Monhamady KABORE
Ph.D. in Electrical and Computer Engineering
Advisor: Assoc. Prof. Samet GÜLER
December 2024

Multi-robot systems (MRS) can accomplish complex tasks extremely challenging for a single robot. For instance, in applications such as cooperative transportation, area coverage and search and rescue operations, MRS can be the best choice. MRS divides tasks into simpler commands and assigns them to individual robots. This structure favored decentralized approaches, which provide significant advantages, including scalability and robustness to single points of failure. Formation control in MRS relies on strong robot positioning, which is particularly challenging in environments where GPS is not available and external infrastructure does not exist. In outdoor environment, GPS can provide absolute positioning but may be insufficient for swarm robotics in environments such as indoors or tunnels. Indoor localization solutions, such as motion capture systems, come at a high cost and require additional infrastructure setup procedure. These limitations highlight the need for robust, onboard localization systems suitable for swarm robotics applications. This work presents a novel decentralized, marker-free localization framework for MRS that relies entirely on onboard capabilities, eliminating dependence on external infrastructure. To find a localization solution for MRS, our method combines cooperative localization algorithms powered by deep learning detections with formation control mechanisms. We validate the framework through extensive simulations and real-world experiments to validate the effectiveness of the proposed framework. We explore the system's scalability by adapting to varying team sizes, showcasing its effectiveness in applications. This work also provides an open-source dataset for ground robots, thereby promoting further research in MRS.

Keywords: Relative localization, Mobile robotics, Formation control, Deep learning, Sensor fusion

ÖZET

ÇOKLU ROBOT SİSTEMLERİ İÇİN LOKALİZASYON ALGORİTMASI TASARIMI VE GERÇEKLEŞMESİ

Kader Monhamady KABORE
Elektrik ve Bilgisayar Mühendisliği Anabilim Dalı Doktora
Tez Danışmanı: Assoc. Prof. Samet GÜLER
Aralık 2024

Çok robotlu sistemler (MRS), tek bir robot için son derece zorlayıcı olan karmaşık görevleri gerçekleştirebilir. Örneğin, iş birliğiyle taşıma, alan kapsama ve arama-kurtarma operasyonları gibi uygulamalarda, MRS en iyi seçenek olabilir. MRS, görevleri daha basit komutlara bölerek bireysel robotlara atar. Bu yapı, ölçeklenebilirlik ve tek bir hata noktasına karşı dayanıklılık gibi önemli avantajlar sağlayan merkezi olmayan yaklaşımlara ilgiyi artırmıştır. MRS'deki formasyon kontrolü, özellikle GPS'in bulunmadığı ve dış altyapının olmadığı ortamlarda güçlü robot konumlandırmasına dayanır. Dış ortamlarda GPS mutlak konumlandırma sağlayabilir ancak kapalı alanlar veya tüneller gibi ortamlarda sürü robotları için yetersiz kalabilir. Hareket yakalama sistemleri gibi kapalı alan konumlandırma çözümleri, yüksek maliyetli olup ek altyapı kurulum prosedürleri gerektirir. Bu sınırlamalar, sürü robotikleri uygulamaları için uygun, dayanıklı ve dahili konumlandırma sistemlerine olan ihtiyacı vurgulamaktadır. Bu çalışma, tamamen dahili yeteneklere dayanan, dış altyapıya bağımlılığı ortadan kaldıran yeni bir merkezi olmayan, işaretleyicisiz konumlandırma çerçevesi sunmaktadır. MRS için bir konumlandırma çözümü bulmak amacıyla, yöntemimiz, derin öğrenme ile güçlendirilmiş iş birliği temelli konumlandırma algoritmalarını formasyon kontrol mekanizmalarıyla birleştirmektedir. Önerilen çerçevenin etkinliğini doğrulamak için kapsamlı simülasyonlar ve gerçek dünya deneyleri gerçekleştirilmiştir. Sistem ölçeklenebilirliği, farklı ekip boyutlarına uyum sağlayarak test edilmiştir ve uygulamalardaki etkinliği gösterilmiştir. Bu çalışma ayrıca yer robotları için açık kaynaklı bir veri seti sunarak MRS alanında daha fazla araştırmayı teşvik etmektedir.

Anahtar kelimeler: Göreceli lokalizasyon, Mobil robotik, Formasyon kontrolü, Derin öğrenme, Sensör füzyonu

Acknowledgements

Thanks to my advisor. On a long and difficult journey, you always need guidance. Assoc. Prof. Samet GÜLER has always been there to provide me with this light of guidance.

Thanks to my Jury members who always brought comment to improve my research.

Thanks I would like to express my heartfelt gratitude to Dr. Akrema, Dr. Ruby Phul, Dr. Ehsan Soheyli, Dr. Rim Zgueb, Engineer Aziz Kirakoya and to my fellow Ph.D. colleagues, Zeynep and Şebnem for their camaraderie, support, and the countless enriching discussions that made this journey more meaningful and enjoyable.

A special thanks goes to my wife for her unwavering support, patience, and understanding throughout my research. Your belief in me has been my anchor.

To my firstborn and my new best buddy, you've brought immense joy and inspiration into my life, and for that, I am forever grateful.

Finally, I would like to thank the Scientific and Technological Research Council of Turkey (TÜBİTAK) for providing the PhD Scholarship under the 2232 and 2211-A Research Program.

Thanks to everyone who assisted me on this journey.

TABLE OF CONTENTS

1. INTRODUCTION	1
1.1 RESEARCH CONTEXT AND BACKGROUND	1
1.2 MOTIVATION FOR THE RESEARCH.....	2
1.3 RESEARCH QUESTIONS AND HYPOTHESES	3
1.4 SIGNIFICANCE OF THE RESEARCH	4
1.5 THESIS STRUCTURE.....	5
2. PRELIMINARIES AND NOTATION	7
2.1 PRELIMINARIES AND NOTATION	7
2.1.1 <i>Graph Theoretical Modeling of Multi-Vehicle Systems</i>	7
2.1.2 <i>Adjacency and Laplacian Matrices</i>	7
2.1.3 <i>Rigidity and Persistence of Multi-Vehicle Formations</i>	8
2.2 FORMATION CONTROL TECHNIQUES	9
2.2.1 <i>Distance-Based Formation Control</i>	9
2.2.2 <i>Bearing-Based Formation Control</i>	10
2.2.3 <i>Orientation Alignment-Based Approaches</i>	11
2.2.4 <i>Consensus in Multi-Robot Systems</i>	11
2.3 LITERATURE SUMMARY	12
2.3.1 <i>Literature on Localization for MRS</i>	13
2.3.2 <i>Formation Control for MRS</i>	17
3. PERCEPTION MECHANISM FOR A THREE-LEADER SYSTEM.....	19
3.1 PROBLEM FORMULATION.....	20
3.2 REAL-TIME DISTRIBUTED FORMATION CONTROL.....	23
3.2.1 <i>Equation model of the closed-loop system</i>	23
3.3 LOCALIZATION FRAMEWORK.....	25
3.3.1 <i>Measurement Model</i>	25
3.3.2 <i>Estimation Model</i>	26
3.4 DRONE DETECTION	26
3.4.1 <i>Detection with Deep Learning</i>	26
3.4.2 <i>YOLO as a CNN architecture</i>	27
3.4.3 <i>YOLO Training</i>	28
3.4.4 <i>A comparison of detection methods</i>	29
3.5 EVALUATION OF THE FRAMEWORK.....	30
3.5.1 <i>Simulations</i>	30
3.5.2 <i>Experiments</i>	32
3.6 <i>Conclusion</i>	35
4. FORMATION ACQUISITION IN NONHOLONOMIC MULTI-VEHICLE SYSTEMS	36
4.1 NOTATIONS AND SYSTEM MODELING.....	37
4.2 PROBLEM FORMULATION.....	39
4.3 SENSING NETWORK	40
4.4 A TUTORIAL ON DEEP LEARNING FOR OBJECT DETECTION: TRAINING AND IMPLEMENTATION	41
4.4.1 <i>Dataset Generation</i>	41
4.4.2 <i>YOLOv3 and YOLOv4</i>	44
4.4.3 <i>YOLOv8</i>	46
4.5 LOCALIZATION.....	48
4.6 BASE FORMATION CONTROL LAWS	49
4.7 <i>Control Law without Magnetic Sensing</i>	51

4.8 EVALUATION	52
4.8.1 <i>Simulation</i>	53
4.8.2 <i>Experiments</i>	55
4.9 CONCLUSIONS	58
5. DESIGNING A FULLY SCALABLE FRAMEWORK FOR DECENTRALIZED SYSTEMS	59
5.1 THE PROPOSED APPROACH: AN OVERVIEW	60
5.2 TARGET TRACKING	61
5.3 FORMATION GRAPH CONSTRUCTION UNDER SENSING CONSTRAINTS	65
5.4 SYSTEM EVALUATION.....	68
5.4.1 <i>Simulation</i>	68
5.4.2 <i>Indoors Experiments</i>	71
6. ANALYSIS AND DISCUSSION	74
6.1 THE THREE-COLEADER CYCLIC AGENT MODEL ANALYSIS	74
6.1.1 <i>Analysis</i>	74
6.1.2 <i>Comparison with Other Approaches</i>	75
6.2 EMPOWERING CONVOLUTIONAL NEURAL NETWORKS WITH UWB FOR ENHANCED LOCALIZATION AND DETECTION	77
6.2.1 <i>Sensor Characteristics and Distance and Bearing Angle Estimation</i>	78
6.3 COMPARING PERFORMANCE	80
6.3.1 <i>Qualitative Comparison</i>	80
6.3.2 <i>Quantitative Comparison</i>	82
6.4 LIMITATIONS.....	84
7. CONCLUSIONS AND FUTURE PROSPECTS	85
7.1 CONCLUSIONS.....	85
7.2 SOCIETAL IMPACT AND CONTRIBUTION TO GLOBAL SUSTAINABILITY.....	86
7.2.1 <i>Societal and Scientific Impacts</i>	86
7.2.2 <i>Sustainable Development Goals</i>	87
7.3 FUTURE PROSPECTS: TOWARDS RL AND GNN.....	87
8. BIBLIOGRAPHY	89
9. CURRICULUM VITAE.....	98

LIST OF FIGURES

Figure 3.1 The local bearing angle γ_1 , the heading angle ψ_1 , and the absolute bearing angle ϕ_1 for robot R_1 are shown together with the body frame Σ_{b1} and the inertial frame Σ_1	21
Figure 3.2 The block diagram of robot R_i	22
Figure 3.3 The internal layers of the YOLOv3-Tiny with a sample input image and its detection output from an experimental environment [110].	28
Figure 3.4 (Left) An image segmented into 13×13 grids ($s = 13$); (Right) An example of two anchor boxes ($b = 2$) with the network output vector. Our version of YOLOv3-Tiny uses six anchors for each grid cell and has a single class [110]. ...	28
Figure 3.5 The DJI (left) and the F450 (right) used in the experiments [110]	32
Figure 3.6 The drones' trajectories in a simulation for $d_i = 5$ and $k = 0.3$ [110].....	34
Figure 3.7 Absolute distance errors e_i and bearing errors γ_i for $d_i = 5$ and $k = 0.3$; (c) Average distance and bearing errors for different initial conditions.[110].....	34
Figure 3.8 The EKF result of the closed-loop perception-decision framework [110]....	35
Figure 4.13 Sensors view in experiment: a) A detection from Jetson nano: A view from R_1 b) A scan from the LDS LIDAR seen by robot R_2 . [100]	52
Figure 4.14 Formation acquisition by expansion simulation: a) Trajectories of the agents, b) the inter-robot distance errors. [100].....	54
Figure 4.15 Formation acquisition by contraction simulation: a) Trajectories of the agents, b) the inter-robot distance errors.[100]	54
Figure 4.16 The robots and hardware used in the experiment.[100]	54
Figure 4.17 Contraction experiment: a) Trajectories of the agents, b) the inter-robot [100].....	56
Figure 4.18 Expansion Experiment. a) Trajectories of the agents. b) The relative distance errors.[100]	57
Figure 4.19 a) The error on R_1 during a formation contraction without magnetometers, b) A comparison of UWB sensor data with ground truth from Vicon capture system.[100].....	58
Figure 5.1 A block diagram illustrating the mechanisms integrated within the proposed framework for a four-robot system. The components related to the localization and control layers are given with the orange and blue backgrounds, respectively.	61
Figure 5.2 GM-PHD Experiments in a simulation environment: Each column represents a series of experiments at a specific distance while each row represents either the relative distance d_{ij} or the bearing β_{ij} from the follower agent $i = 0$ to the leaders $j = \{1, 2\}$ [111].....	64
Figure 5.3 a) Four Turtlebot robot agents in the Webots simulation; b) The underlying L-FF graph among the robots [111].	65
Figure 5.4 Vertex and edge addition operation in the Henneberg construction method. Two directed edges are added towards the previous vertices with every new vertex.[111].....	65
Figure 5.5 The Henneberg construction for three robots considering the visibility constraints. The camera of robot R_1 is directed towards its only leader, R_0 , while the camera of the second follower robot R_2 is directed towards its two leaders, R_0 and R_1 . The second follower robot R_2 is positioned on the line perpendicular to the line segment connecting its two leaders [111].....	67

Figure 5.6 Leader-Follower task with GM-PHD: a)-b) The measurement and prediction of the leader position on the relative x and y axes, respectively; c) The measurement of the relative distance, its desired value, and the filter output.	69
Figure 5.7 The trajectories of the agents in the simulations [111].	69
Figure 5.8 Simulation results: a) The error e_{10} between the First Follower R_1 and the leader R_0 ; b) The errors e_{20}, e_{21} ; and c) The errors e_{31}, e_{32} [111]	70
Figure 5.9 a) The agents in the webots simulation environment; b) The trajectories of the agents in the simulations with six robots. The dotted lines are the edges of the graph at the starting and end positions of the robots [111].....	71
Figure 5.10 The robots used in the experiment: a) The leader with no camera and the last follower with a camera and UWBs. b) Two Turlebots with UWBs, Jetson Nano, and camera. c) The four agents in formation in an experiment [111].	72
Figure 5.11 The trajectories of the agents during the indoor experiment. The dotted lines are the edges of the graph at the starting and end positions of the robots [111].....	73
Figure 5.12 The errors during the simulations: a) The error between the First Follower R_1 and the leader R_0 , b) The plot e_{20}, e_{21} during the indoors experiment c) The plot e_{31}, e_{32} during the indoors experiment [111]	73
Figure 6.1 (a) The distance measurement obtained with the approach of [15] (the experimental setup seen in the embedded image); (b) Comparison of the distance acquired by the UWB sensor (blue), the distance calculated by the algorithm of [16] (green), and the ground truth (yellow); (c) A sample distance-based relative position estimation framework [110].....	76
Figure 6.2 Detections at different distances in the simulation environment.	78
Figure 6.3 Detections at various distances in a room [111].	78
Figure 6.4 Distance measurement comparison between YOLO and UWB sensors. Left: The area of the detected bounding box versus the true distances between the robots, Right: Mean values of the UWB measurements versus the true distances between the robots [111].	80
Figure 6.5 Variances of the converted distance measurements with YOLO. Left: Behind configuration, Right: Diagonal configuration [111].	80
Figure 6.6 Comparison of our detection against Aruco marker: a) Two Aruco markers, b) The result from Aruco-based approach. C) The detection from a CNN-based approach [111]	81
Figure 6.7 The Comparison and applications: a) The sensors and methods used in the comparison: 1-LDS Lidar, 2-Intel RealSense Depth Camera D435i, 3-A whycon circular marker 4 -Ours: A camera and UWB [111].	82

LIST OF TABLES

Table 3.1 A comparison of the YOLOv3 and YOLOv3-Tiny.....	30
Table 3.2 RMSE of the distance and bearing errors for eight values of the design constant k [110].....	31
Table 3.3 RMSE of the distance and bearing errors for eight values of the design constant k [110].....	31
Table 4.1: A Comparison Table of four CNN architectures.....	44
Table 4.2 Advancements in YOLO CNN: From YOLO to YOLOv8.....	46
Table 6.1 The quantitative comparison between methods [111]......	83



LIST OF ABBREVIATIONS

CNN	Convolutional Neural Network
DL	Deep Learning
EKF	Extended Kalman Filter
FOV	Field of View
GNN	Graph Neural Network
GPS	Global Positioning System
IMU	Inertial Measurement Unit
MRS	Multi-Robot System
RL	Reinforcement Learning
ROS	Robot Operating System
SIFT	Scale-Invariant Feature Transform
SURF	Speeded-Up Robust Features
UAV	Unmanned Aerial Vehicle
UWB	Ultra-Wideband
YOLO	You Only Look Once (Object Detection Framework)



To my family

Chapter 1

Introduction

1.1 Research Context and Background

Significant contributions have been made to robotics in the last couple of decades. Indeed, robots are being increasingly employed in various industries and applications. However, among the many branches of robotics, Multi-Robot Systems (MRS) is one of the most promising research and application areas. MRS is a framework composed of multiple robots that work in a coordinated way to achieve common goals. It offers robustness, efficiency, and scalability advantages that are difficult to attain with single-robot systems [1]. MRS is beneficial when the tasks are too complex for a single robot system, require redundancy, and must be performed in parallel. An example is search and rescue. While a single robot may face a huge challenge to achieve a search of a large area, an MRS framework allows multiple robots to cover large areas simultaneously [2-3]. It is also applied in industrial automation, where robots can collaborate to perform complex assembly tasks more efficiently than a single robot would be able to.

Nonetheless, real-world applications of MRS face multiple challenges in their practical deployment. From the various problems, we can list a particular issue: a robust localization framework for swarm systems—an accurate solution for determining the position and orientation of each robot within the work environment. Precise localization is critical for robot coordination, collision avoidance, and mission task execution. Traditional localization methods, such as GPS-based ones, usually fail in indoor or GPS-denied environments [2, 4]. Moreover, depending on a single point of control or external infrastructure, centralized localization systems can be prone to failures and are challenging to scale for more significant numbers of robots.

In such challenges, there is a growing interest in developing decentralized localization systems that can rely only on onboard sensors without the need for external infrastructure. This dissertation discusses recent advances in sensor technology, computer

vision, and machine learning with regard to the state of the art of decentralized localization for MRS.

1.2 Motivation for the Research

Despite their vast application areas and advantages, the robot swarms are far from online implementations in real life. Some reasons include the lack of safe, reliable, and robust frameworks which can be applied on real swarms. Many localization algorithms depend on fixed infrastructures such as mocap systems and GPS sensors. Therefore, there is a huge need for a relative localization system framework that can be applied in any environmental setting. The proposed algorithms and frameworks in this thesis are expected to help bring the robot swarms closer to real life.

This research aims at overcoming the shortcomings of the current localization systems used in multi-robot applications. Most of the localization approaches developed for MRS rely on some sort of external infrastructure, such as GPS or fiducial markers, which may not always be available or practical in every environment. In particular, GPS-based systems are unreliable in indoor settings. Within a close environment, the GPS signals may be blocked or degraded. Similarly, systems based on physical markers are difficult to set up, and generalizing can be an issue in all environments [5-6].

Moreover, many existing localization systems are centralized, meaning all localization data is processed at a central point. This centralization introduces a single point of failure and limits the system's scalability, as the computational burden and communication overhead increase with the number of robots. In contrast, an onboard system, where each robot processes its localization data using solely its onboard sensors, can scale more effectively and is less vulnerable to single-point failures [7].

Recent advances in the field of Neural Networks and Deep Learning offer new opportunities to improve the accuracy and robustness of decentralized localization systems. For instance, convolutional neural networks (CNNs) have shown outstanding results in object detection and image recognition tasks. We found that this solution can be applied to relative localization in MRS. By integrating vision-based approaches with range sensor data (e.g., ultra-wideband (UWB) sensors), it is possible to develop a localization system that is both accurate and robust, even in challenging environments [8].

Motivated to achieve a decentralized localization framework, we propose a framework that can overcome the limitations of current systems and leverage the latest technology advances to improve performance. This framework would allow MRS to be more widely and effectively used in various applications, from industrial automation to real word scenarios response.

1.3 Research Questions and Hypotheses

We consider a multi-robot system (MRS) of N robot vehicles $R_1, R_2 \dots R_N$. Our aim is to design and implement a relative perception and coordination mechanism between the robots. The robots can be ground vehicles (e.g., a non-holonomic ground robot) or aerial vehicles such as drones. We classify the overall problem into the perception mechanism design and control mechanism design. In the perception part, our aim will be to design and implement an estimation framework where each drone has a situational awareness about its surrounding. In the control part, our goal will be to design and implement distributed formation control frameworks for the entire system so that the MRS moves as a whole in a given formation.

Given an N -robot MRS system $\{R_1, R_2 \dots R_N\}$ where each robot R_i has certain motion dynamics and sensing characteristics. We present the problems of interest as follows:

Problem 1: Design a sensing/communication graph among the MRS and relative state estimation mechanisms so that each robot R_i can localize itself in the MRS by using onboard sensors solely without any stationary infrastructure.

Problem 2: Design distributed formation control algorithms based on onboard perception feedback of the robots.

We focus on two types of sensor modalities: Vision and distance. Due to its low-cost, high signal-to-noise-ratio (SNR), and physical properties, we assume that each robot is equipped with a UWB sensor. Thus, we obtain the inter-robot distances from the UWB sensors onboard. Furthermore, we equip each robot with a vision sensor. Stereo cameras provide depth information together with image frames at the expense of additional computational complexity. On the other hand, monocular cameras possess favorable physical properties such as small size and weight. We will explore the advantages and downsides of the monocular and stereo cameras onboard of the robots.

Solution approaches to Problem 1 differ from the traditional localization methods where a set of beacons of a mocap system are placed at known locations of a room, and the position of a robot with a tag sensor is calculated at a ground station. Then, this location information is broadcast to the robots in the room. The differences stem from several factors. First, in a typical scenario of Problem 1, there is no stationary (fiducial) landmark or infrastructure that may help computing the robot locations. Therefore, the robots should take advantage of their onboard sensors or objects which are temporarily detected in the environment. Second, the onboard sensors are prone to drifts over time in the absence of reference landmarks. Therefore, the proposed frameworks should tackle possible drifts in measurements.

Although Problem 2 has been treated in the control theory literature extensively, only a small portion of the proposed approaches are applicable to robots in real life. For instance, many algorithms exist for the consensus and leader-follower formation control problems. However, most of these approaches are implemented in primitive simulation setups by assuming point agent or double-integrator agent kinematics. Drone motion can be modeled by a double-integrator kinematics under certain assumptions but real-time sensing modalities and characteristics affect significantly the original problem definition. For instance, the primary leader-follower formation control algorithms which satisfy rigidity constraints need to be modified extensively if the camera field-of-view and object detection characteristics are considered. To prove our hypothesis, we will study the integration of the vision and distance sensors on robots in realistic simulation environments and in real experimental setups.

1.4 Significance of the Research

This work provides significant outcomes for robotics and also for other related domains. In developing a decentralized, marker-free localization system, the work addresses some of the critical challenges that currently limit the deployment of multi-robot systems in real-world applications. Due to the independence of the proposed system from external infrastructures, such as GPS or fiducial markers, high versatility would be achieved, and adaptation to environments as diverse as industrial structured environments and unstructured outdoor environments, where GPS signals may become weak or unavailable, would be possible.

Using CNNs for object detection and tracking, the system can achieve better accuracy and robustness in challenging conditions and environments. This research adds to the knowledge of decentralized systems and brings new insights and techniques that can be applied to other areas in robotics and autonomous systems. The results from this research can enhance the autonomy and reliability of multi-robot systems so that more complex and coordinated tasks can be performed in various situations. This may benefit multiple robotic-based applications, such as search and rescue missions, which can use this localization framework for cases without solid GPS signals. The same can be said for industrial automation, where multiple robots performing tasks together can drastically improve efficiency and outcomes.

The development of this decentralized localization system is an important step toward the more widespread adoption of multi-robot systems in real-world settings, as it answers one of the basic challenges that has been preventing their deployment.

1.5 Thesis Structure

This thesis consists of seven chapters, commencing with an introduction chapter that establishes the groundwork for the research.

Chapter 2 presents a comprehensive introduction of the essential theories regarding multi-agent systems and formation control techniques. The document encompasses a comprehensive literature assessment of current research on formation control and relative localization, emphasizing significant gaps and limitations within the domain.

Chapter 3 examines the distinct issue of triadic leadership inside multi-agent systems. A proof-of-concept for our suggested system is provided, illustrating its capability to resolve this enduring issue. This chapter demonstrates how our architecture provides practical and scalable solutions to this issue.

In Chapter 4, we expand our paradigm to encompass nonholonomic vehicles, demonstrating its relevance to formation control issues. The chapter explores the incorporation of deep learning methodologies for image-centric training and the application of essential localization data, highlighting the pragmatic execution of our solutions.

Chapter 5 emphasizes the demonstration of our framework's scalability. An effective filtering mechanism is implemented to manage bigger systems, accompanied

with a novel method for graph generation based exclusively on camera field-of-view (FOV) data for relative localization. This chapter highlights the flexibility and effectiveness of our methodology in dynamic situations.

Chapter 6 presents a comparative analysis, examining our suggested paradigm in relation to analogous and existent research. It rigorously assesses the merits, drawbacks, and contributions of our methodology in comparison to the current state of the art.

Chapter 7 ultimately summarizes the thesis by encapsulating the principal findings and contributions. It delineates prospective research trajectories, highlighting the wider ramifications and possible advancements of this study in decentralized multi-agent systems.



Chapter 2

Preliminaries and Notation

2.1 Preliminaries and Notation

For the purpose of analyzing and modeling Multi-Robot Systems (MRS), this section addresses the fundamental concepts and methods that will be applied across the entirety of this work. These preliminary discussions include elements from formation attributes, such as graph theory, rigidity, and persistence, which are essential for decentralized control and formation maintenance in MRS. The following section is mainly derived from these books [9-10].

2.1.1 Graph Theoretical Modeling of Multi-Vehicle Systems

A general framework for modeling MRS is provided by graph theory. In this framework, each robot is represented as a node, and the edges indicate the interactions or communication links that exist between robots. It is possible to represent a MRS in a formal sense by using an undirected graph $G = (V, E)$, where $V = \{v_1, v_2, \dots, v_n\}$ is the set of nodes (robots) with $|V| = n$, $E \subseteq V \times V$ is the set of edges, where each edge $e_{ij} = (v_i, v_j) \in E$ represents a communication or sensing link between robots i and j .

The connectivity of the graph G plays a significant role in the information flow across the MRS. A connected graph ensures that there is a communication path between any two robots, enabling the sharing of localization, control, and perception information among the robots. The degree of each node d_i reflects the number of robots that are directly connected to robot i , which characterizes the local interaction between groups of robots.

2.1.2 Adjacency and Laplacian Matrices

The interactions that take place between robots in an MRS can be represented in matrix form which provides a formal method when formulating control rules and carrying out stability analysis. The adjacency matrix $A \in \mathbb{R}^{n \times n}$ in an MRS is defined as:

$$A_{ij} = \begin{cases} 1, & \text{if } (v_i, v_j) \in E, \\ 0, & \text{otherwise.} \end{cases} \quad (2.1)$$

The adjacency matrix expresses the interactions among the pairs of nodes in a graph and is symmetric for undirected graphs, where $A_{ij} = A_{ji}$. Additionally, the Laplacian matrix $L \in \mathbb{R}^{n \times n}$ is defined as:

$$L = D - A \quad (2.2)$$

Where the diagonal matrix $D \in \mathbb{R}^{n \times n}$, $D_{ii} = d_i$, is the degree matrix. The Laplacian matrix shows how connected the graph is, and its features, e.g., eigenvalues, play an important role for consensus and formation control algorithms. For instance, for a connected graph, the smallest eigenvalue of L is zero, with the corresponding eigenvector being 1_n (a vector of all ones), whereas all other eigenvalues are positive. Particularly, the second smallest eigenvalue of L determines convergence properties of consensus-based control systems.

2.1.3 Rigidity and Persistence of Multi-Vehicle Formations

In order to accomplish several types of tasks with an MRS, the robots must move in a certain shape or formation. Rigidity refers to the maintenance of the shape of a team of robots as they move. Several formation control architectures use the ideas and methods from the rigidity theory to ascertain that robots stay in a certain shape while performing a given task. Rigidity theory states that if the distances between specific robot pairs don't change with motion, the structure of the formation will be maintained. Therefore, by using the rigidity theory, one can assess if the form of the robot team will be modified or not during motion.

There are two categories of rigidity: Global rigidity and local rigidity. If a formation maintains its shape in the Euclidean plane under any arrangement, it is said to be globally rigid. In a globally rigid formation, any two realizations of the graph, obeying the distance constraints on the edges, differ by only translation, rotation, and reflection. Thus, the entire realization moves as a rigid entity as long as the distance constraints are satisfied, and it is impossible to distort the entire structure without altering the distance between at least one pair of robots. Contrarily, in a locally rigid graph, the rigidity is

maintained around a realization, and it is not guaranteed that the rigidity will be maintained for all realizations.

The rigidity properties of a matrix can be determined using a rigidity matrix $R \in \mathbb{R}^{|E| \times 2n}$, which holds the constraints about the distances between robots in the graph. For a 2D formation with n robots and m edges, the rigidity matrix R is given as:

$$R_{ij} = \begin{cases} (p_i - p_j) / \|p_i - p_j\|, & \text{if } e_{ij} \in E \\ 0, & \text{otherwise} \end{cases} \quad (2.3)$$

A framework is locally rigid if $\rho(R) = 2n - 3$ on a 2D plane, where $\rho(\cdot)$ denotes the rank of the matrix. indicating that constraints are satisfied to maintain the base structure. Rigidity is an important feature in formation control because it can be used to maintain a formation's shape by focusing only on the desired constraints between the robots.

It is also important to maintain the constraint persistence. Persistence encompasses rigidity and constraint persistence. It is equally important to maintain the structural integrity of the graph in MRS and maintain oriented graph constraints between agents. It allows under motion to preserve the geometry constraint given the graph theoretic criteria. Persistence is especially important in leader-follower formations, where designated robots (leaders) set the movement trajectory while followers maintain certain relative locations and orientations. Persistence in such formations is critical for maintaining the overall structure as it navigates across changing terrain or obstacles.

2.2 Formation Control Techniques

Formation control is needed in Multi-Robot Systems (MRS) to make sure that a group of robots stays in a certain shape as they move through an area. Formation control can be done in a number of ways, such as using distance-based, bearing-based, or orientation alignment-based methods. Each method has its own benefits and uses, and different combinations of these methods are often used to make the overall control more effective and flexible.

2.2.1 Distance-Based Formation Control

The basic goal of distance-based formation control is to maintain certain distances between surrounding robots while achieving the required formation configuration. Each robot changes its location based on the relative distances between its neighbors within the

graph. The distance between two robots i and j , denoted as d_{ij} , is maintain according to a predefined desired distance d_{ij}^* . The error distance between robots i and j is defined as:

$$e_{ij} = \|p_i - p_j\| - d_{ij}^*, \quad (2.4)$$

where p_i and p_j are the positions of robots i and j , respectively, and $\|\cdot\|$ denotes the Euclidean norm. The control input u_i for each robot i can be designed to minimize the distance error with respect to its neighbors. A common control law is:

$$u_i = -k_d \sum_{j \in \mathcal{N}_i} e_{ij} \frac{p_i - p_j}{\|p_i - p_j\|}, \quad (2.5)$$

where k_d is a positive control gain that determines the convergence speed, $\mathcal{N}_i$ represents the set of neighbors of robot i , e_{ij} is the distance error between robots i and j . This control law ensures that each robot moves in the direction that reduces the distance error between itself and its neighbors, eventually converging to a formation where all inter-robot distances match the desired distances d_{ij}^* . Distance-based control is advantageous for maintaining specific geometric shapes and is commonly used in applications where formation rigidity is essential.

2.2.2 Bearing-Based Formation Control

Formation Based on Bearing measurements control prioritizes preserving particular bearings between robots over only distance distances data. This method is advantageous in contexts where distance measurements are inconsistent, but bearing data are available in systems such as in camera-based systems. Let ϕ_{ij} denote the relative bearing angle between two robots i and j . The bearing control is to maintain a desired bearing angle ϕ_{ij}^* between the two robots. The bearing error can be defined as:

$$e_{ij}^\phi = \phi_{ij} - \phi_{ij}^*, \quad (2.6)$$

where ϕ_{ij} is the estimated bearing angle, and ϕ_{ij}^* is the desired bearing angle. The control input each agent i is designed to reduce the bearing error with respect to its neighbors. A typical control law for bearing-based formation control can be illustrated as follows:

$$u_i = -k_\phi \sum_{j \in \mathcal{N}_i} e_{ij}^\phi (p_i - p_j), \quad (2.7)$$

where k_ϕ is a positive control gain. Bearing-based formation control is particularly useful for tasks that require alignment in direction, such as coordinated surveillance or following a specific heading, and is often employed when robots rely on visual sensing systems to maintain formation.

2.2.3 Orientation Alignment-Based Approaches

Orientation alignment-based approaches aim to line up the heads or orientations of all robots in the MRS. This method is needed in applications that require a common direction of during the mission, such as maintaining a parallel formation. We can define θ_i as the orientation of robot i . The goal is for each agent to align its orientation θ_i with the desired orientation of its neighbors. The orientation error e_i^θ for robot i relative to its neighbors can be illustrated as follows:

$$e_i^\theta = \theta_i - \bar{\theta}_i, \quad (2.8)$$

where $\bar{\theta}_i$ is the orientation of the neighbors agent i is calculated as:

$$\bar{\theta}_i = \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} \theta_j \quad (2.9)$$

The control input u_i^θ for orientation alignment is defined to minimize the global orientation within the MRS. A control law for orientation alignment can be written as:

$$u_i^\theta = -k_\theta e_i^\theta \quad (2.10)$$

where k_θ is the control gain for the convergence rate. Orientation alignment-based control is especially useful in leader-follower formations, where a designated leader robot defines the desired orientation, and all other robots align themselves accordingly. This method allows the MRS to maintain a coherent formation direction, which is critical for tasks requiring collective movement in a specified direction.

2.2.4 Consensus in Multi-Robot Systems

Consensus algorithms are important in MRS because they allow robots to agree on common states or objectives like location, velocity, and heading. Decentralized MRS achieves agreement through local interactions, in which each robot adjusts its state based on information received from its neighbors. For a group of robots with positions $x_i \in \mathbb{R}^d$, the consensus protocol can be expressed as:

$$\dot{x}_i = - \sum_{j \in \mathcal{N}_i} (x_i - x_j), \quad (2.11)$$

where $\mathcal{N}_i$ denotes the set of neighbors for robot i . This technique ensures that all robots converge to a common position over time, assuming the graph G is connected. Consensus-based control is especially beneficial in formation control, when each robot adjusts its position to match its desired constraints to its neighbors. Consensus algorithms'

convergence speed is impacted by the graph's algebraic connectedness, represented by the second smallest eigenvalue of the Laplacian matrix L . Higher algebraic connection indicates faster convergence; hence the graph structure is an important component in the performance of consensus algorithms.

2.3 Literature Summary

Multi-robot systems (MRS) can accomplish demanding objectives which may be intractable for a single robot. Recently, MRS have been employed in many civilian and industrial tasks such as cooperative transportation of objects and choreography illustrations. Area coverage is another application example where a MRS perform superior to a single robot [11]. In a MRS application, the objective can be divided into simpler commands and assigned to the individual robots based on a predefined graph structure. While MRS can be controlled by a central unit which can broadcast the control action for each robot, decentralization offers unique opportunities such as scalability and robustness to single point of failure. Moreover, decentralized and distributed methods are more appropriate for outdoor applications.

MRS operations necessitate a robust and resilient coordination algorithm among the robots, which calls for accurate positioning of robots during an operation. In outdoor environments, global positioning systems (GPS) may be used on each robot with high precision. However, while GPS can provide absolute position in outdoor environments, the GPS sensors may not provide sufficient accuracy in many swarm applications (± 3 meter precision). Also, the GPS signals are not trustable in GPS-denied regions such as indoors, inside tunnels, and confined urban locations. In indoor environments, motion capture (mocap) systems can be used as a localization solution. These systems include a set of visual sensors (e.g., infra-red cameras) or distance sensors (e.g., ultrawideband, Bluetooth, or wi-fi sensors) together with a ground station which computes the 3D pose of a vehicle. Although such systems provide centimeter-level accuracy for agent localization, they are entirely centralized, demand existing infrastructures, and are immobile. Therefore, robotic agents in an MRS cannot trust such infrastructures in the environment for positioning: If a multi-drone system is designed to operate both indoors and outdoors, then the drones must exploit their onboard capabilities to execute a given task. We divide the related literature into two categories: Localization and control.

2.3.1 Literature on Localization for MRS

MRS localization can be posed as the localization of individual robots in the MRS according to a common coordinate frame. For instance, a team of N robots can calculate their absolute positions in a room by using a mocap system and use this information in their controllers. However, such a framework would not be distributed nor decentralized since the entire framework relies on a central infrastructure. On the other hand, the robots can employ their onboard sensors to calculate their relative states with respect to their neighbor robots, referred to as onboard localization. Due to its versatility and robustness, we focus on the decentralized and distributed architectures in this thesis. Various decentralized localization methods/frameworks based on onboard sensors have been investigated for reliable MRS navigation in the literature. There are several considerations in choosing the right type of sensors for a particular application. We summarize some recent works below classified based on the type of the onboard sensors used.

Basiri et.al. [12] used ultrasonic sensors as a way for relative bearing estimation. By mounting a microphone array on each drone in the MRS, the time difference between the arrival times of chirp signals on the microphones are used to estimate the relative bearing between the agents. Although the validity of the proposed approach is demonstrated on a group of drones with onboard resources, the motor and acoustic noises affect significantly the localization performance. In [13], a relative localization method is presented where the agents communicate the state variables such as velocity, height, orientation with each other by using wireless transceivers onboard. Further, the authors used ultrawideband (UWB) sensors for inter-robot distance measurements instead of using the received signal strength (RSS) and demonstrated improved performance in [14]. UWB sensors offer unique advantages such as small size and weight, low-cost, omnidirectional measurements, high signal-to-noise ratio (SNR), and non-line-of-sight (NLOS) measurement capability [15-18]. The authors equip robots with UWB sensors to estimate the relative positions between robots in a two-robot system. Particularly, three UWB anchors are installed on a robot which serves as a mobile anchor platform to estimate the location of another robot with a single UWB tag sensor. A series of successful experiments on both ground robots and drones validate the effectiveness of the framework as a viable infrastructure-free solution for both indoors as well outdoors. The frameworks in [19] present a similar approach with varying numbers of UWB sensors on the robots such as two anchors and two tags.

Due to their small size and practicality, cameras have long been employed on robots in various applications. Camera feedback can be used to localize a robot in a given environment or generate a map of the environment. Cameras can provide excessive useful information about the environment such as the size and scale of the objects around the robot. The physical properties of cameras such as low weight and small size make it an ideal candidate for micro-aerial vehicle applications. A leading work in this direction demonstrated cooperative control of nonholonomic mobile robots based on omnidirectional cameras onboard [20]. This approach relied on a ground station which constantly communicated videos and control signals with the robots. The images acquired on the robots are calibrated and analyzed with digital image processing techniques for edge detection and collision avoidance. Camera calibration can be time-consuming for multi-robot systems (MRS), and omnidirectional cameras may not be available for all agents. The authors in [21] relaxed the calibration requirement in leader-follower formations of nonholonomic mobile robots by using the unscented Kalman filter (UKF). This concept was revised in [22] where a formation control is introduced based on an omnidirectional or perspective onboard camera without the need for calibration. Fidan et.al. [23] proposed a single-view distance estimation scheme in which a camera with limited FOV is used on each ground robot to localize the leader robots in the team for satisfying rigid formation constraints. The FOV restriction of a single camera is addressed in [24] in which a set of ground robots with onboard omnidirectional cameras observe and guide a team of ground robots.

A typical approach for the visual localization of mobile robots is to equip the robots with distinguishable patterns. The leader robot(s) are mounted with feature patterns onboard to be captured by the follower robot(s). These features can be in the form of specific features or colors. An adaptive vision-based leader-follower control framework for mobile robots was investigated in [25]. Another common strategy in the literature is to include fiducial markers onboard the robots or in the environment [26-28]. Due to its simple identification, circular patterns were of particular interest in these studies. In [29], [30-31], patterns with circles in grayscale are used to estimate the relative positions among mobile robots. This approach is robust and fully decentralized but prone to illumination conditions. Also, the FOV specifications put an important restriction on the formation configuration. In [32], ultraviolet (UV) blinking markers are mounted onboard the robots for visual relative localization of multi-drone systems. A downside of this

approach lies in the sensing mechanism itself: UV markers may be unobservable in some environmental settings.

Visual-inertial localization and mapping have been studied as well in multi-robot applications where an external measurement infrastructure is unavailable. Montano et.al. [33] addresses the localization problem for multi-UAV systems. The UAVs are equipped with a downward facing camera, and the images acquired are compared among UAVs to compute the relative positions between the robots. Visual-inertial odometry is used to coordinate 12 quadrotor robots in [34]. The robots are assumed to be informed with their initial position and the features on the map to maintain particular formations. A partially decentralized approach for the localization of ground robots by a set of drones is presented in [35], where a reference image on the ground is used for localization.

With the advancements of the neural networks, marker-less object detection has become possible. Schilling et.al. [36] proposed a fully decentralized swarm that can learn to coordinate agents by imitating. A controller derived based on a CNN which predicts the velocity commands for the formation. Although this approach provides a completely marker-less approach, it requires omnidirectional cameras mounted on the drones which may not be suitable for micro-aerial vehicles (MAV). In [37], a neural network is trained to separate MAVs from the environment. This architecture is further validated for vision-based multi-MAV localization in [38]. The authors designed a coupled probabilistic data association filter for localization of multiple agents using vision and IMU sensors. Vrba and Saska investigate a marker-less drone detection with an onboard camera by incorporating a CNN algorithm [39]. The bounding boxes obtained from the detection were used to determine the bearing angle, and the relative distance was inferred using geometry from the known UAV dimensions. This approach puts several constraints on the physical dimensions of the UAVs to be detected. The conventional computer vision techniques have focused on designing algorithms that best describe the features on the acquired images. This line of research centered around extracting informative visual features that can provide a robust description of objects in an image. In mobile robotics, such features are designed as edges, corners, and color-related properties of objects in the environment. These features are extracted through several fundamental techniques such as color thresholding, edge detection, corner detection, and visual segmentation. Some prominent methods used for feature extraction include scale-invariant feature transform (SIFT) [40], speeded-up robust features (SURF) [41], features from accelerated segment test (FAST) [42], and histogram of oriented gradient (HOG) [43].

In contrast to the conventional computer vision methods, the machine learning algorithms derive logical patterns from a given set of data by training weights of a neural network. As an advanced machine learning technique, deep learning algorithms use multiple hidden layers in the network to extract higher level features from data [31]. The deep learning methods require large amount of data and high computational resources. With the rapid development in the computational power of computers, deep learning has attracted huge interest in the last decade. The development in the parallel computing systems such as the graphics processing units (GPU) has enabled the applications of the deep learning methods. Many large datasets were collected and recorded as benchmarks for deep learning algorithms [32]. Several variations have emerged in the deep neural network (DNN) family such as convolutional neural network (CNN), recurrent neural network (RNN), and deep belief network (DBN). Particularly, CNNs have been used heavily in computer vision and robotic perception [44].

Production of small-size, powerful GPU boards have enabled the use of CNN algorithms on mobile robots. In a MRS, a robot can detect another robot and infer the inter-robot relative states. Drone-vs-bird detection challenge was one of the main thrusts for real-time drone detection research [45]. In this challenge, an image dataset containing both drones and birds in the scenes was supposed to be trained so that the resulting model would be capable of detecting drones as true positives. Saqib et.al. [41] used a transfer learning method and utilized a Faster RCNN architecture and tuned the model with VGG16 for the classification. In the same challenge, the authors of [43] used three steps to accurately detect and segment the drones. First, they used U-Net to segment candidates from the background, and then targets are passed to a CNN classifier ResNetV2 to classify the class of the targets. Finally, they add another layer where they used multiple frames to track the object and apply thresholding to eliminate false targets. To enrich the drone dataset and improve detection performance, Aker and Kalkan [46] derived a method to artificially construct a dataset by combining UAV photos, environment background, and additional items in the scene. In the same direction, a method for synthetically generating the images for training the network was investigated in [47]. Spatial and temporal context information fusion was employed to detect the flying UAVs in [48-49]. To highlight a flying object in a sequence of images, both appearance and motion shift can be utilized. Filtering algorithms can be associated to compensate for the motion errors. A slightly different method is proposed to differentiate UAVs from aircraft in [50], which aims at recognizing UAVs by utilizing two cameras with intersecting FOVs. A customized

version of YOLO is integrated into a Kalman filter-based estimation algorithm. Moreover, the conventional computer vision methods such as SIFT and SURF can be combined with machine learning to take advantage of both approaches [51].

2.3.2 Formation Control for MRS

Early pioneers focused on the problem of triangle formation control. While a group of two agents can only represent a line on a plane, adding a third agent allows the agents to attain a triangular shape. As a result, initial work in the field emphasized the concepts of three agents on a plane. Reference [52] is a primary work that laid some theoretical foundations of the concepts on multi-agent formations. Their work defined the problem of formation control in a triangular shape. They demonstrated a stable rigid formation framework of point vehicles and proposed possible controls laws for nonholonomic vehicles as well. Similarly, in [53], the authors reshaped the framework to an equilateral polygon. They suggested a control law based on gradient descent to converge to a non-collinear equilibrium.

In the same direction, [54] pointed out the problem of cyclic triangular graphs, known as the co-leaders. Unlike previous contributions, they introduced the concepts of consistencies between agents. They highlighted a directed sensing graph between agents in a cyclic framework. They first defined a fixed heading direction for all agents. Then, with this assumption, the authors used work the relative distances and the inner angles which are co-related to the fixed headings to prove the stability of the triangle framework. Although this work was effective, the inner angles were considered as additional variables in this triangular formation since the constraints were defined as the relative distance between agents. Moreover, obtaining these angles was merely linked to the fixed heading constraint restricted on the framework. Therefore, the later work [55-56] presented the framework of a rigid graph with only distance constraints between agents. These studies are effective in demonstrating the formation of 3 agents. However, the scope is restricted to a triangular shape and more generalized approaches had to be investigated to adapt to more agents. [57-59] considered control laws that are adaptable to more than three-agent cases. While their work illustrated the particular instance of a triangle formation control, still they established the expansion to different shape.

For an arbitrary number of robots, a polygon shape can be constructed. Yet, this leads to more constraints within the framework. As a result, the paradigm shifted from the formation shapes to the sensing and communications in the graph. Fundamentals

concepts such as directed, persistence, and graph rigidity became crucial in the definition of the problem framework. Excellent reviews journals in [60-61] explain strongly these notions. Therefore, we can categorize the literature into the distance-based and the bearing-based approaches.

In this section, the realization of the control framework is based on the relative distance between agents. The works [60], [62] studied the purely distance-based formation of points agents. An interesting point of these papers is that the authors directly derived the equations based on the matrix rigidity theory. References [63-64] further addressed the issue of the cyclic and acyclic structure of the graph and the persistence requirement of the distance-based graphs. They also highlighted the concepts of agents with different degrees of freedom. The framework is divided into a leader with degree 0, a first follower with degree 1, and ordinary followers out-degree larger or equal to 2. These concepts are well detailed in [54].

In parallel to the distance-based approaches, a relative bearing between agents is an alternative sensing vector. This approach was theoretically formulated in the works [65-66] and gained large attention lately due to various factors. First, there is a significant potential of bearing derivation using a camera on mobile robots. This allows an easy transition from the theoretical control's laws to practical experiments. Secondly, bearing rigidity is conserved from a 2D space to a 3D space without extra parameters on the matrix [67]. Thirdly, bearing allows formation scaling of the formation while preserving the bearing constraints. This feature is useful for obstacle avoidance of multi-agents [68]. In [69-70], the authors proposed a leaderless bearing-based formation framework. In their analysis, they assumed that each agent has means to measure their relative bearing and features in the environment. In [65], [67], the authors explored the bearing rigidity concepts and a theoretical comparison to distance rigidity. They demonstrated the stability of the controls with and without global reference frames. A leader-first follower structure was introduced in [71] using only bearing measurements. The authors proved the global stability of the formation and demonstrated a rotation and scaling of the entire formation using the leader orientation.

This thesis will focus on the distance and bearing in the sensing and communication graphs. Thus, we have presented a short survey of formation control categorized mainly into two groups distance and bearing. However, other forms of sensing modalities exist in the literature such as Orientation Based formation Control [72], subtended based formation Control [73].

Chapter 3

Perception Mechanism for a Three-Coleader System

The design of a coordination framework for a multi-robot system (MRS) requires analyzing task suitability and feasibility, constructing a formation graph among the robots, proposing a reliable perception-action mechanism, and conducting system evaluations. Although each of these four steps warrants independent analysis, they are all interdependent. For instance, the design of a perception-action mechanism cannot be separated from the formation graph structure, and experimental system evaluations may necessitate reanalyzing the feasibility of the entire system. Therefore, the steps of the MRS design process must be assessed collectively.

The main goal of this thesis is to design reliable, flexible, and robust perception-action mechanisms for MRS by using the state-of-the-art sensor technology. We aim to propose task-agnostic frameworks that can be implemented on several types of robotic platforms to tackle variety of missions with minimum modifications. Evidently, such perception mechanisms should take advantage of multiple sensing resources. Particularly, the mechanisms devised here rely on the combination of distance and vision sensors.

In this chapter, we devise a perception mechanism integrated into the three-coleaders control problem which was originally studied in [54]. In [54], three-point agents maintain a triangle formation with desired edge lengths within a cyclic constraint network. References [74] and [110] extend the result of [54] to the cyclic formation of three quadcopter drones. This modification in the problem setup leads to the need for redesigning the perception and action mechanisms for the drones. In this chapter, we revisit the results of [74] and [110] and provide the detailed analysis of this modification.

We propose a framework solution to this theoretical problem using kinematics and real-time perception. Specifically, we use a bearing angle controller to control the heading

angles of the drones, and a deep learning-based detection technique to establish their relative positions. Assuming that distance and bearing measurements are noisy, we investigate the formation control framework's stability and convergence, develop a multimodal localization algorithm based on sensor fusion, and test it on real hardware.

We provide the findings of a full simulation study of the entire algorithm, as well as a series of real-world trials on a prototype system. The suggested technique results in a closed-loop architecture in which the deep learning-based detection output is used as a feedback signal by a control algorithm. We demonstrate that, due to the exponential convergence property of the proposed formation control method, the formation constraints are met in a finite time up to a small error, even with noisy feedback. Furthermore, the entire system is modularly constructed, allowing each module to use a variety of different perceptual and decision-making methodologies. As a result, our design provides a viable end-to-end solution for future drone swarm applications.

This chapter study resulted into a journal published in the IEEE/ASME Transactions on Mechatronics [110], and the content are copyrighted © 2022 by IEEE.

3.1 Problem Formulation

We propose a distributed perception-decision framework for a three-drone system that forms a digraph. We divide the primary goal into two categories: formation control algorithm design and localization algorithm design. The formation control algorithm will produce control inputs for each drone in order to illustrate swarm behavior independent of external localization infrastructures. The localization method uses each drone's onboard sensors to provide relative location estimates, which are then used as feedback in the formation control algorithm.

Consider three drones $\{\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3\}$ each with a first-order motion model:

$$\dot{\zeta}_i(t) = \begin{bmatrix} \mathbf{u}_x^i(t) \\ \mathbf{u}_y^i(t) \\ \omega_i(t) \end{bmatrix} = \mathbf{R}_z(\psi_i(t)) \begin{bmatrix} {}^{b_i}\mathbf{u}_x^i(t) \\ {}^{b_i}\mathbf{u}_y^i(t) \\ \omega_i(t) \end{bmatrix} \quad (3.1)$$

where $\zeta_i = [x_i, y_i, \psi_i]^T$ is the state vector, x_i, y_i are the planar coordinates, ψ_i is the heading angle, and $\mathbf{u}_x^i, \mathbf{u}_y^i$ are the control inputs in the x, y axes of drone $\mathcal{R}_i, i = (1, 2, 3)$, all with respect to the inertial frame Σ^I , $\mathbf{R}_z(\psi_i) \in \text{SO}(3)$ is the rotation matrix and ω_i is the angular velocity about the inertial z -axis, respectively. The signals ${}^{b_i}\mathbf{u}_x^i, {}^{b_i}\mathbf{u}_y^i$ denote the velocity inputs in the body-attached frame Σ^{b_i} of $\mathcal{R}_i$.

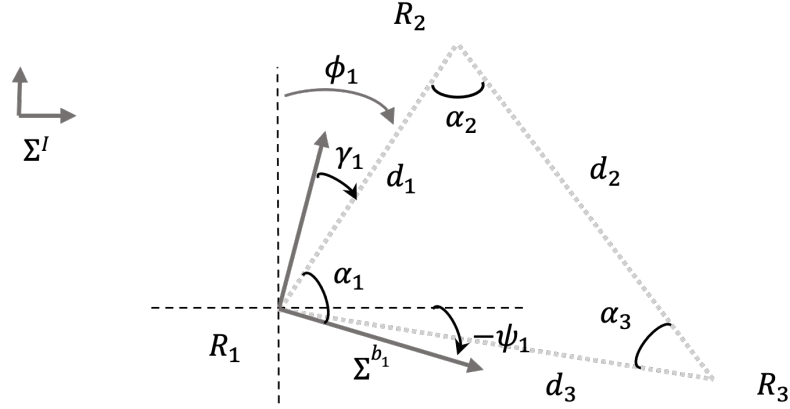


Figure 3.1 The local bearing angle γ_1 , the heading angle ψ_1 , and the absolute bearing angle ϕ_1 for robot $\mathcal{R}_1$ are shown together with the body frame Σ^{b_1} and the inertial frame Σ^I

An aerial vehicle can be modeled as a six degree-of-freedom (6-DOF) rigid body with three linear (x, y, z) and three angular (roll, pitch, yaw) velocities. In this work, we assume that an onboard flight controller regulates the roll and pitch angles and maintains the vehicle altitude (z) at a certain height. Therefore, the complete drone model can be reduced to (3.1) to facilitate the controller synthesis process.

We assume that the drones do not have magnetometers. The fundamental reason for this assumption is that magnetometers are known to deliver measurements with low signal-to-noise ratio in demanding conditions, affecting system reliability. Thus, we extract the control inputs in the body frames Σ^{b_i} of the drones, which has various advantages in real-world applications. Because the drones do not require external infrastructure to operate, we can create completely distributed formation control algorithms. We model the distance and bearing constraints of the three-drone system $\{\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3\}$ as a digraph $\mathcal{D}(\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = \{v_1, v_2, v_3\}$, $\mathcal{E} = \{(v_1, v_2), (v_2, v_3), (v_3, v_1)\}$, where v_i denote the nodes of the graph. Notably, this design creates a cyclic sensing and constraint graph, where each drone $\mathcal{R}_{i-1}$ should maintain the relative constraints toward $\mathcal{R}_i$, $i \in \{1, 2, 3\}$. We name each drone $\mathcal{R}_i$ the coleader of $\mathcal{R}_{i-1}$. Let $\bar{d} = \{\bar{d}_1, \bar{d}_2, \bar{d}_3\}$ be a constraint consistent desired distance set for $\mathcal{D}$, i.e., $\mathcal{D}$ is realizable with $\bar{d}$. We denote the distance between robot $\mathcal{R}_i$ and its coleader $\mathcal{R}_{i+1}$ by

$$d_i = \|\mathbf{p}_i - \mathbf{p}_{i+1}\|, \quad i \in \{1, 2, 3\}. \quad (3.2)$$

We assume that there exists a sufficiently small $\epsilon > 0$ such that the following conditions are satisfied for all $i, j, k = (1, 2, 3), i \neq j, j \neq k, k \neq i$:

$$|\bar{d}_i - \bar{d}_j| < \bar{d}_k - \epsilon, \quad |\bar{d}_i + \bar{d}_j| > \bar{d}_k + \epsilon \quad (3.3)$$

Notably, by imposing (3.3), we exclude the collinear drones case in the desired configuration. Furthermore, we assume that $d_i(t) > d_{\min}$ for some arbitrary $d_{\min} > 0$ and all $t \geq 0$, which can be guaranteed with a simple collision avoidance mechanism among the robots. Denoting by l_i the line segment from $\mathbf{p}_i$ to $\mathbf{p}_{i+1}$ (Fig. 3.1), the relation between the absolute bearing angle $\phi_i \in [0, 2\pi)$ and the local bearing angle $\gamma_i \in [0, 2\pi)$ can be found as [23]:

$$\gamma_i = \psi_i + \phi_i \quad (3.4)$$

$$\phi_i = \frac{\pi}{2} - \text{atan}(\mathbf{p}_i, \mathbf{p}_{i+1}), \quad (3.5)$$

where $i = \{1, 2, 3\}$, $\psi_i \in [0, 2\pi)$ is the heading angle of robot $\mathcal{R}_i$, and $\text{atan}(\cdot, \cdot): \mathcal{R} \rightarrow [0, 2\pi)$ is the arc tangent function evaluated with respect to the inertial x -axis. Also, denote by α_i the internal angle of the triangle formed by the drones at node v_i . Let β_{fov} be the FOV of the cameras on drones.

We define our key objective as follows. Consider the three-drone system $\{\mathcal{R}_1, \mathcal{R}_2, \mathcal{R}_3\}$ with the associated cyclic constraint graph $\mathcal{D}(\mathcal{V}, \mathcal{E})$. Assume that the initial configuration of the robots are not collinear and that each robot $\mathcal{R}_i$ measures constantly the noisy distance and the local bearing angle

$$d_i^m = d_i + \eta_i^d, \quad \gamma_i^m = \gamma_i + \eta_i^\gamma, \quad (3.6)$$

where $|\eta_i^d(t)| \leq \bar{\eta}^d$, $|\eta_i^\gamma(t)| \leq \bar{\eta}^\gamma$ for arbitrary $\bar{\eta}^d, \bar{\eta}^\gamma$ and all $t \geq 0$. The first problem is to design a distributed control algorithm for each drone so that the configuration of the underlying graph $\mathcal{D}$ satisfies all distance constraints in $\bar{d}$. The second problem is to propose a filtering algorithm to estimate the controlled variables d_i, γ_i .

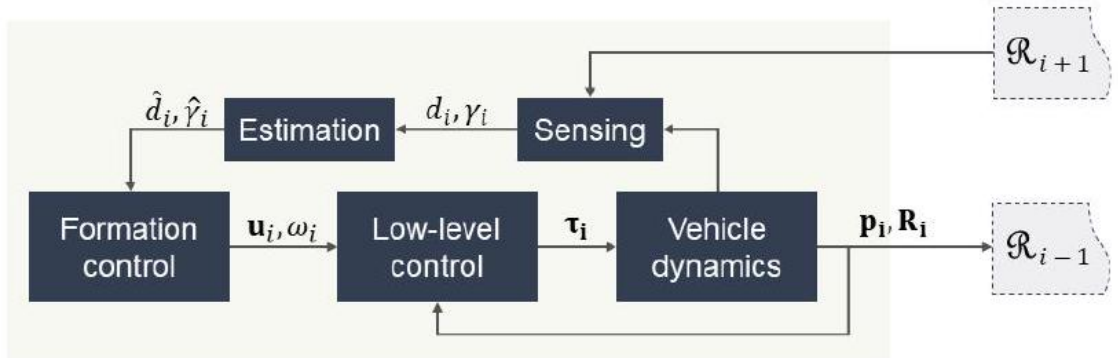


Figure 3.2 The block diagram of robot $\mathcal{R}_i$.

3.2 Real-time Distributed Formation Control

In this section, we provide a high-level overview of our design approach, provide a formation control method, and evaluate its stability properties. Fig. 3.2 depicts a block diagram of the system. The Low-level Control block functions as the drone's autopilot, generating the needed torque instruction for the motors based on the drone's IMU feedback. This Formation Control block provides the velocity commands $\mathbf{u}, \omega$ based on the filter output $\hat{d}, \hat{\gamma}$ which are produced by the Estimation block. Because we want to develop a distributed control method, we construct the control laws for each robot separately. Our solution tries to achieve the desired configuration of the three-robot system by managing each robot's planar motion and local bearing. Hence, robot $\mathcal{R}_i$ tries to preserve the distance between itself and robot $\mathcal{R}_{i+1}$ at its desired distance $\bar{d}_i$ and to control the bearing angle γ_i . We use the following control law for drone $\mathcal{R}_i$ which was proposed in [74] and [110]:

$${}^{\mathbf{b}_i}\mathbf{u}_x^i(t) = e_i(t)\sin(\gamma_i(t)) \quad (3.7)$$

$${}^{\mathbf{b}_i}\mathbf{u}_y^i(t) = e_i(t)\cos(\gamma_i(t)) \quad (3.8)$$

$$\omega_i(t) = -k\gamma_i(t) \quad (3.9)$$

$$e_i(t) = d_i(t) - \bar{d}_i \quad (3.10)$$

where $k > 0$ is a design constant and $e_i(t)$ is the distance error. The control law (3.7)-(3.8) aims at regulating the distance error e_i by actuating the robot in the x and y axes of the body frame $\Sigma^{\mathbf{b}_i}$ whereas the regulation term (3.9) modifies the heading of the robot such that the positive y axis of the frame $\Sigma^{\mathbf{b}_i}$ aligns with the line segment l_i in the steady-state.

The controller generates a fully decentralized algorithm because drone $\mathcal{R}_i$ derives its control inputs by using the variables related to its neighbor $\mathcal{R}_{i+1}$ only, without using any additional means. In [54], the planar control law (3.7)-(3.8) was developed to control point agent kinematics using the absolute bearing angle ϕ_i with respect to global north. Our algorithm does not rely on the initial North axis, and it uses the local bearing angle γ_i which is a fully onboard solution using camera measurement. Therefore, it provides a significant advantage in instances where magnetometer data is erroneous.

3.2.1 Equation model of the closed-loop system

We now investigate the stability issues of the control algorithm (3.7)-(3.9). The directed constraint graph produces a coupled kinematic structure, necessitating a comprehensive examination of the entire system. Define $\mathbf{E} = [\mathbf{e}^\top, \mathbf{\Gamma}^\top]^\top \in \mathbb{R}^6$ as the combined error vector, where $\mathbf{e} = [e_1, e_2, e_3]^\top$ and $\mathbf{\Gamma} = [\gamma_1, \gamma_2, \gamma_3]^\top$ denote the distance error vector and the bearing error vector, respectively. Evidently, $\mathbf{E} = \mathbf{0}$ is an equilibrium of the coupled system (3.1) under the control law (3.7)-(3.9), where the heading of robot $\mathcal{R}_i$ is directed toward its leader $\mathcal{R}_{i+1}$ and all distance constraints are satisfied, i.e., $d_i = \bar{d}_i$ for all i . Assuming that the desired distance set $\bar{d}$ is constant, the distance dynamics for robot $\mathcal{R}_i$ are derived as follows:

$$\dot{e}_i = -e_i - e_{i+1} \cos(\alpha_{i+1}) \quad (3.11)$$

which gives the following coupled nonlinear system:

$$\dot{\mathbf{e}} = \begin{bmatrix} -1 & -\cos(\alpha_2) & 0 \\ 0 & -1 & -\cos(\alpha_3) \\ -\cos(\alpha_3) & 0 & -1 \end{bmatrix} \mathbf{e} \quad (3.12)$$

Theorem VII.1 in [110] guarantees the existence and boundedness of solutions for initial conditions except some certain configurations. Particularly, if the agents start from collinear configurations, the existence of solutions and stability properties may not be valid. Because of this, we may reasonably assume that the drones can be started in configurations that are suitable for us, and we can also rule out the possibility of collinear initial conditions. This requirement can be satisfied by adjusting the experimental scenarios so that the drones take off and hover at an appropriate altitude by receiving external inputs to design their initial conditions on the plane accordingly.

The following proposition states the main properties of the proposed algorithm [74] and [110]:

Proposition 1 [110] Consider the three-robot system with the motion models (3.1) and control law (3.7)-(3.9) under the assumptions imposed in sections 3.1 and 3.2. If the initial configuration satisfies (i) $\alpha_i < \frac{\pi}{2} - \epsilon$, (ii) $|\gamma_i| < \frac{\beta_{\text{fov}}}{2} - \delta$ for sufficiently large $\epsilon, \delta > 0$, then, under the noise-free measurement case, $\mathbf{E}$ is globally convergent with $\mathbf{e}$ converging exponentially.

Proof: Refer to [110].

The distributed control algorithm (3.7)-(3.9) steer the drones $\mathcal{R}_i$ toward the desired configurations provided that the distance and bearing measurements are obtained continuously with small error. However, in real life conditions, these sensors are prone to several faults, which may lead to deterioration of the convergence guarantees and the

formation configuration. In the following section, a localization framework is proposed as a solution to this problem.

3.3 Localization Framework

Each drone solely $\mathcal{R}_i$ utilizes its onboard means for the control inputs and estimate the position of its coleader drone $\mathcal{R}_{i+1}$. In the following, our filtering algorithm is presented which combines the bearing and distance measurements of the drones.

3.3.1 Measurement Model

The low-level controller of a drone uses the IMU measurements to stabilize its attitude and altitude. The formation control algorithm (3.7)-(3.9), in particular, requires that the pitch and roll angles remain constant at the origin so that the drones can always move at a specific altitude parallel to the inertial frame. Furthermore, a drone can be equipped with numerous other exteroceptive sensors, such as optical flow and laser range finders, to achieve accurate low-level motion control outdoors. Each drone also has a low-cost monocular camera, a precise omnidirectional distance sensor (such as UWB), and a GPU-aided companion computer. These sensors generate the exteroceptive measurements d_i, γ_i for localization, which are fused in a filter to produce the relative position hypothesis toward the coleader drone. The monocular camera sensor is mounted on the front side of each drone and detects its coleader drone using a CNN-based detection technique.

In the preceding section, we showed that the suggested cyclic formation control framework (3.7)-(3.9) produces a stable system that regulates error signals exponentially. This convergence guarantee ensures rapid and reliable performance that is resistant to minor variances in system modeling and measurement uncertainty. On the other hand, the controller is primarily reliant on distance and bearing measurement feedback, both of which are intrinsically noisy. In a variety of demanding settings, the CNN algorithm achieves good detection performance. However, the detection is sensitive to ambient light conditions and may result in false positives or outliers. The UWB sensors produce trustworthy distance measurements at a fast data rate, but numerous factors, such as multi-path effects, can reduce the precision and accuracy of those readings. Systems with rapid dynamics, such as drones, may experience divergence under the presence of measurement

errors. In the following section, we develop a Bayesian filter to solve potential issues caused by measurement uncertainty.

3.3.2 Estimation Model

Keeping with our goal of developing a distributed framework for the system, we propose a filter for each drone in its own coordinate frame. In our framework, drone $\mathcal{R}_i$ is not informed by the location or velocity of its coleader $\mathcal{R}_{i+1}$. Assuming robot $\mathcal{R}_{i+1}$ is stationary, we consider the following motion model between robots $\mathcal{R}_i$ and $\mathcal{R}_{i+1}$ [110]:

$$\dot{\mathbf{x}}_i = \begin{bmatrix} -\sin(\mathbf{x}_{i,2}) & -\cos(\mathbf{x}_{i,2}) & 0 \\ -\frac{1}{\mathbf{x}_{i,1}} \cos(\mathbf{x}_{i,2}) & \frac{1}{\mathbf{x}_{i,1}} \sin(\mathbf{x}_{i,2}) & 1 \end{bmatrix} \begin{bmatrix} \mathbf{u}_{\mathbf{x}}^i \\ \mathbf{u}_{\mathbf{y}}^i \\ \omega_i \end{bmatrix} + \eta_p, \quad (3.13)$$

where $\mathbf{x}_i = [d_i, \gamma_i]^\top \in \mathbb{R}^2$ is the state vector and $\eta_p \in \mathbb{R}^2$ is the process noise vector. Notably, $\mathbf{u}^i$ and ω_i denote the linear and angular velocities in the body frame of drone $\mathcal{R}_i$, respectively. To see the derivation of (3.18), one can form a virtual frame $\hat{\Sigma}^{\mathbf{b}_i}$, obtained by rotating $\Sigma^{\mathbf{b}_i}$ by $\frac{\pi}{2} - \gamma_i$, with positive x -axis in the direction of the line segment l_i . Then, the velocity along the l_i will represent the distance dynamics while the velocity along the line perpendicular to l_i will represent the bearing dynamics. The measurement model can be posed as follows:

$$\mathbf{y}_i = \mathbf{x}_i + \eta_i, \quad (3.14)$$

where $\eta_i = [\eta_i^d, \eta_i^\gamma]^\top$ is defined in (3.6).

Assuming that the measurement bias has been adjusted, the distance measurement noise signal can be well-modeled as a zero-mean Gaussian with a suitable variance. In our evaluations, CNN algorithm gives the bearing angle γ_i measurements, such that the uncertainty can be characterized as Gaussian with a variance value correlated to the detection performance metrics. Therefore, we model $\eta_i \sim \mathcal{N}(\mathbf{0}, \mathbf{S})$ as a zero-mean Gaussian vector with a diagonal covariance matrix $\mathbf{S}$. We apply the standard EKF algorithm to observe the distance and bearing values as described in [77]. As a result, the filter is supposed to produce a smooth approximation of the measured variables while minimizing the effects of outliers.

3.4 Drone Detection

3.4.1 Detection with Deep Learning

In the fields of computer vision and robotics, object detection has been a topic of active research in the last decade. The process of object detection was traditionally derived as a pipeline consisting of invariant feature extractions, pattern matching, and, eventually, the categorization of the item [78]. Therefore, it is possible that neither a pretraining nor a dataset is necessary for the standard techniques. It is possible to utilize a template of the object as a sliding window in the image in order to match the region that corresponds accordingly. Cascade classifiers, SIFT, and SURF are just a few examples of the many different approaches that can be taken in this route. On the other hand, extracting the features of a drone is a difficult task that might not result in highly effective performance. Alternatively, markers that have distinguishing characteristics, such as circular patterns or ultraviolet lights, are attached to drones in order to identify drones that are flying at a high data rate [79], [80].

As part of the framework that has been proposed, the objective of the algorithm for detecting drones is to determine the bearing angle that exists between the robots. Because poor detection performance can easily lead to divergence for nimble vehicles such as drones, the precision of the algorithm plays a vital role in the success of the overall framework. Accordingly, the implementation of the framework must have high success rate. When compared to the traditional methods, the CNN-based object identification methods exhibit superior precision and accuracy in terms of the amount of computational complexity involved. CNN architectures, in their most fundamental form, are designed to solve concurrently the classification and localization of the object in the image under consideration. Some pioneering designs like RCNN [81], Fast-RCNN, and Faster-RCNN, are programmed to extract regions of proposals in the image before propagating each region into the detection's network [82]. These architectures are designed to extract regions of proposals. Consequently, despite the fact that these architectures demonstrate higher performance in comparison to the traditional techniques, they are significantly slower. In recent times, a CNN architecture known as YOLO (you only look once) has demonstrated exceptional performance in terms of both speed and precision [83]. This design resolves the areas of suggestions by utilizing grids and anchors.

3.4.2 YOLO as a CNN architecture

In a YOLO architecture, an image is first divided into $s \times s$ grids where each grid has b number of anchors which aim at detecting the bounding boxes by implementing a

series of convolutional layers and maxpooling layers (Fig. 3.3). As in the other CNN architectures, each convolutional block is defined as a chain of convolution kernels with a batch normalizer and an activation function (Fig. 3.3). We implement the YOLOv3-Tiny version which provides robust performance while demanding less computational power compared to the other CNN architectures [84]. The tiny version is a multi-scale predictor with two outputs at 13×13 and 26×26 and has eight convolution layers with 3×3 kernels and three convolution layers with 1×1 kernels. Six max-pooling layers reduce the network parameters after each convolution operation. Although it has less network layers compared to the YOLOv3, YOLOv3-Tiny remains a suitable candidate for light aerial platforms with limited onboard computing resources.

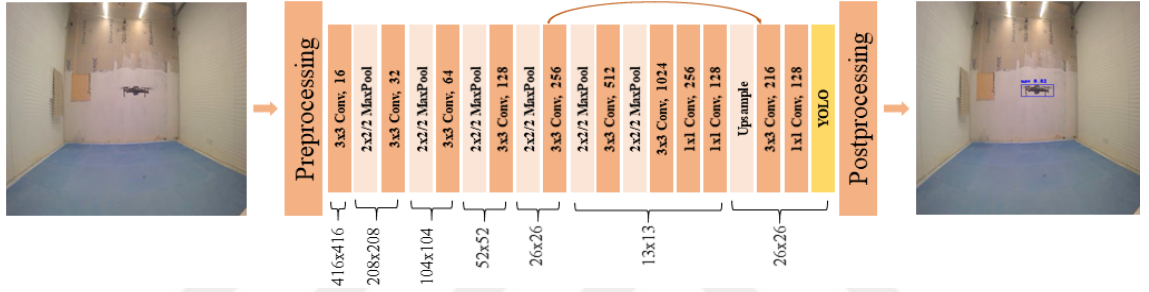


Figure 3.3 The internal layers of the YOLOv3-Tiny with a sample input image and its detection output from an experimental environment [110].

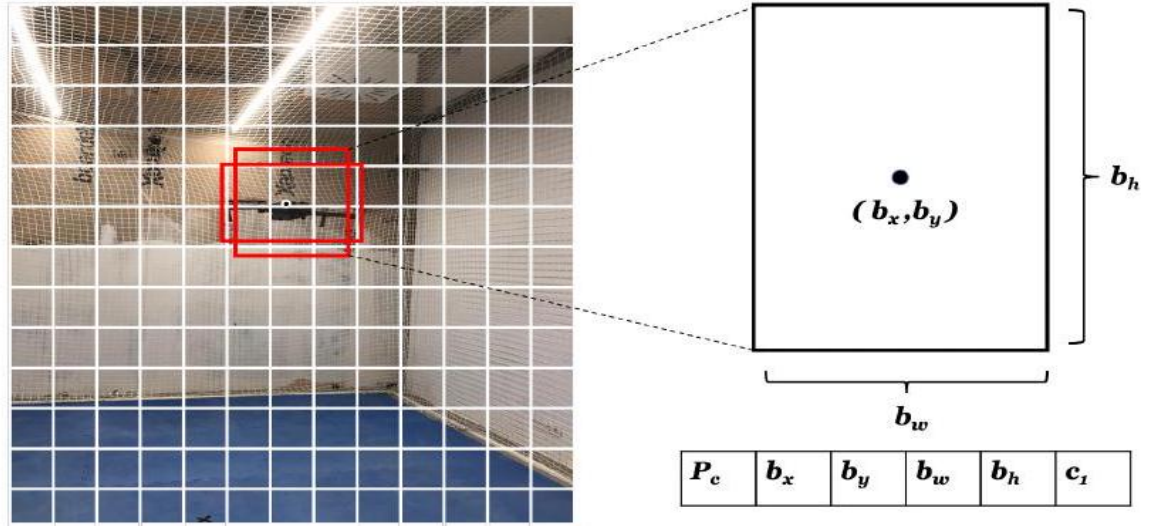


Figure 3.4 (Left) An image segmented into 13×13 grids ($s = 13$); (Right) An example of two anchor boxes ($b = 2$) with the network output vector. Our version of YOLOv3-Tiny uses six anchors for each grid cell and has a single class [110].

3.4.3 YOLO Training

We made the following changes to the base YOLOv3Tiny. The network parameters were changed and adjusted for single-class detection. To reduce processing time, we resized the input image to the algorithm's usual size of 416×416. We created a custom-built dataset for the training procedure. For simulations, we took 1500 pictures of an Iris quadcopter in the Gazebo simulation environment from various angles and backgrounds. For real-world tests, we created a new set of 3360 DJI Mavic drone photos, including images from web searches with video recordings of the drone flying in the lab. Training was conducted on a super micro PC equipped with two multi-core processors and two NVIDIA graphics cards. We divided the training dataset into three categories: training (80%), validation (10%), and verification (10%). We trained the model using 2000 iterations for simulation photos and 5000 iterations for experiment photographs. As a result, we achieved a mean average precision (mAP@50) of 92.93% in the simulation and 99.41% in the trials. Fig. 3.3 shows a sample that was successfully detected. Following the detection procedure, the bearing angle is determined as the difference between the bounding box center and the image center, scaled by a constant value derived from the camera's intrinsic features.

3.4.4 A comparison of detection methods

We compared YOLOv3-Tiny with two other object detection methods in terms of detection speed, size, and precision. We used 348 images from the validation set which were not included in the training set. First, YOLOv3 and YOLOv3Tiny were compared on the super micro-computer used for the training. Both architectures were trained on the same DJI drone dataset mentioned in previous section with the same hyper-parameters. Both models showed excellent precision and recall. However, although the original YOLO yields higher precision and recall for applications involving multiple classes compared to the tiny version, it demands more time for both training and tests (Table 3.1). Notably, there was a factor of 10 difference in the training time. The second comparison was performed between the traditional template matching (TM) of OpenCV v.4.5 and YOLOv3-Tiny on the Jetson Nano. Template matching is a simple OpenCV framework that slides a window over the image to generate a comparison. We used a patch template from a common DJI image for the algorithm. Although the average detection time for TM (69.8 ms) was less than the YOLOv3-Tiny (82 ms), the precision of the TM was lower than the precision of the YOLOv3-Tiny.

Table 3.1 A comparison of the YOLOv3 and YOLOv3-Tiny.

	YOLOv3	YOLOv3-Tiny
Training Time [hr]	27	2.5
Average Detection Time [ms]	100.1	8.8
Size [MB]	246.4	34.8

3.5 Evaluation of the Framework

We tested the suggested framework on a three-drone system in the ROS-based realistic simulation environment Gazebo, as well as a prototype two-drone system in real-world trials. In this section, we show the simulation and testing results [110].

3.5.1 Simulations

We illustrate the performance of the proposed framework using three Iris quadrotor drones in a Gazebo simulation environment. Each drone $\mathcal{R}_i$ is equipped with a Pixhawk autopilot to control the internal dynamics and a monocular camera with a field of view of $\beta_{\text{fov}} = 1.39\text{rad}$. The camera mounted on drone $\mathcal{R}_i$ detects robot $\mathcal{R}_{i+1}$ and obtains the bearing measurement γ_i (Fig. 3.6). To replicate the UWB sensors, we applied Gaussian noise to the measured distances d_i . We analyzed 26 simulation tests. In all tests, all drones were set to hover at the altitude $z_i = 3$ m. Next, the control algorithm was implemented to each drone. In the first set of simulations, we evaluate the drones' behavior with the integrated perception-decision algorithm, in which the localization algorithm output was fed back to the formation control algorithm. We illustrate a sample case in Fig. 3.6 where the drones' planar trajectories are depicted for 280 seconds for $\bar{d}_i = 5$ m and $k = 0.3$. Although the drones' initial planar positions do not meet the intended limitations, they converge to a desired configuration with a modest error in a short time. The relative distance and bearing errors are illustrated in Fig. 3.7. We note the exponential convergence of the distance errors e_i and the bearing errors γ_i in the first 30 seconds. The fluctuations in the steady-state occur due to the following reasons. First, the measurements are noisy, and the YOLO algorithm occasionally misses the identification of coleader drones, resulting in tiny deviations from the equilibrium. Second, unlike ground robots, the drones are agile vehicles and subject to drifts on the planar coordinates $(x - y)$ even when they hover under the velocity-level control. Thus, even relatively

small noise levels in IMU data can produce oscillations and drifts in drone motion if the flight control unit is not tuned to position-level control.

Furthermore, we examined the convergence and robustness of the closed-loop system for several initial conditions of the controlled variables d_i, γ_i . In this set of simulations, we set the initial inter-robot distances $d_i(0)$ for all i to a value in the set $\{4, 4.5, 5.5, 6, 6.5, 7, 8\}$ m with $\bar{d}_i = 5$ m in each run, and measured the average of the absolute distance errors $e_a(t) = \frac{1}{3} \sum_{i=1}^3 |e_i(t)|$. We repeated the same process for the bearing and set the initial bearing values $\gamma_i(0)$ for all i to each of $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7\}$, and measured the average of the absolute bearing errors $\gamma_a(t) = \frac{1}{3} \sum_{i=1}^3 |\gamma_i(t)|$. We observe that the distance and bearing errors converge exponentially in a short time in all cases, and $e_a < 0.8$ m and $\gamma_a < 0.06$ rad in the steady-state.

Finally, we examined the effect of the design parameter k on the distance and bearing errors. We set $k = (0.1, 0.3, 0.5, 0.8, 1, 1.2, 1.5, 1.8)$ in each run. As the performance metric, we used the root mean square errors (RMSE) of each drone:

$$e_{\text{RMSE}}^i = \left(\frac{1}{T} \sum_{t=1}^T |e_i(t)|^2 \right)^{1/2}, \gamma_{\text{RMSE}}^i = \left(\frac{1}{T} \sum_{t=1}^T |\gamma_i(t)|^2 \right)^{1/2}.$$

Table 3.2 shows the average of the RMSE $e_{\text{R,a}} = \frac{1}{3} \sum_{i=1}^3 |e_{\text{RMSE}}^i|$ and $\gamma_{\text{R,a}} = \frac{1}{3} \sum_{i=1}^3 |\gamma_{\text{RMSE}}^i|$, where T is the total number of time steps in each test. As expected, we observed that the lower values of k lead to slow convergence of the bearing error γ_i and increase $\gamma_{\text{R,a}}$ but do not have a direct effect on the distance RMSE $e_{\text{R,a}}$.

Table 3.2 RMSE of the distance and bearing errors for eight values of the design constant k [110].

	Design constant k						
	0.1	0.3	0.5	0.8	1.0	1.2	1.5
$e_{\text{R,a}}[\text{m}]$	0.327	0.336	0.317	0.391	0.363	0.312	0.316
$\gamma_{\text{R,a}}[\text{rad}]$	0.075	0.035	0.025	0.039	0.027	0.014	0.020

Table 3.3 RMSE of the distance and bearing errors for eight values of the design constant k [110].

Test number

	1	2	3	4	5
$e_{R,a}[\text{m}]$	0.279	0.365	0.378	0.294	0.358

3.5.2 Experiments

We ran multiple experiments on a two-drone prototype system to ensure that the suggested technique is feasible and effective on real hardware. While the formation control algorithm was developed in the previous section using a three-drone system, we present a subset of the complete formation by taking advantage of its distributed nature due to the fly zone's limited space. We used an F450 frame-based drone which was to satisfy the distance and bearing angle constraints toward a DJI Mavic drone (fig. 3.5). The F450 drone was equipped with a Pixhawk Cube flight controller, a DecaWave DWM1001 UWB sensor, an NVIDIA Jetson Nano board, and a monocular camera (LIIMX) with $\beta_{\text{fov}} = 2.53\text{rad}$. The DJI drone was equipped with an UWB sensor. The formation control and perception algorithms were run at 20 Hz on the Jetson board using the Robot Operating System (ROS) interface. To ensure a safe flying in a small room, the F450 drone was operated in position control mode by the autopilot, which received velocity control inputs created by the suggested algorithm on the Jetson board. The ground-truth motion data was captured using a VICON motion capture device.

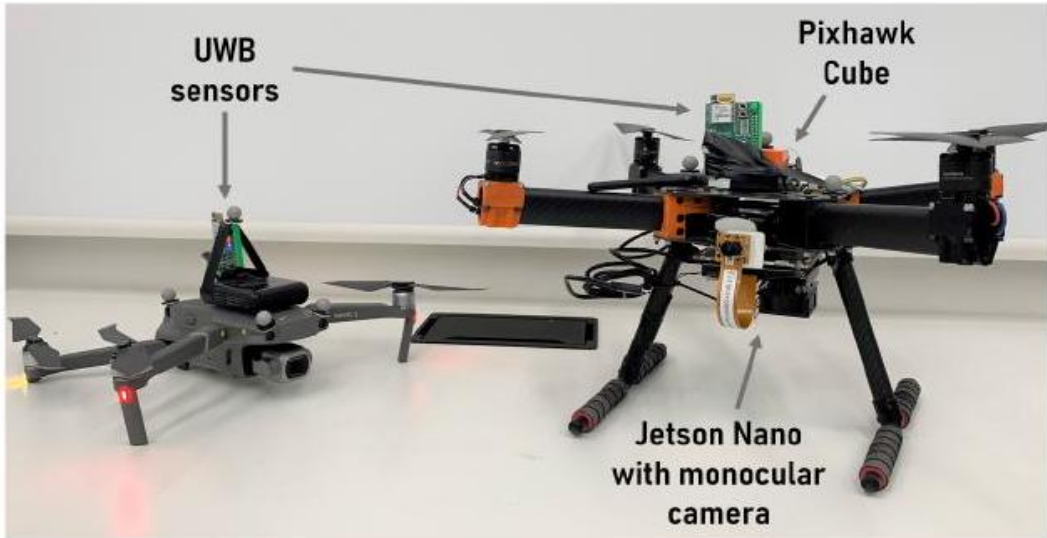


Figure 3.5 The DJI (left) and the F450 (right) used in the experiments [110]

Assuming a line-of-sight (LOS) measurement case, we used the following model for the UWB range measurements [85]:

$$d_k = \bar{d}_k + \eta \quad (3.15)$$

where $\bar{d}_k$ denotes the true distance between the sensors at time step k , and η denotes additive noise which can be well modeled with a Gaussian with bias $\mu_b(\bar{d}_k)$ dependent on the true distance. We conducted a set of calibration experiments where we collected a set of measurement data at certain locations ranging from 1 m to 8 m. We observed that the bias and standard deviation of η can be approximated by $\mu_{\text{bias}} = 0.024\bar{d}_k$ and $\sigma^2 = 0.04$. The UWB sensor produced the distance readings at 50 Hz. Notably, our system satisfies the LOS measurement assumption since each coleader must constantly be in the following drone's camera's line of sight. A more thorough experimental investigation incorporating NLOS scenarios and taking into account alternative noise models is necessary for a highly accurate modeling of a UWB sensor. We found that the achieved UWB measurement accuracy with the suggested calibration approach was adequate compare to external factors like drifts and wind effects that cause the drone motion to be far more uncertain than the UWB sensor precision.

We tested the performance of the perception-execution integration by relaying the perception algorithm outcome to the formation control algorithm. In these tests, we moved the DJI drone in the room manually so that the F450 drone does not collide with the boundaries of the room while maintaining the inter-robot distance. For high estimation performance, the EKF initial condition was set as the first set of measurements $(d(0), \gamma(0))$. A test result is illustrated in Fig. 3.8 where the desired distance was set to $\bar{d} = 1.8$ meters. The EKF produced smooth estimations and reduced the effects of the UWB outliers. The little jumps in the distance figure correlate to the time instants when the DJI was manually moved, and the two spikes denote the outliers where the distance readings were out of range. The zero readings $\gamma(t) = 0$ seen in the bearing graph correspond to the time instants where the CNN algorithm could not detect the DJI. We observed that the F450 maintained the inter-robot distance around its desired value and regulated the bearing angle within an acceptable error margin. The EKF yielded satisfactory estimation performance mainly due to the observability of both states.

We show the distance RMSE $e_{R,a}$ of five experiments in Table 3.3. In these tests, we set $\bar{d} = 1.8$ m and $k \in [15, 30]$ and initiated the measurement covariance matrix of the EKF with $\mathbf{S} = \text{diag}\{\sigma_d^2, \sigma_\gamma^2\}$ where σ_d is found with UWB calibrations and σ_γ , the bearing detection variance, is found empirically to obtain the best observed performance.

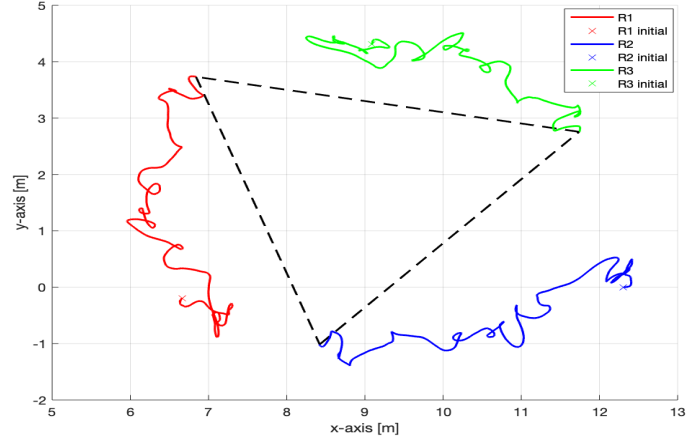


Figure 3.6 The drones' trajectories in a simulation for $\bar{d}_i = 5$ and $k = 0.3$ [110]

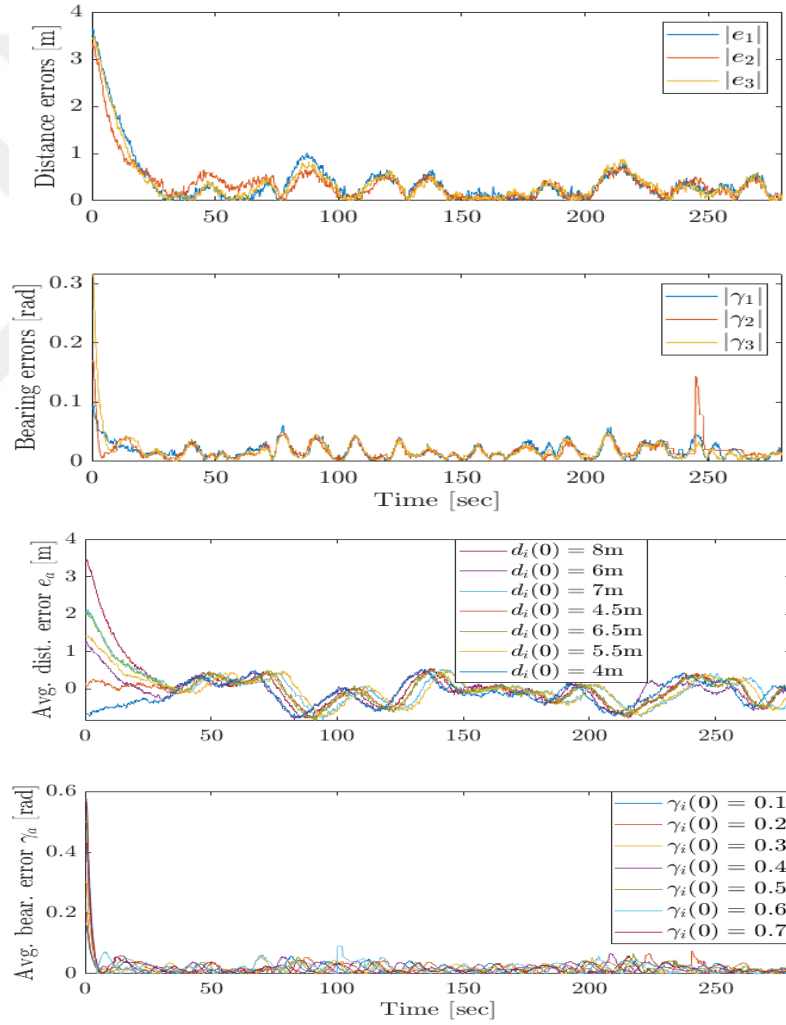


Figure 3.7 Absolute distance errors $|e_i|$ and bearing errors $|\gamma_i|$ for $\bar{d}_i = 5$ and $k = 0.3$; (c) Average distance and bearing errors for different initial conditions.[110]

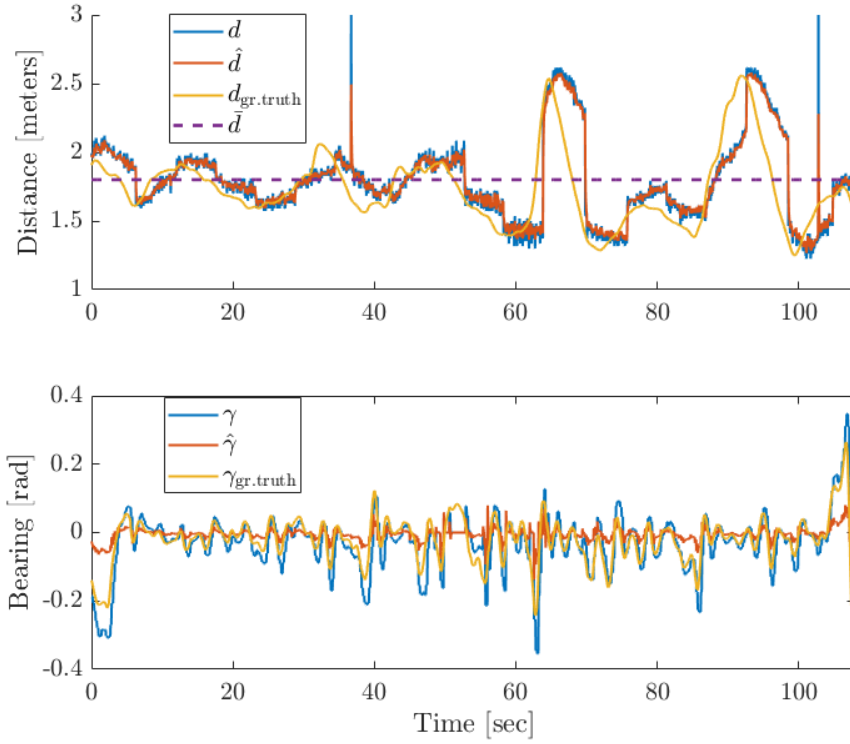


Figure 3.8 The EKF result of the closed-loop perception-decision framework [110]

3.6 Conclusion

We have presented a distributed and decentralized perception-decision framework for a three-drone system in a cyclic coleader configuration. The perception mechanism consists of an EKF algorithm which combines the UWB distance measurements with the bearing detections from monocular cameras. A major advantage of the proposed scheme is that the drones do not need to employ a sensory mechanism which provides absolute measurements such as GPS, mocap, and magnetometer. Indeed, each drone can implement the proposed framework with an inertial measurement unit (IMU), an autopilot, a monocular camera, a GPU-based computing board, and an UWB distance sensor. Such a distributed architecture enables the system to operate both indoors and outdoors and offers a key advantage in critical missions where the drones might be subject to adversarial attacks by outside entities.

Chapter 4

Formation Acquisition in Nonholonomic Multi-Vehicle Systems

In the previous chapter, we have presented a perception and coordination mechanism on a three-drone system. The proposed framework uses vision and distance sensing modalities in the relative positioning algorithm to coordinate the drones based on the inter-agent constraints defined. In this chapter, we apply the relative position estimation structure derived to a formation acquisition problem of multiple non-holonomic ground vehicles. While in the previous chapter the drones are assumed as holonomic agents on the xy -plane and each agent is a leader in a three-coleader coordination, in this chapter we tackle a leader-follower scenario to build a formation by a team of non-holonomic unmanned ground vehicles (UGV).

Formation acquisition is the process by which a group of robots moves from a random or disordered starting configuration to a desired formation shape. Based on the control strategy applied, during formation acquisition each robot moves to a location determined by some constraints with respect to its neighbor robots. We emphasize that acquiring a formation ensures that the position of every robot aligns with the desired configuration, which is the very first step for further formation control tasks. Thus, achievement of formation acquisition will prove useful to validate our framework. We propose two formation acquisition types: Contraction and expansion. While contraction refers to the scenario where agents decrease the inter-agent distances to form the desired shape, in the expansion case, they move away from each other for the same goal.

In this chapter, we revisit the results of [100], providing a comprehensive analysis and explanation. We first formulate the formation acquisition problem by introducing the notations used in the chapter. Then, we introduce the relative positioning mechanism for two different sensing modalities: A combination of vision and UWB distance measurements and a two-dimensional (2D) LIDAR. Similar to Chapter 3, our vision

solution relies on a CNN architecture to generate the bearing angle between two agents. Together with the UWB distance measurements, the inter-agent relative position is estimated. For the other sensing modality, we apply point clustering method on a series of 2D LIDAR data to estimate the relative positions between three agents. Since these estimations take place on the robot hardware without any inter-robot communication or a base station, the entire system retains the distributed property. In addition, a base formation control law is proposed for three types of agents in a hierarchical structure to solve the formation acquisition problem. Finally, we present the simulation and experimental results and discuss about the efficacy of the solution.

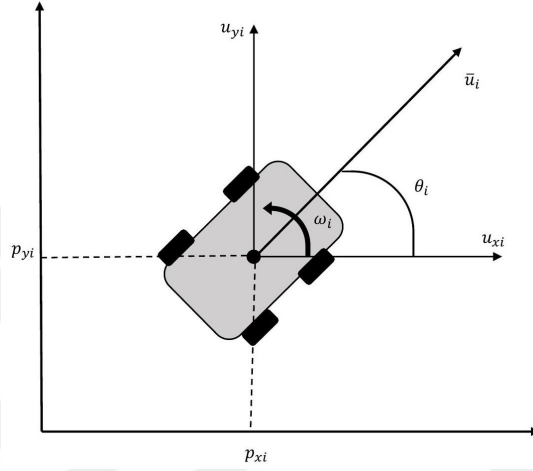


Figure 4.1 The nonholonomic robot kinematic model

4.1 Notations and System Modeling

Consider N robotic agents in a 2D Cartesian workspace environment $\mathbb{R}^2$. The agent motions are modeled as the unicycle model as follows (Fig. 4.1):

$$\begin{aligned}\dot{x}_i(t) &= \bar{u}_i \cos \theta_i(t) \\ \dot{y}_i(t) &= \bar{u}_i \sin \theta_i(t) \\ \dot{\theta}_i(t) &= \omega_i(t)\end{aligned}\tag{4.1}$$

where $p_i(t) = [x_i(t), y_i(t)]^T \in \mathbb{R}^2$ is the position vector, $\theta_i(t)$ is the heading angle (i.e., orientation), $\bar{u}_i$ is the linear speed, and $\omega_i(t)$ is the angular speed of agent $R_i, i \in \{1, \dots, N\}$. Defining $u_i(t) = \dot{p}_i(t) = [u_{xi}(t), u_{yi}(t)]^T \in \mathbb{R}^2$ as the velocity of the agent in the 2D Cartesian workspace, we have that $\|u_i(t)\| = \bar{u}_i$, where $\|\cdot\|$ denotes the Euclidean norm.

Each agent R_i obtains the measurements of its own heading $\theta_i(t)$ at any time t with its onboard sensors. However, the agents do not have the capability to measure or

acquire their positions $p_i, i \in \{1, \dots, N\}$ in a global frame. This assumption reflects the usual real-life robotics application scenarios where the workspace may not contain any global localization aid such as a motion capture system and GPS, and thus, the robots must utilize their onboard sensors for localization and motion control.

The robotic agents aim to build a formation and move in this formation from a start configuration to a goal configuration. Thus, a graph is defined to allocate the motion constraints within the robotic team. Particularly, two graph structures can be employed for this purpose: Directed and undirected. Due to its versatility and robustness to agent failures, we model the present framework as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \dots, N\}$ and $\mathcal{E} = \{\dots, (i, j), \dots\} \subset \mathcal{V} \times \mathcal{V}$ are respectively the set of the vertices and the edges in the graph. A vertex $i \in \mathcal{V}$ represents an agent and an edge (i, j) defines a constraint between agent i and agent j such that agent j is to fulfill a relative position constraint toward agent i . The set of neighbors of agent i is defined by $\mathcal{N}_i = \{j \in \mathcal{V} \mid (i, j) \in \mathcal{E}\}$.

Utilizing the graph and agent model proposed above, we define the formation framework by $\mathcal{F} = (\mathcal{S}, \mathcal{G}, \mathcal{Q})$, where $\mathcal{S} = \{R_0, R_1, \dots, R_N\}, N \geq 2$, denotes the swarm formed by the robotic agents, and $\mathcal{Q} = \{q_{ij} \mid (i, j) \in \mathcal{E}\}$ denotes the desired (constraint consistent) distance configuration. To minimize the required constraint links and the sensing requirements, the constraint graph $\mathcal{G}$ is designed to be minimally persistent. Minimally persistent graphs contain the minimum number of constraint links among the agents and can be generated using the Henneberg construction method; we refer to [60] for a detailed discussion. Accordingly, a leader-first follower-second follower (LFF) formation is considered. In this graph, vertex $\mathcal{V}_0$, represented as R_0 in Fig. 4.2, has no outgoing edge. Thus, it possesses two degrees-of-freedom (2-DOF) and does not have a mobility restriction in $\mathbb{R}^2$. The second vertex $\mathcal{V}_1$ has one outgoing edge connection toward $\mathcal{V}_0$ which corresponds to 1-DOF in $\mathbb{R}^2$, i.e., it is desired to satisfy one constraint toward agent R_0 . Each of the remaining vertices $\mathcal{V}_i, i \in \{2, \dots, N\}$ (the ordinary followers) has two outgoing edges toward its neighbors, which leads to 0-DOF. For instance, the vertex $\mathcal{V}_2$ is desired to satisfy two constraints toward the leader $\mathcal{V}_0$ and the first-follower $\mathcal{V}_1$. One can continue building the constraint sets of the remaining follower agents as $\mathcal{N}_i = \{i - 2, i - 1\}, i \in \{3, \dots, N\}$. Such a graph design yields a minimally persistent graph $\mathcal{G}$ where the entire vertex set can have translation and rotation motion in $\mathbb{R}^2$ as long as all distance

constraints in Q are satisfied. Thus, the entire formation can move as a single entity in a graph shape defined by Q .

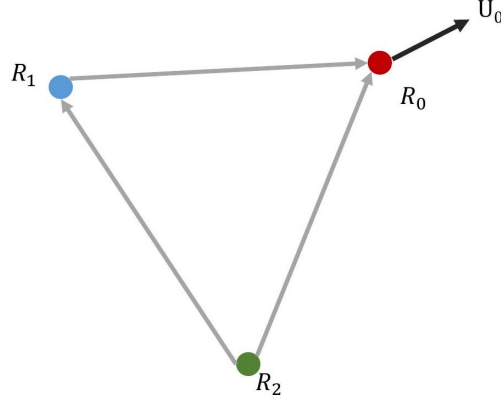


Figure 4.2 The leader-follower network (R_0 : Leader, R_1 : First follower, R_2 : Second follower).

4.2 Problem Formulation

We assume that all robots rely completely on their onboard perceptual and computing resources, with no global localization aids or central ground stations accessible. When there is no central computational unit or global localization system, the robots must rely on the sensors that are mounted onboard. We define our key goal as follows.

Consider the swarm $S = \{R_0, R_1, \dots, R_N\}$, $N \geq 2$, and the leader-first follower formation framework $F = (S, G, Q)$ defined in Section 4.1. We assume that all agents in the swarm rely only on their onboard resources. There is no ground station, and explicit inter-robot communication is not available. Agent R_0 is equipped with a UWB anchor sensor whereas robot R_1 is mounted with a monocular camera and a UWB sensor. Each of the remaining robots $\{R_2, \dots, R_N\}$ is equipped with a 2D LIDAR sensor. Hence, the robot's sensing mechanism is heterogeneous. The purpose is to create a relative localization solution for the given system, as well as a distributed control law for the robots to converge from suitable initial conditions to a desirable formation configuration defined by $F = (S, G, Q)$.

4.3 Sensing Network

We now present the relative localization solution and study the features of the sensors mounted on the robots, namely ultrawideband (UWB) sensors, monocular camera, and 2D LIDAR sensors. We model the hybrid usage of these sensors in our localization system which is solely dependent on onboard sensors and does not exploit any external infrastructure. Distance and vision measurements can be exploited in a complementarily and efficiently manner to acquire the characteristics of the localized target. In particular, UWB sensors and cameras can operate as dual sensors to provide a robust framework for multi-robot localization. Accordingly, we use a pair of UWB sensors, with one anchor on robot R_0 and one tag on robot R_1 , together with a monocular camera on R_1 to provide an efficient localization system for the follower robot R_1 . The distance measurement q_{01} between R_0 and R_1 is obtained from the UWB sensors, and the monocular camera is used to determine the bearing angle ϕ_1 (Fig. 4.3).

2D LIDARs provide a practical detection option for small robots with low weight and energy consumption. Furthermore, because the motion capability of nonholonomic robots is limited by nonlinear constraints, the broad field-of-view of the 2D LIDARs provides a key advantage over this type of robots. Therefore, we choose a 2D LIDAR sensor on the second following robots $R_i, i \in \{2, \dots, N\}$. A second follower robot R_i is to maintain desired distances to its leaders $\{R_{i-2}, R_{i-1}\}$. Thus, it calculates the relative location toward $\{R_{i-2}, R_{i-1}\}$ using a laser scan point cloud and an iterative clustering technique. The goal is to highlight the target agent inside the sensor's range scan and determine their relative positions p_{02}, p_{12} in the body frame of robot R_i . We first convert the raw scan data to point cloud in 2D to differentiate the robots R_0, R_1 from the surroundings. The robot point clouds are then segmented into clusters using an empirically determined threshold, and the remaining laser scans are eliminated (Fig. 4.13b).

We present a CNN model to detect the leader robot, R_0 , with a monocular camera on robot R_1 . We restructure the network to fit our single class output and modify the network hyperparameters (Fig. 4.8). We determine the relative position p_{01} in the body frame of robot R_1 by integrating the UWB distance measurements with the bearing data given by the YOLO v3-Tiny detector. The object detection module generates bounding boxes for observed items, and the picture size can be retrieved from the monocular

camera. Using this idea, we calculate the relative bearing from the target using the central pixel of the bounding box. (Fig. 4.11 and Fig. 4.12).

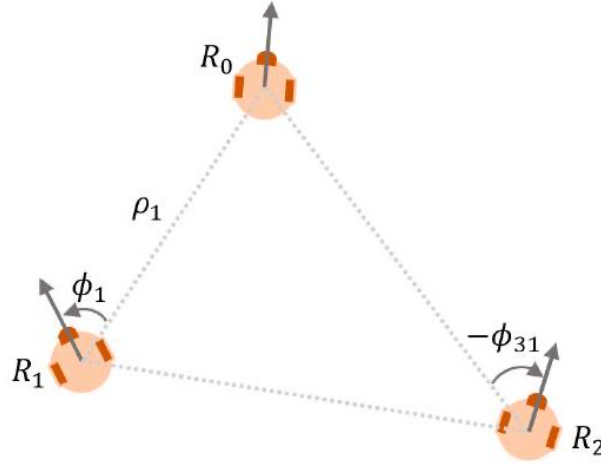


Figure 4.3 The top-view of three non-holonomic robots in formation. [100]

4.4 A Tutorial on Deep Learning for Object Detection: Training and Implementation

With a focus on the You Only Look Once (YOLO) framework, we show in this section how to train a Convolutional Neural Network (CNN) for object detection. YOLO's great speed and accuracy have made it among the most often used deep learning models for real-time object recognition. We aim to provide a thorough tutorial on configuring and training various YOLO versions for an object recognition task using several versions of YOLO to show their performance, and efficiency improvement. Our method guides the user through the processes needed to prepare data, build the model, and train the YOLO network on a specific dataset. We compare the architecture and performance of several YOLO variants—YOLOv3, YOLOv4, and YOLOv8.

4.4.1 Dataset Generation

We have built multiple datasets throughout our bearing detection using object detection. Since the performance of a deep learning model highly depends on the quality of its training data, and no open-source dataset was available for mobile robots, our deep learning process started with a generation of a dataset. First, we always built two image datasets (one for simulation and one for experiments) for training the model. In the

simulation, we collected images from a series of image captures from Gazebo simulations, while for experiments, the robots were randomly moved in an indoor environment, and the camera frames were saved in a pool of images.

First, we built two image datasets (one for simulation and one for experiments) for training the model. In the simulation, we collected images from a series of image captures from Gazebo and Webots simulations, while for experiments, the robots were randomly moved in an indoor environment, and the camera frames were saved in a pool of images. The simulation dataset contained 1451 images with a single class output whereas the real experiment dataset contained 2130 images with two classes (which correspond to two robot shapes), in both closed lab environment and outdoor environment (Fig. 4.4, Fig. 4.5, Fig. 4.6). Statistics of the training datasets can be seen in Fig. 4.7. These images were labeled manually, which consisted of drawing the bounding boxes of the target on each image frame and assigning the detection to the proper class. As usual, the images were split into three datasets before the training: Training (80%), validation (10%), and testing (10%).

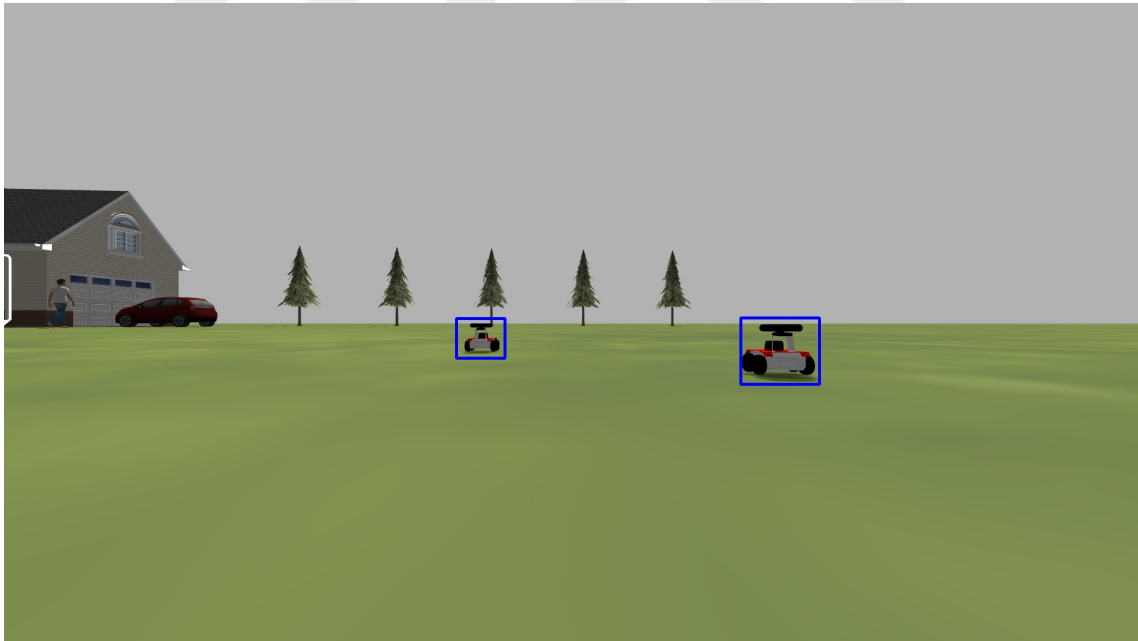


Figure 4.4 Detection of Rosbots in simulation

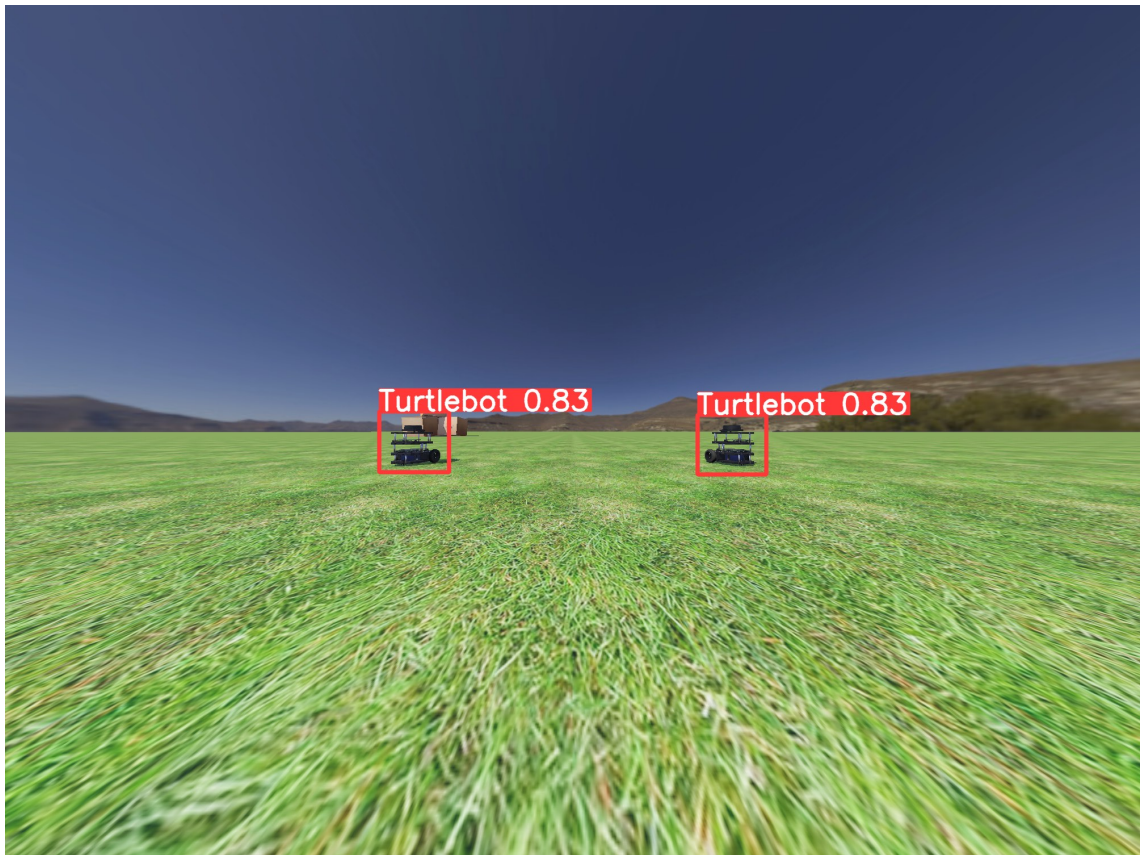


Figure 4.5 Detection of Turtlebots in simulation

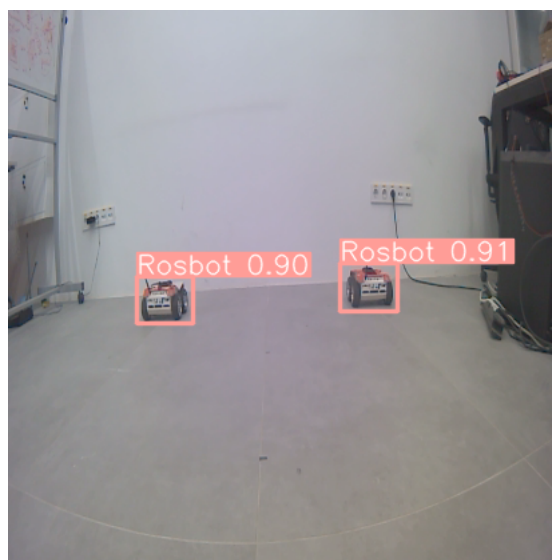


Figure 4.6 Detection of Rosbots in the closed lab environment

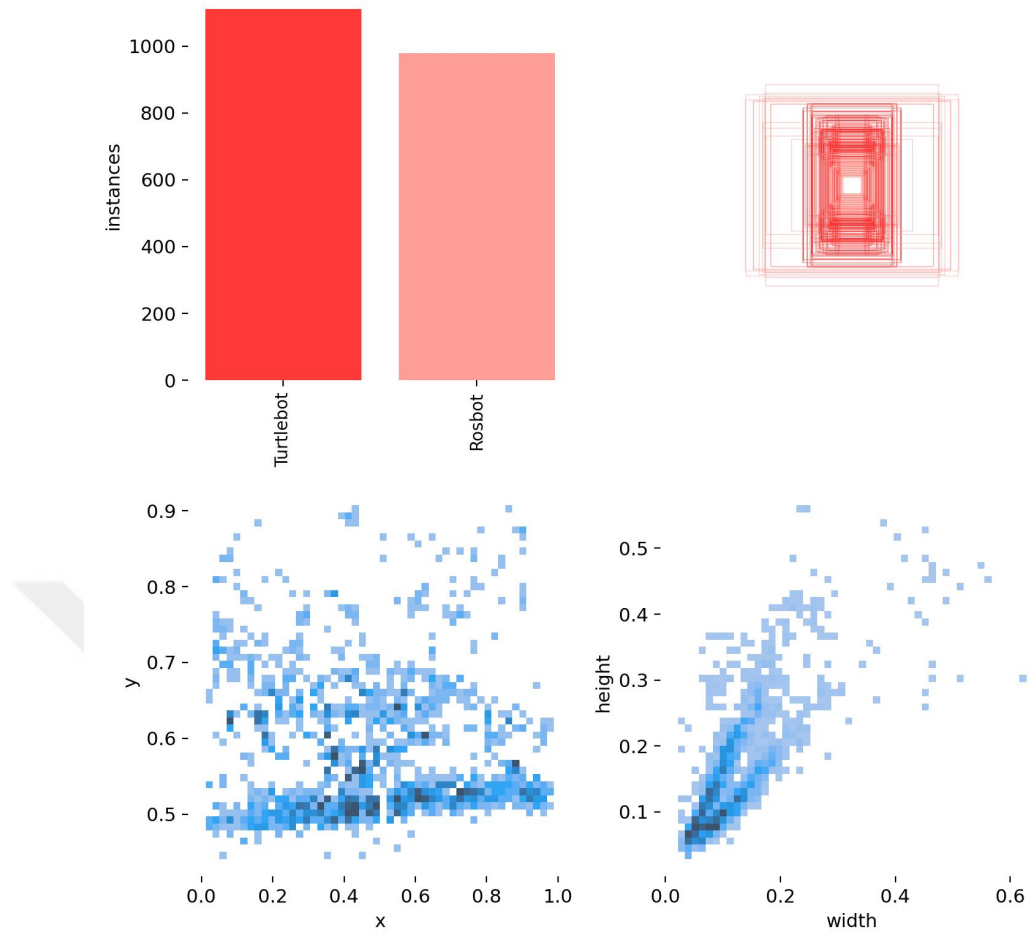


Figure 4.7 The statistics of the dataset used for real experiments: Top) The descriptions of the images and the object classes; Bottom) The correlation graphs of the items in the datasets.

Table 4.1: A Comparison Table of four CNN architectures [111].

Models	Size (MB)	Inference Time (ms)	F1-score	Average IoU (%)
YOLOV3	246.3	2.110	0.98	78.29
YOLOV3-Tiny	34.7	1.718	0.92	77.21
YOLOV4	256.0	2.126	0.70	47.91
YOLOV4-Tiny	23.5	1.768	0.95	79.24

4.4.2 YOLOv3 and YOLOv4

Four CNN architectures were selected for performance comparison, namely, YOLOv3, YOLOv3-Tiny, YOLOv4, YOLOv4-Tiny from Darknet [87]. For each model,

we followed the pattern of the best-used cases and customized the parameters to our problem. The network parameters were redefined to fit the dataset and the input sizes. For instance, for the YOLOv3-Tiny in the simulation dataset, the input size was set to 416×416 , the convolutional filters were defined as 18, and the momentum was selected as 0.9 in the validation process, with a learning rate of 0.01. A comparison of the selected networks on the simulation dataset with a single class is given in Table 4.1. It can be observed that YOLOv4-Tiny has the smallest model size and the highest average intersection of union (IoU). In terms of the F1-score, the original YOLOV3 demonstrated the best score. In terms of the inference time, a critical parameter when performing solely onboard, YOLOv3-Tiny was regarded as the best option. Remarkably, the YOLOv4 demanded exceptional parameter values, such as a batch of 64, which was beyond the performance of the machine used for the training. Instead, the YOLOv4 was trained with a batch size of one and subdivisions of one.

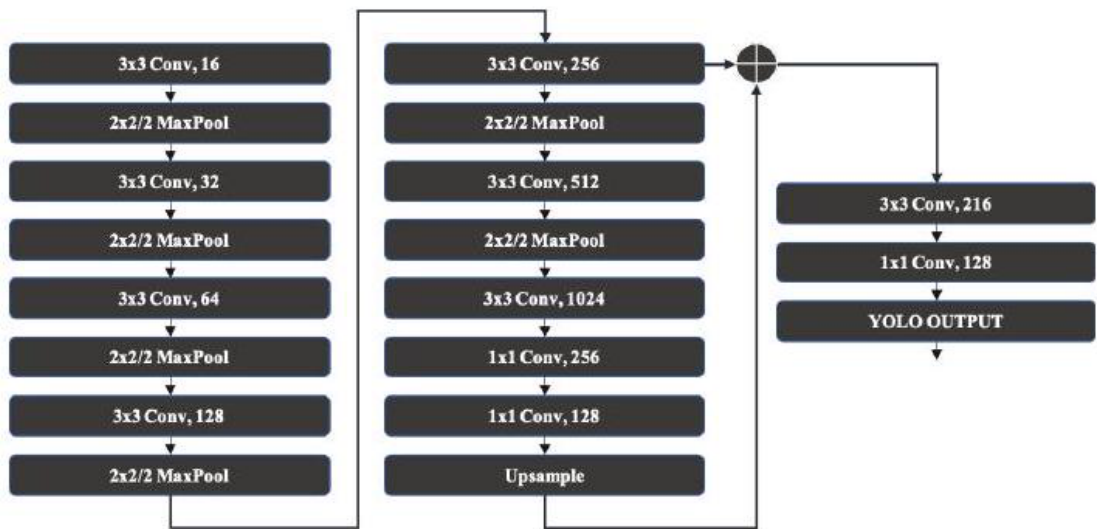


Figure 4.8 The YOLO Tiny architecture backbone used in the training.

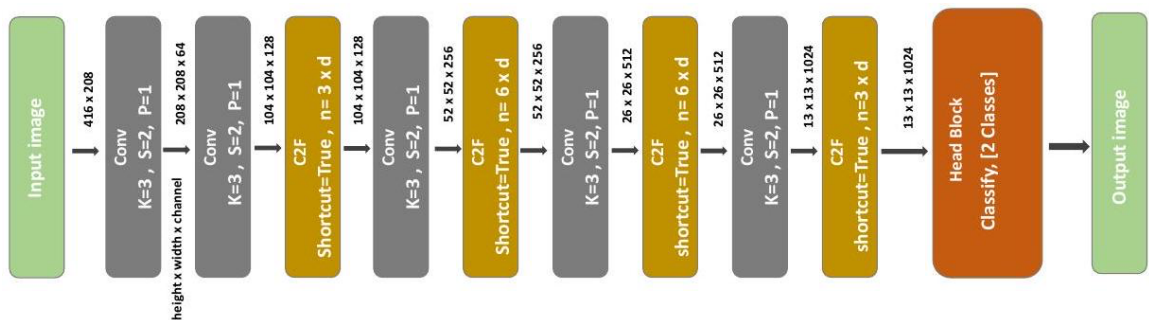


Figure 4.9 YOLOv8 Nano image classification model backbone which forms the base architecture of our custom detection model [111].

4.4.3 YOLOv8

We have also employed the Ultralytics framework [92] for the training, opting for the YOLOv8 nano architecture to enhance detection speed during operation Fig. 4.9. For each of the simulation and experimental environments, an individual CNN training was conducted on a desktop computer with a GeForce GTX 1660 Ti GPU to produce the required model weights. To expedite the training process, we initialized the network weights by using the YOLOv8 nano model pre-trained with the COCO dataset [14] and used the Stochastic Gradient Descent (SGD) algorithm, augmented with momentum for improved convergence. Key training parameters included a momentum setting of 0.937, a batch size of 16, and an initial learning rate of 10^{-2} . We continued the training until there was no further obvious improvement in the loss scores on the test dataset (Fig. 4.10). Subsequently, we converted the model to TensorRT format to ensure compatibility with the Jetson GPU models used on the robots in the experiments. This conversion allowed for faster inference on the Jetson platform, as it is optimized for the NVIDIA architecture.

Table 4.2 Advancements in YOLO CNN: From YOLO to YOLOv8

Version	Year	Key Features	Backbone	Innovations and Improvements
YOLO	2016	Real-time object detection	Darknet	Single evaluation approach, grid-based detection
YOLOv2 [86](YOLO900)	2016	Improved speed and accuracy	Darknet-19	Batch normalization, anchor boxes, multi-scale training, high-resolution classifier
YOLOv3 [87]	2018	Enhanced performance	Darknet-53	Feature Pyramid Network (FPN) for multi-scale detection, binary cross-entropy loss
YOLOv4 [88]	2020	Optimized for single GPU use	CSPDarknet3	Mosaic data augmentation, SAT (Self-Adversarial Training), Bag of Freebies & Specials
YOLOv5 [89]	2020	User-friendly and customizable	Custom (PyTorch-based)	Multiple model sizes (s, m, l, x), ease of training and deployment in PyTorch

YOLOv6 [90]	2022	Industry-focused optimization	Custom	Efficient backbone and neck structure for faster inference
YOLOv7 [91]	2022	GPU-optimized real-time performance	Custom	Reparameterization, task-specific layers, optimized GPU-based architecture
YOLOv8 [92]	2023	Versatile, supports multiple tasks	Custom	Supports object detection, segmentation, pose estimation; improved loss function; enhanced training and documentation

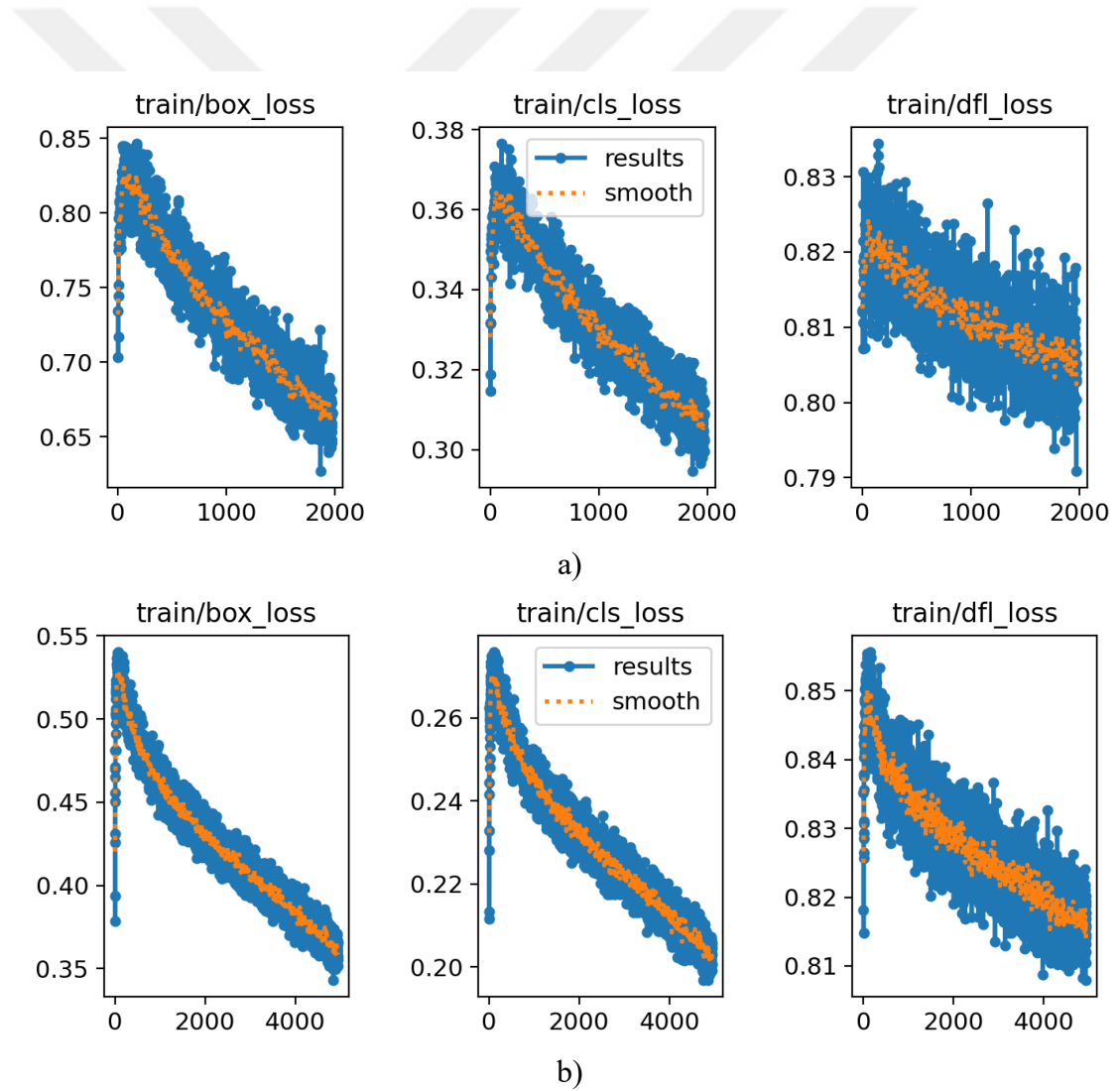


Figure 4.10 The results of the CNN training on the datasets: The loss of the boxes, the object classification, and the Directional Feature Learning (DFL) a) On simulation dataset, b) On experimental Dataset.

4.5 Localization

This section introduces our relative localization algorithm which is based on vision and distance data. The main purpose is to derive the relative bearing angle from the detection coordinates on the image frame. Referring to Fig. 4.12, denote the size and the center of the image plane by $I_x \times I_y$ and $M_i = (I_x/2, I_y/2)$ in pixels, respectively. A CNN detection produces bounding boxes as $B = (x, y, w, h)$, x, y are the coordinates of the left-bottom corner, and w, h are the width and height of the detected box in pixels. Thus, the horizontal center of the box is found as $m_i = x_i + w/2$. Finally, the metric value p_y can be obtained by $p_y = (M_i - m_i)\mu$, where μ represents the camera's pixel size, which may be determined after calibration as follows:

$$\frac{I_{dx}}{I_{px}} = \frac{f_{dx}}{f_{px}}, \quad (4.2)$$

where I_{dx} is the distance in mm , I_{px} is the pixel value, f_{dx} is the focal length of the camera in mm and f_{px} is obtained from the camera calibration matrix. Thus, utilizing the p_y value, we can extract the bearing angle β_i with the distance value, where $\bar{d}_i$ is acquired from the range sensor (either UWB sensors or RGBD cameras). As a result, the combined distance $\bar{d}_i$ and the bearing angle β_i define the relative position between a robot R_i and its neighbor R_j in the local frame of robot R_i .

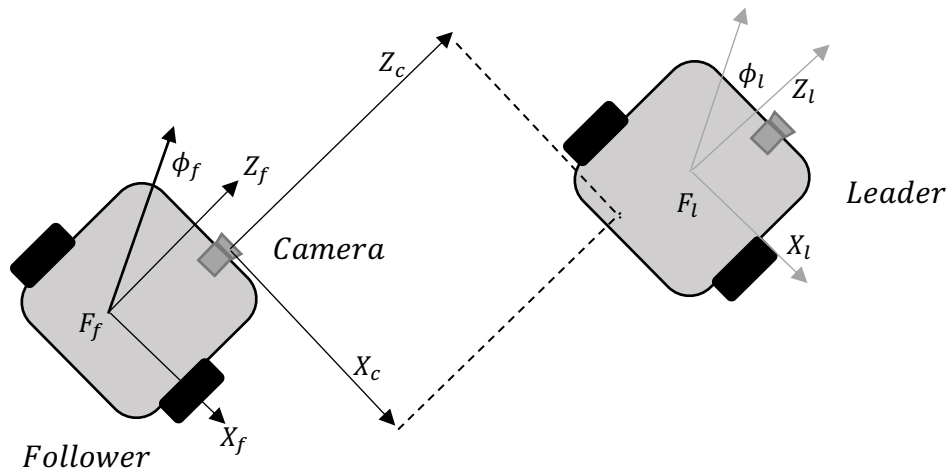


Figure 4.11 The leader-follower network from the camera perspectives;

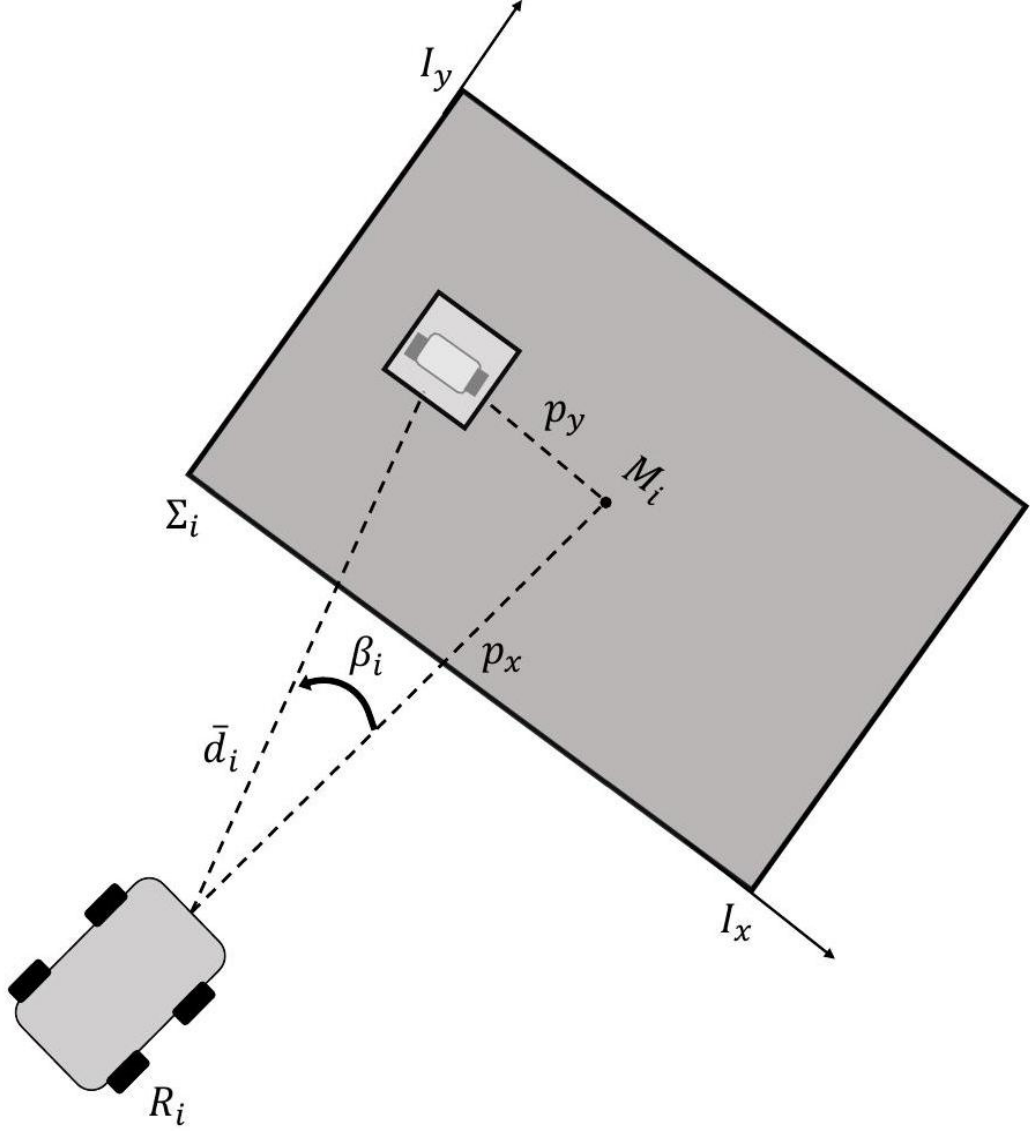


Figure 4.12 The bearing derivation from the camera perspective

4.6 Base Formation Control Laws

The non-holonomic vehicle kinematics (1) poses a challenge for the motion control design as the robot configuration dimension ($\mathbb{R}^3$) is higher than the controllable DOF ($\mathbb{R}^2$). Inspired by the control design of [23], we propose the following individual control laws for a cohesive maneuver of the swarm of nonholonomic agents. First, denote the desired velocity vector and the regulating angular velocity for agent R_i by

$$\begin{aligned} u_i^d(t) &= \begin{bmatrix} \bar{u}_i \cos(\theta_i^d(t)) \\ \bar{u}_i \sin(\theta_i^d(t)) \end{bmatrix} \\ \omega_i(t) &= k_\omega \left(\text{mod} \left((\theta_i(t) - \theta_i^d(t)) + \pi, 2\pi \right) - \pi \right) + \dot{\theta}_i^d \end{aligned} \quad (4.3)$$

where u_i^d is the control input to drive the robot to the desired state, $k_\omega < 0$ is a proportional gain, $\theta_i^d(t) = \text{atan}(u_{i,y}^d(t)/u_{i,x}^d(t))$ is the desired heading of robot R_i , $\text{atan}(\cdot)$ is the four-quadrant arc-tangent function, and

$$\dot{\theta}_i^d = \begin{cases} \frac{\dot{u}_y^d u_x^d - \dot{u}_x^d u_y^d}{(u_x^d)^2 + (u_y^d)^2} & \text{if } u^d \neq 0 \\ 0 & \text{otherwise.} \end{cases} \quad (4.4)$$

This generic control scheme (5)-(6) can drive a point agent kinematics toward its goal location with the linear speed of $\bar{u}_i(t) = \|u_i^d(t)\|$, while regulating its heading angle to its desired value θ_i^d . Before the controller derivations, we define the relative position error $z_{ij} \in \mathbb{R}^2$ and the distance error $e_{ij} \in \mathbb{R}$ as [100]:

$$\begin{aligned} z_{ij} &= p_i - p_j \quad \forall (i,j) \in \mathcal{E} \\ e_{ij} &= \|z_{ij}\|^2 - d_{ij}^2 \quad \forall (i,j) \in \mathcal{E} \end{aligned} \quad (4.5)$$

where d_{ij} is the desired constraint-consistent distance between robot i and robot j . The purpose of the follower robots is to steer the distance errors to zero, i.e., when $e_{ij}(t) = 0$ for all $(i,j) \in \mathcal{E}$, the cohesive flocking objective is achieved up to translation and rotation. In the following, we derive the distributed control laws for each of the three agent types by modifying the control law (5)-(6) for nonholonomic agents.

Leader: In the proposed leader-follower graph, the leader R_0 has two DOF, which means it has no directed distance constraints to other robots in the swarm. Therefore, to invoke a flocking motion, the leader is assigned a purely translational motion with constant speed, i.e., $\bar{u}_0 = \alpha, \omega_i = 0$, where the design constant $\alpha > 0$ is the flocking constant. We assume that α is not known by the follower robots.

First Follower: With one DOF, robot R_1 has one outer degree constraint in the framework. Thus, it needs to maintain the desired distance constraint to the leader robot R_0 , by utilizing the UWB distance measurements and the bearing measurements obtained from the vision sensor. We propose to use the following controller in robot R_1 [100]:

$$\bar{u}_1(t) = \|u_1^d(t)\| = \|k_v e_{01}(t) z_{01}(t)\| \quad (4.6)$$

along with (4.3), where:

$$\theta_1^d(t) = \theta_1(t) - \phi_{01}(t) + \theta_{off}. \quad (4.7)$$

Here, $k_v < 0$ is the proportional gain, $\phi_{01} \in [-\pi, \pi)$ is the bearing angle derived from the object detection, and $\theta_{off} \in [-\pi, \pi)$ is a constant value that represents the offset between the robot body frame and the camera frame. The offset parameter θ_{off} allows the sensor to have a different orientation than the robot body frame.

Second Follower: Having zero DOF, the second followers robot R_i has two motion constraints in the constraint graph $\mathcal{G}$ and aims to maintain the distance constraints $d_{i-2,i}$ and $d_{i-1,i}$ toward robots R_{i-2} and R_{i-1} , respectively. We use the following control law for R_i [100]:

$$\bar{u}_i(t) = \|u_i^d(t)\| = \left\| k_v \sum_{j \in \mathcal{N}_i} e_{ij} z_{ij} \right\| \quad (4.8)$$

along with (4.3), where

$$\theta_i^d(t) = \sum_{j \in \mathcal{N}_i} (\theta_i(t) - \phi_{ij}(t)) + \theta_{off} \quad (4.9)$$

$k_v < 0$ is the proportional gain, $\mathcal{N}_i$ is the neighbor set of R_i in the graph $\mathcal{G}$, and $\phi_{ij} \in [-\pi, \pi)$ is the bearing angle between R_i and R_j measured in the body frame of the second follower R_i .

4.7 Control Law without Magnetic Sensing

In the preceding part, we developed a distributed control system for the three types of robots in the MRS. The control laws for robots R_1 and R_2 heavily rely on the odometer data which is produced by fusing the IMU and magnetometer measurements. However, in some workstation conditions, magnetometer sensors can be influenced by severe electromagnetic changes, making readings inaccurate and low in quality. In such cases, the entire system may exhibit poor localization and control performance. To avoid this difficulty, one needs to derive the control laws directly from the exteroceptive sensors. Assuming the leader robot R_0 is stationary, consider the polar dynamics between robots R_0 and R_1 as follows:

$$\begin{aligned} \dot{\rho}_1 &= -\bar{v}_1 \cos(\phi_1) \\ \dot{\phi}_1 &= \frac{\bar{v}_1}{\rho_1} \sin(\phi_1) + \omega_1 \end{aligned} \quad (4.10)$$

where $\rho_1 = \|q_{01}\|$ and $\phi_1 = \text{atan}(x_0 - x_1, y_0 - y_1)$ is the relative bearing angle (Fig. 4.1). Thus, the dynamics (4.10) allows to control the distance and bearing angle directly by adjusting the control inputs $\bar{v}_1, \omega_1$ at the expense of reducing the controllable degree-of-freedom to two. Thus, the formation objective can be achieved, but the robot cannot be steered to any location on the plane. However, because formation acquisition can be accomplished by simply changing the distance and bearing to the desired values, the dynamics model remains appropriate for our primary goal. We want to create a control

law that will ensure that the distance and bearing toward the leader converge to the correct values.

We propose the following control law for the first follower [100]:

$$\begin{aligned}\bar{v}_1 &= \text{sat}(K_v |e_\rho|, 0, v_{\text{up}}) \\ \omega_1 &= \text{sat}(K_\omega e_\phi, -\omega_{\text{up}}, \omega_{\text{up}})\end{aligned}\quad (4.11)$$

where $K_v, K_\omega, v_{\text{up}}, \omega_{\text{up}} > 0$ are design constants, $\text{sat}(a, \bar{a}, \underline{a})$ is a saturation function such that if $a \geq \bar{a}$, then $a = \bar{a}$, and if $a \leq -\bar{a}$, then $a = -\bar{a}$, and

$$e_\rho = \rho_1 - \rho_1^{\text{des}}, \quad e_\phi = \phi_1 - \phi_1^{\text{des}} \quad (4.12)$$

with $\rho_{\text{des}} > 0$ and ϕ_{des} being the desired distance and bearing values, respectively. If $\phi_1(t) \in [-\pi/2, \pi/2)$, the controller (4.11) steers R_1 toward the desired location imposed by the constraints $\rho_1^{\text{des}}, \phi_1^{\text{des}}$. We suppose that initially, the robot R_0 is in the FOV of the of robot R_1 . If such a setup is not possible, one rotates R_1 with a constant angular velocity in one direction until robot R_0 enters the FOV of the camera of robot R_1 then apply (4.11) for formation acquisition. Thus, we expect the robot to converge to its desired location under the control law (4.11).

4.8 Evaluation

In this section, we demonstrate the results of our simulations and real world experiments for distance and 2D LIDAR.

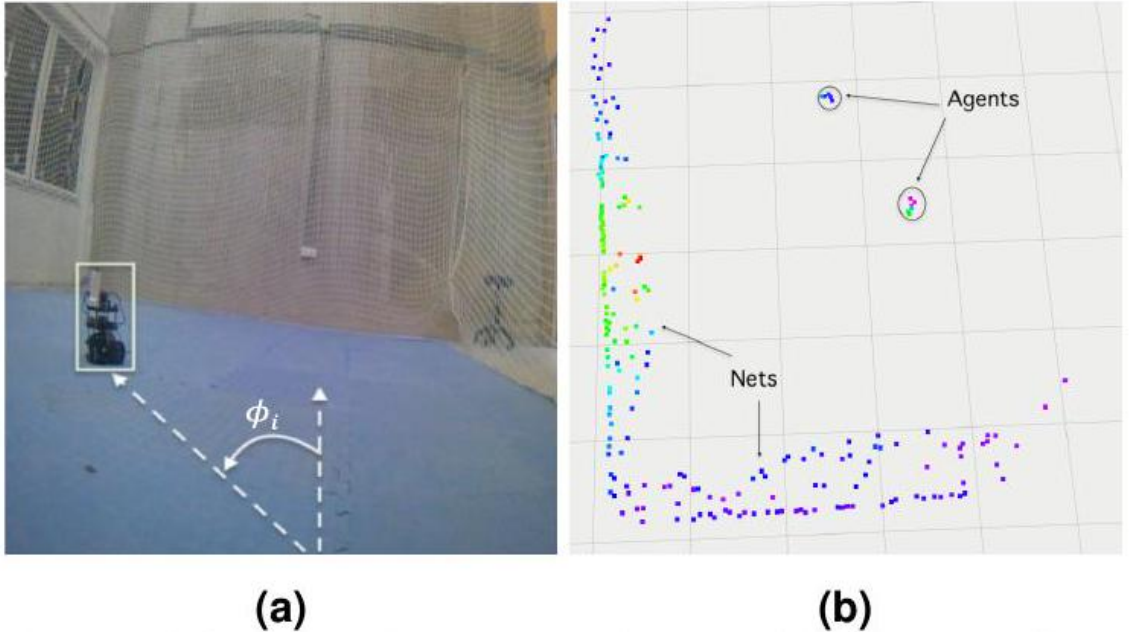


Figure 4.1 Sensors view in experiment: a) A detection from Jetson nano: A view from R_1 , b) A scan from the LDS LIDAR seen by robot R_2 . [100]

4.8.1 Simulation

We simulated our proposed system in Gazebo, a realistic and powerful environment that permits the visualization of robotic systems with identical physical attributes. We used three Turtlebot 3 burger robots and onboard sensors with the following characteristics: A monocular RGB camera with FOV of 2.45 radians on R_1 and a 2D LIDAR with 360 FOV and a range of 6.28 meters on R_2 . We processed the LIDAR data using the point cloud package from robot operating system (ROS) and used OpenCV [96] for post-processing the YOLO model inference.

In the simulations, we assessed the setup for two formation objectives: Contraction and expansion. These two crucial behaviors demonstrated the agents' adaptability in adapting to different settings regardless of the beginning conditions. The first assessment asks the robots to construct a formation with smaller relative distances than the original values, whereas the second evaluation begins with a configuration in which the distances are smaller than the required values. The trajectories and relative distance errors are shown in Fig. 4.14 and Fig. 4.15. We observed an exponential convergence to the target configuration and rapid decrease of the distance errors. In all simulations, we achieved a CNN detection rate of more than 90%, confirming the efficacy of the suggested localization technique.

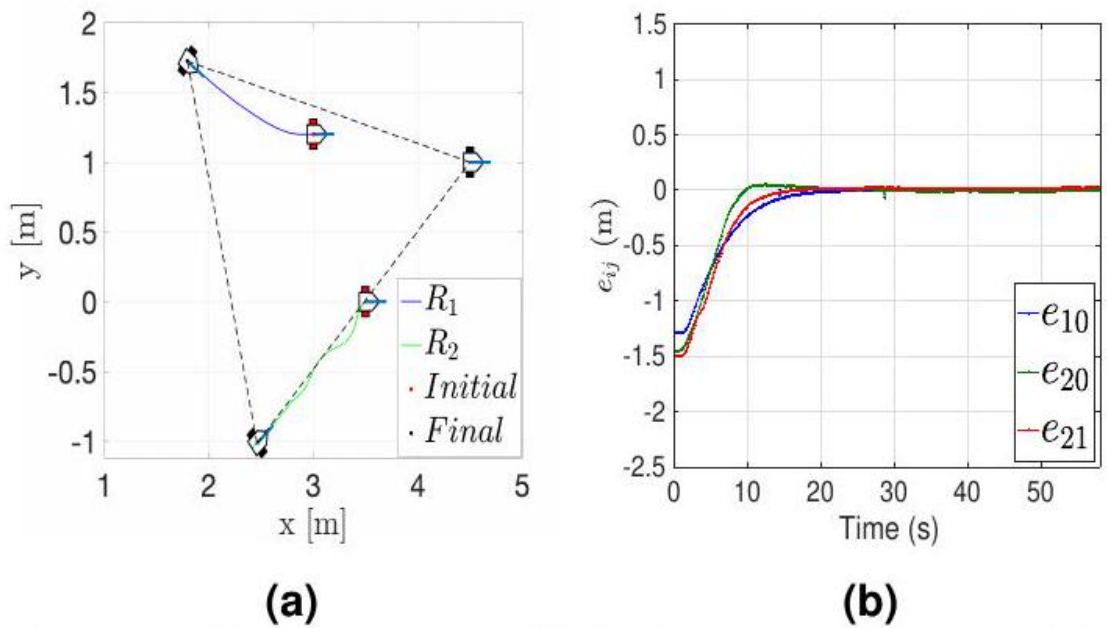


Figure 4.2 Formation acquisition by expansion simulation: a) Trajectories of the agents, b) the inter-robot distance errors. [100]

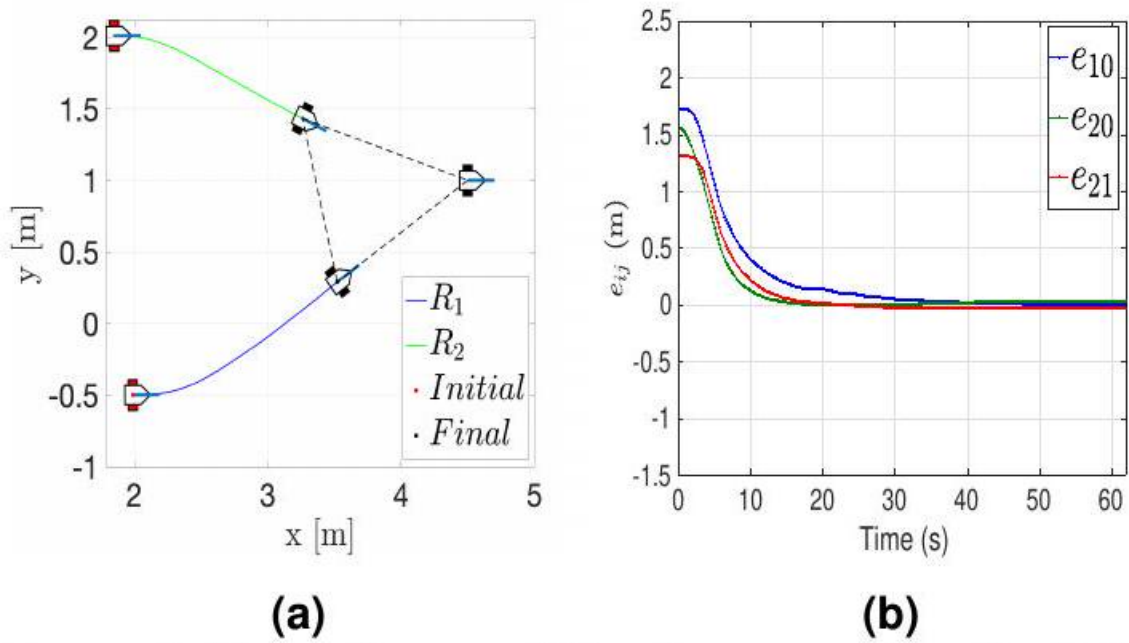


Figure 4.3 Formation acquisition by contraction simulation: a) Trajectories of the agents, b) the inter-robot distance errors.[100]

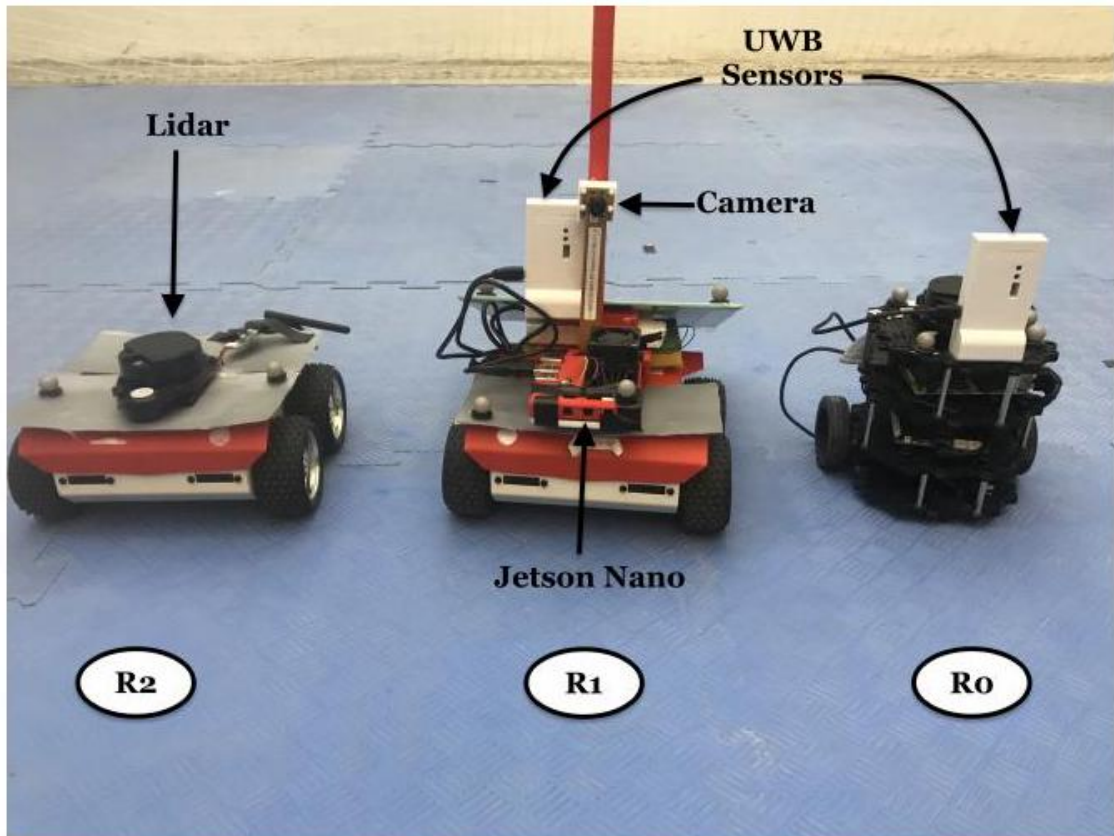


Figure 4.4 The robots and hardware used in the experiment.[100]

4.8.2 Experiments

We ran a number of tests in an indoor lab setting where GPS was not available. The three-robot system comprised a Turtlebot3 as the leader (R_0) and two Rosbots as the first-follower (R_1) and the second follower (R_2), respectively (Fig. 4.16). We provided the robots with the sensor network described above. We installed a pair of DWM1001 UWB modules on the leader and first follower to calculate the relative distance between them. In addition, we installed a CSI camera on the first follower via an Intel Jetson Nano computer to detect bearings. We calibrated the camera to remove distortions and lowered the image size from (3280×2464) to (416×416) for input to the CNN model. Finally, we equipped the second Rosbot with a 2D LDS LIDAR with a 360-degree field of view and a maximum range of 3.5 meters. For illustration purposes, we recorded the robots' ground truth data using a VICON motion capture device. None of our algorithms used data from the motion capture system in experiments.

We evaluated the proposed framework on the three-robot system defined above relying on their onboard resources. The results of a contraction experiment are illustrated in Fig. 4.17a. In this experiment, the robots were initiated from $p_0(0) = [0.02, 0.62, 0.0]^T$ meters, $p_1(0) = [1.14, -1.33, 0.0]^T$ meters, and $p_2(0) = [-1.13, -1.41, 0.0]^T$ meters with different heading angles which are also denoted in Fig. 4.17a. Initially, the leader robot R_0 was not in the FOV of robot R_1 . Thus, R_1 first rotated with zero linear speed and waited for the leader to enter its camera FOV. Once R_1 is detected with the CNN at the 36th second, it started approaching R_0 based on the control rule (4.3). On the other hand, robot R_2 segmented the LIDAR scans to generate clusters for robots R_0 and R_1 . After building the clusters, robot R_2 generated the distance estimates q_{02}, q_{12} . We assumed that initially the robots R_0 and R_1 were present within the laser range. We depict a sample point cloud view of the scans in Fig. 4.13 b. There were nets on the left and rear sides of R_2 which can be clearly seen in Fig. 4.13 b, which we removed with a simple preprocessing in the algorithm. We observed that the robots converged to the desired configuration which are $d_{ij} = 0.5$ meters for all i, j (Fig. 4.17 b), and that the distance errors remained bounded for sufficiently long time (around 400 seconds). We conjecture that the relatively small error in the distances were due to the sensor noises. For instance, the UWB distance readings from this experiment with its ground truth are given in Fig.

4.19, which shows the noise with a variance of around 0.05 meters. Nevertheless, the framework exhibited high performance in terms of formation acquisition. A similar performance can be seen in the formation expansion experiment in Fig. 4.18.

Furthermore, we evaluated the case where the magnetometer is not available for the first follower (Fig. 4.19a). In this experiment, robot R_1 utilized the controller and was able to converge on the desired setting. This result demonstrates the adaptability of the proposed architecture for magnetometer-free environments. Although not displayed here, another controller that does not rely on a magnetometer sensor can also be created for the second follower robot, which is one of our future goals.

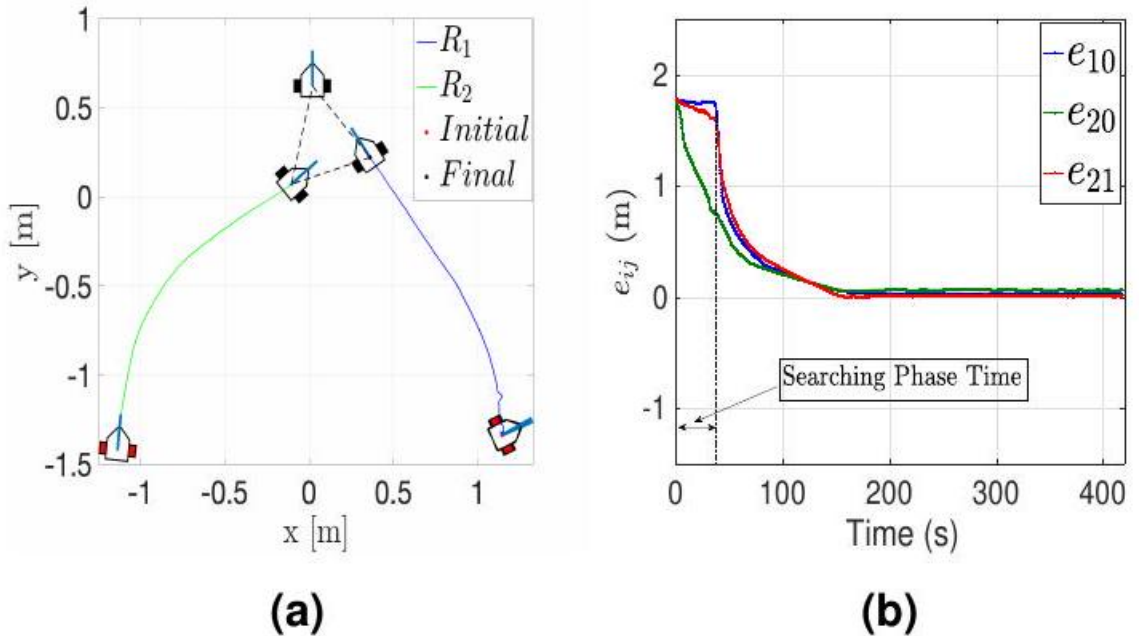


Figure 4.5 Contraction experiment: a) Trajectories of the agents, b) the inter-robot [100]

We will go over some practical elements that were necessary when shifting from simulations to experiments. In our framework, the second follower robot R_2 relies on LIDAR solely. However, when LIDAR identifies another object in the workplace, the two clusters for locating the robots R_0 and R_1 may not be created precisely, resulting in decreased performance. As a result, we highlight the assumption that the workplace must be free of all other objects. If such a criterion cannot be met, an advanced post-processing of LIDAR measurements is required. Furthermore, although we have provided a general framework for $N + 1$ -robot swarms, we have focused on a three-robot setup in evaluations. When there are more than one second-follower robots, i.e., when robots

$R_i, i \in \{3, \dots, N\}$ are added to the system, a more advanced LIDAR post-processing would be necessary to discriminate the two leader robots for each second-follower.

Furthermore, while UWB sensors offer accurate distance data in most settings, they can produce low signal-to-noise ratio if not correctly calibrated. We recommend calibrating the UWB sensors to determine the optimum bias and noise variance values. In our trials, we employed a basic calibration method that required us to deploy the UWB pair at specific distances and gather data for a set amount of time. However, UWB distance estimations might vary greatly depending on the angular location of the sensors and other environmental conditions. We emphasize the necessity of UWB calibration for more accurate results.

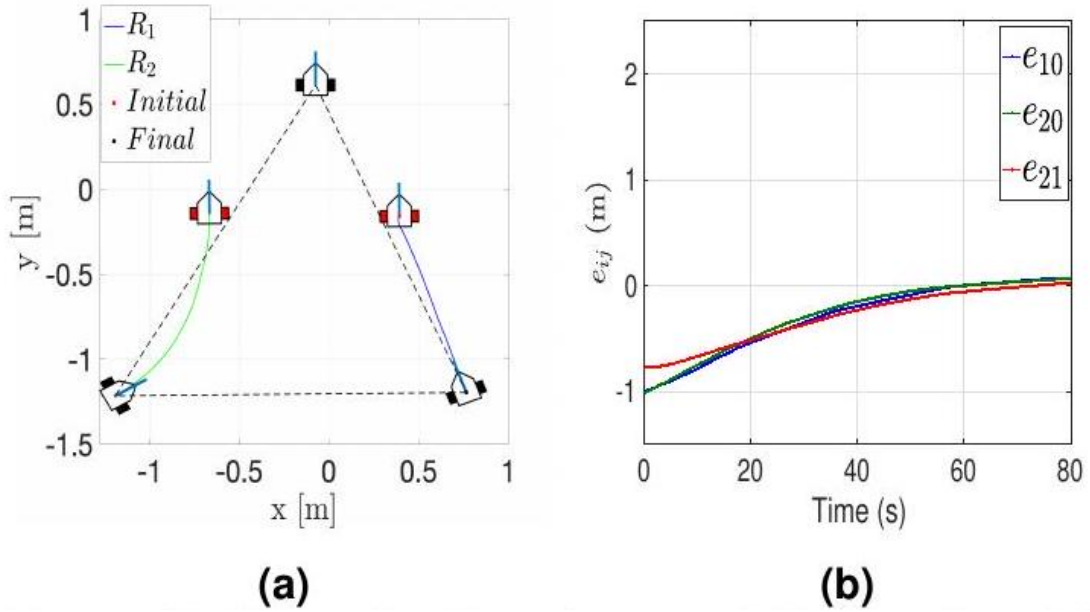


Figure 4.6 Expansion Experiment. a) Trajectories of the agents. b) The relative distance errors.[100]

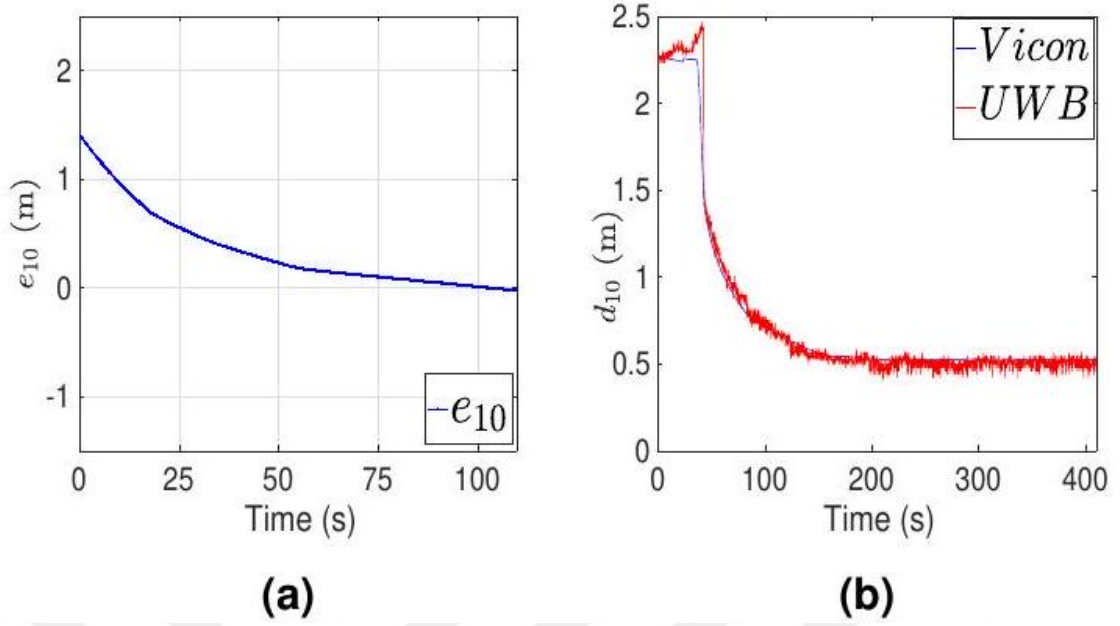


Figure 4.7 a) The error on R_1 during a formation contraction without magnetometers, b) A comparison of UWB sensor data with ground truth from Vicon capture system.[100]

4.9 Conclusions

We have analyzed the practical formation control problem for multi-robot systems employing onboard sensors. We have proposed an extensive framework that includes a localization module and a control module. The localization module includes a bearing angle computation employing a YOLOv3-Tiny architecture with a monocular camera and a methodical clustering technique for a 2D LIDAR. The control module consists of a set of distributed formation control algorithms for three groups of robots inside the leader-follower hierarchy. Furthermore, we have introduced a control system that operates without a magnetometer sensor which is known for generating inaccurate sensor readings under specific conditions. We have evaluated the framework for nonholonomic ground robots for two types of formation acquisition goals (contraction and expansion) in both simulations and experiments. The exceptional results seen in a series of experiments indicate the viability of the proposed method. Future research may encompass the enhancement of the framework for dynamic path planning and obstacle evasion for the entire swarm, leveraging onboard resources. Moreover, the post-processing of LIDAR data to distinguish the leader robots inside the field of vision necessitates examination to improve swarm scalability.

Chapter 5

Designing a Fully Scalable Framework for Decentralized Systems

In Chapters 3 and 4, we have addressed the relative localization problem in MRS to build a multi-modal perception capability among the robots by focusing on a three-robot system. Chapter 3 demonstrates a unique perception-decision framework for a three-drone system whereas Chapter 4 studies the formation acquisition problem for three unmanned ground vehicles (UGV). Both examine the benefits of utilizing a combination of omnidirectional distance sensors and monocular cameras on the robots for the ultimate goal of formation control designs. Through a comparative analysis, we demonstrate that the two sensing modalities can be used as complementary to each other for a unified perception model. While a vision sensor can provide extensive information about the neighbor robots in a team, ultrawideband (UWB) sensors can produce accurate inter-robot distance measurements which are crucial for formation control purposes. By integrating these two sensing modalities, we have demonstrated that a MRS can function effectively without any global localization infrastructure.

When the number of robots in the MRS is increased, critical issues need to be addressed for the feasibility of the designed perception mechanisms. The most important factor is the efficient tracking of the leader agents (UGV or drone) by the follower agents. For the UWB sensors, this issue is handled by their natural data association property as data is communicated over UWB sensors with their associated sensor identity. Thus, all inter-robot distance data are identified whenever they are acquired. However, the robot detections by the CNN algorithms must be discriminated on the image frames acquired by the robots' cameras. For this purpose, we propose a Gaussian mixture probability hypothesis density (GM-PHD) filter in this chapter. GM-PHD combines the two sensing models to generate a robust relative positioning between the robots and successfully tracks the detected robots on follower robots' images.

Another concern with the increased number of agents is the feasible formation configurations among the robots that are consistent with the perception models. The key factor that determines the feasibility of a formation shape is the field-of-view (FOV) of the monocular cameras onboard. Therefore, we provide suitable formation configurations that suit the needs for the proposed localization module. Several simulations and real-world experiments verify that the proposed mechanism based on the two distinct sensing modalities performs well in real environmental conditions.

Most of the results of this chapter were presented in [111].

5.1 The Proposed Approach: An Overview

A block diagram illustrating the proposed framework for a four-robot system $\{R_0, R_1, R_2, R_3\}$ is given in Fig. 5.1. It shows the main components of each robot and the interrelationships among them. We opt for a distributed and decentralized approach, where each robot is responsible for estimating and controlling the relative positions to their neighbor robots in a formation graph structure. As will be explained in Section 6, since robot R_0 takes the leader role and guides the rest of the team, it does not possess a visual sensor for the formation control purposes.

Each robot consists of a localization layer and a control layer. To address our goal, we propose a specialized sensing suite for the robots by using two sensing modalities. Particularly, the localization layer estimates the relative positions between a robot and its neighbors by utilizing an UWB module and a camera sensor. The acquired sensor data are processed with customized CNN (particularly, YOLO) and tracking (GM-PHD) algorithms to generate an estimation of the relative positions between robots (Fig. 5.2). These estimated values are fed to the high-level control block that computes the control signals (linear and angular speeds) required for maintaining the formation and achieving the objective. This two-step approach aims to ensure a cohesive and coordinated motion of the robots within the swarm, adhering to the specified requirements. The low-level controller, which is assumed to be embedded in the robots, controls the low-level force/torque dynamics to match the speed outputs with their desired values.

In the design process of the high-level controller, we propose a method to build the formation configuration that respects the constraints imposed by the designed perception mechanism. Particularly, the distances between two neighbor robots should

obey the camera field-of-view (FOV) constraints and the reliable object detection range limitations to ensure continuous operation.

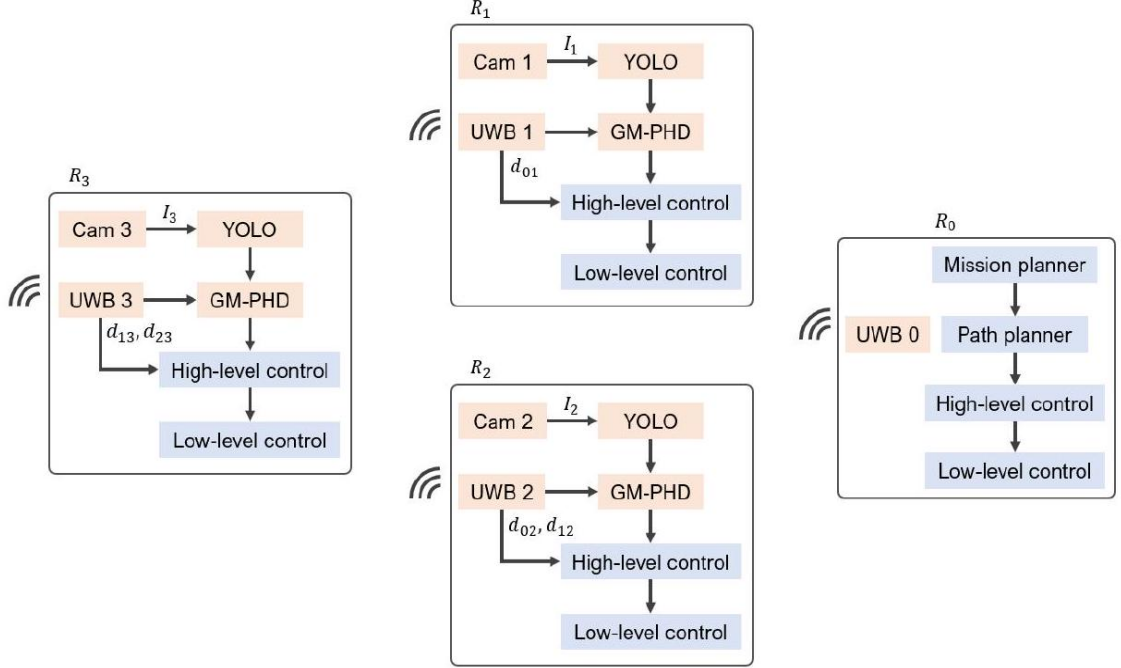


Figure 5.1 A block diagram illustrating the mechanisms integrated within the proposed framework for a four-robot system. The components related to the localization and control layers are given with the orange and blue backgrounds, respectively [111].

5.2 Target Tracking

In the previous section, the relative position estimation mechanism between two robots is given, which utilizes the camera and UWB sensors to generate the estimations of the distance and bearing angles using a single image frame and a single measurement. To implement a distributed MRS, accurately tracking of the detected robots in a stream of the image frames is equally important. In this section, we present our target tracking approach which aims to track the detected targets continuously. Since we are concerned with a scalable MRS of more than two robots, the multiple target tracking problem is investigated.

Gaussian Mixture Probability Hypothesis Density (GM-PHD) filter allows for the estimation of multiple targets in a dynamic environment. It is based on the idea of representing the probability distribution of multiple targets using a mixture of Gaussian components. By utilizing the GM-PHD algorithm, accurate and efficient tracking of multiple targets in a dynamic environment becomes possible. By combining the principles

of probability hypothesis density and Gaussian mixture modeling, the GM-PHD can effectively estimate the number of targets as well as their individual states without the need for explicit data association [97] [98]. In our system, GM-PHD is applied to track the relative position between the robots by using camera frames and distance measurements.

The GM-PHD filter processes a set of relative localization measurements, denoted as $\mathcal{Z}_k = \{\mathbf{z}_{k,1}, \dots, \mathbf{z}_{k,M_k}\}$, to compute an output set of agent states $X_k = \{\mathbf{x}_{k,1}, \dots, \mathbf{x}_{k,J_k}\}$ at each discrete time step k . These states are represented as a single intensity, $v_k(\mathbf{x}_k)$, expressed as a weighted sum of Gaussian components:

$$v_k(\mathbf{x}_k) = \sum_{i=1}^{J_k} w_k^{(i)} \mathcal{N}(\mathbf{x}_k; \mathbf{m}_k^{(i)}, \mathbf{P}_k^{(i)}) \quad (5.1)$$

where $w_k^{(i)}$ signifies the weight associated with each of the J_k Gaussian components, defined by mean $\mathbf{m}_k^{(i)}$ and covariance $\mathbf{P}_k^{(i)}$. Each Gaussian component undergoes propagation through a prediction and update step. The prediction step is formalized as follows:

$$v_{k|k-1}(\mathbf{x}_k) = \sum_{i=1}^{J_k} w_{k|k-1} \mathcal{N}(\mathbf{x}_k; \mathbf{m}_{k|k-1}^{(i)}, \mathbf{P}_{k|k-1}^{(i)}) + \gamma(\mathbf{z}_k) \quad (5.2)$$

with corresponding weight, mean, and covariance given by:

$$w_{k|k-1}^{(i)} = p_{s,k} w_{k-1}^{(i)} \quad (5.3)$$

The mean $\mathbf{m}_{k|k-1}^{(i)}$ and the covariance $\mathbf{P}_{k|k-1}^{(i)}$ follows an Extended Kalman Filter (EKF) with a transition model $\mathbf{m}_{k|k-1}^{(i)} = f(\mathbf{x}_{k-1}, \mathbf{u}_k)$ with a state model $\mathbf{x}_k = [x_k, y_k, \theta_k]^\top$, velocity input $\mathbf{u}_k = (v_k, \omega_k)$ and the linearized Jacobian matrix :

$$\mathbf{F}_{k-1} = \frac{\partial f(\mathbf{x}_{k-1}, \mathbf{u}_k)}{\partial \mathbf{x}_{k-1}} = \begin{bmatrix} 1 & 0 & v_k \delta k \cos \theta_k & \theta_k \\ 0 & 1 & v_k \delta k \sin \theta_k & \theta_k \\ 0 & 0 & 1 & \end{bmatrix}. \quad (5.4)$$

$\mathbf{Q}_{k-1}$ is the process noise covariance and the survival probability after the prediction step is denoted by $p_{s,k}$. The birth model $\gamma_k(\mathbf{z}_k)$ generate new Gaussian components with weight w_γ , mean $\mathbf{m}_\gamma$, and covariance $\mathbf{P}_\gamma$.

The update step is formulated as:

$$v_k(\mathbf{x}_k) = (1 - p_{d,k}) v_{k|k-1}(\mathbf{x}_k) + \sum_{\mathbf{z}_k \in \mathcal{Z}_k} \sum_{i=1}^{J_k} w_k^{(i)}(\mathbf{z}_k) \mathcal{N}(\mathbf{x}_k; \mathbf{m}_{k|k}^{(i)}(\mathbf{z}_k), \mathbf{P}_{k|k}^{(i)}) \quad (5.5)$$

with corresponding weight, mean, and covariance given by:

$$\begin{aligned}
w_k^{(i)}(\mathbf{z}_k) &= \frac{p_{d,k} w_{k|k-1}^{(i)} q_k^{(i)}(\mathbf{z}_k)}{\kappa_k(\mathbf{z}_k) + \sum_{j=1}^{J_{k|k-1}} p_{d,k} w_{k|k-1}^{(j)} q_k^{(j)}(\mathbf{z}_k)} \\
\mathbf{m}_{k|k}^{(i)}(\mathbf{z}_k) &= \mathbf{m}_{k|k-1}^{(i)} + \mathbf{K}_k^{(i)}(\mathbf{z}_k - \mathbf{H}_k \mathbf{m}_{k|k-1}^{(i)}) \\
\mathbf{P}_{k|k}^{(i)} &= (\mathbf{I} - \mathbf{K}_k^{(i)} \mathbf{H}_k) \mathbf{P}_{k|k-1}^{(i)}
\end{aligned} \tag{5.6}$$

and

$$\begin{aligned}
q_k^{(i)}(\mathbf{z}_k) &= \mathcal{N}(\mathbf{z}_k; \mathbf{H}_k \mathbf{m}_{k|k-1}^{(i)}, \mathbf{H}_k \mathbf{P}_{k|k-1}^{(i)} \mathbf{H}_k^\top + \mathbf{R}_k) \\
\mathbf{K}_k^{(i)} &= \mathbf{P}_{k|k-1}^{(i)} \mathbf{H}_k^\top (\mathbf{H}_k \mathbf{P}_{k|k-1}^{(i)} \mathbf{H}_k^\top + \mathbf{R}_k)^{-1}
\end{aligned} \tag{5.7}$$

where $\mathbf{H}_k$ is the measurement matrix and $\mathbf{R}_k$ is the measurement noise covariance. The observation model is nonlinear and consists of measurements of range and bearing:

$$h(\mathbf{x}_k) = \begin{bmatrix} r(x_k, y_k) \\ \beta(x_k, y_k) \end{bmatrix} = \begin{bmatrix} \sqrt{x_k^2 + y_k^2} \\ \text{atan2}(y_k, x_k) \end{bmatrix} \tag{5.8}$$

where we use the two-argument function atan2 to avoid ambiguities in the conversion from cartesian to polar coordinates. We linearize the measurement model by computing the Jacobian with respect to the state as:

$$\mathbf{H}_k = \frac{\partial h(\mathbf{x}_k)}{\partial \mathbf{x}_k} = \begin{bmatrix} \frac{x_k}{\sqrt{x_k^2 + y_k^2}} & \frac{y_k}{\sqrt{x_k^2 + y_k^2}} & 0 \\ \mathbf{0}_2 & \frac{y_k^2}{x_k^2 + y_k^2} & 0 \end{bmatrix} \tag{5.9}$$

The detection probability during the update step is denoted by $p_{d,k}$, and false positive detections are modeled as clutter κ_k .

In the filtering process, the number of Gaussian components increases with each iteration, leading to potential computational challenges. To maintain efficient tracking performance, we apply a three-step pruning strategy:

Pruning: Components with a weight below a certain threshold, T , are discarded. This reduces the number of components to a manageable level without significantly impacting the overall accuracy.

Merging: Components that are very close to each other, as determined by a Mahalanobis distance less than threshold U , are merged. This step helps in reducing redundancy among the components.

Selection: Out of the remaining components, only the top components, based on their weights, are retained. This step ensures that only the most significant components are considered, enhancing the efficiency of the process. This streamlined approach helps in managing the complexity of the filter, ensuring fast and effective tracking performance.

To specifically evaluate the performance of the Gaussian Mixture Probability Hypothesis Density (GMPHD) filter, we designed a tracking scenario where we measured

the filter's performance at multiple distances. The agents consisted of a follower and two leader robots. The two leaders were particularly placed at equidistant relative distances from the follower robot on the x -axis respectively 0.76 m and 0.76 m, creating a setup where the measurements varied based on their y -axis relative distances. During the evaluation, all three robots were given a constant velocity as input. We then executed the GMPHD filter on this scenario and recorded the results. The performance of the GMPHD filter was evaluated by analyzing its ability to accurately track the positions of the follower robot's relative distance and bearing to the two leader robots at 0.5 m, 1.5 m, and 2.5 m. We have compared the performance of the GMPHD filter at different distances by analyzing its tracking output to the ground truth (GT) and the measurement (meas). The performance can be observed from the following plots in Fig. 5.2, which shows the results of the evaluation at multiple distances.

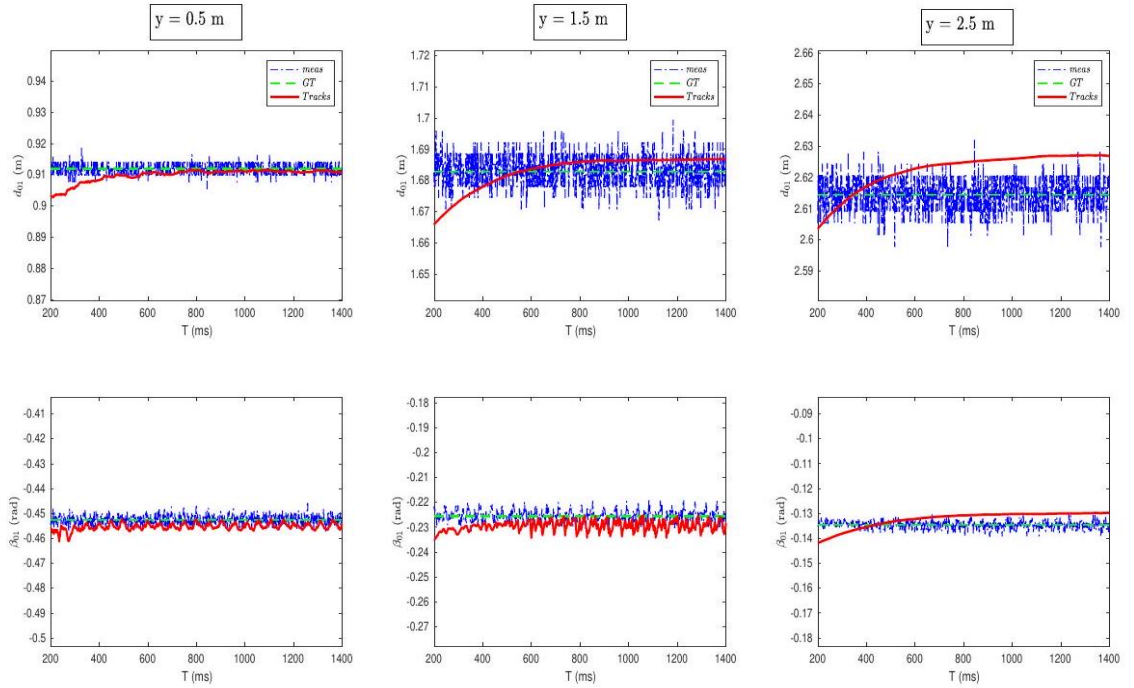


Figure 5.2 GM-PHD Experiments in a simulation environment: Each column represents a series of experiments at a specific distance while each row represents either the relative distance d_{ij} or the bearing β_{ij} from the follower agent $i = 0$ to the leaders $j = \{1, 2\}$ [111].

In conclusion, the evaluation of the GM-PHD filter at multiple distances demonstrated its ability to accurately track the positions of the follower robot relative to the two leader robots. The analysis of the filter's tracking outputs at 0.5 m, 1.5 m, and 2.5 m distances provided valuable insights into its performance. From the plots, each column

depicts a series of evaluations at a specific distance, while each row represents a measurement from the first leader robot.

5.3 Formation Graph Construction under Sensing Constraints

The robotic agents in the MRS aim to move in formation from a start configuration to a goal configuration. Formation configuration plays a significant role in achieving formation tasks successfully. While numerous formation graph structures have been proposed with appealing theoretical properties for multi-agent systems, most of these structures cannot be implemented directly on real-world systems because they ignore the sensor constraints in the graph construction stage. Henneberg-based directed graph construction method provides an effective solution to this problem, enabling the modeling and optimization of agent paths while accounting for the field-of-view (FOV) constraints of each sensor. Therefore, we apply a Henneberg-based graph construction method to satisfy the feasibility of the proposed relative localization approach.

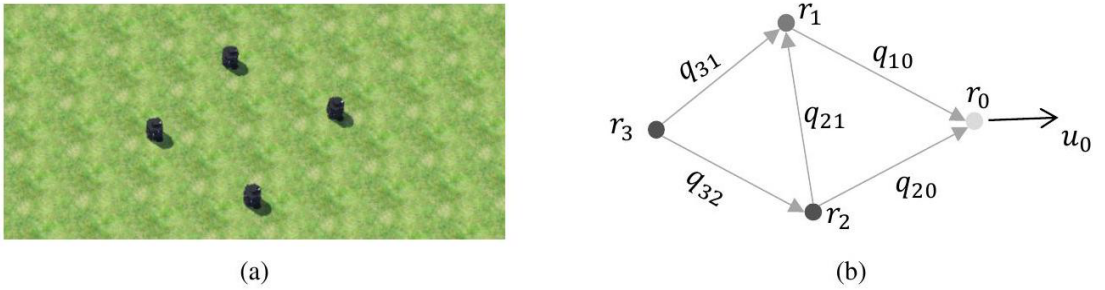


Figure 5.3 a) Four Turtlebot robot agents in the Webots simulation; b) The underlying L-FF graph among the robots [111].

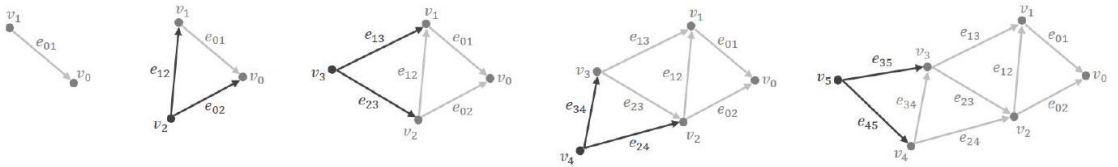


Figure 5.4 Vertex and edge addition operation in the Henneberg construction method. Two directed edges are added towards the previous vertices with every new vertex [111].

We define the formation framework by $\mathcal{F} = (\mathcal{S}, \mathcal{G}, \mathcal{Q})$, where $\mathcal{S} = \{R_0, R_1, \dots, R_N\}$, $N \geq 2$, denotes the swarm formed by the robotic agents, $\mathcal{G}$ denotes the underlying graph among the agents, and $\mathcal{Q} = \{q_{ij} \mid \overrightarrow{(i,j)} \in \mathcal{E}\}$ denotes the desired distance configuration. To minimize the required constraint links and the sensing requirements, the constraint graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \dots, N\}$ and $\mathcal{E} = \{\dots, (i,j), \dots\} \subset \mathcal{V} \times \mathcal{V}$ are respectively the set of the vertices and the edges in the graph, is designed to be minimally persistent (directed, minimally rigid, and constraint-consistent) which has the property of containing the minimum number of constraint links among the agents. A vertex $i \in \mathcal{V}$ represents an agent and an edge (i,j) defines a constraint between agent i and agent j such that agent j is to fulfill a relative position constraint toward agent i while agent i does not have any constraint regarding agent j . The set of neighbors of agent i is denoted by $\mathcal{N}_i = \{j \in \mathcal{V} \mid (i,j) \in \mathcal{E}\}$.

Minimally persistent graphs can be generated using the Henneberg construction method [60]. A sample graph construction is represented in Fig. 5.4, where one starts with the vertices $\mathcal{V}_0, \mathcal{V}_1$ and the edge $(0,1)$ and adds further vertices and the corresponding edges cumulatively. With the addition of vertex $\mathcal{V}_2$, the edges $(0,1)$ and $(0,2)$ are defined from $\mathcal{V}_2$ to $\mathcal{V}_0$ and $\mathcal{V}_1$, denoting the motion constraints. Then, the vertices $\mathcal{V}_3, \mathcal{V}_4$ and $\mathcal{V}_5$ are added such that each has two outgoing edges towards their leaders. One can continue building the constraint sets of the remaining follower agents so that vertex $\mathcal{V}_i$ has the neighbor set $\mathcal{N}_i = \{i-2, i-1\}$, $i \in \{3, \dots, N\}$, resulting in a minimally persistent graph $\mathcal{G}$ with $2N - 3$ edges.

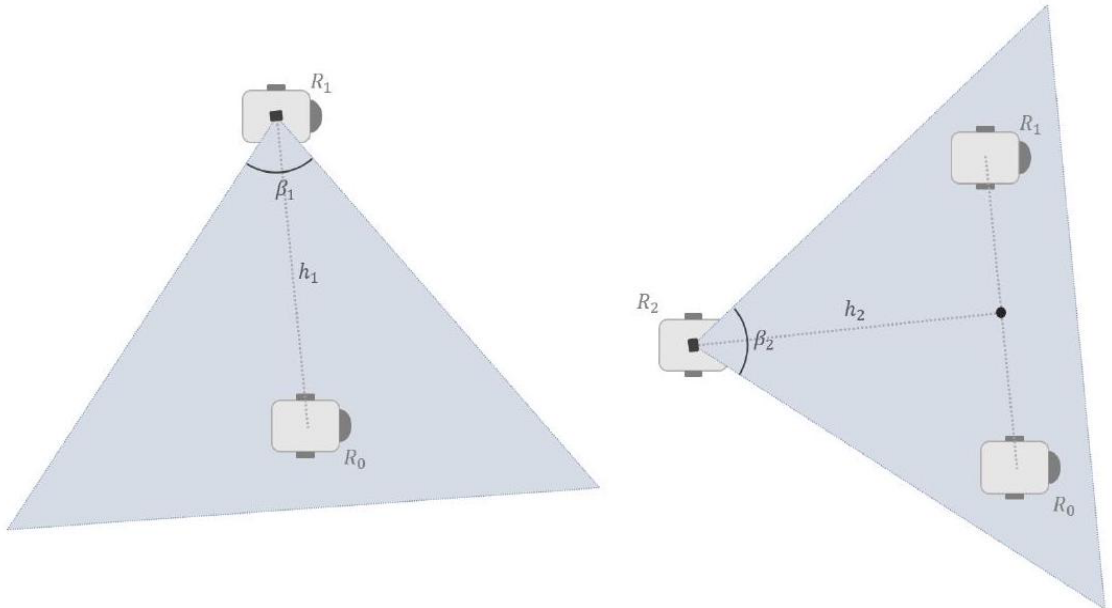


Figure 5.5 The Henneberg construction for three robots considering the visibility constraints. The camera of robot R_1 is directed towards its only leader, R_0 , while the camera of the second follower robot R_2 is directed towards its two leaders, R_0 and R_1 . The second follower robot R_2 is positioned on the line perpendicular to the line segment connecting its two leaders [111].

The embedding of the minimally persistent graph G in the formation framework $\mathcal{F}$ evokes a leader-first follower-second follower (L – FF) formation. The leader robot R_0 , corresponding to vertex $\mathcal{V}_0$, has no outgoing edge and possesses two degrees of freedom (2-DOF) without a mobility restriction in $\mathbb{R}^2$. The second vertex $\mathcal{V}_1$ denotes the first follower robot R_1 with one outgoing edge connection toward $\mathcal{V}_0$, which corresponds to 1-DOF in $\mathbb{R}^2$, i.e., robot R_1 is desired to satisfy one constraint toward robot R_0 . Each of the remaining vertices $\mathcal{V}_i, i \in \{2, \dots, N\}$ (the ordinary followers) has two outgoing edges toward its neighbors, which leads to 0 -DOF. For instance, robot R_2 is desired to satisfy two constraints, one toward the leader R_0 and one toward the first-follower R_1 . Thus, the entire MRS can move as a single entity, i.e., the entire vertex set can have translation and rotation motion in $\mathbb{R}^2$ as long as all distance constraints in 2 are satisfied.

To ensure that the formation $\mathcal{F}$ meets the requirements imposed by the sensing limitations, we integrate the FOV constraints into the Henneberg construction. Let β_i denote the FOV of the camera of robot $R_i, i \in \{2, \dots, N\}$. To maximize the visibility of the leader robots, the cameras are positioned so that they are directed towards their leader robots, regardless of their heading angles. The first follower robot R_1 can be positioned anywhere on the plane, and its camera is directed towards the leader R_0 . The second follower robot R_2 is positioned on the line intersecting perpendicularly to the midpoint of the line segment which connects its two leaders. Accordingly, the camera of R_2 is directed towards the midpoint of its two leaders, R_0 and R_1 , to improve the detection performance of the YOLO algorithm. Let h_2 denote the distance between the center of R_2 and the midpoint of the line segment connecting its two leaders. In this configuration, the visibility constraint imposes that $h_2^{\min} \leq h_2 \leq h_2^{\max}$, where $h_2^{\min}, h_2^{\max}$ denote the minimum and maximum limits for h_2 . The minimum $h_2^{\min}$ is caused by the geometrical constraint that if $h_2 < h_2^{\min}$, none of the leaders will be visible on the image frame. On the other hand, $h_2^{\max}$ determines the maximum distance where the YOLO algorithm can detect the robots reliably. Afterwards, the other second follower robots $R_i, i \in \{3, \dots, N\}$ are positioned in the same manner, and their cameras are directed towards the midpoint

of their two leaders, R_{i-2} and R_{i-1} . Therefore, the camera visibility constraints are handled with this scheme at the expense of restricting the allowed configuration sets of the robots.

Remark (1). In the proposed scheme, the camera of a robot can be oriented irrespective of the robot's heading, which provides flexibility in designing the formation shape.

Remark (2). The proposed formation graph construction rules under sensing constraints apply to other static MRS formation graphs, such as line, cyclic, and undirected graphs. The primary objectives in defining these rules are (1) placing the robot(s) to be detected in the camera image frame and (2) ensuring that the inter-robot distances obey the reliable detection range of the YOLO algorithm.

5.4 System Evaluation

Our experimental investigation comprehensively assesses the efficacy of the proposed framework on a four-robot system $\mathcal{S} = \{R_0, R_1, R_2, R_3\}$ through both simulations and real-world experiments. Notably, this setup is scalable, allowing for seamless integration into larger swarms with multiple second followers through minimal adjustments. The effectiveness of the proposed perception method and formation configuration setup was assessed with regards to accomplishing objectives.

5.4.1 Simulation

Webots simulation platform was employed with ROS2 for a detailed performance evaluation. It facilitates the realistic emulation of robot kinematics and various environmental dynamics with its advanced physical engine (Fig. 5.2a). Our experimental setup consisted of four Turtlebot3 robotic agents, each equipped with an RGB camera sensor. They relied on CNN-based detection from the RGB cameras, and the inter-robot distance measurements were emulated within the simulation by adding Gaussian noise to the GPS signals provided by Webots.

First, to specifically evaluate the performance of the Gaussian Mixture Probability Hypothesis Density (GM-PHD) filter, we tested a leader-follower scenario with robots R_0 and R_1 , where the follower robot R_1 was programmed to maintain a predetermined distance (1.5 m) from the leader R_0 . The relative position estimation on the x -axis and y -

axis, as well as the relative distance d_{01} , are shown for 37 seconds in Fig. 5.6. The results clearly demonstrated that the GM-PHD filter effectively filtered and smoothed out noise from the measurements, leading to more accurate and stable tracking of the relative position. This scenario underscores the GM-PHD filter's efficacy in enhancing measurement precision in dynamic leader-follower interactions.

The next task involved a formation maneuver performed by the four robots, where they were required to maintain the specified formation constraints while executing a linear motion path. This ensured that the robots adhered to both the desired spatial relationships and the non-holonomic motion requirements of the formation.

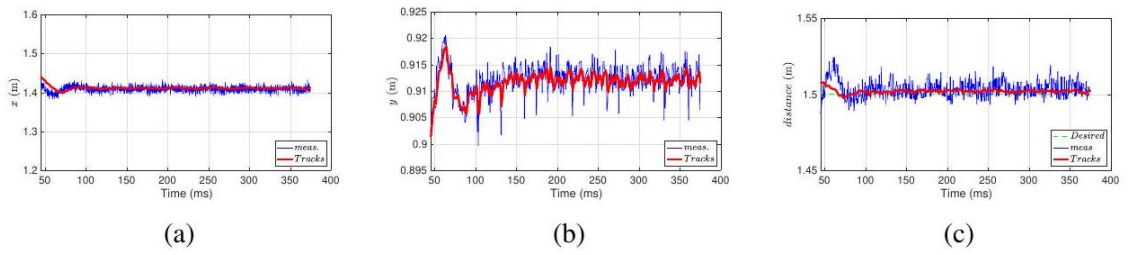


Figure 5.6 Leader-Follower task with GM-PHD: a)-b) The measurement and prediction of the leader position on the relative x and y axes, respectively; c) The measurement of the relative distance, its desired value, and the filter output [111].

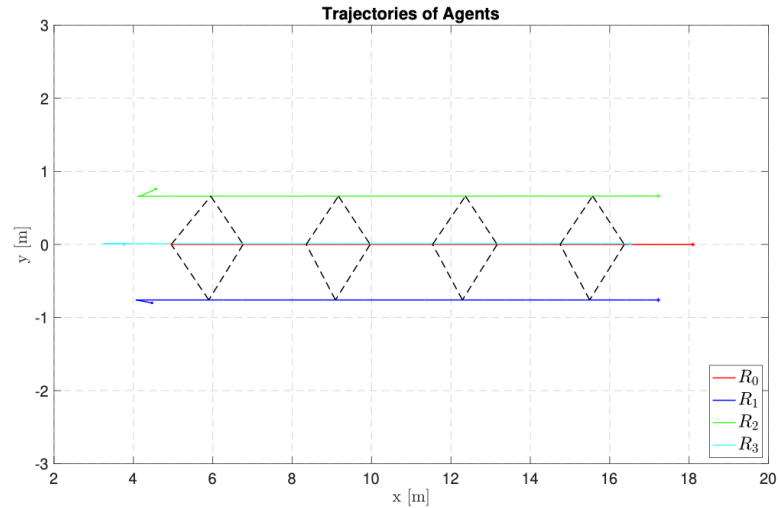


Figure 5.7 The trajectories of the agents in the simulations [111].

The trajectories generated by the robots during this maneuver are depicted in Fig. 5.7 which confirms that the agents successfully achieved formation acquisition and subsequently maintained the predetermined distance constraints. Furthermore, the convergence of the error metrics is demonstrated in Fig. 5.8, which clearly indicates a

rapid compliance with the necessary constraints. Notably, although the robots' initial positions were slightly off of the desired inter-robot distances (up to 0.2 m), the errors converged in the first few seconds.

Significantly, the framework demonstrated robust adaptability across various tests involving different speeds and initial conditions. These tests consistently showed that the robots were capable of both achieving and maintaining formation, provided the follower robots could detect their leaders within the camera's field of view. This robust performance under diverse conditions underscores the practical effectiveness of our framework in managing formation maneuvers.

Finally, the scalability of the proposed framework was evaluated by a six-robot system, where the formation graph was generated employing the proposed Henneberg construction technique (Fig. 5.9a). This scenario positioned the agents at their predefined locations relative to the target within the swarm. The primary task involved a coordinated formation maneuver, with the swarm's leader assigned a velocity of $\bar{u}_0 = 0.1$. The effectiveness of this scenario is evident in the trajectories depicted in Fig. 5.9b. Here, it is observable that the agents successfully maintained their formation constraints throughout the maneuver. This demonstration validates our framework's scalability and highlights its robustness in managing larger swarms while adhering to precise formation dynamics.

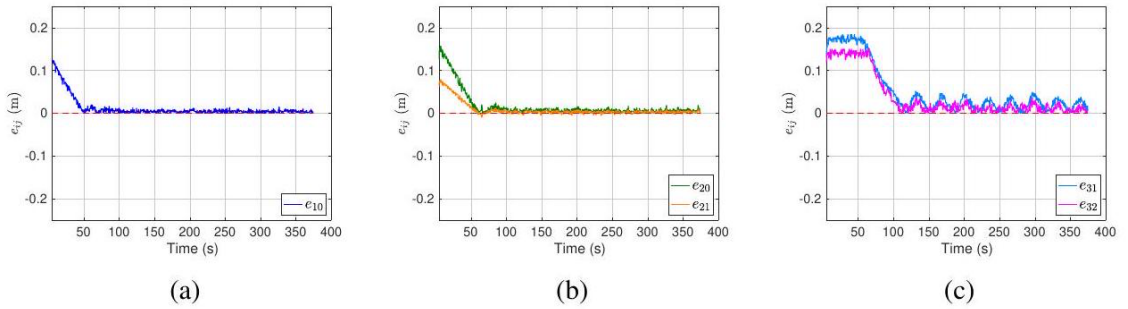


Figure 5.8 Simulation results: a) The error e_{10} between the First Follower R_1 and the leader R_0 ; b) The errors e_{20}, e_{21} ; and c) The errors e_{31}, e_{32} [111].

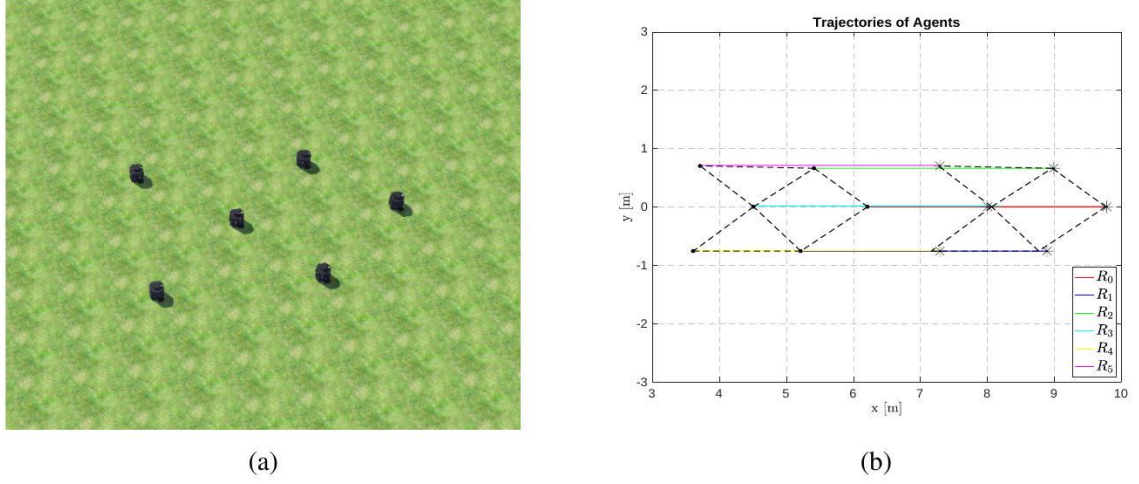


Figure 5.9 a) The agents in the webots simulation environment; b) The trajectories of the agents in the simulations with six robots. The dotted lines are the edges of the graph at the starting and end positions of the robots [111].

5.4.2 Indoors Experiments

To ascertain the practicality of our proposed framework, we executed a series of experiments in an indoor space of dimensions $4.5 \text{ m} \times 5.5 \text{ m}$. Our setup comprised a heterogeneous ensemble of four nonholonomic robots: a RosBost equipped with an Asus Tinker board and Rockchip RK3288 as the leader (R_0), a Turtlebot3 Burger robot as the first follower (R_1), another Turtlebot3 Burger as the second follower (R_2), and a second RosBost with identical specifications as the ordinary follower (R_3) Fig. 5.10. The sensor suite in our experimental setup was meticulously chosen to optimize performance. It comprised four Decawave UWB modules and three monocular cameras, each enhanced with a Jetson Nano board integrated into the RosBot and Turtlebot systems. This addition was specifically aimed at accelerating the CNN inference process.

To ensure the highest accuracy in our visual data, all monocular cameras have undergone a thorough calibration process. This step was crucial to eliminate distortions in the final visual outputs and was accomplished using the ROS perception package. Additionally, our approach to range sensing involved a meticulous calibration process to establish the initial measurement noise, commonly referred to as the bias. The inherent noise present in UWB sensors was addressed through a series of precise measurements.

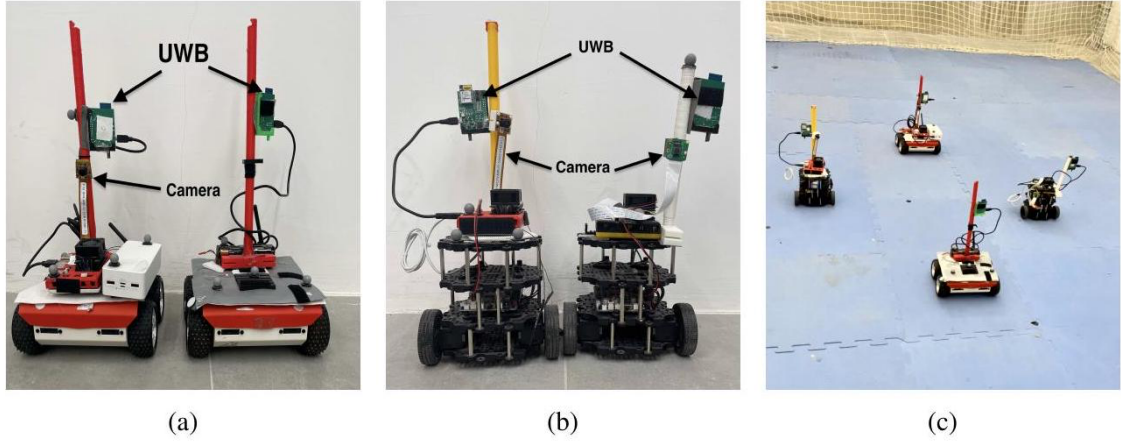


Figure 5.10 The robots used in the experiment: a) The leader with no camera and the last follower with a camera and UWBs. b) Two Turlebots with UWBs, Jetson Nano, and camera. c) The four agents in formation in an experiment [111].

This process enabled us to approximate and subsequently correct the noise impact, ensuring the accuracy of the projected range values. This comprehensive calibration and correction methodology was instrumental in enhancing the overall reliability and precision of our sensor configuration. For ground truth validation, a Vicon mocap camera system was used to track the robots' absolute positions within a global frame in the experimental arena. It is important to note that the mocap system was solely for validation purposes and not integrated into the robots' onboard control mechanisms. This setup allowed for precise and accurate ground truth measurements, essential for a rigorous assessment of our system's performance.

In our tests, we predefined specific desired inter-robot distances: $d_{10} = 0.9$ m, $d_{20} = 1.2$ m, $d_{21} = 1.4$ m, $d_{31} = 1.5$ m, and set a linear velocity for the leader R_0 at $\bar{u}_0 = 0.08$ m/s. The results, as depicted in Fig. 5.11 and Fig. 5.12, demonstrate successful convergence of the robots to the flocking motion, adhering to the distance constraints. Overall, our experiments validate the proposed framework's viability for controlling swarm motion, notably achieved without reliance on any specific positioning infrastructure. This aspect underscores the framework's adaptability to various indoor and outdoor environments.

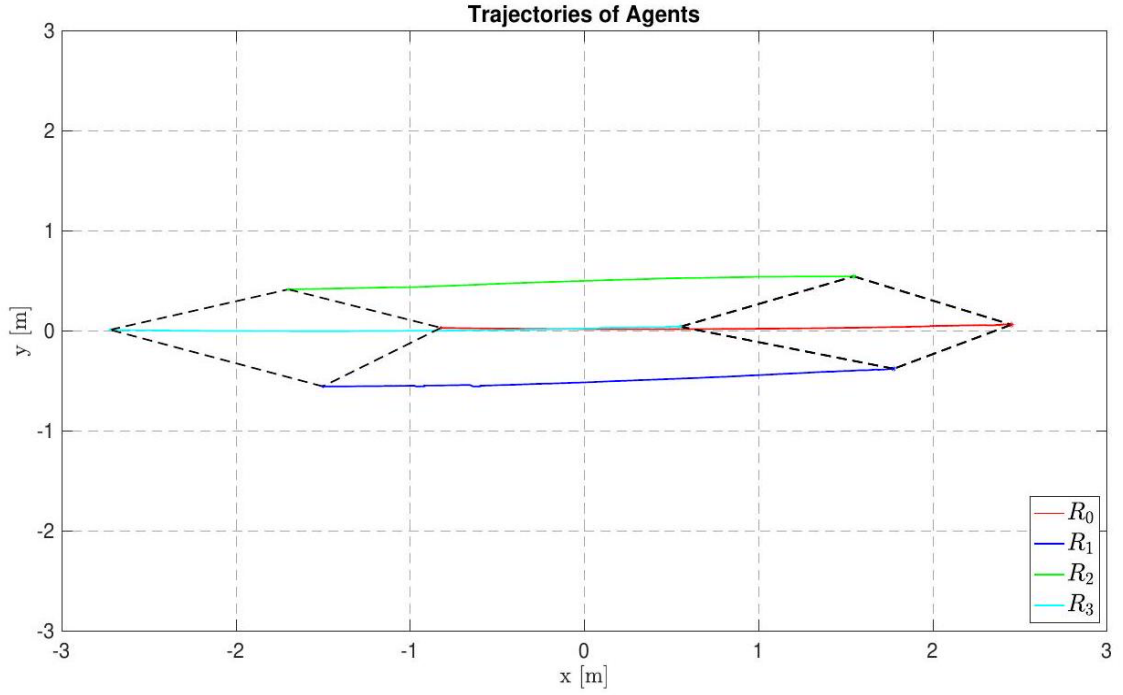


Figure 5.11 The trajectories of the agents during the indoor experiment. The dotted lines are the edges of the graph at the starting and end positions of the robots [111].

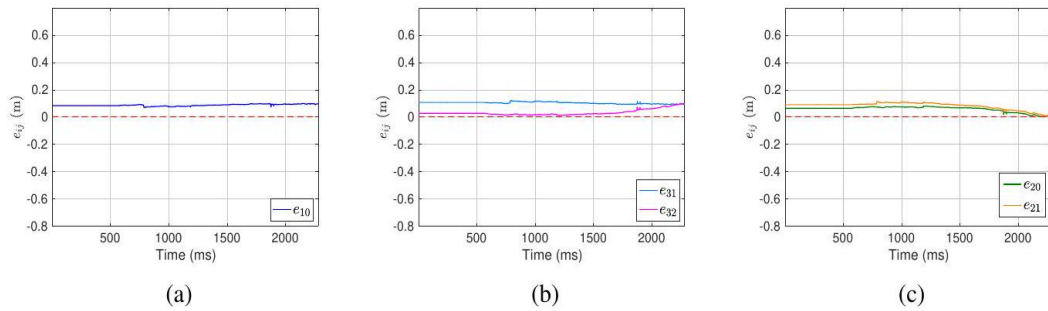


Figure 5.12 The errors during the simulations: a) The error between the First Follower R_1 and the leader R_0 , b) The plot e_{20}, e_{21} during the indoors experiment c) The plot e_{31}, e_{32} during the indoors experiment [111].

Chapter 6

Analysis and Discussion

We now analyze the results and discuss several points regarding the implementation of our framework. Firstly, we emphasize that the proposed closed-loop system differs from the usual formation control algorithms in that our control law depends on the onboard perception solely. The proposed framework does not take advantage of any external sensing or communication mechanism such as GPS and mocap. Therefore, it results in a completely distributed approach.

6.1 The Three-Coleader Cyclic Agent Model Analysis

6.1.1 Analysis

The distributed nature of the system allowed for the demonstration of the complete perception-decision mechanism on two drones via experimental methods. Moreover, our solution operates independently of the magnetometer. If velocity flocking or another type of formation control algorithm is relevant, the formation control law may be expanded to incorporate the absolute north as well. The absence of a common reference frame, such as the global north, results in the gradual drift of the entire three-drone system, even while adhering to the specified distance and bearing constraints. In our experiments, this drift manifested as a rotation of the follower drone around its coleader. Such rotations are anticipated, as the control laws ensure the convergence of the distance error but do not regulate the planar trajectory.

The wind disturbance hindered the drone's ability to execute precise velocity control inputs, thereby amplifying the resultant distance error. The DJI drone was maneuvered slowly within the room to avoid collisions with the boundaries, which consequently increased the reported distance errors. Wind disturbance is an inherent factor in typical outdoor experimental setups. Our indoor experiments demonstrated the applicability of the proposed algorithm under windy conditions.

The effectiveness of the entire algorithm is contingent upon the CNN algorithm employed; thus, the user must ensure adequate detection precision. A substantial quantity of training images was utilized, diversified according to several key factors including ambient lighting, background, and image perspectives. The presence of a single object class, specifically the drone, enables the CNN algorithm to achieve high detection performance across various scenarios. The primary factor facilitating the experiments on real hardware was the detection speed, which ranged from 60 to 85 ms per iteration on the lightweight Jetson Nano board, yielding approximately 14 Hz. To ensure safe flight, the motion control loop of a drone must operate at a minimum frequency of 20 Hz; thus, we implemented the estimation algorithm to address this requirement.

The proposed scheme establishes a closed-loop architecture in which the detection results from deep learning are transmitted to a control algorithm as feedback. The proposed formation control algorithm demonstrates an exponential convergence property, ensuring that formation constraints are satisfied in a short time, even in the presence of noisy feedback. The system is designed modularly, allowing for the utilization of various alternative perception and decision approaches within each module. A leader-follower constraint graph may serve as a substitute for the cyclic graph, allowing for the derivation of an appropriate formation control algorithm. Consequently, our architecture presents a viable end-to-end solution for prospective drone swarm applications.

6.1.2 Comparison with Other Approaches

This section presents a comparative examination of the proposed framework against several existing methodologies. Many current methodologies execute distributed formation control algorithms in laboratory settings utilizing a motion capture system, which fails to establish a genuinely distributed framework for two reasons. The drones consistently acquire accurate positional data at elevated data rates, significantly enhancing low-level control. Secondly, while a ground station disseminates the positions of all drones to every drone, each drone possesses access to the positions of all other drones inside the network, so contradicting the tenets of the distributed approach. Consequently, we compare our methodology with references [79], [6], and [85], which are entirely dependent on onboard sensors.

Saska et al. affix circular-shaped marks on drones for detection by the cameras of follower drones in a series of papers [79]. They employ a series of computer vision techniques to identify the markers and compute the three-dimensional inter-drone relative

position. Despite the method demonstrating good performance with an approximate detection time of 1 ms, its efficacy is contingent upon the size of the marker. We executed the software package in [10] and analyzed the distance calculations depicted in Fig. 6.1a. The pattern on the DJI, with a diameter of 8.5 cm, was undetected beyond 2 m by our camera. The zero readings can be mitigated by utilizing the last significant reading or applying a filter as outlined in [79], although the marker size imposes a limitation on the algorithm’s usefulness. Conversely, UWB distance sensors may yield accurate distance measurements of up to around 10 meters, making them a feasible choice for both interior and outdoor applications.

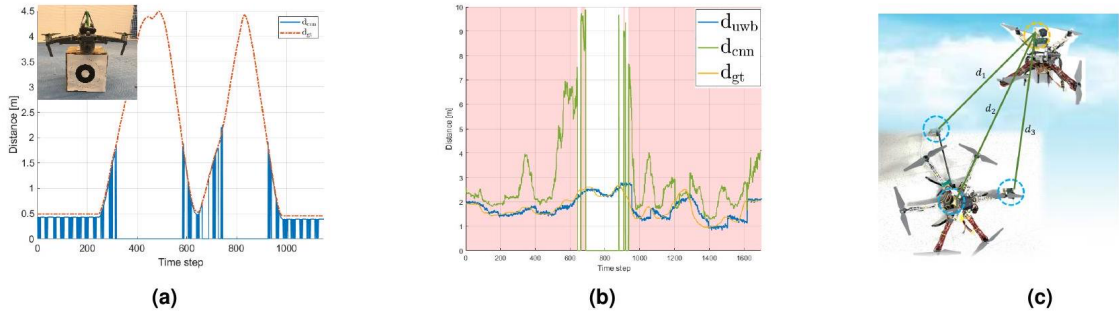


Figure 6.1 (a) The distance measurement obtained with the approach of [15] (the experimental setup seen in the embedded image); (b) Comparison of the distance acquired by the UWB sensor (blue), the distance calculated by the algorithm of [16] (green), and the ground truth (yellow); (c) A sample distance-based relative position estimation framework. [110]

In [16], a drone adheres to a leader drone utilizing a leader-follower formation control law, facilitated by the 3D relative location estimation generated by YOLO. The entire system relies on perceptual feedback and is decentralized. In contrast to our method, the 3D relative position is determined exclusively from the bounding box surrounding the identified drone. We employed the methodology of [6] in a bespoke configuration wherein a stationary camera on a Jetson Nano board identifies the DJI drone in flight. The method in [6] produces incorrect distance calculations, as the algorithm relies entirely on the dimensions of the detected bounding box. Fig. 6.1 depicts the distance estimation from [6], the obtained UWB distances, and the ground truth distance data from a sample test, highlighting the significant mistake in the distance estimation of [6]. The moments when the DJI drone is spotted are illustrated with a pink background. The projected distance may vary significantly even when the drone remains stationary. Due to the unreliability of the distance estimations, we refrained from conducting actual

experiments with the algorithm referenced in [6]. Despite offering a genuinely distributed methodology reliant solely on camera feedback, the algorithm in [6] may not be suitable for all formation control rules at minimal distances. Results for a minimum inter-drone distance of 5 meters are documented in [6].

The distance-based method aims to ascertain the relative positioning of drones using just distance measurements. In [85], a drone equipped with three UWB sensors endeavors to monitor a leader drone that is equipped with a UWB sensor. The following drone employs a particle filter to evaluate its relative position to the leader drone, achieving a root mean square error of around 1.6 meters. Although the reported outcomes appear promising and comparable to our methods, the system in [85] is impeded by scalability challenges. To execute the three-coleader-based control law, each drone must be equipped with at least three UWB sensors, resulting in a total of nine sensors, potentially causing signal loss and multipath effects. Furthermore, the UWB sensors must be installed on the drones with a minimum spacing of half a meter, so imposing an additional design constraint. Additionally, to implement a three-drone system using this approach, the drones must be positioned at different elevations to satisfy the line-of-sight criterion.

The inability of the aforementioned methods require the incorporation of distance and vision sensors to get high precision in various environmental situations, both indoor and outside. In our methodology, distance and vision measurements serve as complementing elements; distance data is utilized when the CNN detection algorithm fails to yield the bearing, and vice versa. In a tunnel search and rescue operation, consistent detection of the leader drone (or its markers) by camera systems cannot be guaranteed due to lighting conditions; but, UWB distance data can be acquired as long as the two drones remain in close proximity (e.g., less than 20 m).

6.2 Empowering Convolutional Neural Networks with UWB for Enhanced Localization and Detection

In a MRS application, as the number of agents or the spatial area expands, relying on a centralized positioning infrastructure can become prohibitively costly. Therefore, utilizing formation control aided by onboard sensors becomes imperative. To design a distributed and decentralized framework and to address our goal to propose a localization

approach based on onboard distance and vision data, acquired from low-cost monocular cameras and UWB sensors on the robots.

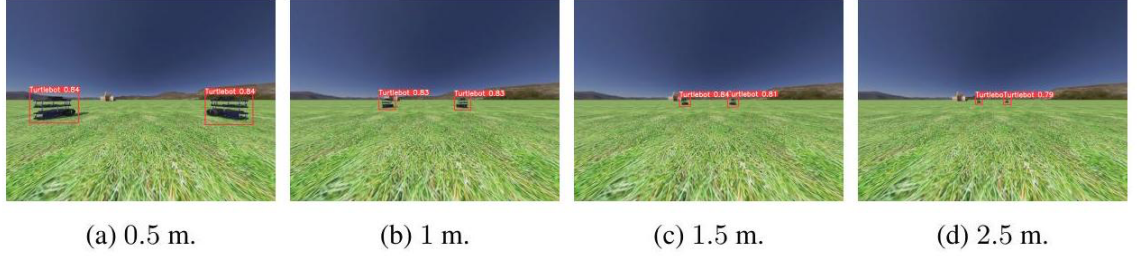


Figure 6.2 Detections at different distances in the simulation environment [111].

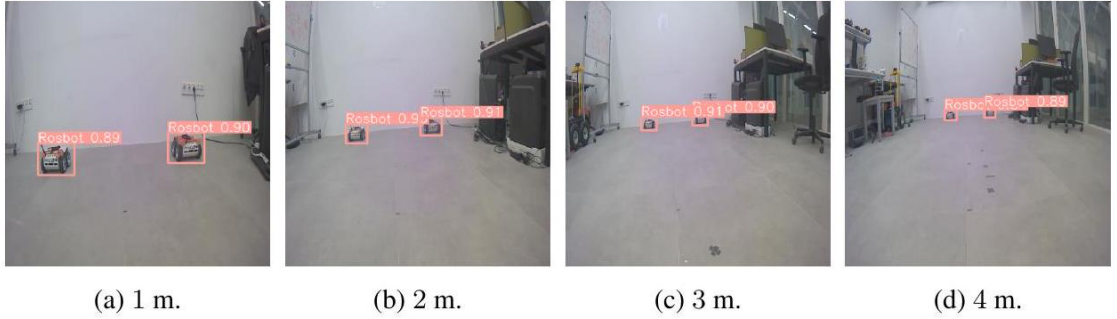


Figure 6.3 Detections at various distances in a room [111].

6.2.1 Sensor Characteristics and Distance and Bearing Angle Estimation

The goal of the localization algorithm is to produce the relative positions between each pair of robots in a MRS, which can be achieved by measuring the distance and bearing angles between the robots. On one side, a UWB sensor pair can produce omnidirectional distance measurements by applying the time-based algorithms such as time-of-flight (TOA) and time-difference-of-arrival (TDOA). As a result of communicating at large bandwidths, they generate distance data remarkably robust to noise. On the other side, camera sensors provide enormous information from image views. Remarkably, when integrated with the recent deep learning methods, they can be used for generating the relative bearing angle between robots. In this part, we aim to present the advantages of fusing distance and vision sensing modalities on the system performance by providing a comparative analysis.

The YOLO algorithm generates a bounding box around the detected robot as explained in Chapter 4. Thus, the presence of a neighbor robot can be detected on a robot's

camera. We conducted a set of experiments to analyze further if this detection result can be used to calculate the distance between the two robots and how it compares with the UWB distance measurements in terms of the error performance. In the first test series, we placed the robots in a line configuration where the detected robot resides in front of the detecting robot. In the second test series, the detecting robot was placed at a diagonal configuration so that both sides of the detected robot could be seen by the detecting robot. The distances between the robots were set to between 0.6 m and 4.8 m, with 0.2 m increments. At each distance configuration, the areas of the detected bounding boxes were recorded in 500 frames. The mean values of these areas versus the true distances are shown in Fig. 8a. Also, the variances of the 500 recordings can be seen in Fig. 6.4.

From Fig. 6.4a and Fig. 6.4, one can deduce that the variances of the recordings are so small that a function can be fit so that a distance estimation can be obtained for a detected box area. However, there are two drawbacks of this approach. First, since the detection does not give any clue about the robots' configuration (behind or diagonal), inconsistency can easily occur in distance estimations. For instance, for a detected box area of 400, the behind configuration produces 3 m, whereas the diagonal configuration produces 3.4 m (Fig. 6.5a). This inconsistency increases as the inter-robot distance increases. Second, the difference in the pixel values dramatically reduces as the inter-robot distance increases, which requires discriminating two distances by a few pixel differences. The performance gets even worse when we consider the different configurations of the two robots. On the other hand, the UWB sensor's distance profile shows almost linear behavior at all evaluated distance values, thanks to the TOA mechanism (Fig. 6.5b). Therefore, we use UWB sensors for distance measurements as it remains a more viable and robust approach compared to bounding box postprocessing.

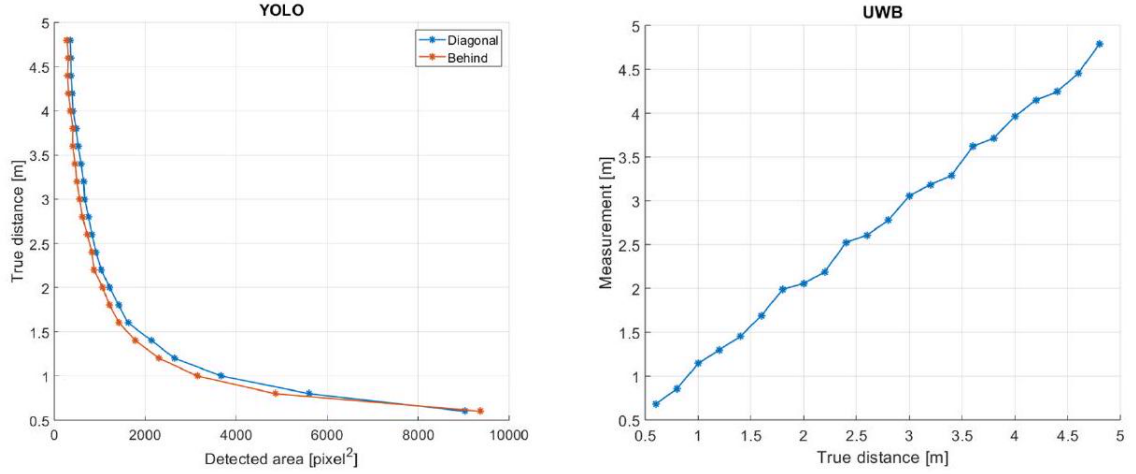


Figure 6.4 Distance measurement comparison between YOLO and UWB sensors. Left: The area of the detected bounding box versus the true distances between the robots, Right: Mean values of the UWB measurements versus the true distances between the robots [111].

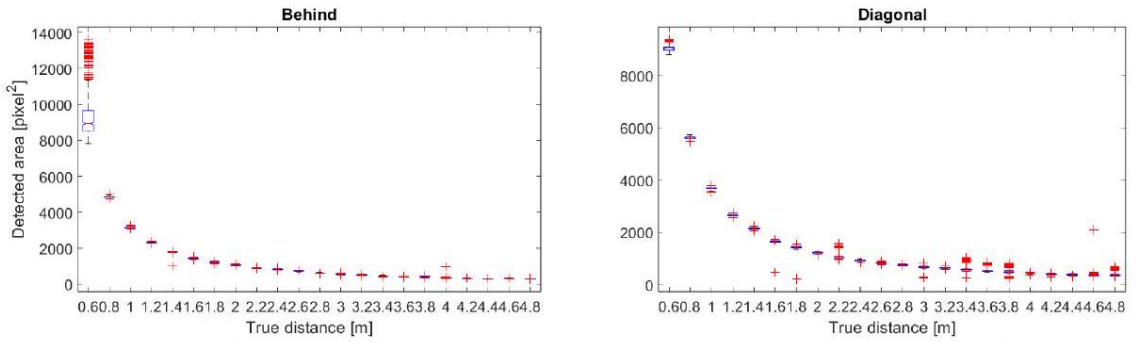


Figure 6.5 Variances of the converted distance measurements with YOLO. Left: Behind configuration, Right: Diagonal configuration [111].

6.3 Comparing Performance

6.3.1 Qualitative Comparison

This section shows a comparative analysis of our localization framework against prevalent existing methods, elucidating our approach's distinctive advantages and applicability.

A widely used onboard method is the visual marker-based technique, where each robot is outfitted with a distinct marker carrying an encoded ID and a specific size. This information facilitates relative localization among agents. Nonetheless, this method poses practical challenges, particularly in scenarios involving multiple agents. It necessitates

individual marker assignment and customization for each agent, which becomes cumbersome and less feasible, especially in swarm operations. Moreover, while effective in controlled laboratory settings, the marker-based approach encounters significant hurdles when transitioning to real-world environments. For instance, reference [6] utilized circular markers to differentiate between agents, a process that may not be as straightforward in diverse operational settings. Our framework, however, employs a marker-free approach, leveraging robust deep-learning detection techniques. This method seamlessly adapts to various environments and scales efficiently to larger numbers of agents without the need for additional markers.

Another limitation of marker-based systems is their reliance on the camera's field of view (FOV). The effectiveness of these systems is contingent upon having the entire marker within the camera's FOV, as partial marker visibility impedes accurate data extraction (Fig. 6.6b). In contrast, our deep learning-based detection demonstrates resilience in scenarios of partial visibility, still accurately providing relative positioning information (Fig. 6.6c).

Furthermore, our proposed sensor network approach offers significant advantages in terms of hardware efficiency. Traditional methods, such as those detailed in [99], often require redundant hardware for effective operation. For example, [99] employed a leader-follower model necessitating multiple ultrawideband (UWB) sensors for range measurements. Scaling such a system to include more agents leads to a proportional increase in the required hardware. Our framework, conversely, necessitates only a single range measurement between each target and follower, thereby ensuring a more streamlined and cost-effective solution.

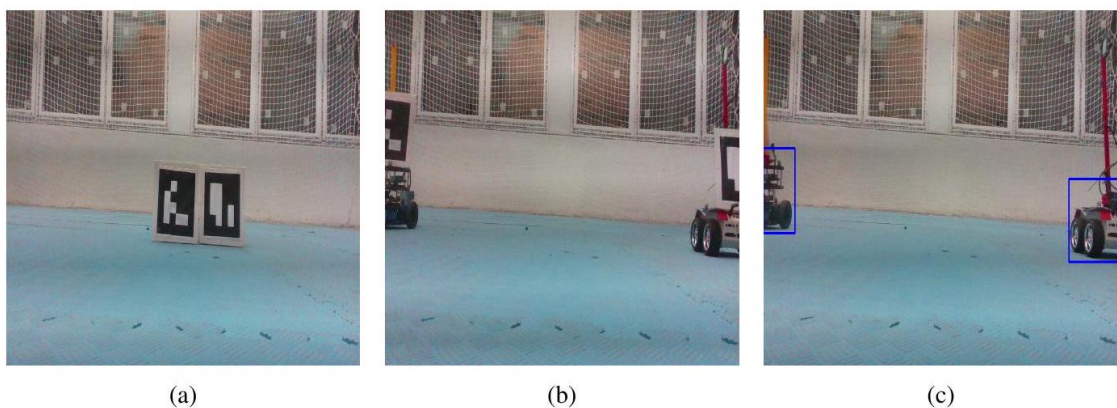


Figure 6.6 Comparison of our detection against Aruco marker: a) Two Aruco markers, b) The result from Aruco-based approach. C) The detection from a CNN-based approach [111].

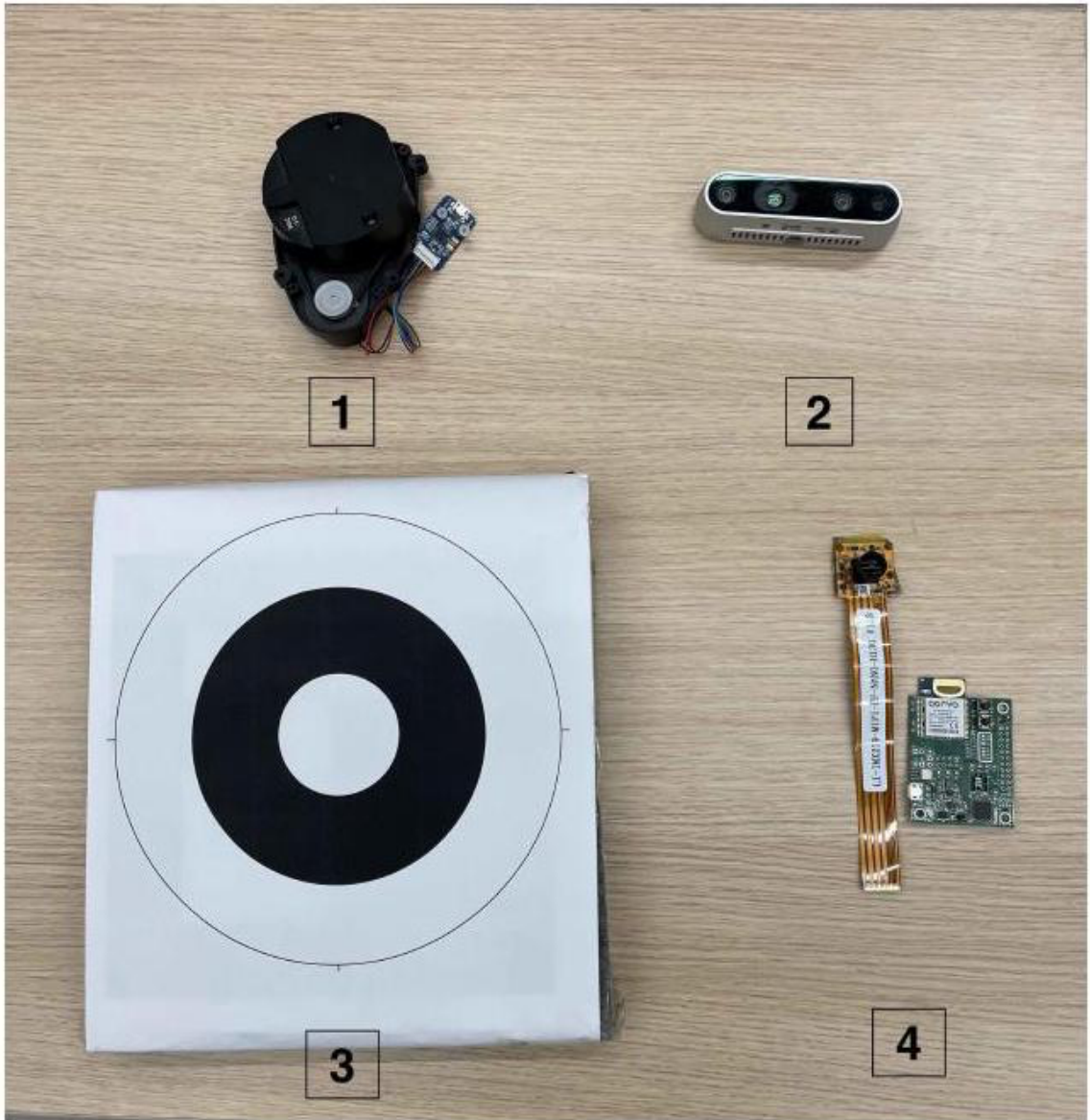


Figure 6.7 The Comparison and applications: a) The sensors and methods used in the comparison: 1LDS Lidar, 2-Intel RealSense Depth Camera D435i, 3-A whycon circular marker 4 -Ours: A camera and UWB [111].

6.3.2 Quantitative Comparison

We further conducted a quantitative comparison to assess the efficacy of our approach against some established localization methods. Our focus was on sensors commonly used in mobile robot localization. We selected two primary sensors and a marker-based method for this comparison.

The first sensor evaluated was the 2D LIDAR, a popular choice for non-holonomic vehicles such as Turtlebots and Rosbots. 2D LIDARs, like the LDS-01, are

preferred in small-sized robots for their practicality, offering minimal payload impact and low energy consumption. The LDS-01 features a 360degree angular range and a 1° angular resolution. We implemented a method to convert LIDAR scans into point clouds, subsequently clustering them to determine object distances [100].

Table 6.1 The quantitative comparison between methods [111].

Methods	Measurements									
	0.5	1	1.5	2	2.5	3	3.5	4	4.5	5
LDS Lidar	0.48	1.02	1.47	1.95	2.52	-	-	-	-	-
Whycon	0.49	0.98	1.47	1.91	2.43	-	-	-	-	-
RGB-D	0.47	0.98	1.48	1.97	2.46	2.92	-	-	-	-
Ours	0.53	1.04	1.45	1.91	2.53	3.07	3.59	4.03	4.46	5.05

The second sensor in our evaluation was the Intel RealSense D435i depth camera, which provides both image and pixel range data. This camera was used to calculate distances between agents, using its RGB capabilities for object detection and point cloud depth for distance measurements.

For the marker-based approach, we replicated the method available [101], using markers with an outer diameter of 122 mm and an inner diameter of 50 mm . The same camera employed in our framework was used for marker detection. Measurements were conducted at various distances, specifically at $d = \{0.5, 1.0, 1.5, 2.0, 3.0, 3.5, 4.0, 4.5, 5.0\}$ meters, to thoroughly assess each sensor and the marker-based system’s performance. The cost is defined in terms of the financial expenditure required to acquire and replicate the system.

Our analysis demonstrates that our framework not only provides a more economical solution but also achieves a superior operational range compared to the methods we evaluated. Specifically, the LDS LIDAR and the Whycon system were limited to a maximum range of 2.5 m, and the depth camera’s effectiveness did not extend beyond 3 m. In contrast, our approach shows potential for effective performance at distances exceeding 5 m, provided that it is supported by datasets encompassing long-range images. For this study, we focused on a dataset capturing mobile robots at distances up to 5.5 m.

6.4 Limitations

Despite its advantages, our localization framework utilizing cameras and UWB technology has limitations. First, the framework heavily relies on the quality and relevance of the image dataset. A well-curated and accurately labeled dataset is essential for robust performance, as it must capture the relative distances between agents effectively. For example, if formation control requires agents to maintain a 4-meter spacing, the dataset should include sufficient data collected at this specific distance to ensure high localization accuracy. To address this dependency, a more automated approach to dataset collection and labeling could be employed. Instead of relying solely on real image data, artificial datasets generated through simulation or augmentation techniques could balance and enhance the dataset for critical distance ranges.

Second, the camera's dependency on lighting conditions presents challenges in environments with poor or inconsistent illumination. In dark or dimly lit areas, the camera's ability to capture accurate visual data diminishes, potentially affecting the overall performance of the system. This limitation makes the framework less effective for nighttime operations or poorly lit indoor environments. However, since our approach primarily uses the camera to extract bearing angles, alternative sensors capable of providing bearing information, such as stereo cameras or infrared sensors, could be integrated to adapt to different lighting conditions. Monocular cameras could also be substituted with other camera types optimized for low-light environments, enhancing the framework's versatility.

Third, while UWB offers precise range measurements, it is susceptible to noise and signal degradation, particularly as the number of agents in the system increases. Higher agent density can lead to interference and reduced reliability of localization data. Nevertheless, since our framework primarily utilizes UWB for distance measurements, it offers the flexibility to replace UWB with other sensors capable of providing better range accuracy in specific scenarios. For example, different types of UWB devices or alternative range sensors, such as LIDAR or ultrasonic sensors, could be integrated to address scalability issues and improve performance.

Overall, our framework's adaptability allows it to overcome these challenges by incorporating alternative sensing modalities and approaches to match real-world circumstances, ensuring continued effectiveness across diverse environments and scenarios.

Chapter 7

Conclusions and Future Prospects

7.1 Conclusions

We have presented a distributed, marker-free localization and formation maneuvering system intended to function efficiently across several contexts, including both indoor and outdoor ones. This framework provides a more flexible and scalable solution fit for more application scenarios than conventional marker-based systems, which can depend on optical markers or GPS and so have limits.

Our main contribution is to the adaptability and scalability of the suggested Multi-Robot Systems (MRS) architecture. We present a strong relative localization mechanism using two complementing sensory modalities, hence improving the accuracy and durability of the framework in demanding conditions with few or inaccurate visual clues. Maintaining formation in dynamic and congested environments depends on redundancy, which this hybrid technique not only increases but also enhances localization accuracy.

Our solution is based on a new formation graph building technique that uses the limitations of the localization process. This approach enables robots inside the system to build and preserve a dynamic graph structure, therefore allowing them to adjust their relative positions in response to changes in formation shape or size. Furthermore, guaranteeing optimal communication and interaction among robots, the formation graph supports local and worldwide control strategies inside the MRS.

Extensive simulations and real-world tests have helped us to completely validate the suggested approach. While the actual trials gave important insights on its practical relevance and performance, the simulations let us test the robustness of the framework

under several conditions, including high degrees of noise, occlusions, and communication delays. These tests show that, even in difficult settings where conventional techniques may fail, our methodology can attain consistent localization and steady formation control.

Furthermore, we have made an open-source dataset especially for ground robots available for MRS uses. For localization and development activities, this dataset comprises labeled data, thereby offering a useful resource for academics and developers trying to evaluate or improve their own algorithms.

By improving the localization and formation capabilities of robotic systems, therefore opening the path for more effective, resilient, and flexible robotic partnerships, our study greatly advances the state-of-the-art in MRS. Where effective and flexible multi-robot formations are essential, this framework has possible uses including warehouse automation, search and rescue operations, and autonomous exploration.

7.2 Societal Impact and Contribution to Global Sustainability

7.2.1 Societal and Scientific Impacts

The last decade has witnessed huge developments in the field of aerial robotics. Unmanned aerial vehicles (UAV) offer unparalleled benefits in a wide range of domains. Although primarily military applications were the driving force for the UAV research, many civilian applications have emerged as well in the recent years. For instance, exploration and monitoring of risky locations such as volcanoes can be realized by UAVs. Small-sized, agile UAVs can be controlled remotely to collect visual measurements in such dangerous environments [102]. UAVs are revolutionizing the field of agriculture by providing instructive feedback about the crops and fields [103]. They can reduce the cost of maintenance in traffic monitoring and building inspection [104-105]. Thanks to their flexible and modifiable structures, UAVs can be designed for challenging tasks such as cleaning of windows of tall buildings [106]. Moreover, UAVs can be miniaturized with the state-of-the-art, small-sized equipment to give them the capability of performing in areas which might be impossible for humans to access.

The proposed frameworks can be integrated to small-size drones operating outdoors. Since the proposed frameworks do not depend on a fixed infrastructure, the

drone team can operate both indoors and outdoors in several application domains. For instance, border patrolling can be performed with a drone team easier and more efficiently than a single drone. Mobility and easy integration of the proposed frameworks into other formation control algorithms is another significant property. For instance, the relative localization frameworks can be well integrated to a coverage algorithm to enable drones perform coverage both indoors and outdoors without any modification.

7.2.2 Sustainable Development Goals

The thesis is related to the AGU’s Research Focus Area “Cities and societies”, which are set based on the United Nations Sustainable Development Goals. The drone swarms will be present in every aspect of the urban lifestyle in several domains. For instance, a U.S. based e-commerce company is currently developing “delivery drones” which deliver packages from warehouses to residents in a short time. The weight a drone can carry is currently limited; however, delivery of heavy packages would be possible with drone teams.

Furthermore, drone teams can be implemented as auxiliary internet service providers after disasters. It is a known fact that such services are prone to catastrophic events such as earthquakes and floods because these disasters affect significantly their infrastructure. A drone team can be developed for after-disaster scenarios such that each drone can carry low-weight communication sources onboard. By placing the drones at critical locations in the city, the communication infrastructure can be improved.

Teams of drones can also be employed to improve agriculture efficiency. A team of drones can patrol on an agricultural field for fertilization, which could take much more time and effort with humans.

7.3 Future Prospects: Towards RL and GNN

Combining Graph Neural Networks (GNN)[107], [108] and Reinforcement Learning (RL)[109] with our distributed localization system presents interesting developments for multi-robot systems (MRS), hence enabling more flexible, autonomous, intelligent multi-agent behavior.

Graph neural networks (GNNs) are ideal for MRS structured data processing since every robot is modeled as a node compiling information from its neighbors. Each robot can learn to make distributed decisions depending on local interactions by employing

GNNs, therefore enabling scalable formation control without depending on global communication. MRS can thus adjust to changes in team size or environmental circumstances. GNNs, trained on different production jobs, enable robots to sustain forms—such as lines, circles, or more intricate shapes—robustly across many environments. GNNs help the system to generalize to new circumstances by learning from much facts, therefore providing flexibility and dependability.

By learning policies that optimize rewards for desired behaviors—such as maintaining formation, avoiding collisions, and conserving energy— Reinforcement Learning (RL) can equip robots to navigate and maintain formation independently. Learning by interaction with its surroundings, every robot functions as an autonomous RL agent. Defining reward functions that give formation integrity and safety top priority helps RL let the robots learn and change with dynamic surroundings. RL lets robots change their formations depending on factors like grouping together in confined locations or dispersing out over open areas.

MRS capacities could be improved even more by a hybrid GNN-RL system. Under this method, RL decides on best behaviors while GNNs offer a structured knowledge of each robot’s local surroundings. Real-time decision-making made possible by this mix lets every robot make cooperative, data-based changes that increase formation stability, efficiency, and adaptability in challenging environments.

We see a more autonomous and flexible MRS competent of robust formation control, dynamic reconfiguration, and efficient job management in applications like search and rescue, environmental monitoring, and warehouse automation by combining GNN and RL with our localization framework. The development of intelligent multi-robot systems advances significantly with the synergy between distributed localization and powerful machine learning methods.

BIBLIOGRAPHY

- [1] H.-S. Ahn, *Formation Control*, Springer Nature 2020, doi: 10.1007/978-3-030-15187-4.
- [2] S. J. Chung, A. A. Paranjape, P. Dames, S. Shen, and V. Kumar, ‘A Survey on Aerial Swarm Robotics’, *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 837–855, Aug. 2018, doi: 10.1109/TRO.2018.2857475.
- [3] G. Silano, T. Baca, R. Penicka, D. Liuzza, and M. Saska, ‘Power Line Inspection Tasks With Multi-Aerial Robot Systems Via Signal Temporal Logic Specifications’, *IEEE Robot Autom Lett*, vol. 6, no. 2, pp. 4169–4176, 2021.
- [4] M. Abdelkader, S. Güler, H. Jaleel, and J. S. Shamma, ‘Aerial Swarms: Recent Applications and Challenges’, *Current Robotics Reports 2021 2:3*, vol. 2, no. 3, pp. 309–320, Jul. 2021, doi: 10.1007/S43154-021-00063-4.
- [5] J. Horyna *et al.*, ‘Decentralized swarms of unmanned aerial vehicles for search and rescue operations without explicit communication’, *Auton Robots*, pp. 1–17, Oct. 2022, doi: 10.1007/S10514-022-10066-5/FIGURES/12.
- [6] M. Vrba and M. Saska, ‘Marker-Less Micro Aerial Vehicle Detection and Localization Using Convolutional Neural Networks’, *IEEE Robot Autom Lett*, vol. 5, no. 2, pp. 2459–2466, 2020.
- [7] J. C. Trujillo, R. Munguia, S. Urzua, and A. Grau, ‘Cooperative Visual-SLAM System for UAV-Based Target Tracking in GPS-Denied Environments: A Target-Centric Approach’, *Electronics 2020, Vol. 9, Page 813*, vol. 9, no. 5, p. 813, May 2020, doi: 10.3390/ELECTRONICS9050813.
- [8] S. Güler, M. Abdelkader, and J. S. Shamma, ‘Infrastructure-free multi-robot localization with ultrawideband sensors’, in *2019 American Control Conference (ACC)*, 2019, pp. 13–18.
- [9] M. Mesbahi and M. Egerstedt, ‘Graph Theoretic Methods in Multiagent Networks’, *Princeton Series in Applied Mathematics*, Jul. 2010, doi: 10.1515/9781400835355.
- [10] H.-S. Ahn, ‘Moving Formation’, *Studies in Systems, Decision and Control*, vol. 205, pp. 3–26, 2020, doi: 10.1007/978-3-030-15187-4_1.
- [11] K. Kondak *et al.*, ‘Unmanned aerial systems physically interacting with the environment: Load transportation, deployment, and aerial manipulation’, *Handbook of Unmanned Aerial Vehicles*, pp. 2755–2785, 2015.
- [12] M. Basiri, F. Schill, P. Lima, and D. Floreano, ‘On-Board Relative Bearing Estimation for Teams of Drones Using Sound’, *IEEE Robot Autom Lett*, vol. 1, no. 2, pp. 820–827, 2016, doi: 10.1109/LRA.2016.2527833.
- [13] M. Coppola, K. N. McGuire, K. Y. W. Scheper, and G. C. H. E. de Croon, ‘On-board communication-based relative localization for collision avoidance in Micro Air Vehicle teams’, *Auton Robots*, vol. 42, no. 8, pp. 1787–1805, 2018, doi: 10.1007/s10514-018-9760-3.

- [14] S. van der Helm, M. Coppola, K. N. McGuire, and G. C. H. E. de Croon, ‘On-board range-based relative localization for micro air vehicles in indoor leader–follower flight’, *Auton Robots*, vol. 44, no. 3–4, pp. 415–441, Mar. 2020, doi: 10.1007/S10514-019-09843-6/FIGURES/33.
- [15] F. Mazhar, M. G. Khan, and B. Sällberg, ‘Precise Indoor Positioning Using UWB: A Review of Methods, Algorithms and Implementations’, *Wirel Pers Commun*, vol. 97, no. 3, pp. 4467–4491, 2017, doi: 10.1007/s11277-017-4734-x.
- [16] S. Guler, M. Abdelkader, and J. S. Shamma, ‘Infrastructure-free multi-robot localization with ultrawideband sensors’, *Proceedings of the American Control Conference*, vol. 2019-July, pp. 13–18, 2019, doi: 10.23919/acc.2019.8814678.
- [17] S. Guler, J. Jiang, A. A. Alghamdi, R. I. Masoud, and J. S. Shamma, ‘Real Time Onboard Ultrawideband Localization Scheme for an Autonomous Two-robot System’, *2018 IEEE Conference on Control Technology and Applications, CCTA 2018*, pp. 1151–1158, 2018, doi: 10.1109/CCTA.2018.8511568.
- [18] S. Guler, M. Abdelkader, and J. S. Shamma, ‘Peer-to-Peer Relative Localization of Aerial Robots with Ultrawideband Sensors’, *IEEE Transactions on Control Systems Technology*, vol. 29, no. 5, pp. 1981–1996, Sep. 2021, doi: 10.1109/TCST.2020.3027627.
- [19] T. M. Nguyen, A. H. Zaini, C. Wang, K. Guo, and L. Xie, ‘Robust Target-Relative Localization with Ultra-Wideband Ranging and Communication’, *Proc IEEE Int Conf Robot Autom*, pp. 2312–2319, Sep. 2018, doi: 10.1109/ICRA.2018.8460844.
- [20] A. K. Das, R. Fierro, V. Kumar, J. P. Ostrowski, J. Spletzer, and C. J. Taylor, ‘A vision-based formation control framework’, *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 813–825, 2002, doi: 10.1109/TRA.2002.803463.
- [21] G. L. Mariottini, F. Morbidi, D. Prattichizzo, G. J. Pappas, and K. Daniilidis, ‘Leader-follower formations: Uncalibrated vision-based localization and control’, *Proc IEEE Int Conf Robot Autom*, pp. 2403–2408, 2007, doi: 10.1109/ROBOT.2007.363679.
- [22] D. Guo, H. Wang, W. Chen, M. Liu, Z. Xia, and K. K. Leang, ‘A unified leader-follower scheme for mobile robots with uncalibrated on-board camera’, *Proc IEEE Int Conf Robot Autom*, no. 5, pp. 3792–3797, 2017, doi: 10.1109/ICRA.2017.7989438.
- [23] B. Fidan, V. Gazi, S. Zhai, N. Cen, and E. Karatas, ‘Single-view distance-estimation-based formation control of robotic swarms’, *IEEE Transactions on Industrial Electronics*, vol. 60, no. 12, pp. 5781–5791, 2013, doi: 10.1109/TIE.2012.2236996.
- [24] M. Aranda, Y. Mezouar, G. Lopez-Nicolas, and C. Sagues, ‘Partially distributed multirobot control with multiple cameras’, *Proceedings of the American Control Conference*, pp. 6308–6314, 2013, doi: 10.1109/acc.2013.6580827.
- [25] H. Wang, D. Guo, X. Liang, W. Chen, G. Hu and K. K. Leang, ‘Adaptive Vision-Based Leader–Follower Formation Control of Mobile Robots,’ *IEEE Transactions on Industrial Electronics*, vol. 64, no. 4, pp. 2893–2902, April 2017, doi: 10.1109/TIE.2016.2631514.

- [26] Štěpán, P., Krajník, T., Petrлік, M., & Saska, M. (2019). Vision techniques for on-board detection, following, and mapping of moving targets. *Journal of Field Robotics*, 36(1), 252-269.
- [27] P. Lightbody, T. Krajník, and M. Hanheide, 'An efficient visual fiducial localisation system', *ACM SIGAPP Applied Computing Review*, vol. 17, no. 3, pp. 28–37, 2017, doi: 10.1145/3161534.3161537.
- [28] Lightbody, P., Krajník, T., & Hanheide, M. (2017, April). A versatile high-performance visual fiducial marker detection system with scalable identity encoding. In *Proceedings of the symposium on applied computing* (pp. 276-282).
- [29] M. Saska *et al.*, 'System for deployment of groups of unmanned micro aerial vehicles in GPS-denied environments using onboard visual relative localization', *Auton Robots*, vol. 41, no. 4, pp. 919–944, 2017, doi: 10.1007/s10514-016-9567-z.
- [30] T. Krajník *et al.*, 'A Practical Multirobot Localization System', *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 76, no. 3–4, pp. 539–562, 2014, doi: 10.1007/s10846-014-0041-x.
- [31] J. Faigl, T. Krajník, J. Chudoba, L. Preucil, and M. Saska, 'Low-cost embedded system for relative localization in robotic swarms', *Proc IEEE Int Conf Robot Autom*, pp. 993–998, 2013, doi: 10.1109/ICRA.2013.6630694.
- [32] V. Walter, N. Staub, A. Franchi, and M. Saska, 'UVDAR System for Visual Relative Localization with Application to Leader-Follower Formations of Multirotor UAVs', *IEEE Robot Autom Lett*, vol. 4, no. 3, pp. 2637–2644, 2019, doi: 10.1109/LRA.2019.2901683.
- [33] E. Montijano, E. Cristofalo, D. Zhou, M. Schwager, and C. Sagues, 'Vision-Based Distributed Formation Control Without an External Positioning System', *IEEE Transactions on Robotics*, vol. 32, no. 2, pp. 339–351, 2016, doi: 10.1109/TRO.2016.2523542.
- [34] A. Weinstein, A. Cho, G. Loianno, and V. Kumar, 'Visual inertial odometry swarm: An autonomous swarm of vision-based quadrotors', *IEEE Robot Autom Lett*, vol. 3, no. 3, pp. 1801–1807, 2018, doi: 10.1109/LRA.2018.2800119.
- [35] M. Aranda, G. Lopez-Nicolas, C. Sagues, and Y. Mezouar, 'Formation Control of Mobile Robots Using Multiple Aerial Cameras', *IEEE Transactions on Robotics*, vol. 31, no. 4, pp. 1064–1071, 2015, doi: 10.1109/TRO.2015.2452777.
- [36] F. Schilling, J. Lecoer, F. Schiano, and D. Floreano, 'Learning Vision-Based Flight in Drone Swarms by Imitation', *IEEE Robot Autom Lett*, vol. 4, no. 4, pp. 4523–4530, 2019, doi: 10.1109/LRA.2019.2935377.
- [37] T. Nguyen *et al.*, 'MAVNet: An Effective Semantic Segmentation Micro-Network for MAV-Based Tasks', *IEEE Robot Autom Lett*, vol. 4, no. 4, pp. 3908–3915, 2019, doi: 10.1109/lra.2019.2928734.
- [38] Nguyen, T., Mohta, K., Taylor, C. J., & Kumar, V. (2020, May). Vision-based multi-MAV localization with anonymous relative measurements using coupled probabilistic data association filter. In *2020 IEEE International Conference on Robotics and Automation (ICRA)* (pp. 3349-3355). IEEE.
- [39] M. Vrba, D. Hert, and M. Saska, 'Onboard Marker-Less Detection and Localization of Non-Cooperating Drones for Their Safe Interception by an

- Autonomous Aerial System', *IEEE Robot Autom Lett*, vol. 4, no. 4, pp. 3402–3409, 2019, doi: 10.1109/LRA.2019.2927130.
- [40] D. G. Lowe, 'Distinctive Image Features from Scale-Invariant Keypoints', *International Journal of Computer Vision* 2004 60:2, vol. 60, no. 2, pp. 91–110, Nov. 2004, doi: 10.1023/B:VISI.0000029664.99615.94.
 - [41] H. Bay, T. Tuytelaars, and L. Van Gool, 'SURF: Speeded Up Robust Features', *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3951 LNCS, pp. 404–417, 2006, doi: 10.1007/11744023_32.
 - [42] Devi, N. B., Kavida, A. C., & Murugan, R. (2022). Feature extraction and object detection using fast-convolutional neural network for remote sensing satellite image. *Journal of the Indian Society of Remote Sensing*, 50(6), 961-973.
 - [43] N. Dalal and B. Triggs, 'Histograms of oriented gradients for human detection', *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. I, pp. 886–893, 2005, doi: 10.1109/CVPR.2005.177.
 - [44] Y. Lecun, Y. Bengio, and G. Hinton, 'Deep learning', *Nature* 2015 521:7553, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.
 - [45] A. Coluccia *et al.*, 'Drone-vs-bird detection challenge at IEEE AVSS2019', *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2019*, 2019, doi: 10.1109/AVSS.2019.8909876.
 - [46] C. Aker and S. Kalkan, 'Using deep networks for drone detection', *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2017*, no. August, 2017, doi: 10.1109/AVSS.2017.8078539.
 - [47] J. Peng, C. Zheng, P. Lv, T. Cui, Y. Cheng, and L. Si, 'Using images rendered by PBRT to train faster R-CNN for UAV detection', *Computer Science Research Notes*, vol. 2802, no. May, pp. 13–18, 2018, doi: 10.24132/CSRN.2018.2802.3.
 - [48] Z. Zhang, Y. Cao, M. Ding, L. Zhuang, and Z. Wang, 'Spatial and Temporal Context Information Fusion Based Flying Objects Detection for Autonomous Sense and Avoid', *2018 International Conference on Unmanned Aircraft Systems, ICUAS 2018*, pp. 569–578, 2018, doi: 10.1109/ICUAS.2018.8453295.
 - [49] A. Rozantsev, V. Lepetit, and P. Fua, 'Detecting Flying Objects Using a Single Moving Camera', *IEEE Trans Pattern Anal Mach Intell*, vol. 39, no. 5, pp. 879–892, 2017, doi: 10.1109/TPAMI.2016.2564408.
 - [50] E. Unlu, E. Zenou, N. Riviere, and P. E. Dupouy, 'Deep learning-based strategies for the detection and tracking of drones using several cameras', *IPSI Transactions on Computer Vision and Applications*, vol. 11, no. 1, 2019, doi: 10.1186/s41074-019-0059-x.
 - [51] T. Connie, M. Al-Shabi, W. P. Cheah, and M. Goh, 'Facial Expression Recognition Using a Hybrid CNN–SIFT Aggregator', *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10607 LNAI, pp. 139–149, Nov. 2017, doi: 10.1007/978-3-319-69456-6_12.
 - [52] J. Baillieul and A. Suri, 'Information Patterns and Hedging Brockett's Theorem in Controlling Vehicle Formations', *Proceedings of the IEEE Conference on*

- Decision and Control*, vol. 1, pp. 556–563, 2003, doi: 10.1109/CDC.2003.1272622.
- [53] Smith, S. L., Broucke, M. E., & Francis, B. A. (2006, July). Stabilizing a multi-agent system to an equilateral polygon formation. In *Proc. of the 17th International Symposium on Mathematical Theory of Networks and Systems* (pp. 2415–2424).
 - [54] B. D. O. Anderson, C. Yu, S. Dasgupta, and A. Stephen Morse, ‘Control of a three-coleader formation in the plane’, *Syst Control Lett*, vol. 56, no. 9–10, pp. 573–578, 2007, doi: 10.1016/j.sysconle.2007.04.004.
 - [55] M. Cao, A. S. Morse, C. Yu, B. D. O. Anderson, and S. Dasgupta, ‘Controlling a triangular formation of mobile autonomous agents’, *Proceedings of the IEEE Conference on Decision and Control*, pp. 3603–3608, 2007, doi: 10.1109/CDC.2007.4434757.
 - [56] B. D. O. Anderson, M. Cao, S. Dasgupta, A. S. Morse, and C. Yu, ‘Maintaining a directed, triangular formation of mobile autonomous agents’, *Commun Inf Syst*, vol. 11, no. 1, pp. 1–16, 2011, doi: 10.4310/CIS.2011.V11.N1.A1.
 - [57] M. Cao, C. Yu, A. S. Morse, B. D. O. Anderson, and S. Dasgupta, ‘Generalized Controller for Directed Triangle Formations’, *IFAC Proceedings Volumes*, vol. 41, no. 2, pp. 6590–6595, Jan. 2008, doi: 10.3182/20080706-5-KR-1001.01111.
 - [58] F. Dörfler and B. Francis, ‘Geometric analysis of the formation problem for autonomous robots’, *IEEE Trans Automat Contr*, vol. 55, no. 10, pp. 2379–2384, Oct. 2010, doi: 10.1109/TAC.2010.2053735.
 - [59] M. C. Park, B. D. O. Anderson, Z. Sun, and H. S. Ahn, ‘Stability analysis on four agent tetrahedral formations’, *Proceedings of the IEEE Conference on Decision and Control*, vol. 2015-February, no. February, pp. 631–636, 2014, doi: 10.1109/CDC.2014.7039452.
 - [60] Anderson, B. D., Yu, C., Fidan, B., & Hendrickx, J. M. (2008). Rigid graph control architectures for autonomous formations. *IEEE Control Systems Magazine*, 28(6), 48–63.
 - [61] K. K. Oh, M. C. Park, and H. S. Ahn, ‘A survey of multi-agent formation control’, *Automatica*, vol. 53, pp. 424–440, Mar. 2015, doi: 10.1016/J.AUTOMATICA.2014.10.022.
 - [62] C. Yu, B. D. O. Anderson, S. Dasgupta, and B. Fidan, ‘Control of minimally persistent formations in the plane’, *SIAM J Control Optim*, vol. 48, no. 1, pp. 206–233, 2009, doi: 10.1137/060678592.
 - [63] M. C. Park, B. Y. Kim, K. K. Oh, and H. S. Ahn, ‘Control of inter-agent distances in cyclic polygon formations’, *2012 IEEE Multi-Conference on Systems and Control, MSC 2012*, pp. 951–955, 2012, doi: 10.1109/ISIC.2012.6398247.
 - [64] M. C. Park, Z. Sun, K. K. Oh, B. D. O. Anderson, and H. S. Ahn, ‘Finite-time convergence control for acyclic persistent formations’, *2014 IEEE International Symposium on Intelligent Control, ISIC 2014*, pp. 1608–1613, Nov. 2014, doi: 10.1109/ISIC.2014.6967627.
 - [65] S. Zhao and D. Zelazo, ‘Bearing Rigidity and Almost Global Bearing-Only Formation Stabilization’, *IEEE Trans Automat Contr*, vol. 61, no. 5, pp. 1255–1268, May 2016, doi: 10.1109/TAC.2015.2459191.

- [66] G. Michieletto, A. Cenedese, and A. Franchi, ‘Bearing rigidity theory in $SE(3)$ ’, *2016 IEEE 55th Conference on Decision and Control, CDC 2016*, pp. 5950–5955, Dec. 2016, doi: 10.1109/CDC.2016.7799182.
- [67] S. Zhao and D. Zelazo, ‘Bearing Rigidity Theory and Its Applications for Control and Estimation of Network Systems: Life beyond Distance Rigidity’, *IEEE Control Syst*, vol. 39, no. 2, pp. 66–83, Apr. 2019, doi: 10.1109/MCS.2018.2888681.
- [68] S. Zhao, ‘Affine Formation Maneuver Control of Multiagent Systems’, *IEEE Trans Automat Contr*, vol. 63, no. 12, pp. 4140–4155, Dec. 2018, doi: 10.1109/TAC.2018.2798805.
- [69] R. Tron, J. Thomas, G. Loianno, K. Daniilidis, and V. Kumar, ‘Bearing-only formation control with auxiliary distance measurements, leaders, and collision avoidance’, *2016 IEEE 55th Conference on Decision and Control, CDC 2016*, pp. 1806–1813, Dec. 2016, doi: 10.1109/CDC.2016.7798527.
- [70] R. Tron, ‘Bearing-Based Formation Control with Second-Order Agent Dynamics’, *Proceedings of the IEEE Conference on Decision and Control*, vol. 2018-December, pp. 446–452, Jan. 2019, doi: 10.1109/CDC.2018.8619393.
- [71] M. H. Trinh, S. Zhao, Z. Sun, D. Zelazo, B. D. O. Anderson, and H. S. Ahn, ‘Bearing-Based Formation Control of a Group of Agents with Leader-First Follower Structure’, *IEEE Trans Automat Contr*, vol. 64, no. 2, pp. 598–613, Feb. 2019, doi: 10.1109/TAC.2018.2836022.
- [72] K. K. Oh and H. S. Ahn, ‘Orientation alignment based formation control in multi-agent systems’, *Proceedings of 2012 8th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications, MESA 2012*, pp. 42–45, 2012, doi: 10.1109/MESA.2012.6275534.
- [73] M. C. Park, H. K. Kim, and H. S. Ahn, ‘Rigidity of distance-based formations with additional subtended-angle constraints’, *International Conference on Control, Automation and Systems*, vol. 2017-October, pp. 111–116, Dec. 2017, doi: 10.23919/ICCAS.2017.8204427.
- [74] K. M. Kabore and S. Güler, ‘Deep Learning Based Formation Control of Drones’ in A. Koubaa, A.T. Azar, (eds) *Deep Learning for Unmanned Systems, Studies in Computational Intelligence*, vol. 984. Springer, Cham, 2021, pp. 383–413. https://doi.org/10.1007/978-3-030-77939-9_11.
- [75] B. D. O. Anderson, C. Yu, S. Dasgupta, and A. Stephen Morse, ‘Control of a three-coleader formation in the plane’, *Syst Control Lett*, vol. 56, no. 9–10, pp. 573–578, Sep. 2007, doi: 10.1016/J.SYSCONLE.2007.04.004.
- [76] E. D. Sontag, ‘Remarks on stabilization and input-to-state stability’, *Proceedings of the IEEE Conference on Decision and Control*, vol. 2, pp. 1376–1378, 1989, doi: 10.1109/CDC.1989.70365.
- [77] S. Thrun, *Probabilistic robotics*. Cambridge, Massachusetts: The MIT Press, 2006.
- [78] N. O’Mahony *et al.*, ‘Deep Learning vs. Traditional Computer Vision’, in *Advances in Intelligent Systems and Computing*, Springer International Publishing, 2019, pp. 128–144. doi: 10.1007/978-3-030-17795-9_10.

- [79] M. Saska *et al.*, ‘System for deployment of groups of unmanned micro aerial vehicles in GPS-denied environments using onboard visual relative localization’, *Auton Robots*, vol. 41, no. 4, pp. 919–944, 2017.
- [80] V. Walter, M. Saska, and A. Franchi, ‘Fast Mutual Relative Localization of UAVs using Ultraviolet LED Markers’, *2018 International Conference on Unmanned Aircraft Systems, ICUAS 2018*, pp. 1217–1226, Aug. 2018, doi: 10.1109/ICUAS.2018.8453331.
- [81] R. Girshick, ‘Fast r-cnn’, in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.
- [82] S. Ren, K. He, R. Girshick, and J. Sun, ‘Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks’, *arXiv:1506.01497v3*, Accessed: Jan. 29, 2022. [Online]. Available: <https://github.com/>
- [83] J. Redmon and A. Farhadi, ‘YOLOv3: An Incremental Improvement’, *arXiv preprint arXiv:1804.02767*, 2018.
- [84] Redmon, J. (2016). YOLO: Real Time Object Detection, [online]< <https://github.com/pjreddie/darknet/wiki>. YOLO:-Real-Time-Object-Detection>[Accessed 07 June 2019].
- [85] S. Guler, M. Abdelkader, and J. S. Shamma, ‘Peer-to-Peer Relative Localization of Aerial Robots With Ultrawideband Sensors’, *IEEE Transactions on Control Systems Technology*, pp. 1–16, 2020, doi: 10.1109/tcst.2020.3027627.
- [86] J. Redmon and A. Farhadi, ‘YOLO9000: Better, Faster, Stronger’, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 7263–7271.
- [87] J. Redmon and A. Farhadi, ‘YOLOv3: An Incremental Improvement’, 2018, [Online]. Available: <http://arxiv.org/abs/1804.02767>
- [88] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, ‘YOLOv4: Optimal Speed and Accuracy of Object Detection’, *arXiv preprint arXiv:2004.10934*, 2020.
- [89] G. Jocher, A. Chaurasia, and J. Qiu, ‘YOLOv5’, 2020. [Online]. Available: <https://github.com/ultralytics/yolov5>
- [90] Meituan, ‘YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications’, *arXiv preprint arXiv:2209.02976*, 2022.
- [91] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, ‘YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors’, *arXiv preprint arXiv:2207.02696*, 2022.
- [92] Ultralytics, ‘YOLOv8: Next Generation Object Detection Model’, 2023. [Online]. Available: <https://github.com/ultralytics/yolov8>
- [93] M. S. de Queiroz, X. Cai, and M. Feemster, ‘Formation control of multi-agent systems: a graph rigidity approach’, p. 190, Accessed: Jan. 31, 2022. [Online]. Available: <https://www.wiley.com/en-us/Formation+Control+of+Multi+Agent+Systems%3A+A+Graph+Rigidity+Approach-p-9781118887448>
- [94] S. M. Kang, M. C. Park, B. H. Lee, and H. S. Ahn, ‘Distance-based formation control with a single moving leader’, *Proceedings of the American Control Conference*, pp. 305–310, 2014, doi: 10.1109/ACC.2014.6858587.

- [95] H.-S. Ahn, ‘Moving Formation’, *Studies in Systems, Decision and Control*, vol. 205, pp. 271–295, 2020, doi: 10.1007/978-3-030-15187-4_11.
- [96] ‘OpenCV - Open Computer Vision Library’. Accessed: Nov. 12, 2024. [Online]. Available: <https://opencv.org/>
- [97] B. N. Vo and W. K. Ma, ‘The Gaussian mixture probability hypothesis density filter’, *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4091–4104, Nov. 2006, doi: 10.1109/TSP.2006.881190.
- [98] A. Wasik, P. U. Lima, and A. Martinoli, ‘A robust localization system for multi-robot formations based on an extension of a Gaussian mixture probability hypothesis density filter’, *Auton Robots*, vol. 44, no. 3–4, pp. 395–414, 2020, doi: 10.1007/s10514-019-09860-5.
- [99] T.-M. Nguyen, Z. Qiu, M. Cao, T. H. Nguyen, and L. Xie, ‘Single Landmark Distance-Based Navigation’, *IEEE Transactions on Control Systems Technology*, vol. 28, no. 5, pp. 2021–2028, Sep. 2020, doi: 10.1109/test.2019.2916089.
- [100] K. M. Kabore and S. Güler, ‘Practical Formation Acquisition Mechanism for Nonholonomic Leader-follower Networks’, *Proceedings of the 19th International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, pp. 330–339, 2022, doi: 10.5220/0011320200003271.
- [101] ‘lrse/whycon: Vision-based external localization system. NOTE: not currently being developed’. Accessed: Nov. 13, 2024. [Online]. Available: <https://github.com/lrse/whycon>
- [102] C. D. Melita, D. Longo, G. Muscato, and G. Giudice, ‘Measurement and exploration in volcanic environments’, in *Handbook of Unmanned Aerial Vehicles*, Springer Dordrecht, 2015, pp. 2667–2692.
- [103] M. Kulbacki *et al.*, ‘Survey of drones for agriculture automation from planting to harvest’, in *2018 IEEE 22nd International Conference on Intelligent Engineering Systems (INES)*, 2018, pp. 353–358.
- [104] K. Kanistras, G. Martins, M. J. Rutherford, and K. P. Valavanis, ‘A survey of unmanned aerial vehicles (UAVs) for traffic monitoring’, in *2013 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2013, pp. 221–234.
- [105] L. Merino, J. R. Martínez-de Dios, and A. Ollero, ‘Cooperative unmanned aerial systems for fire detection, monitoring, and extinguishing’, *Handbook of Unmanned Aerial Vehicles*, pp. 2693–2722, 2015.
- [106] B. Convens, K. Merckaert, M. M. Nicotra, R. Naldi, and E. Garone, ‘Control of fully actuated unmanned aerial vehicles with actuator saturation’, *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 12715–12720, 2017.
- [107] A. Prorok, ‘Graph Neural Networks for Learning Robot Team Coordination’, May 2018, Accessed: Nov. 13, 2024. [Online]. Available: <https://arxiv.org/abs/1805.03737v2>
- [108] L. Zhou *et al.*, ‘Graph Neural Networks for Decentralized Multi-Robot Target Tracking’, *SSRR 2022 - IEEE International Symposium on Safety, Security, and Rescue Robotics*, pp. 195–202, 2022, doi: 10.1109/SSRR56537.2022.10018712.

- [109] H. Nguyen and H. La, ‘Review of Deep Reinforcement Learning for Robot Manipulation’, *Proceedings - 3rd IEEE International Conference on Robotic Computing, IRC 2019*, pp. 590–595, Mar. 2019, doi: 10.1109/IRC.2019.00120.
- [110] K. M. Kabore and S. Güler, “Distributed Formation Control of Drones with Onboard Perception,” *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 5, pp. 3121-3131, Oct. 2022, doi: 10.1109/TMECH.2021.3110660.
- [111] K. M. Kabore and S. Güler, “Efficient relative localization and coordination system for unmanned ground vehicle formations under directed graph structure”, *Robotica*, 2025 (To appear).



CURRICULUM VITAE

2012 – 2016	B.Sc., Computer Engineering, Erciyes University, Kayseri, TURKEY
2016 – 2018	M.Sc., Electrical and Computer Engineering, Abdullah Gül University, Kayseri, TURKEY
2019 – 2024	Doctoral Candidate, Electrical and Computer Engineering, Abdullah Gül University, Kayseri, TURKEY
2024	Ph.D., Electrical and Computer Engineering, Abdullah Gül University, Kayseri, TURKEY

SELECTED PUBLICATIONS AND PRESENTATIONS

- J1)** Kader M. Kabore and Samet Güler, “Distributed formation control of drones with onboard perception”, *IEEE/ASME Transactions on Mechatronics*, vol. 27, no:5, pp. 3121-3131, 2021.
- J2)** Kader M. Kabore and Samet Güler, “Efficient relative localization and coordination system for unmanned ground vehicle formations under directed graph structure”, *Robotica* (To appear).
- C1)** Kader M. Kabore and Samet Güler, “Practical Formation Acquisition Mechanism for Nonholonomic Leader-follower Networks”, 19th International Conference on Informatics in Control, Automation and Robotics (ICINCO), Lizbon, Portekiz, 2022.

Book Chapter

- B1)** Kader M. Kabore and Samet Güler, “Deep Learning Based Formation Control of Drones”, in A. Koubaa, Ahmed T. Azar (edt), *Deep Learning for Unmanned Systems*, Springer Nature, pp. 383-413, 2020.